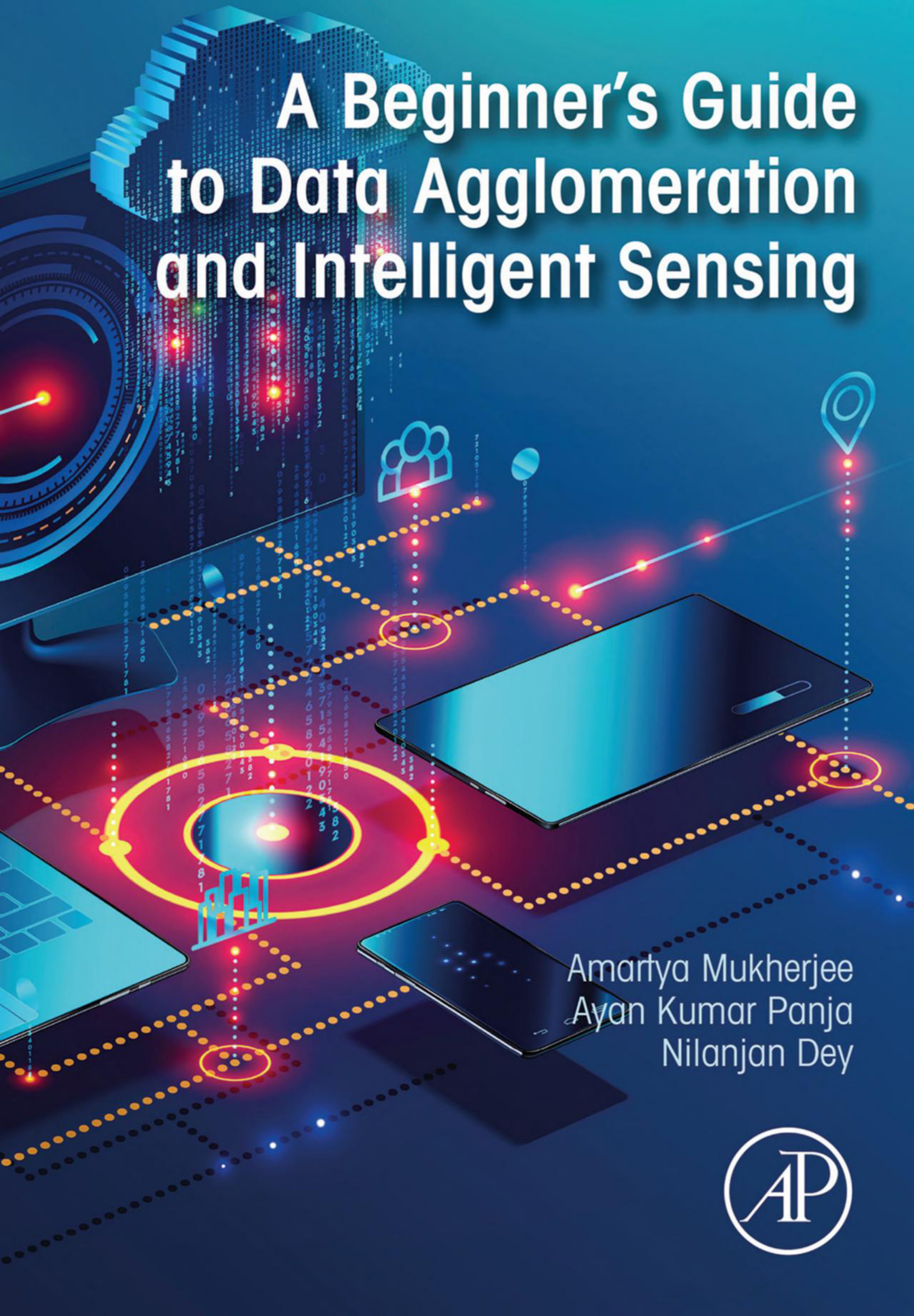
A Beginner's Guide to Data Agglomeration and Intelligent Sensing



Amartya Mukherjee
Ayan Kumar Panja
Nilanjan Dey

A Beginner's Guide to Data Agglomeration and Intelligent Sensing

A Beginner's Guide to Data Agglomeration and Intelligent Sensing

AMARTYA MUKHERJEE
Department of Computer Science & Engineering and BSH,
Institute of Engineering and Management,
Kolkata, India

AYAN KUMAR PANJA
Department of Computer Science & Engineering and BSH,
Institute of Engineering and Management,
Kolkata, India

NILANJAN DEY
Department of Information Technology,
Techno International New Town, Kolkata, India

ELSEVIER

ACADEMIC PRESS
An imprint of Elsevier

Academic Press is an imprint of Elsevier
125 London Wall, London EC2Y 5AS, United Kingdom
525 B Street, Suite 1650, San Diego, CA 92101, United States
50 Hampshire Street, 5th Floor, Cambridge, MA 02139, United States
The Boulevard, Langford Lane, Kidlington, Oxford OX5 1GB, United Kingdom

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library

Library of Congress Cataloging-in-Publication Data
A catalog record for this book is available from the Library of Congress

ISBN: 978-0-12-820341-5

Publisher: Mara Conner
Acquisitions Editor: Chris Katsaropoulos
Editorial Project Manager: John Leonard
Production Project Manager: Swapna Srinivasan
Cover Designer: Mark Rogers

Typeset by MPS Limited, Chennai, India

Contents

About the authors

Amartya Mukherjee is an assistant professor at the Institute of Engineering and Management, Salt Lake, Kolkata, India. He holds a bachelor's degree in Computer Science and Engineering from West Bengal University of Technology and a master's degree in Computer Science and Engineering from the National Institute of Technology, Durgapur, and West Bengal, India. His primary research interest is in embedded application development, including mobile ad-hoc networking, delay-tolerant networks, and Internet of things and machine learning. He has written several research articles, books, and book chapters in the field of wireless networking and embedded systems in various journals and publication houses such as Springer, CRC Press (Taylor & Francis Group), Elsevier, IEEE, World Scientific, and IGI Global. His book *Embedded systems and robotics with open source tools* is one of the bestselling books in the field of embedded application development.

Ayan Kumar Panja is an assistant professor of computer science at Institute of Engineering & Management, Salt Lake, Kolkata, India. He has done his Bachelor of Computer Science from University of Calcutta, MSc in Computer Science from St. Xavier's College, Kolkata, and Master of Technology from University of Calcutta. His primary research interests include localization, sensor network, and sensor cloud architecture. He has written papers on data gathering and load balancing in sensor cloud. He is the organizing committee member of several international conferences and played active role as the coconvenor of the conferences.

Nilanjan Dey is an Assistant Professor in the Department of Information Technology at Techno India College of Technology, Kolkata, India. He is a Visiting Fellow of the University of Reading, United Kingdom. He is a Visiting Professor at the Wenzhou Medical University, China, and Duy Tan University, Vietnam. He was an honorary Visiting Scientist at Global Biomedical Technologies Inc., CA, United States (2012 – 15). He was awarded his PhD in 2015 from Jadavpur University.

He has authored or edited more than 45 books with Elsevier, Wiley, CRC Press, and Springer and published more than 300 papers. He is the Editor-in-Chief of *International Journal of Ambient Computing and Intelligence*, IGI Global and Associate Editor of *IEEE Access* and *International Journal of Information Technology*, Springer. He is the Series Coeditor of *Springer Tracts in Nature-Inspired Computing*, Springer Nature, and Series Coeditor of *Advances in Ubiquitous Sensing Applications for Healthcare*, Elsevier; and Series Editor of *Computational Intelligence in Engineering Problem Solving and Intelligent Signal processing and data analysis*, CRC.

His main research interests include medical imaging, machine learning, computer aided diagnosis, data mining, and so on. He is the Indian Ambassador of International Federation for Information Processing (IFIP)—*Young ICT Group*. Recently, he has been awarded as one among the top 10 most published academics in the field of Computer Science in India (2015 – 17).

(Website: https://sites.google.com/view/drnilanjandey/home)

Preface

The wireless sensor network is significantly popular in modern engineering research. The advancement of the high-end sensing devices and micro electro mechanical systems has made a step forward to build and deploy the sensor networks for various applications such as smart city, health monitoring, ubiquitous computing, and various other fields of engineering. The sophisticated sensor equipment, network communication devices, and related routing protocol take a key role in the synthesis of a modern intelligent network system. The deployment of the smart wireless sensor network is done by maintaining various algorithms. Second, the acquisition and the agglomeration of the data in a sensor network are also a great challenge, which, perhaps, involves effective algorithm of data aggregation, localization, and efficient sensor data-mining techniques.

The fundamental objective of this book is to give the reader an insight of the current trends of intelligent sensor networks and their fundamental features such as routing, mobility design, localization, data acquisition and agglomeration, mining, and various other aspects of sensor networks. Chapter 1, Introduction to sensors and system, fundamentally describes various sensors and their functionalities. Chapter 2, Real-life application of sensors and systems, covers some important aspects of sensors and systems along with the real-life application design paradigms. Some of the well-known simulation and the experimental framework have also been discussed in this chapter. In Chapter 3, Wireless sensor network: principle and the application, the fundamental design principle and the application of the wireless sensor network have been discussed. The autonomous node deployment in the form of Unmanned Aerial Vehicle (UAV) is also a key point of interest in this chapter. Chapter 4, Overview of sensor cloud, gives an insight into the sensor cloud concept, whereas Chapter 5, Sensor data accumulation methodologies, discusses the concepts of various sensor data accumulation techniques. Chapter 6, Intelligent sensor network, on the other hand, fundamentally describes the principles of intelligent sensing. Here several case study-based approaches have also been discussed. The book itself is written in such a way that the readers who are not much familiar with the concept of sensor network and willing to do research on this can easily understand the fundamental insights of the sensor networks and systems. The book will definitely serve as a guide to the graduates and research students to get a certain level of dimension on intelligent wireless sensor networks.

Amartya Mukherjee, Ayan Kumar Panja and Nilanjan Dey

Acknowledgment

This book is itself an acknowledgment of the modern sensor network base intelligent applications. We are thankful to the researchers and their enormous research effort toward the modern sensor network and intelligent network applications.

We are also thankful for our students, coresearchers, especially *Mr. Agniva Chatterjee*, Institute of Engineering & Management, Kolkata, for his dedicated effort and contribution. We are also thankful to our *wives, children, and family members* for their continuous support.

Authors

CHAPTER ONE

Introduction to sensors and systems

Contents

Everything that we see, hear, feel, and sense is information. Interaction with the environment is essential, and interaction can only happen by the process of input and output operation. All living beings have a sense which aids them to interact with the universe. The senses collect data and the brain does the processing and coordinates our output to the universe accordingly. Every mechatronic system needs a subsystem to collect data from the environment. The information so collected is processed using a processing unit such as a microcontroller and a microprocessor. Such subsystem can be broadly classified as sensor, transducers, and signal processing devices. The sensed data are processed and accordingly actions are taken or the output is generated.

A Beginner's Guide to Data Agglomeration and Intelligent Sensing.
DOI: https://doi.org/10.1016/B978-0-12-820341-5.00001-1

1.1 Fundamentals of sensors/transducers

Sensors are defined as an electronic device or a part of the system used for sensing data, detecting events or changes, and relaying those data to a processing system (Fig. 1.1). A transducer, on the other hand, is a bit different; transducers use the principle of transduction. The transducer converts information/signal from one form of energy to another form. To specifically inculcate a comparison between a sensor and transducer, one can say a sensor senses the information from the environment and transducers convert the sensed data into a usable form so that processing can be properly done on them.

To properly understand, let us take an example of a smart water pump. The word smart is used here because this is a special pump which monitors the level of water the tank is holding and accordingly makes a decision when to switch on the pump and when to switch it off automatically.

A sensor which is capable of monitoring the level of water is fixed into the tank. The sensor data are relayed to processing specifically a microcontroller. Instructions are stored in the memory of the microcontroller and are processed accordingly. The microcontroller is connected to a relay which is connected to the switch of the water pump. The water level is sent to the microcontroller in the form of signals. The program/instruction in the microcontroller makes the decision to control the relay switch. If the water level dips below a certain level, the pump is switched on and if the water level rises above certain levels the instruction switches off the relay switch. Hence the working of the water pump is modulated using a smart controller.

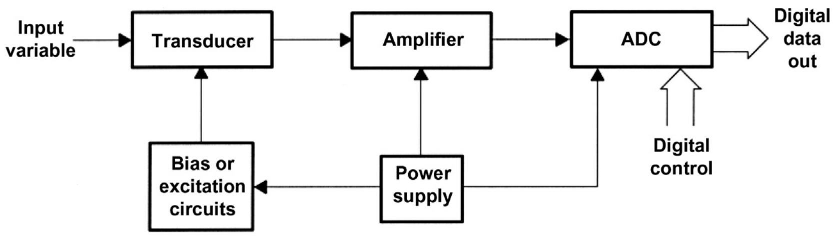


Figure 1.1 Fundamental architecture of a sensor module.

1.2 Principles and properties

The principle of sensing the data from the environment used to be physical or chemical in nature but nowadays even various types of sensors are coming into existence that not only covers physical or chemical aspects but also something more. We now live in a world where we cannot avoid these tiny little electronic gadgets. They are everywhere; one can find sensor devices in every day-to-day life, be it in offices, restaurants, industries, and so on. One such example is a shopping mall; now most of the shopping malls or complex's entrance doors are made automatic. That is whenever a person approaches or departs, the door gets automatically opened or closed accordingly. So, there is a sensor device attached that monitors the proximity of the person approaching and the direction in which he or she is approaching or departing. The processing component, specifically the microcontroller, makes the decision accordingly.

Processing is done on the information gathered from the environment. Hence the type of data that is needed to be extracted depends on the type of sensors and actuators we are using. The physical property of sensing can be weight, temperature, pressure, percentage composition, electric magnetic or electromagnetic, position and orientation, force, and so on. Depending on the type of data that needs to be extracted the sensors are created accordingly.

1.3 Classification of sensors

Now the classification of sensors not only necessarily depend on the type of information the system needs to gather for processing but also on many other factors. Sensors can be broadly classified as into:

1. Active sensors
2. Passive sensors

An active sensor uses an external or self-generated signal for measurement. Passive sensors change their properties for various occurring events, and they detect and respond to some types of input from the physical environment. Another way of classifying the sensors can be application-based, which is a very convenient way to show the segmentation in a very broad manner. With respect to application-based classification, sensors can be classified as

1. Industrial based such as automation and process control measurement
2. Nonindustrial based such as automobiles, hospitals, consumer electronics, and so on.

Now before proceeding with the type of sensors one first needs to know the type of processing, that is, various types of systems. Systems can be broadly classified as

1. Real-time system
2. Batch processing system

A real-time system is a reactive system, that is, the system reacts immediately within a specified time, that is, time-bound with respect to the various environmental stimulus generated. Example of such a system can be a traffic system, weather monitoring system. A batch processing system is one where real-time monitoring or processing is not required and the information is processed in a set of inputs. With respect to the type of system, the sensors are selected accordingly. Table 1.1 illustrates several factors for sensor design.

In the current world scenario, we have various types of sensor. And according to the need of the industries and projects, many types of sensor devices are created. We have proximity sensors that are capable of detecting the presence of nearby objects without any physical contact. Proximity sensors use the property of sonar by sending ultrasound and monitoring the time in which the sound again reverts back. A temperature sensor monitors the temperature.

There are various open-source platforms used for building electronics projects. They consist of a physical programmable circuit board and a piece of software that runs on the computer. One of such popular open-source platform is Arduino. With the help of these open-source platforms, one can develop various projects on sensor technologies. On the latter part of the book, we will discuss these technologies in detail and also explain how one can develop an intelligent system.

Table 1.1 Different factors for sensor design approach.

Environmental factors	Economic factors	Sensor characteristics
Temperature	Cost	Sensitivity
Humidity	Availability	Range
Proximity	Lifetime	Stability
Power consumption		Response time
Self-test capability, etc.		Error

1.4 Networking methodology

The sensors' duty is just to sense the data and modify them into a form in which the microcontroller can perform the processing. One of the major concern remains as to how the data will be relayed to the processing unit. If we take into consideration a complex system containing many subsystems, communication is very essential so that a proper coordination can be achieved between the subsystems. The selection and use of telecommunication protocol and computer hardware for using and managing the network, and the establishment of operation policies and procedure related to the network.

The network methodology can be broadly classified as

1. Wired technology
2. Wireless technology

In wired technology, Sensors and the subsystems are connected using Ethernet, Fiber optic, etc. We know the network operates using various other connecting units that includes computers, routers, hubs, gateways, and access points. We know networks can be broadly classified into local area network, wide area network, and metropolitan area network (MAN). IEEE 802 is the family of IEEE standards dealing with local area networks and MAN. Data are transmitted in the form of packets, packets are fixed or variable-sized. In the case of wireless technology, information is transferred in the form of electromagnetic wave. The set of media access control (MAC) for wireless local area network is IEEE 802.11.

1.4.1 Fifth-generation technology

The most emerging mobile technology is the fifth-generation technology (5G) [1], and mobile communication can also be addressed in this context. The 5G technology has been designed in such a way that intelligent sensor network [Internet of things (IoTs)] and all other smart communication infrastructure are combined together in order to make a highly robust and efficient network communication among everything. One of the most effective approaches, in order to achieve extremely high speed, is the millimeter wave. In this case, the frequency of the signal varies from 3 to 300 GHz. There is an opportunistic offloading mechanism is allowed here to utilize unlicensed spectrums like 5G Wi-Fi. In the current scenario, the carrier frequency of the network saturated within 750–2.6 GHz for the universal mobile telecommunications system (UMTS) and long term evolution (LTE) networks. The 5G technique therefore mostly used the underutilized frequency channels of the physical layer including multiple input, multiple output (MIMO), traffic offloading, beamforming, and cloudification of the wireless radio resources. By 2020, the concept of vehicular ad-hoc network (VANET) and flying ad-hoc network (FANET) [2−4] will grow exponentially, and they will integrate with cellular networks and give a VANET and FANET cloud service that will ensure a highly robust ground, air, and deep-space transportation system. Cisco Inc. published a white paper in 2015 that stated that the monthly global mobile data traffic will pass 25 exabyte by 2019. Under these circumstances, the use of the 5G network is the relevant and obvious choice. The fundamental objective of this technology is to achieve coverage of 200 m in a single cell with an operating carrier frequency of 38 GHz. It has also been observed that in 99% of root mean square (RMS) delay for the urban environment is 129 ns. It is obvious that the design of the 5G technology serves macro layer coverage by using 30 times higher sampling frequency. In 5G design, the minimum bandwidth is considered as 1000 MHz. As the frequency is high the signal will carry the huge amount of information and perhaps the signal may attenuate fast. One of the advantages of this transmission is that the sizes of the antenna get reduced drastically as we are using millimeter waves. The design of the 5G network, therefore, is emphasized on deploying multiple antennas in a single handset and the distributed micro antenna array in order to cover the big region. This can be done in two popular ways.

1. **Beamforming:** The beamforming or spatial filtering is one of the significant approaches in order to get better connectivity and reduced latency in the network. Fig. 1.2 shows a practical beamforming scenario. In this approach, a single access point uses various antennas to transmit the exact same signal. By using this management technique the wireless networks can easily modify the transmitted signals and also get the ideal path of its transmission. The algorithm was designed in such a way that the beam gets set into the direction toward the receiver. The technique has various advantages such as a high signal-to-noise ratio (SNR), and therefore it offers an improved transmission rate in open space as well as indoor. The network efficiency is extremely high as it prevails over internal and external cochannel interference. The technology is highly suitable for vehicular networks and the self-driving car that can be controlled remotely with high precision. In such a case, the street light poles can be used as transmission

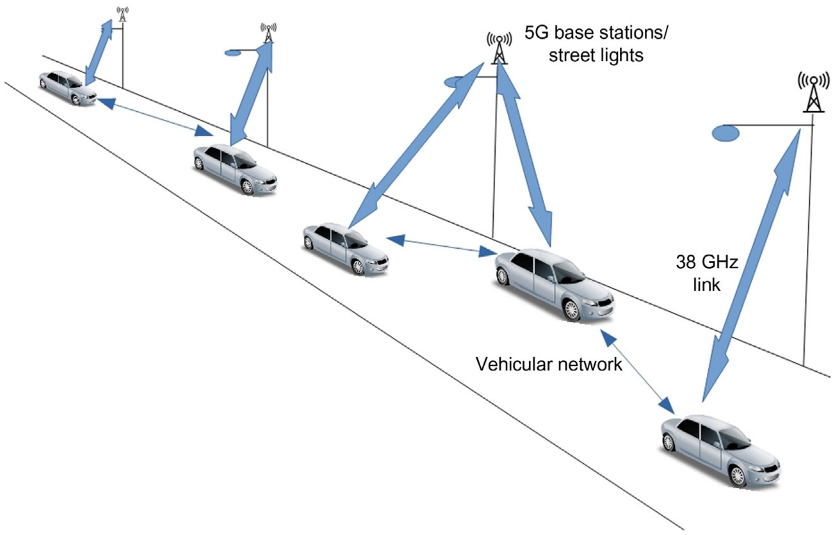


Figure 1.2 Beamforming scenario.

antenna or base station (BS) subsystems. The vehicle will also form an ad hoc communication among each other to transfer real-time information, and as a result, the transmission speed increases drastically and the data delivery latency will be decreased.

2. **Massive MIMO technique:** The main idea behind the deployment of the 5G network is to implement massive antenna and ultra-dense access point service. This results in the design of the massive multiple-input multiple-output framework [5]. In this concept hundred BS units serves couple to users in the same time-frequency resource block. A large number of antennas, in this case, provide an effective way to avoid channel fading, and the beamforming also becomes sharp. The small cell configuration significantly improves the density of the access points, and the inter distance between the transmitter unit and receiver unit will also reduce.

The network structure, in this case, is mostly of heterogeneous type because here the target is to build an ultra-dense small cell infrastructure that overlays the existing system. It is obvious that the deployment of hundreds of antennas and the terminal itself is a challenge. This requires an advanced processing capability for each BS node. One of the potential solutions for this is to deploy a hybrid cell pattern where a single cell may comprise both the macro BS system as well as the locally

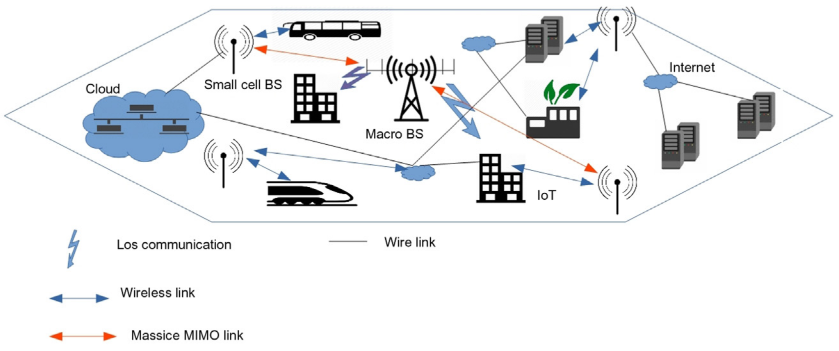


Figure 1.3 Massive MIMO-based 5G infrastructure.

clustered small cell BS. By doing this, the network coverage and the efficiency will increase drastically and thereby the performance will also increase. The massive MIMO-based 5G network structure is depicted in Fig. 1.3.

1.4.2 Bluetooth low energy

Bluetooth low energy (BLE) is a paramount technology that can operate in significantly low power and performs wireless communication. BLE [6] is basically designed to address the trade-off between latency, power consumption, and the throughput of the network in a piconet scenario. BLE technology is basically an enhancement of the classical bluetooth technology, and the protocol stack of the same is also redefined. Fig. 1.4 shows the modified protocol stack. According to a prediction it is expected that the BLE technology will be used by millions of handheld devices in the near future. The protocol stack of the BLE is quite similar to the classic bluetooth protocol stack that consists of a host and a controller. The physical layer and the MAC layer (link layer) are typically designed as in the form of system on chip (SoC). The communication between the host and the controller has been designed in the form of a host controller interface mechanism.

The physical layer of BLE deals with the signal and the frequency spectrums. Typically, in this case, the operation frequency is 2.4 GHz industrial, scientific, and medical (ISM) band. It offers 40 separate channels with 2 MHz band separation. Two types of the channel are reserved. First is the advertisement channel, whose sole responsibility is the device discovery, and the second is the data channel for transferring of the

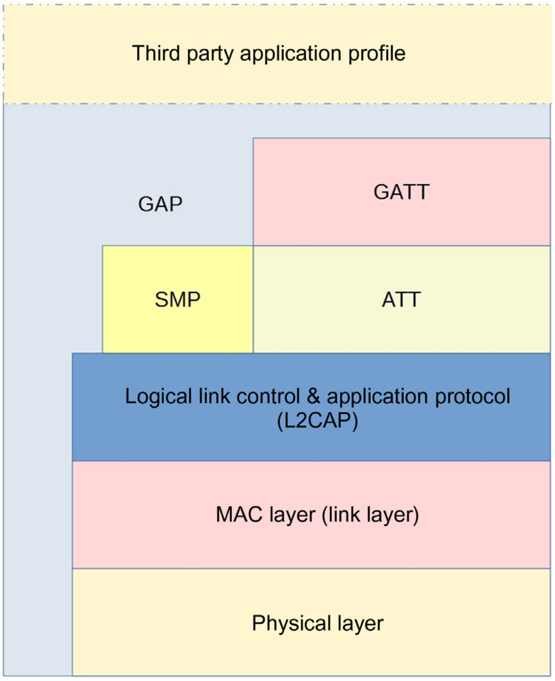

Third party application profile
GAP
GATT
SMP
ATT
Logical link control & application protocol (L2CAP)
MAC layer (link layer)
Physical layer

Figure 1.4 Bluetooth tooth low energy protocol stack.

information. The Gaussian frequency shift keying technique is primarily used by the physical layer.

Link layer (MAC layer) performs the connection control operation. The primary target, in this case, is to connect the devices in a master—slave mode and in order to save the energy the slave devices have been set to sleep mode by default. The medium access coordination, in this case, has been performed by using time division multiple access mechanisms.

L2CAP layer in BLE is an extended version of classic bluetooth L2CAP layer. The main job here is to multiplex the data and make available into the next higher layer. In this layer, the best-effort approach is used without the retransmission and the flow control operation.

Attribute protocol (ATT) on the other hand defines the communication between the client and the server devices and generic attribute profile provides a way that uses by ATT for service discovery.

The job of security manager protocol is to ensure the security on the protocol level using cipher block-chaining message authentication codes. The Message integrity check has also been done in this case.

1.5 Types of sensors

The word "sensor" is very common nowadays. The use of sensors in different cutting edge application is highly remarkable with the introduction of wireless sensor networks (WSNs), IoTs, Internet of vehicle, smart computing devices, smart grid, and airborne sensor networks. In the near future the advancement of system on chip (SoC) and network on chip (NoC) technology guaranties micro-level sensor design. In this chapter, our main topic of discussion is about the various classes of sensors based on sensing under different energy sources. The different classes of sensors are taxonomically represented in Fig. 1.5.

1.5.1 Sound energy-based sensing

There are two classes of sensors that fall under this category. The first one is the sound sensors that typically capture the sound waves under an audible range between 20 Hz and 20 kHz. These classes of sound sensor modules act as a dynamic microphone with an additional LM386 amplifier. The operating voltage of these sensors is within the range of 3.33–5 V. The sensitivity of the microphone is typically 52–48 dB and the SNR is 54 dB.

In the second case, the most popular and perhaps widely used sensing device is the ultrasonic sensor. These sensors are mainly used for range

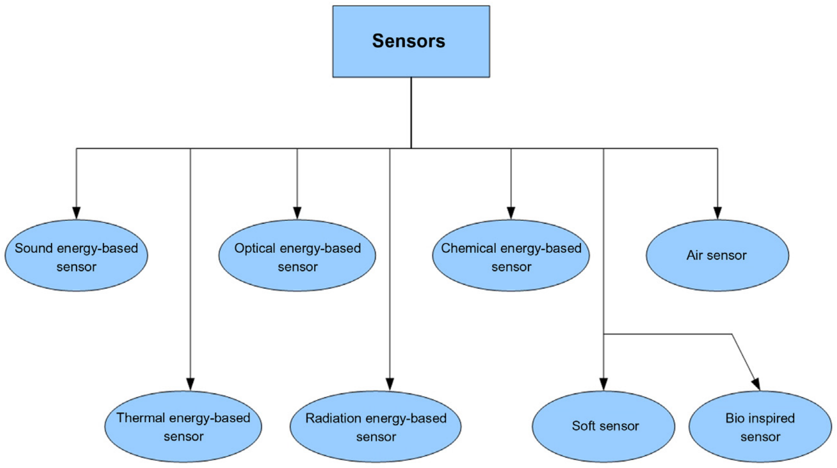

Sensors
Sound energy-based sensor
Optical energy-based sensor
Chemical energy-based sensor
Air sensor
Thermal energy-based sensor
Radiation energy-based sensor
Soft sensor
Bio inspired sensor

Figure 1.5 Classification of various kinds of sensors based on different energy sources.

finding and as distance sensors. Such sensors are feasible to be deployed underwater to create underwater WSNs [7]. Deployment of the low-cost ultrasound sensor in an underwater environment is a challenge in various hazardous environments. There are lots of practical uses of such sensors and of course, the senor networks, such as undersea ecosystem monitoring, ocean current prediction, and underwater disaster, these kinds of networks are typically useful for offshore engineering applications like oil drilling from the sea bed. Most of the remotely operated underwater vehicles use the ultrasound sensing module in order to detect the distance of the sea bed and also nearby hazardous objects. An autonomous underwater vehicle, on the other hand, navigates by considering the set of commands and instruction generated based on the sensor reading. In such a case, most of the module like smart anchor gives high precision location information based on the supplied data by ultrasonic sensors. One of the modern applications in this context is a 3D mapping of the ocean floor. In most of the cases, a single beam sonar system, which is an ultrasonic transducer, mounted on the bottom of the ship generates an ultrasound pulse. The pulse is then reflected by the floor of the ocean and received by the ultrasound receiver. In this way an echo-sounding has been done, which is similar to the mapping of the seafloor. A sample 3D seafloor mapping is depicted in Fig. 1.6.

Multi-beam sonar, on the other hand, produces a more accurate map of the seafloor. In this case, the sound energy emits in the shape of the dome and the sound signal diverges downwards. This technique is also used in signal processing often called beamforming, which is mainly used to determine the direction information from the returning signal. These

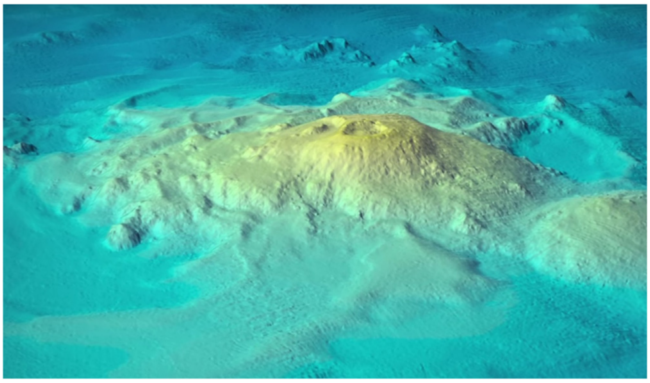


Figure 1.6 3D ocean floor mapping with ultrasound.

classes of the acoustic sensor are having wide use in biomedical sensing applications [8].

1.5.2 Thermal energy-based sensing

Temperature sensing and sensors played a vital role in various industrial and research applications. As an example, we can consider the temperature of the boiler that is a very critical application and has to be maintained and monitored in a mission-critical fashion. Not only that in the case of medicines and food supply chain management the utility of the temperature sensing is essentially required in order to maintain the quality and the safety of the food and the medicine. The fundamental thermal sensor that is popularly used is the temperature sensor LM35 and temperature–humidity sensor modules [9]. Another pretty important temperature measurement can be done by using image sensors. In this approach, a dark current noise corresponding to the image gets a sense and based on that the temperature of the image is computed.

On the other hand, the temperature–humidity sensor module DHT11 is one of the popular temperature modules that is widely used in industrial, agriculture, research, and scientific operation. Based on the temperature and the humidity, the ambient weather can be defined. For example, in 30°C if the humidity is 90%, then it may feel like 35°C. Humidity is also a crucial parameter in order to operate sensitive equipment. The most common types of humidity sensors are the capacitive humidity sensors. It primarily measures relative humidity (RH). It consists of dielectric material whose permittivity changes with the change of humidity. In the case of capacitive RH sensor, the capacitor must be filled with appropriate dielectric material called isolator. The hygroscopic polymer films are widely used as dielectric, and the construction of the two electrode layers on either side of the dielectric film is shown in Fig. 1.7(a).

Capacitive RH sensors are also useful for checking the humidity of medical equipment and medicines such as oral pills and tablets. In this case, RH sensor observes the frequency change of the oscillator caused by the test sample as dielectric material (Fig. 1.7(b)).

- **Infrared thermometer system**

  An infrared thermometer is a special type of sensor that is specifically built to detect infrared radiation, which is below the visible radiation spectrum. The major components of an infrared sensor are the

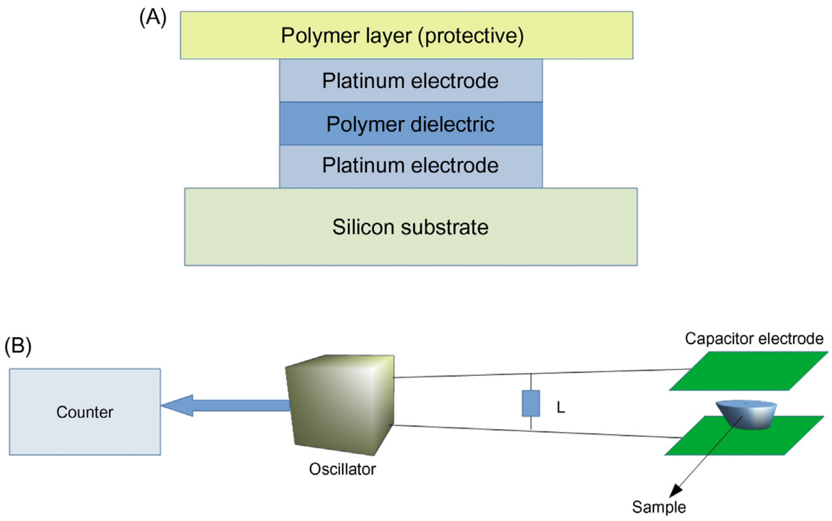


Figure 1.7 (A) Humidity temperature sensing layers. (B) The complete system with a test sample.

lens and a detector. The job of the lens is to focus on the thermal radiation toward the detector. As a result, the radiation falls into the detector and gets converted into an electrical signal. This is the way by which the temperature of an object can be detected without touching the object.

Various classes of sensors are available nowadays such as

1. **Spot infrared thermometer.** This measures the temperature of the top surface of the objects.
2. **Infrared scanner.** This class of thermometer scans larger area in comparison to spot thermometer that have a rotating mirror in it.
3. **Infrared imaging camera**. This technology is the amalgamation of the software- and hardware-based imaging technique. Here the infrared radiation is captured by the lens and converted into a two-dimensional thermogram [10].

The optical lens of the camera converges the infrared radiation into the sensor array. Each pixel of the array reacts on an infrared signal and thus generates an electrical pulse. Camera processor then takes the pulse and performs some mathematical computation to generate a color map corresponding to the object. Each temperature value, in this case, is represented by a specific color. Fig. 1.8 shows a block diagram of an infrared imaging device.

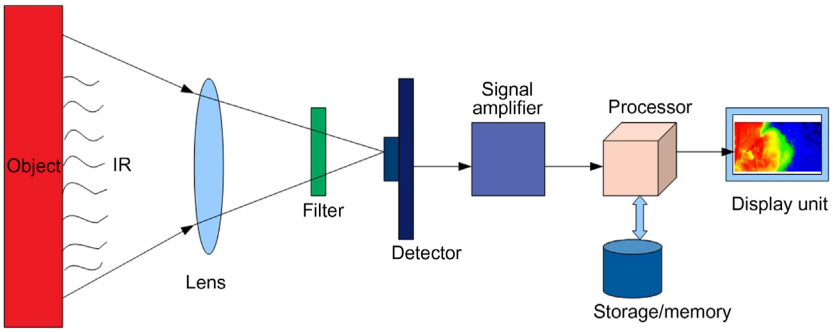


Figure 1.8 Infrared imaging devices.

IR-fusion technology is widely used to combine a normal image with an infrared image. It generates the resulting image by combining pixel to pixel alignment.

1.5.3 Optical energy-based sensing

Optical sensors are most popularly used for biometric authentication, system fault detection, gas, and liquid sensing, and many other applications. The fundamental principle of such sensor is based on light. The object that has been captured by the visible light is actually sensed by the receiver transducer. One of the popular optical sensors is fingerprint sensor system. This sensing device includes a touch panel, and inside that an integrated touch sensor circuitry is situated. The circuit generates a responsive signal pulse by detecting the contact surface associated with the fingerprint. Finally, the signal is sensed in the form of an image. The sensing circuitry further has connected with a processing circuit that generates a signal to determine that the contact has been made with a live finger or not. A basic fingerprint sensing device architecture is depicted in Fig. 1.9.

The operation is started by the detection of the finger by the array of the sensor pixel circuit. The capacitance associated with the figure touch implies a fingerprint scan. After this operation, an optical signal associated with light is reflected from the finger. The output optical beam is then detected by a photodetector sensor. Therefore, there are two distinguished images that have been produced, one for capacitive touch and the other for optical sensing, and these two data have further been processed and combined to generate a universal image signal and then the image stored in a persistence storage device.

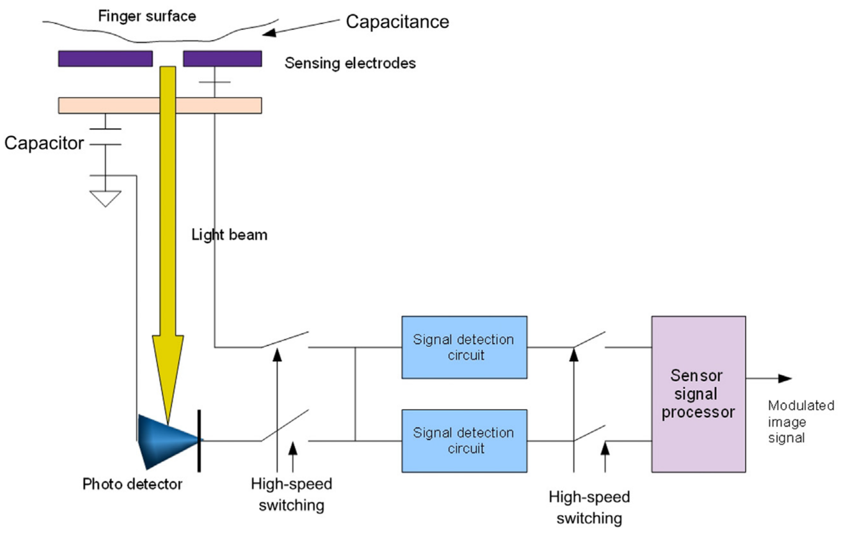


Figure 1.9 A fingerprint sensing module.

Environment monitoring is a crucial task that can be done by using an optical fiber sensing device. These classes of devices are mainly focused on the analysis of physical properties of the environmental parameters. The popular optical fiber sensor used for this scenario is the point base OFS. These sensors provide the measurement data of high sensitivity. There are several configurations of point base OFS available, such as fiber Bragg grating (FBG), long-period fiber grating and Mach–Zehnder interferometer [11].

- **Fiber grating**

  The periodic grating of optical fiber can be used sometimes as narrow-band reflector (Fig. 1.10). This technique is otherwise called FBG. This periodic grating can be done by changing the refractive index of the fiber. This can be achieved in various ways, such as electric arc, chemical etching, and laser irradiation and also by intentional fiber deformation mechanically. Periodic grating in optical fiber causes a reflection of light in a certain wavelength. Wavelength due to Bragg can be described as $\lambda_b = 2n\Lambda$. Where n is the refractive index, Λ is the periodicity. In the case of FBG sensor, based on the change of environmental parameter such as temperature and strain, the wavelength of Bragg also changes. The Bragg wavelength shift can also be measured with the following equation.

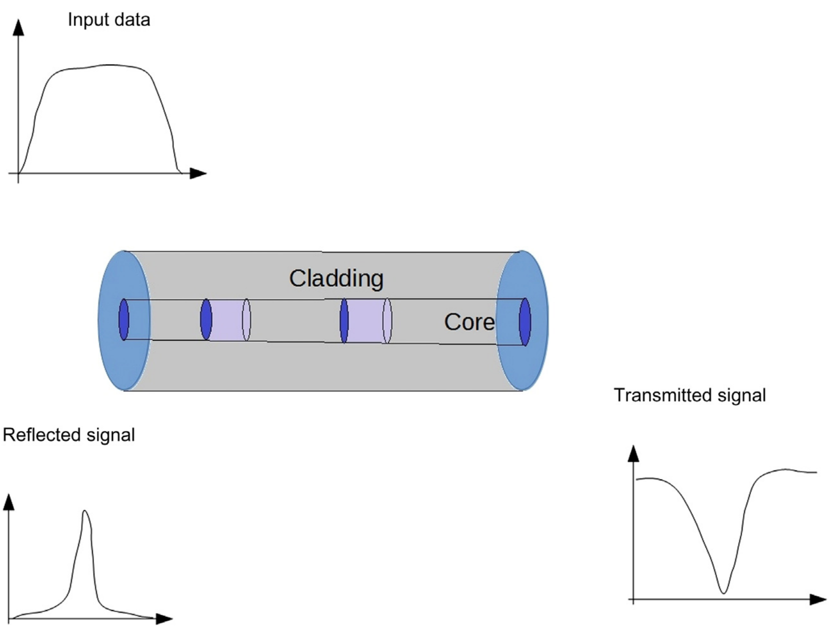


Figure 1.10 Fiber Bragg grating mechanisms.

$$\frac{\Delta\lambda_b}{\lambda_b} = K_S\varepsilon + K_T\Delta T \tag{1.1}$$

Here K_S and K_T are the strain and the temperature constant, ε is a strain, and T is the temperature. In this case, the Bragg wavelength is directly affected by temperature and the strain, therefore, it is difficult to determine the parameters using single fiber; therefore, in most of the practical scenarios, two or more fibers get Bragged. Fig. 1.10 depicts the FBG system in a nutshell.

- **Mach–Zehnder Interferometer systems**

  This class of devices uses the optical fibers as a flexible platform for environmental sensing. The fundamental principle of this sensor involves the light transmission and interference. The single-source light is therefore divided into two paths of optical fibers. The first one is connected with the sensing arm and the second is with reference arm. The job of sensing arm is to measure the environmental parameters as it is exposed to the environment. The light that propagates in two paths has to be multiplexed into a converging point coupler. At this point, the relative phase difference has been measured. The signal with

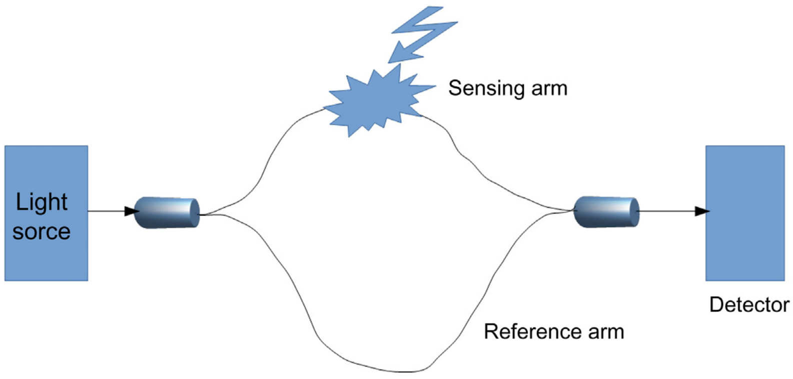


Figure 1.11 MZI sensor system architecture.

phase change contains information about the environmental parameters. There are two categories of Mach-Zehnder interferometer (MZI) systems are widely used in this context. A bulky configuration is used in beam splitting and coupling technique, on the other hand a miniaturized system is used in in-line configuration. In the case of the in-line sensor, the coupler has been designed based on cladding and core size or sometimes core type. In the case of in-line system, the divided light signals moved in core mode and cladding mode. Here core and cladding have been considered for both the sensing and the reference arm. Finally, at the second contact, the two signals get combined. Here the relative phase difference can be measured by the following expression.

$$\phi_{\mathrm{m}} = (2\pi \Delta n_{\mathrm{eff}^m} L)/\lambda \tag{1.2}$$

Here $\Delta n_{\mathrm{eff}^m}$ is the effective refractive index between core and cladding. L is the iteration length and λ is the wavelength of the light. Fig. 1.11 shows the MZI system.

1.5.4 Chemical energy-based sensing

A chemical sensor is a device that can respond with the selectivity of the particular chemical substances. The chemical sensors are basically a complex system and are mainly designed with an optimized manner for a particular substance. Based on the properties of different chemicals, the sensor reacts and produces some electrical signal. Although chemical sensor design research is growing steadily, it gets outpaced due to the diversity of the measurement of the chemicals. Further, chemical sensing is also another significant issue in material limitations. Chemical sensing can be categorized into two distinguished types namely

1. Direct reading based or selective sensing: these types of sensors work based on detecting of some substances and immediately respond to the presence of interferents. This is an idealized sensing and pretty demanding approach.
2. Using chromatographic and electrographic samples: in this case, samples are considered having various chemical reactivity properties. The reactivity includes numerous chemical phenomena, such as size, shape, and the dipolar properties of the molecule. A catalytic reaction cycle is also considered.

One of the practical and perhaps a popular example of chemical property sensing is the pH sensing. pH is the property of the material or solution through which one can identify how acidic or alkaline it is. In general pH, the term can be measured and translated by the concentration of hydrogen ion. In the pH scale, the values 0, 1, and 2 imply acidic and that consists of high concentration of hydrogen ion. The pH value above 7 signifies a small concentration of H^+. This means the solution is alkaline [12].

In order to sense the pH value, three basic components are required. A pH sensor comprises of the sensing electrode, a reference electrode, and a temperature sensing circuit. A preamplifier and an analyzer system are also incorporated. The sensor itself acts as a battery where the anode is the sensing electrode and the cathode is the reference electrode. The sensing anode is sensitive to H^+ ion. It develops a voltage corresponding to H^+ ion and related to the concentration of hydrogen ion. The reference electrode, in this case, provides a stable potential and the difference between the reference and the sensing electrode can be compared. A block diagram for pH sensing operation is shown in Fig. 1.12.

Sometimes the change in temperature also impacts on the change of H^+ ion concentration. In order to rectify such error, the additional temperature sensor is required with a pH sensor. The preamplifier is, on the other hand, performs signal conditioning, stabilize the signal and reduce the noise so that it can be easily understood by the analyzer.

1.5.5 Radioactivity sensing

In order to detect radioactivity of different object, a radiation sensor is necessary. The most used sensor in this context is the Geiger counter [13]. The main objective of the Geiger counter is to detect beta particle and gamma-ray. The main construction of the counter depends upon the tube

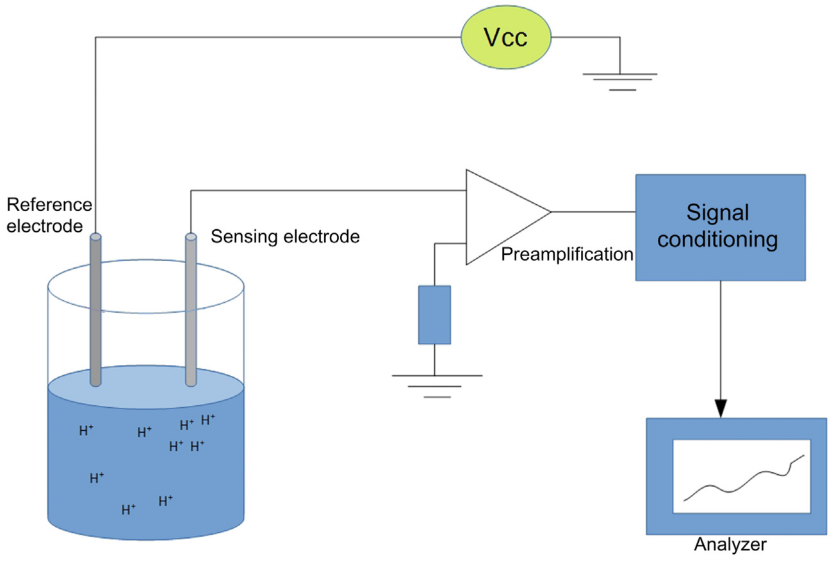


Figure 1.12 pH sensing system components.

chamber that is filled by an inert gas. This gas will be highly conductive of electricity when a high energy particle is impacted on it. When Geiger counter is exposed to radiation, the beta particle penetrates the tube and hits the gas. As a result, more electrons get released and the positively charged ions get free and negative charged electrons are attracted to the high voltage wire. When a number of electrons reach the threshold level, the current flow starts and the signal gets detected. A basic Geiger—Muller tube is depicted in Fig. 1.13.

The Geiger—Muller tube consists of an anode and the metallic wall of the tube acts as the cathode. The front of the tube is generally sealed with a mica window that allows passage of the weak penetrating mica particle. The tube should be filled with neon, argon, and halogen. In the anode terminal, a +500 V DC voltage has to be applied

1.6 Smart sensors and transducers

In today's world, the sensors are the most common and vital part to support ubiquity between different devices. This is however possible by introducing high-end technologies, such as micro-electro-mechanical

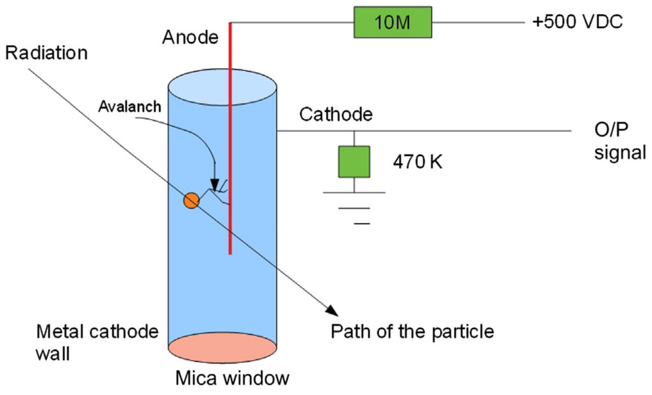


Figure 1.13 Geiger–Muller tube.

systems (MEMS), advanced silicon technology, and nano-technology. These sensor technologies also take part in enhancing the precision and the reliability of the several systems. The major part of sensors nowadays is made of silicon because it is low-cost and it is possible to mass-produce various sensors from a single manufacturing unit. As silicon is a highly versatile material, a wide range of physical phenomena can be exhibited by it so that it realizes the sensors of various properties.

The fundamental objective of the smart sensor is to extend human sensing capability to the next level. In most of the cases, the smart sensors are the dedicated electronic chip, which may be interfaced with an additional electronics unit. For example, gyroscope, accelerometer, and the internal measurement unit systems for navigation and control applications are very common in such class. We can consider an autopilot unit of an aircraft as the best example of the same. In this case, the autopilot checks the roll, pitch, and the yaw angle of the aircraft and if it is found to be beyond the threshold level it immediately corrects the angles. This is done by using gyroscope and the accelerometer sensors. Further, the pilot of the aircraft also can program the aircraft so that it maintains a proper flight path and fly by maintaining accurate direction. The autopilot unit is an open-source device and can be used to design unmanned aircraft systems and navigate with a predefined flight path.

1.6.1 Electronic interfacing for smart sensors

Smart sensors [14] are highly used in home automation, medical sensing, and so many other applications nowadays. In a smart sensor deployment, various techniques have to be combined in the overall design. The operations like sensing, analog-to-digital, and digital-to-analog conversion, signal sampling and quantization, and data processing are some of them. Features like auto-calibration, self-testing, and healing are also crucial parts of the smart sensor, which must be carried out. In order to design a low-cost and high-performance system, the object-oriented approach is best suited. In the case of sensor system design if we consider the cost-effectiveness of the sensing device then the best approach is to select the single-chip integrated circuit approach. In most of the scenarios, the sensor designer maintains either top-down or bottom-up approaches to design the sensor. But this leads to a serious limitation. This is because of the interdisciplinary subsystems that are involved with the sensors. The object-oriented design, however, gives the essence of reusability. This ensures the reuse of the powerful components of the smart sensor such as microprocessor, memory buffer, computing unit, and the subroutines and thereby exploits the auto-calibration and self-testing features. A universal transducer has to be implemented as a front-end interface for several types of sensors. Nowadays, the features of all sensing element can also be merged into a single chip resulting in a smart sensor. Fig. 1.14 shows the smart sensor abstraction based on object-oriented approach.

1.6.2 Universal transducer interface

Universal transducer interface (UTI) is a well-known transducer equipment used for multiple sensing paradigms [15]. This device has a chip-level design and comprises of a number of front-end circuitry. The system is optimized in such a way that it is frequently used for different sensors. The input connection has been connected with sensing elements, reference elements and biasing systems optionally. A data selector or often called a Multiplexer is connected sequentially with each of the modules. The multiplexer (MUX) is further controlled by a phase counter. The job of this counter is to count three, four, or five phase. From them, at least three-phase is required for the auto-calibration job. A sensor mode-specific circuit is connected with a multiplexer that consists of relaxation oscillator which is connected with a modulator that generates a square wave. The period length of the square wave, in this case, is perhaps proportional to the value supplied by the sensing element. An N-counter is also associated with the phase counter which triggered the phase counter for next phase measurement and every measurement is, however, having

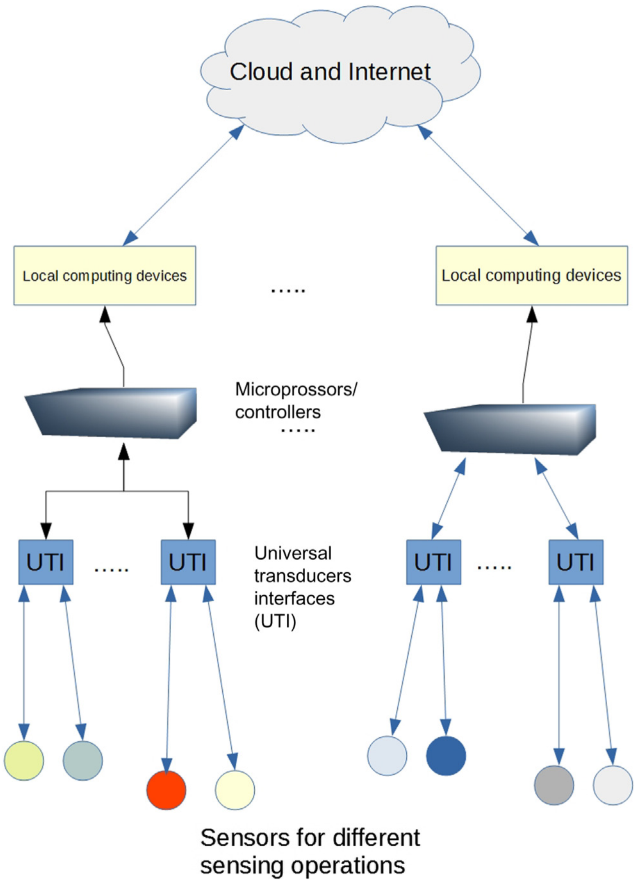


Figure 1.14 An object-oriented approach of smart sensor deployment.

a period of N. a switching is also introduced between the microcontroller and the counter to simplify the detection of the period length. Fig. 1.15 depicts a UTI.

One of the core parts of the interface is the modulator. The modulator circuit performs a determination of the relative value of the sensor capacitor and the reference capacitor present in the module. The output of the modulator actually controls the switches in order to generate square wave over the selected capacitors. There are various circuit modes that are associated with the UTI core front-end circuitry. The first mode is the resistive mode; in this case, a reference resistor is placed in series with a platinum resistor. In this case, the achieved resolution is almost constant over the complete temperature range. In the case of thermistor mode, the

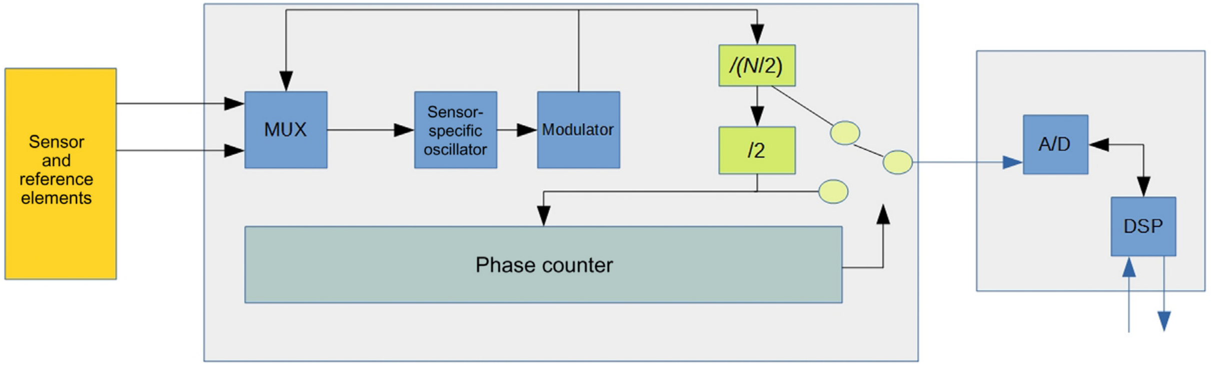


Figure 1.15 A conceptual model of UTI system.

reference resistor is definitely in a series with a thermistor. Calibration is the most crucial thing for smart sensor manufacturer as well as the consumers. Manufacturer of the sensors should maintain a well-defined calibration methodology in order to achieve a level of accuracy. In most of the cases, the smart sensors are supposed to be a plug-and-play device and therefore the calibration procedure is abstracted and encapsulated from the users.

1.6.3 IEEE 1451 smart transducers

These classes of transducers are often called smart transducers [16] that define a set of network independent and common type open communication interface for connecting sensors and the actuators with the microcontroller and several other instruments for interfacing and control. There are several protocols that have been defined in order to monitor and control the distributed application with wired as well as wireless fashion. There are several sets of commands associated with IEEE1451.0 that are highly used for distributed, multihop, and point-to-point communication protocol framework. There are some possible and perhaps popular ways to access actuators.

1. Using the IEEE 1451.0 hypertext transfer protocol.
2. Distributed multihop interface that supports 1451.3—2003.
3. A point-to-point support that fulfills IEEE 1451.2 standard.
4. The interface like Wi-Fi CAN bus, Bluetooth, and Zigbee can also be associated with several standards like IEEE1451.5, IEEE P1451.6, and IEEE P1451.7.

In most case, the transducer interface and the signal conditioning modules are not specified as within the IEEE1451 standard. Some of the interfaces like IEEE1451.4 support the transducer electronic datasheet (TED). This supports a plug-and-play service that contains the crucial information about the interface that characterizes and utilizes the signals from analog sources. The TED concept can be used in two different ways (Fig. 1.16). In the first case, the TED can be stored in the memory of the embedded device like flash memory or EEPROM. Secondly, it can be obtained as virtual service as software that is downloadable from the Internet. This method is useful when the device has no inbuilt memory. Figure 1.16c shows the standard TED content for the transducer.

IEEE 1451.1 standard runs on a common object model. It perhaps contains the interface specification for the networks of smart transducers.

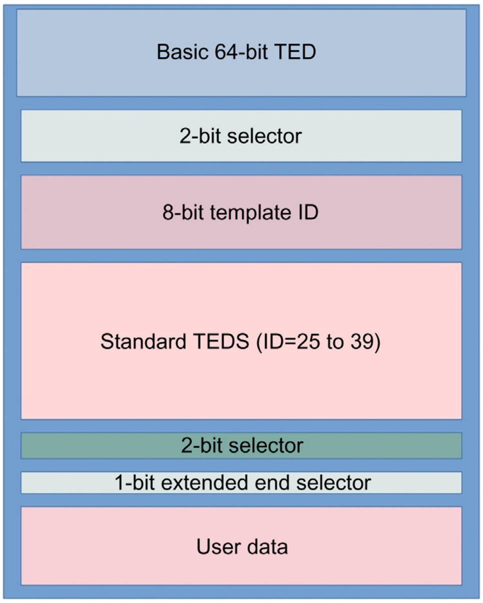


Figure 1.16 Standard transducer electronic datasheet content for transducers.

The architecture of the model has can be defined based on three fundamental models where various softwares and the hardware components worked with each other and perform a collaborative job. The architecture comprises of two sub-modules namely transducer interface module (TIM) and network capable application processor (NCAP). The main job of TIM is to perform signal conditioning and supports mixed-mode interface. Whereas the main task of NCAP is to link transducers by using web services and other protocol such as HTTP. The position of IEEE 1451.1within the IEEE1451 system architecture is shown in Fig. 1.17.

1.7 Summary

This chapter describes the various aspects of sensing and data gathering mechanisms. Numerous sensor devices and transducers have also been discussed along with smart sensors and transducers. The networking methodology using advanced low energy bluetooth and 5G technology is also addressed. The entire chapter, therefore, gives an

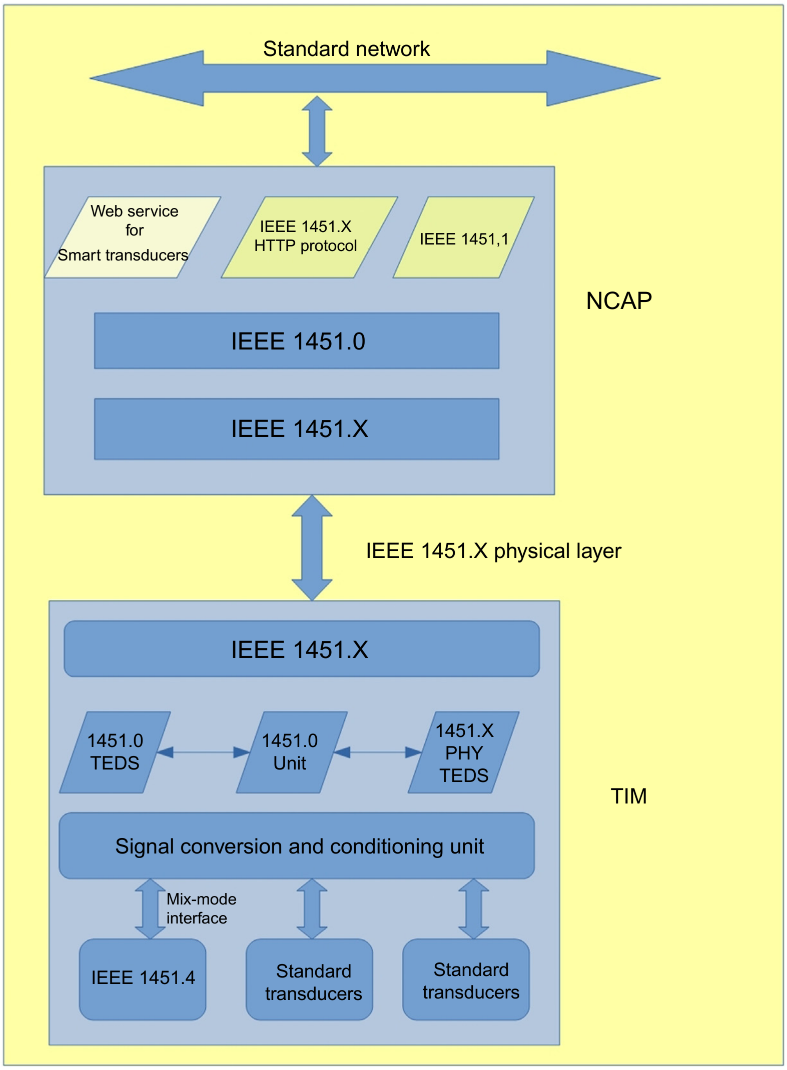


Figure 1.17 IEEE 1451 system architecture.

essence to the building and deployment of smart sensing devices for various application perspectives.

References

[1] S. Chen, F. Qin, B. Hu, X. Li, Z. Chen, User-centric ultra-dense networks for 5G: challenges, methodologies, and directions, IEEE Wirel. Commun. 23 (2) (2016) 78–85.

[2] A. Khan, M. Abolhasan, W. Ni, J. Lipman, A. Jamalipour, A hybrid-fuzzy logic guided genetic algorithm (H-FLGA) approach for resource optimization in 5G VANETs, IEEE Trans. Veh. Technol. (2019).

[3] M.A. Khan, I.M. Qureshi, F. Khanzada, A hybrid communication scheme for efficient and low-cost deployment of future flying ad-hoc network (FANET), Drones 3 (1) (2019) 16.

[4] A. Mukherjee, N. Dey, R. Kumar, B.K. Panigrahi, A.E. Hassanien, J.M.R.S. Tavares, Delay tolerant network assisted flying ad-hoc network scenario: modeling and analytical perspective, Wirel. Netw. 25 (5) (2019) 1–21.

[5] B. Han, Z. Jiang, L. Liang, P. Chen, F. Yang, Q. Bi, Joint precoding and scheduling algorithm for massive MIMO in FDD multi-cell network, Wirel. Netw. 25 (1) (2019) 75–85.

[6] S.M. Darroudi, R. Caldera-Sànchez, C. Gomez, Bluetooth mesh energy consumption: a model, Sensors 19 (5) (2019) 1238.

[7] D.A. Abraham, Introduction to underwater acoustic signal processing, Underwater Acoustic Signal Processing, Springer, Cham, 2019, pp. 3–32.

[8] N. Dey, A.S. Ashour, W.S. Mohamed, N.G. Nguyen, Acoustic sensors in biomedical applications, Acoustic Sensors for Biomedical Applications, Springer, Cham, 2019, pp. 43–47.

[9] C. Ru, Y. Gu, Z. Li, Y. Duan, Z. Zhuang, H. Na, et al., Effective enhancement on humidity sensing characteristics of sulfonated poly (ether ether ketone) via incorporating a novel bifunctional metal–organic–framework, J. Electroanal. Chem. 833 (2019) 418–426.

[10] X. Wang, Q. Peng, L. Liu, J. Yang, X. Du, Y. Li, Scanning distortion analysis of infrared thermal imaging systems, Optical Design and Testing VIII, vol. 10815, International Society for Optics and Photonics, 2018, p. 108150X.

[11] H.-E. Joe, H. Yun, S.-H. Jo, M.B.G. Jun, B.-K. Min, A review on optical fiber sensors for environmental monitoring, Int. J. Precis. Eng. Manuf.-Green Technol. 5 (1) (2018) 173–191.

[12] H. Schlicke, T. Jochum, S.C. Bittinger, T. Vossmeyer, J.S. Niehaus, H. Weller, Nanoparticle composites as functional materials for novel devices: chemical sensing and optoelectronic applications, *2018 IEEE 13th Nanotechnology Materials and Devices Conference (NMDC)*, IEEE, 2018, pp. 1–4.

[13] P. Wang, X.-B. Tang, P. Gong, X. Huang, L.-S. Wen, Z.-Y. Han, et al., Design of a portable dose rate detector based on a double Geiger–Mueller counter, Nucl. Instrum. Meth. Phys. Res. Sect. A. 879 (2018) 147–152.

[14] S. Fong, J. Li, W. Song, Y. Tian, R.K. Wong, N. Dey, Predicting unusual energy consumption events from smart home sensor network by data stream mining with misclassified recall, J. Amb. Intell. Hum. Comput. (2018) 1–25.

[15] V.N. Pugach, E.L. Voronin, Smart transducer with radiomodem, J. Phys.: Conf. Ser. 998 (1) (2018) 012027.

[16] E.Y. Song, M. Burns, A. Pandey, T. Roth, IEEE 1451 smart sensor digital twin federation for IoT/CPS research, *2019 IEEE Sensors Applications Symposium (SAS)*, IEEE, 2019, pp. 1–6.

CHAPTER TWO

Real-life application of sensors and systems

Contents

2.1 Overview of Internet of things

Internet of things (IoT) is a computational concept that comprises a network of physical objects [1]. This mostly comprises sophisticated embedded technology that plans to communicate and sense or may interact with the internal states or the outside environment.

Everything that we feel and look is generally incorporated within that network. The concept of the cloud takes a vital role in making things connected to each other. As technology emerges one day, it might communicate with even all other technologies by taking a very minimal amount of time. The concept of IoT makes the existing technology squeezed and maybe they combined with any other technology [2]. There are several benefits involved of IoT platform. IoT philosophy mostly imbibes the concept of machine to machine. It can mostly visible to the manufacturing industry, power industry, and many more. A device having smart capabilities can be mostly used. A major benefit is its always and anywhere available characteristics. Since all the device is connected all together, it is very easy to keep track the status or even can be controllable from anywhere in the world. Another very important benefit is its scalability. Since it is a kind of loosely coupled system so at a particular instance of time any node can be added or removed very easily.

The system itself is generally guaranteed to be highly robust because of its availability of nodes. The size of the network, in general, is so much dense. And some of the nodes actually physically perform redundant operations. Further, the node density may be different depending on the physical organization of the sensor and the activity of the device. The main phenomenon is that the nodes are heterogeneous in nature. Second, the nodes may be either static or highly dynamic in nature. Mostly the dynamic nodes will follow a certain route and mobility platforms.

2.2 Design perspective

IoT design is highly influenced by a protocol called Message Queuing Telemetry Transport (MQTT) [3]. MQTT protocol was invented in 1999. In 2010 the protocol was released under a free license. The protocol version at that time was 3.1. MQTT is an ISO standard that publishes subscribe-based messaging protocol. It fundamentally works on the top of the transmission control protocol/Internet protocol (TCP/IP) suite. It is a lightweight system design to connect with the remote location and where the bandwidth is limited. Due to its lightweight nature, it is highly adaptable in the IoT ecosystem. The main goal of design and implement MQTT protocol is to ensure message transfer through minimum bandwidth communication channel. It basically works on request-response paradigm and transfers the binary message with low overhead. MQTT supports publisher—subscriber pattern where the message has been decoupled into to part one is a publisher that sends or publishes the

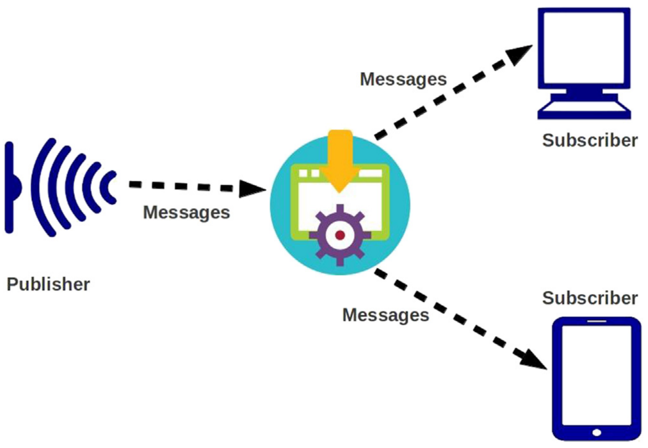


Figure 2.1 MQTT architecture.

message and subscriber that receives the message. It is an asynchronous type of protocol that is why not necessarily connects to publisher and subscriber at the same instant of time.

Fig. 2.1 shows the primary ecosystem of MQTT. As MQTT follows publisher – subscriber-based paradigm, the main component of the same is known as a broker. The job of the MQTT broker is to deliver the message from publisher to subscriber. The messages are delivered in an asynchronous mode through publish-subscribe architecture. It uses a topic filter to filter out the clients those who receive the message. This topic is something like a virtual channel that makes the architecture of the IoT ecosystem more scalable as publisher and scriber who do not need to know each other. There are several control packets involved in the exchanging of the message. Each control packet has a specified purpose, and it is carefully designed to optimize the overhead during the message transmission. The several message format types for MQTT message have been mentioned in Table 2.1.

MQTT protocol is ideal for constrained networks such as low bandwidth, high latency, limited data, and fragile connection. The packet control headers are intentionally made as smaller as possible. Each header contains three subparts: a fixed header, a variable header, and a payload. Fixed header size of each MQTT packet is generally 2 bytes. A payload up to 256 MB can be attached with MQTT packet. Having a significantly small payload overhead makes the protocol enough lightweight for IoT application.

Table 2.1 MQTT control packet formats.

MQTT control	Flow direction	Remarks
CONNECT	Client side to server side	A request made to connect from client to server
DISCONNECT	Client side to server side	Request for disconnect
CONNACK	Client side to server side and vice versa	To make an acknowledgment of connection from the client to server and vice versa
PUBLISH	Client side to server side and vice versa	To publish the message within networks
PUBACK	Client side to server side and vice versa	To make an acknowledgment of publishing from client to server and vice versa
PUBCOMP	Client side to server side and vice versa	Publish completed (signifies an assured delivery part 3)
PUBREC	Client side to server side and vice versa	Publish received (signifies an assured delivery part 1)
PUBREL	Client side to server side and vice versa	Publish released (signifies an assured delivery part 2)
PINGREQ	Client side to server side	Deploy a ping request
PINGRESP	Server side to client side	Response to ping request
SUBSCRIBE	Client side to server side	Subscription request from client
UNSUBSCRIBE	Client side to server side	Unsubscribe request from client
SUBACK	Server side to client side	To make an acknowledgment of subscribing from client to server and vice versa
UNSUBACK	Server side to client side	To make an acknowledgment of unsubscribing from client to server and vice versa

MQTT, Message Queuing Telemetry Transport.

Three quality of service (QoS) methodologies have been followed by the MQTT protocol. In QoS 0, the packet must be delivered with its best effort of the operating environment message loss may occur in such case. In QoS 1, the message is surely delivered, but duplicate can occur. In QoS 2, the message should be delivered strictly at once only.

When MQTT client connects to MQTT server, client defines a topic and a message needs to be published automatically. When a client disconnects unexpectedly, the keep-alive timer of the server detects that a client has not sent any message. Here server generates the will message on the will topic defined by the client.

2.3 Related platform

In a simple sense, an IoT device may connect to another IoT device and application to relay information using transfer protocols. The bridging of the gaps between different sensors and data network is to be done by the IoT platform. Such a platform connects the data network to the sensor arrangement and provides insights using some couple of backend applications to make sense of chunk of data generated by hundreds of sensors.

There are hundreds of startups and ventures nowadays creating their business by taking IoT as the primary technology. The companies such as Amazon and Microsoft are taking a lead role in this domain. In the following section, we are going to discuss such platforms in brief.

2.3.1 Amazon web service

The fundamental feature of Amazon web service (AWS) IoT platform is that it offers secure device gateway and offers registry for recognized device [4]. Some of the software development kit (SDK) is also available as an IoT development environment. It also provides device shadow feature, and an internal rule engine is also available for inbound message evaluation.

According to Amazon philosophy, they provide for easy-to-use IoT platform, which further is modifiable with some hardware vendor as they create some supportable starter kits. Further AWS management console provides a web-based interface for accessing and managing all the AWS IoT resources. AWS starter kit is available to connect with IoT core. AWS SDK takes out the complexity of coding as it provides several application program interfaces (APIs) for numerous AWS services.

The device SDK for the same helps for the easy connection and deployment of hardware. It offers some enhanced feature so that they can efficiently interact with device shadow and gateway, and kick-starting of the project becomes very easy.

Some of the development kits that are supported by AWS are as follows:

Advantech ARM-based single-board computer RSB-4760 is a Qualcomm ARM Cortex A53 APQ8016 having speed up to 1.2 GHz. It is having onboard 2 GB DDR3 memory with 8-GB external memory support. Inbuilt USB, GPIO, I2C, and SPI connectivity are also present.

It also consists of wide voltage range and DC power input. It supports Android, Linux, and Debian file system.

2.3.2 Google Cloud Platform

This platform gives a highly scalable reliable web infrastructure to build test and deploy the web applications. In Google cloud API, several services [5] such as SAP, machine learning, IoT, and so on have been incorporated; the cloud IoT core is the highly secure IoT management service through which we can deploy, connect, and manage several globally connected devices. The cloud IoT core gives various other services with the combination of other APIs. This produces a complete solution for processing, visualizing, and analyzing IoT data in a real-time philosophy. The fundamental architecture of Google cloud IoT has been depicted in Fig. 2.2.

The module cloud pub/sub will aggregate the sparse device data into a single system that integrate seamlessly with Google cloud data analytics services. The advanced IoT services can be used analytics, visualization, and machine learning. A developer can give their inputs and anticipate problems and building better and richer and optimized platform.

A most dominating feature of this service is security. Millions of device can securely be connected through a secure protocol that uses automatic load balancing. A global technical infrastructure has been designed to ensure security through the entire life cycle of information processing.

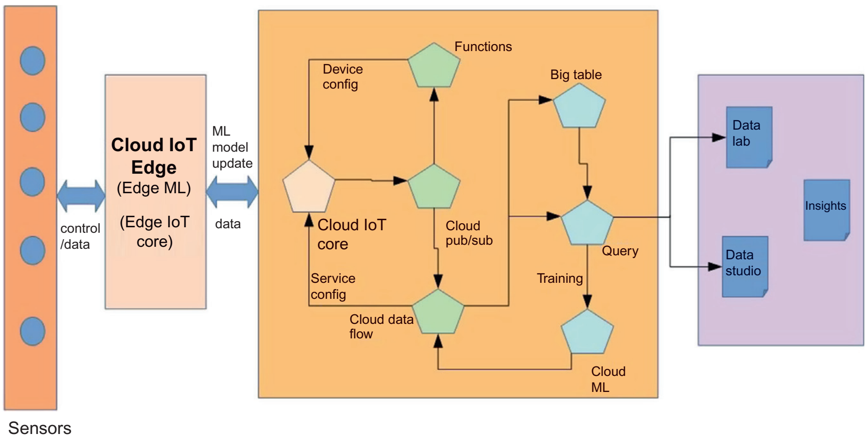


Figure 2.2 Google Cloud IoT platform architecture.

The infrastructure provides secure deployment of the applications and the services, secure data storage, and privacy safeguard. Such infrastructure service is highly responsible for building internet service, consumer service, and the enterprise service as well. The security layer support basically lay on progressive layer support starting form physical security of the data centers to the operational security. Some of the well-known security features of Google service are encryption at rest, login abuse protection, encryption of interservice communication, service identity and integrity, secure boot stack, and machine identity.

Google cloud platform (GCP) security is a major issue as a public cloud platform. In Google cloud platform, the compute engine enables the customer to run their own virtual environment in a form of the virtual machine. The engine offers several logical components. The management control plane is the finest component that deploys external APIs and performs the task like virtual machine management and migration. Various services run under this infrastructure, and it gets the secure boot chain like functional integrity automatically. The customer engagement center (CEC) plane exposes the different APIs via GFE. It also exploits the security feature of denial of service protection, which is centrally managed by secure sockets layer / transport layer security (SSL/TLS) support.

2.4 Real-life examples and implementation

2.4.1 A device to a cloud system

A device to cloud system can be conceptualized when a mobile or other sensor-based system or any other handheld device interacts with a cloud service [6]. This results in the transfer of the information from one device to another or may be from cloud service to device or vice versa. The main advantage of the system is to reallocate and update the information dynamically. For an example, Google docs that provide several services such as a spreadsheet that can be accessed dynamically by *n* number of user at a time. Not only the human but also any mobile or sensor-based intelligent device. In the following sections, we are now going to discuss more a real life.

Application of the device to the cloud can be implemented by setting up a device consist of an Arduino board and Zigbee module. Further, this

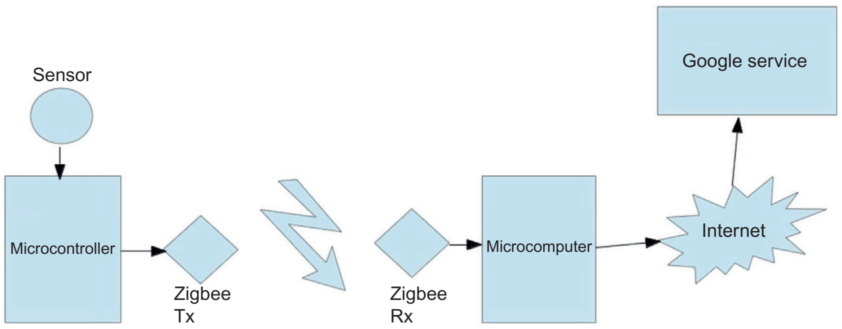


Figure 2.3 The architecture of the device to cloud.

device module could communicate with a Google spreadsheet application. As an intermediate system, we can use a PC or maybe a small raspberry pi microcomputer. The architecture of the device to cloud system is shown in Fig. 2.3.

The procedure involves certain layered approach. In layer 1, we do deploy certain sensor nodes that might take the ambient data and perform accumulation of data by applying data accumulation algorithm. It should also perform communication through a wireless network to the next layer. The layer 2 comprises a computation module and a network receiver that receives the raw data from layer 1. These data now should be redirected to the cloud service by implementing a certain protocol. The layer 3 is the service itself that can take the data from the device. Such service may have some capacity to compute and analysis of the data. When data are pushed to the server, a mandatory authentication has to be done. The layer 1 can be designed in the following way.

A temperature sensor (specifically an lm35) can be interfaced with Arduino UNO board by applying a voltage of 5 V. Lm35 having three pins: a ground, Vcc, and signal in. The signal supplied by lm35 is purely analog. Therefore, one should connect it in the analog input pin of any five pins of Arduino.

In Arduino code, we often use analog reference function with a parameter "INTERNAL." This ensures a reference voltage of 1.1 V. That divided the data samples into 1024 steps. Based on these steps, the value of the analog signal gets quantized.

Zigbee communication setup is another task in layer 1, which will do a communication bridge between the two layers. The Zigbee protocol is an IEEE802.15.4 standard that is widely used in recent days. It generally uses to make Zigbee-based personal area network (PAN). Depending on the range, it may vary from 10 to 100 m with a frequency range of 2.4-GHz ISM band. Mostly the network that can be made using Zigbee is dedicatedly used for short-range communication.

Arduino support for Zigbee communication is mostly used to develop wireless sensor network (WSN) host nodes and the devices that are able to communicate over the network or do interaction with the cloud. Arduino Xbee shield is one of the widely used modules that is highly used for interfacing Xbee adapter with Arduino.

The task of configuring an Xbee module is a bit tricky. An Xbee adapter can be configured by using a hyper terminal program like PuTTY or X-CTU (specifically for Windows).

The PuTTY hyper terminal can be accessed from its programming mode by pressing "+ + + " symbol. Setting up the host module address just uses ATMY command so that it can store a 16-bit address. In the same way, we can also put the destination address by applying ATDL, and ATID command should be used to create the PAN. After putting all commands, write WR to save the command in the module.

The receiver of a node of Zigbee PAN, therefore, must be attached to a PC or a microcomputer. The receiver Xbee module, in that case, may use an Xbee adapter or Arduino Xbee shield. The adapter then connected to PC via USB. We can run a python or processing application to get the data coming from the Xbee receiver.

A simple python code consists of serial library that can easily fetch and print the data from output.

The following python code illustrates the method of printing the data coming at the serial port.

```
import serial

obj = serial.Serial('/dev/ttyUSB0',9600)

dat = obj.readline()

print dat
```

Here, "/dev/ttyUSB0" is the USB port name, and 9600 is the baud rate for communication.

After completion of the Zigbee setup, it is the time to connect with the Google spreadsheet service. There are numerous steps involved in this.

First of all, we have to choose the appropriate platform that has been used to push the data into spreadsheet service. In this case, we are choosing python for that. To interact with Google spreadsheet, we need to use a python API for Google spreadsheet. The API called gdata API and is freely available on the google code repository.

Successful installation of the API, the second task is to configure Google Account so that it can interact with some third-party device. Generally, Google provides an API key to access the device connected with it. However, only the API key is not sufficient to connect the native application. To get fully connected, the two-step authentication procedure should be made, and after doing that, we must seek an application password. As the app password applied to the python code now, the system is ready to communicate with the Google spreadsheet application and can upload data easily.

2.4.2 Home automation system

In this 21st-century world, home automation system is a very common system [7]. Most of the case, an automation system is highly applicable in the industry called industrial automation, and in parallel, nowaday's home automation takes its part to make the social life more smarter. It generally involves an automated household environment. It can be categorized into various types. Some of them can be controlled remotely using a smartphone. Secondly, It can control an actuator, or a motor that ensures security such as door lock.

Most of the home automation solution nowadays is basically an android phone based, where a single smartphone controls all home appliances. A home automation prototype can be easily implemented using Arduino, relay module, Wi-Fi (IEE802.11 b/g/n)/4 G, 5 G, or Bluetooth module with a PC or microcomputer. The following diagram (Fig. 2.4) illustrates the fundamental organization of a simple home automation system.

The number of channel in relay depends on the number of appliances that you want to connect with the automation system. An Arduino board can take power up to 12 V. Or maybe it can be powered through 9-V DC battery. The Bluetooth and the relay module can be powered using Arduino 5 V/3.3 V output pin.

The Bluetooth module can perform data transfer operation in master slave mode with a baud rate of 9600—115,200 and mostly used universal

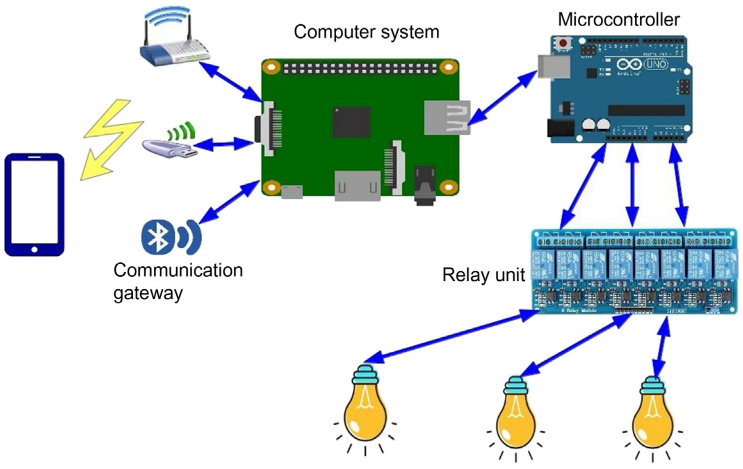


Figure 2.4 Home automation system.

asynchronous receiver/transmitter (UART) asynchronous transfer mode. This mode is used to communicate with another device like laptop, mobile, and so on. In this case, the Bluetooth module has been paired with an Android phone, and a native android app has been used to control the appliances.

The fundamental function of the relay is to control appliances such as fan, light, fridge, or air conditioner. It provides a complete isolation form Arduino and another module to the high voltage line. Its primary coil gets excited by applying 5 V on the coil that is mainly applied from the Arduino board.

2.4.3 Body sensor networks

Due to the emerging IoT technology, the feature of the network architecture has been extended to an integrated multiobject infrastructure. The data that are produced by several devices routed directly through a wireless channel. IoT philosophy fundamentally supports heterogeneous device deployment; "things" and the services are in a common umbrella so the data may come from anywhere and can be redirected to any direction. Under these phenomena, an obvious question may arise that whether the applicability of the Internet of Thing is justified for any mission-critical and tactical application. As the data come from heterogeneous things, the security of the entire network and the service may be compromised to some extent. If attackers smash a single window, they get access to the network that leads to widespread chaos. Data elements such as border surveillance data, Statistics of Army operation in some terrorist infested region, and health statistics of Army personals during the mission are the classified information. Transmission of such information is really a critical issue. Several security measures have been implemented and addressed by considering such tactical scenario. Investigation of the security level of classified data, more formally the classified health information of group leader and other soldiers and finally proposed an optimum method of security of the data from the battlefield to the base station is the main subject of interest here. In this work, our primary focus is to implement a security sublayer in the perception layer of the IoT framework where physical sensors have been deployed. Here the onboard encryption on data has been made in a hostile environment. The encryption has been done over the real-time data stream hare to ensure strong security over the mission-critical information. body sensor network (BSN) system composes of several sublayers [8]. The bottom layer is known as a perception layer where sensor data have been grabbed and sensor fusion has been taking place. Data encryption-decryption can also be performed in this layer. In the second layer (network layer), basically packet routing has been performed. At the final level in the application layer, the data have been received, and all analysis and decryption have been done in that layer. The perception of sensor nodes is the elementary component of the perception layer. Open-source microcontroller can be used to implement it. The sensor node can be deployed with a concept of the device to cloud methodology. Fig. 2.5 depicted the single unit of the node. The node has four major components:

Temperature sensor: Body temperature has been taken using the LM35 temperature sensor. It is a precision integrated circuit-based temperature sensor. The centigrade temperature value given by the sensor is proportional to the output voltage. No additional calibration required for the temperature measurement. However, for more precise operation and to make the temperature sensor work as a body temperature measurement device, we are using a sophisticated calibration methodology. Although TO-92 plastic package of LM35 is having slower response time but works more precisely in the temperature range $-40^{\circ}\mathrm{C}$ to $+150^{\circ}\mathrm{C}$, we can use a metal coating over the TO-92 package to get a more faster response. We have made a taping provision to mount them in various parts of the body.

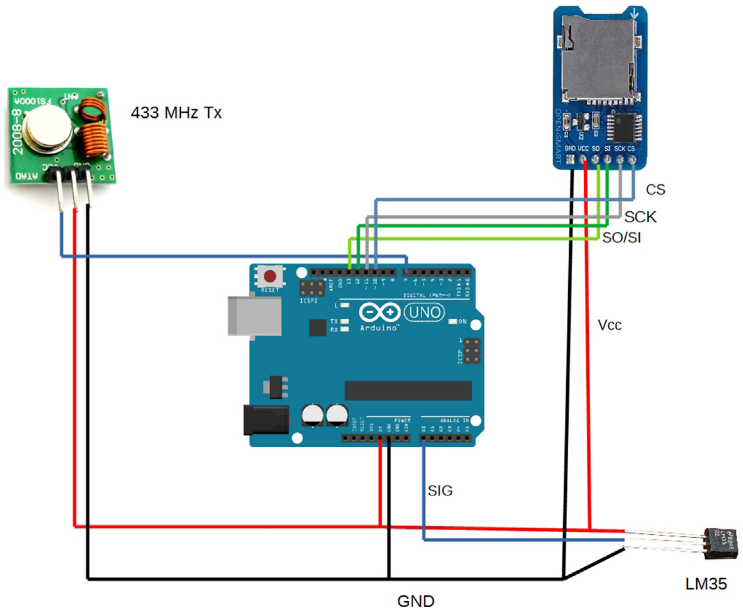


Figure 2.5 Node of a body sensor network.

2.4.3.1 Arduino microcontroller

Arduino microcontroller is a well-known microcontroller board driven by ATMega328P. These classes of the microcontroller are 8-bit microcontrollers consists of 14 digital PWM I/O and 5 analog input pins [9]. It runs with a clock speed of 16 MHz. A more sophisticated version of Arduino can also be used known as Arduino Mega2560. Sensors can be interfaced within digital or analog pins. There are numerous hardware components that can be added to extend the functionality of Arduino. These plug-in components are called shield.

2.4.3.2 Datalogger

This is an additional part of the device that can store the temperature data in storage. The encryption has been done on this data as it stores on the device. After a certain interval of time, the data have been fetched by the encryption module, and it generates an encrypted packet of data that are considered as a stream of characters. These data have to send to the base station using a single-channel 433 MHz radio transmitter unit.

Radio transceiver system is a single-channel 433 MHz radio frequency (RF) transmitter-receiver unit. The maximum range of the unit is 500 m

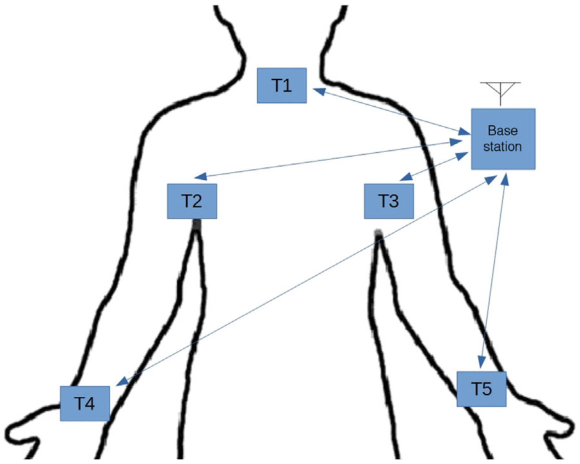


Figure 2.6 Point of sensor deployment for an elementary BSN.

by introducing additional power to the receiving side. The transmitter unit sends the form of packet stream that is nothing but similar to character sets. In the Arduino platform, the virtual wire library has been used to transmit and receive encrypted data stream.

At room temperature of 0°C, the surface of the body temperature varies within 28°C–31°C, whereas at room temperature 28°C, it varies within 31°C–34°C. We have placed the temperature sensors in the location such as wrist, axillaries, and throat. Fig. 2.6 depicted the body map and the sensor deployment point. The device itself is a wearable device, and the data that have been sent by the sensor are analogs. Body temperature that has been taken from the various points of the human body is fused in the microcontroller through analog input pins. Central limit theorem methodology has been used to perform fusion on the temperature data. The theorem says if x_1 and x_2 … x_n are the n sensor measurement with noise variance $\sigma^2{}_1$ and $\sigma^2{}_2$, respectively, and the combination of the measurement can be obtained by $x_n = \sigma_n^2\left(\sigma_1^{-2}x1 + \sigma_2^{-2}x2 + \ldots + \sigma_2^{-2}x_n\right)$ where $\sigma_n^2 = \left(\sigma_1^{-2} + \sigma_2^{-2} + \ldots + \sigma_n^{-2}\right)^{-1}$ is the variance of the combined estimate. The final result of fused value is simply a linear combination of n measurements weighted by their respective variances.

2.4.4 Implantable wireless body area network

Implanted body sensors are of popular use in recent days [10]. The sensors in this case are communicated in between different other modules or controllers inside and outside of the human body. The major difference between a wireless body area network and implantable body area network is the communication method. In the case of implantable sensor networks, the human tissue is considered to be the communication channel. As transmission via an electromagnetic signal through the body is a highly challenging issue, several types of research are carried out to understand the response of human tissue with high-frequency signals. In addition to that, miniaturization and sustainability on the power supply are also a primary requirement for long-term communication system. Several communication methodologies have been studied in this context including the coupling between implanted and surface mounted sensing coils. Further antenna-based communication is another technique that is also used. The research on the emerging field of communication using optical, molecular, and ultrasound-based communication has also been investigated. Lots of challenges and scopes are still existed in this avenue to improve the quality and the sustainability of such implantable technology. One of the emerging medical domains in which the technology has been highly involved was the pacemakers. Further, the cutting edge development of the smart sensors also takes part in the service of diagnosis that measures vital health sign. The devices such as glucose sensor, deep brain activity sensor, pH sensors, and gastrointestinal imagery pills are most popularly used devices for such applications. In the second scenario, a precalibrated stimulus or controlled closed-loop feedback units may be used as stimulation units. These systems are called IMD, and these are primarily used as nerve and muscle stimulation. IMD-assisted devices can even improve the anatomical ability of any healthy people by introducing the stimulants. Other examples such as prosthetic limbs, vision implants, and brain-computer interface are most popular in these contexts.

2.4.5 Application of WBAN in medical cyber-physical systems

Medical cyber-physical systems are an integration of medical instruments through wired or wireless networks [11]. The system ensures high-quality healthcare in hospitals as it deals with high-grade security, interoperability, and high system assurance. Cyber physical systems (CPS) technique has a significant responsibility in social service-based information technology framework, which primarily emphasizes the medical and healthcare application. The performance improvement of the technology needs a precise improvement of the computation and networking framework. Intelligent wearable devices and smart meters are considered to be the emerging platform and act as part of the IoT and social networks. The IoT concept is fundamentally limited with the condition and control of miniature devices in another context the cyber-physical systems not only deals with the device level interaction to monitoring and control within cyberspace. One can view the IoT as the network of the cyber-physical systems. Hence we can consider CPS as the first tire and the IoT as the second tire. CPS with a network connection can be considered as IoT. An improvement of the medical CPS can be done by introducing onboard computing ability and wireless-based distributed computing feature. The pervasive health monitoring systems can, therefore, be included, and its ability can be enhanced with the above-mentioned technology. The medical cyber physical systems (MCPS) itself required a well-organized physical infrastructure. Structure wise, as more components are added in the system, the increment of complexity and reliability is obvious. The CPS has some inherent characteristics such as reading, status, and trust uncertainty; second, it should achieve real-time performance demand. The system must also have input and feedback from the environment without considering any network channel for communication. The system is also separated with wide geographical distribution, and the components involved may prone to a security problem.

In term of communication channel, the MCPS can be conceptualized as a communication between the physical and the virtual space. There should be a bidirectional data flow between the virtual, physical, and intelligent sensor components. A layered approach can be thought of in this context (Fig. 2.7) in which the first part the sensors and actuators as well as the mobile sensing platforms, network stations are also the part of the component. The devices in that layer are connected with the application platform that sends the accumulated data into the intelligent decision system. The data are further processed by this unit, and the service alert has been sent to the service providers and the action has been taken accordingly.

Wireless body area network often termed as a body sensor network is a key component in MCPS. Unlike a general sensor network, a wireless body area network (WBAN) performs a crucial task of health monitoring with some intelligent decision making. Implantable and miniaturized wearable sensors are the fundamental part of the design of a body area network. The sensor in a BSN is designed to measure the crucial component of the body such as temperature, pH level, glucose level, heart rate,

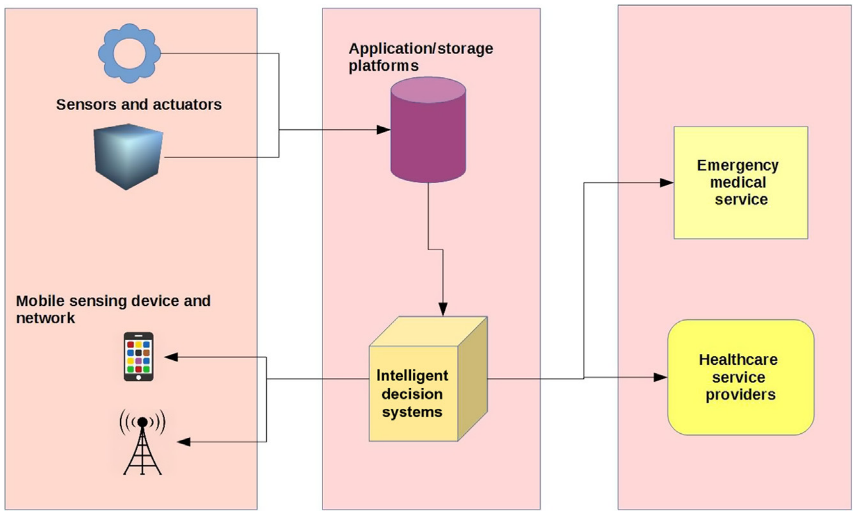

Sensors and actuators
Application/storage platforms
Mobile sensing device and network
Intelligent decision systems
Emergency medical service
Healthcare service providers

Figure 2.7 Layer abstraction of the MCPS system.

respiration rate, and many more. A central microcontroller-based data agglomeration node should be present to grab the data via wired or wireless links. Further, these nodes should connect with a mobile ad-hoc network for further communication and the remote processing. Since the CPS systems mainly emphasize on heterogeneous information flow, intelligent decision making and cross-platform sensor collaboration so the wide variety of embedded sensor network applications can be legitimately implemented.

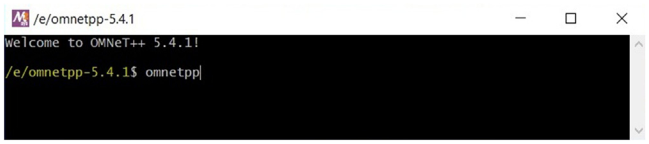
2.5 WSN simulation environments

There are several simulation platforms involves the realization of the different aspects of the WSN. One of the most popular simulators is Omnet++. Further in WSN domain, a major research area in contemporary days is the management of the sparseness of the nodes. Hence the intermittent connectivity of the dynamic WSN can also be addressed as a research problem. In order to visualize the real-world behavior of the intermittently connected sensor nodes, the opportunistic network simulator can be used. There are several other simulation methodologies that are also applicable for WSN simulation like localization of sensor nodes and node tracking using unmanned aircraft systems. In the next subsections, we are going to emphasize both of these two network simulators and their uses to achieve a high-performance output.

2.5.1 OMNeT++ simulator

OMNeT++ is the discrete event-based simulation environment [12]. The fundamental job of this simulator is to create communication network, WSNs, and opportunistic networks. The advanced module of OMNeT++ also makes visualize long term evolution (LTE) networks and WSNs [13]. The base language of the simulator is C++. The simulator supports large-scale simulation and has the reusable components. Use of the OMNeT++ is simple. We can download the windows version of omnet++ from its official website (https://omnetpp.org/download/). After downloading, extract the zip file into c drive and then open the folder containing script file "mingwenv." Just double click it to start the file in the command-line interface and press any key to continue. Now the necessary file for simulator environment gets extracted. After extraction, press any key to continue. Now another command-line window will appear. Here we must type the command "./configure" to configure the OMNeT++ simulation environment. The installation and the configuration process will take 5–6 minutes. After this step in the command line window, we have to type "make." The make command creates all dependency and installs the simulator successfully. It takes around 15 minutes to execute. After completion of this process, we can type "omnetpp" in the "mingwenv.cmd" command console to start the simulator as shown in Fig. 2.8.

Although OMNeT++ is a framework for network simulation, it can be considered as a generic distributed system platform. This provides a scalable and well-organized facility for designing the simulation framework like INET. The fundamental backbone of this simulator is the C++ kernel framework. The simulator itself is highly versatile, and there are provisions for real-time simulation, parallel and distributed simulation,



```
Welcome to OMNeT++ 5.4.1!

/e/omnetpp-5.4.1$ omnetpp
```

Figure 2.8 Mingwenv command console for Omnet++ startup.

network emulation. Various sample simulations are also available. The fundamental simulation file that one can find is a network topology description (.NED) extension file. This file contains the framework of the network that is going to be simulated. One can access it in textual and the graphical form by just double-clicking it. Another very important file is "omnetpp.ini." This file contains the settings of the simulation kernel. While initialization, this file can automatically be read by the simulator kernel. The parameter like simulation-time-limit can be set through ".ini" file so that the simulation never goes for an indefinite time of iteration.

In order to execute the simulator, first of all, a new window named OMNeT++/Qtenv should appear. This is the default IDE for OMNeT++ and one can easily run the simulation in a pretty efficient manner. To run the simulation, one can simply press the run button and doing so an animated simulation will start. To increase the speed, a slider is available, and to temporarily hold the simulation, we can press pause.

The result of the simulation is further been analyzed using an analysis tool. This can be done by getting the result file that is saved in the project directory in the form of ".vec" or ".sca" format. By double-clicking on this result file, one can generate the analysis file ".anf" that are open in Analysis editor tool by default.

- **Components of OMNeT++**

  The various components of OMNeT++ simulation are present, and they work in a collaborative fashion. The major component of OMNeT++ communicates with each other using message passing mechanisms. The basic modules that are the fundamental backbone are called simple module. By combining "*n*" number of simple module, one can create compound modules. In a simulation environment, the simple module can be represented as communication protocol like TCP, network devices or routers like IEEE802.11, or Network Interface Card. The other components are user agents, routing table data structures, movement models, and automatic IP addresses. Simple and the compound components are basically the instances of the module. The network is also simulated as an instance of several module types.

  Another major component is the connection. As the major communication methodology in OMNeT++ considered as message passing, therefore most of the attribute has to be added into the message component such as Timestamp, header, and arbitrary data. The modules typically send the data using gates. There are basically three types of gates involved with the message passing namely input, output, and "inout." Input or output gate or two "inout" gates may be connected to complete the connection module. Connections are considered as the part of components, and it can be structured as a tree-like hierarchy. And due to that structure, the messages routed through connection chain. The major phenomenon such as propagation delay, data rate, and bit error can be assigned to connection as well. Each module has the parameters through which we can pass the configurable parameters. Supported parameter types are string, integer, Boolean, and XML files.

- **A basic example of message transmission for two nodes**

  A standard example for two-node message transmission can be considered as the elementary implementation of a network often called Tic-Toc. In this case, the node 1 will create a data packet and transmit it to node 2. We have to define a network descriptor file (NED) to implement the model. At the initial stage, the NED must contain two simple gates input and output. Now we have to create a module and for which some submodule can be created. In submodule, we must call some simple modules. The connections must also be defined in the module. The following NED code illustrates the basic tow node message transmission system.

```
simple Txc1
{
    gates:
        input in;
        output out;
}

network Tictoc1
{
    submodules:
        tic: Txc1;
        toc: Txc1;
    connections:
        tic.out --> {  delay = 100ms; } --> toc.in;
        tic.in <-- {  delay = 100ms; } <-- toc.out;
}
```

In the design window, the network configuration design is shown in Fig. 2.9.

The first module here is Txc1 is a simple module. This is an active component of the network and also atomic. The behavior of the component can be defined using C++ language. There are gates that

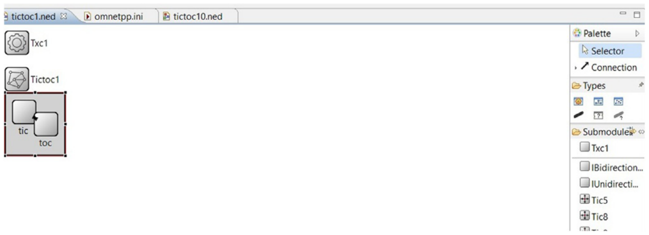


Figure 2.9 Implementation of the two-node message transfer network.

have to be defined in the module name in and out, which are input and output gates, respectively. The second bloc is the network. There are two submodules that are present in this module, namely "tic" and "toc." Both submodules should be module-type Txc1. The output gate and the input gate of the submodule have been cross-connected with each other.

The functionality of the Txc1 module can be implemented by a C++ source file. In the IDE, we can simply do that by adding a new source file in the project. The functionality of the Txc1 module can be described by the following code snippet.

```
#include <string.h>
#include <omnetpp.h>

using namespace omnetpp;
class Txc1 : public cSimpleModule
{
  protected:
    virtual void initialize() override;
    virtual void handleMessage(cMessage *msg) override;
};

Define_Module(Txc1);
void Txc1::initialize()
{

    if (strcmp("tic", getName()) == 0) {
        cMessage *msg = new cMessage("tictocMsg");
        send(msg, "out");
    }
}

void Txc1::handleMessage(cMessage *msg)
{
    send ( msg, "out");
}
```

Here Txc1 is a derived class from the cSimpleModule class. In the network, both modules are basically Txc1 object. The class must be registered in the OMNeT++ environment with Define_Module (Txc1) macro. The initialize () and handleMessage () function are invoked by the simulation engine. Those functions should be redefined before use in this case. After initialization, we should send the message through "out" gate. handleMessage () function, on the other hand, performs the delivery of the message from one node to another node. The message transfer delay can be defined within the NED file for a particular channel. In this case, a standard 100-ms channel propagation delay has been considered. During the cold start of the process, tic module here sends the first message. Therefore the out channel of "tic" object sends the message to the in channel of "toc." The same thing has to be repeated by "toc" module, and as a result, the message gets bounced back to the node repeatedly.

The next important job is to add "omnetpp.ini" file in the project. In order to run the project, an "omnetpp.ini" is mandatory in which we can define simulation time and CPU time. The runtime visualization of the two-node message-passing network is shown in Fig. 2.10.

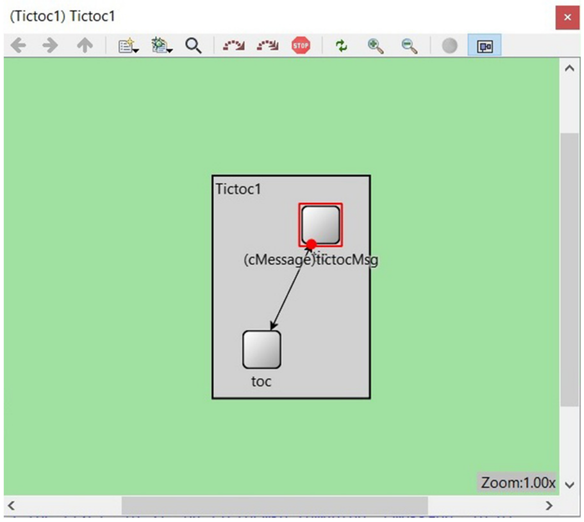


Figure 2.10 Simulation execution for two-node message transfer network.

- **Visualization of message transfer event in the space-time sequence chart**

  During the execution of the simulated network the simulation event can be logged using the event logger system. As the event logging button is clicked in Qtenv runtime simulation window, a file corresponding to the event login ".elog" format gets recorded. The log file is mainly analyzed in the sequential chart tool as a space-time diagram as shown in Fig. 2.11.

  In this case, sequence chart illustrates message transfer event between two nodes within a particular interval of time. Although the chart displayed seems to be pretty simple but for the complex network structure, this is pretty helpful to understand the message transfer between *n* numbers of nodes.

  A more complex network structure has been discussed in the next part, which consists of six nodes. The design will be made in such a way that a single node will create a message and the message transfer to the destination via some intermediate nodes. In order to get that, some changes in the NED need to be incorporated. In this case, the Txc file should have multiple input and output gates. Therefore in the gate declaration, we have to create input and output gates as a vector. In this case, as a submodule, we have to create the vector and hence it will determine the number of nodes. Fig. 2.12 represents the network topology with six nodes. The following NED code shows the initialization of elements.

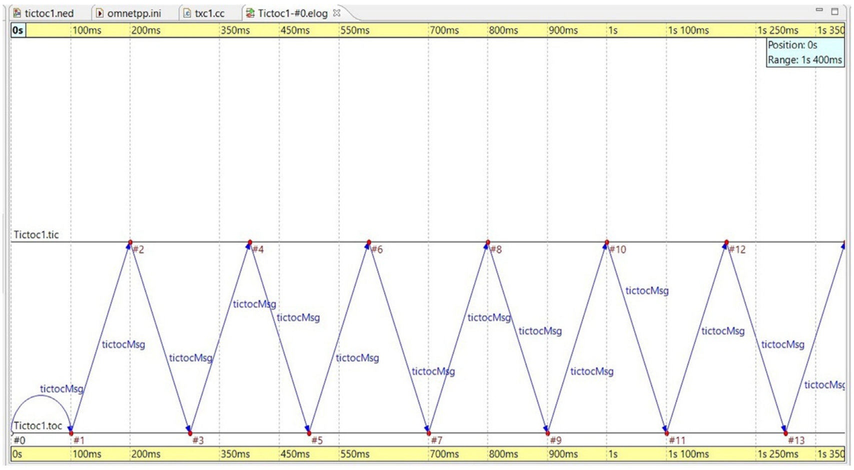
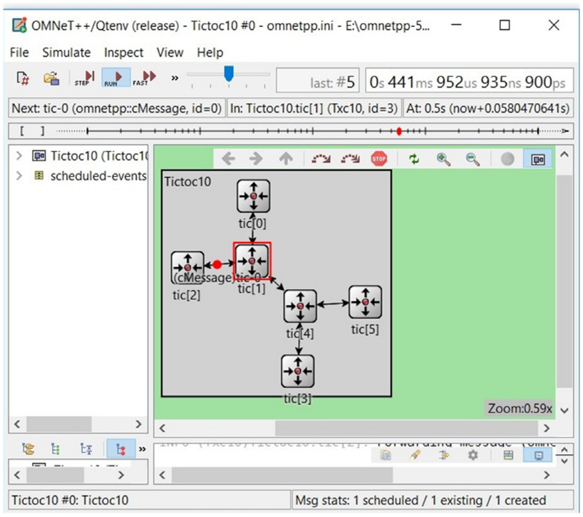


Figure 2.11 Message transfer event in space-time sequence chart.


OMNeT++/Qtenv (release) - Tictoc10 #0 - omnetpp.ini - E:\omnetpp-5...
File Simulate Inspect View Help
STEP RUN FAST
last: #5
0s 441ms 952us 935ns 900ps
Tictoc10 (Tictoc10
scheduled-events
Tictoc10
tic[0]
(cMessage)tic-0
tic[1]
tic[2]
tic[4]
tic[5]
tic[3]
Zoom:0.59x

Next: tic-0 (omnetpp::cMessage, id=0) | In: Tictoc10.tic[1] (Txc10, id=3) | At: 0.5s (now+0.0580470641s)

Tictoc10 #0: Tictoc10 | Msg stats: 1 scheduled / 1 existing / 1 created

Figure 2.12 Network topology scenario for six nodes.

```
simple Txc10
        {
          parameters:
             @display("i=block/routing");
          gates:
             input in[];  //
             output out[];
        }
```

Here @display(i = blocking/routing); shows the default routing icon for module. Further two gate vectors have been defined as input in[] and output out[].

The network can be defined by applying the connection between each node elements in the vector. To do so, a submodule in the network has to be created.

```
network Tictoc10
   {  submodules:
        tic[6]: Txc10;
      connections:
        tic[0].out++ --> {  delay = 100ms; } --> tic[1].in++;
        tic[0].in++ <-- {  delay = 100ms; } <-- tic[1].out++;
            ……
            ……. }
```

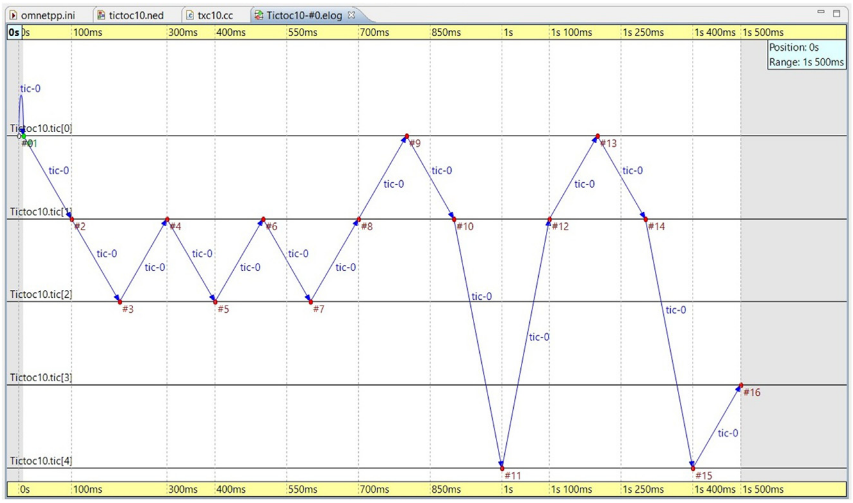


Figure 2.13 Message transfer between node (0) and the node (4).

In this way, the six submodules for in and out have to be created, and the connection has to be made with the propagation delay of 100 ms. In this case, tic(0) generates the message and sends it. The process is carried out using initialize() and getIndex() function. The getIndex() returns the current index of the submodule.

Fig. 2.13 shows space-time sequence of message transfer from the node (0) to the node (4). It is observed that the message has been created by node (0) and send to the node (1), which further sends the message to the node (2). After that, node (2) reverts it to the node (1), and this way after 850-ms time interval, node (1) directly sends the message to the node (4). At the time 1 sec and 400 ms, the event queue becomes empty and hence the simulation is done in this point. A message "No More events, Simulation completed—at 1.5 seconds, event#16" will appear as the completion of the event.

2.5.2 Opportunistic network simulation environment

The node of a sensor network sometimes considered as fixed, and depending on the ease of use, the node may be dynamic. In case of a dynamic sensor network, the topology is not fixed, and therefore the wireless mobile network protocols are applicable for them in order to grab the sensor data. The system becomes more complicated when we consider the degree of the sparseness of the network. In various situations such as disaster management and border area surveillance, we may deploy the robots (flying or ground) that sense the vast geographical territory. Not only that, the concept of deep-space opportunistic sensing and delay-tolerant network-assisted flying ad-hoc networks [14] is one of the primary research challenges nowadays. In such cases, the standard wireless network protocols also failed to achieve optimum performance. In such a scenario, perhaps the best way to realize the sensor network as an opportunistic network [15] is based on a delay-tolerant approach. In order to realize the intermittently connected wireless sensor node, we have to mainly emphasize on the mobility model and routing protocols. There are several routing methodologies that can be applied for such a scenario. Majority of the techniques are greedy-based flooding, where the node floods the data as it encounters with another mobile node. Three most common flooding-based routing algorithms have been used in this scenario, namely epidemic, ProPHET, and spray and wait. Along with routing, the mobility model is also a major concern in order to achieve the optimal message delivery performance. The routing algorithms are discussed as follows.

- *Epidemic routing*: A classical routing strategy, perhaps the simplest routing technique. The scheme works as follows
  1. initialize the message and should be queued in a buffer memory.
  2. assign a Message-ID to the message.
  3. if the node contacted with another do:
     a. Exchange the ID as a summary vector.
     b. Check the summary vector for those messages ID that is undelivered.
  4. Transfer undelivered message.

  The uniqueness of the routing is that it supports random and nonrandom movement methodology and also supports physical locality.
- *ProPHET routing*: ProPHET is basically a probabilistic flooding technique that exploits of transitivity and history phenomena. A unique metric called delivery predictability is mainly responsible in order to get the routing successful. The value of delivery predictability is, however, computed as $0 \leq P(X, Y) \leq 1$ for each X to the known node Y. delivery predictability is computed in such a way that node having higher value will be chosen as a better forwarder of the message

toward the destination. The node, however, chooses better forwarder in the following ways:

- nodes exchange the summary vector and update their entry based on the following formula:

$$P(X,Y) = P(X,Y)_{\text{old}} + \left(1 - P(X,Y)_{\text{old}}\right) X\, P_{\text{enc}} \tag{2.1}$$

Where P_{enc} is the configurable encounter parameter.

- now if X meets the node j more frequently, then there must be a good chance to route the message from X to j and finally j to Y, therefore, this can be done based on

$$P(X,i) = P(X,i)_{\text{old}} + \left(1 - P(X,i)_{\text{old}}\right) X\, P(X,i)\ X\ P(X,Y) \tag{2.2}$$

- in order to confirm symmetry DP value, has to be edged with a constant aging factor γ that produces

$$P(X,i) = P(X,i)_{\text{old}} X\, \gamma^{T} \tag{2.3}$$

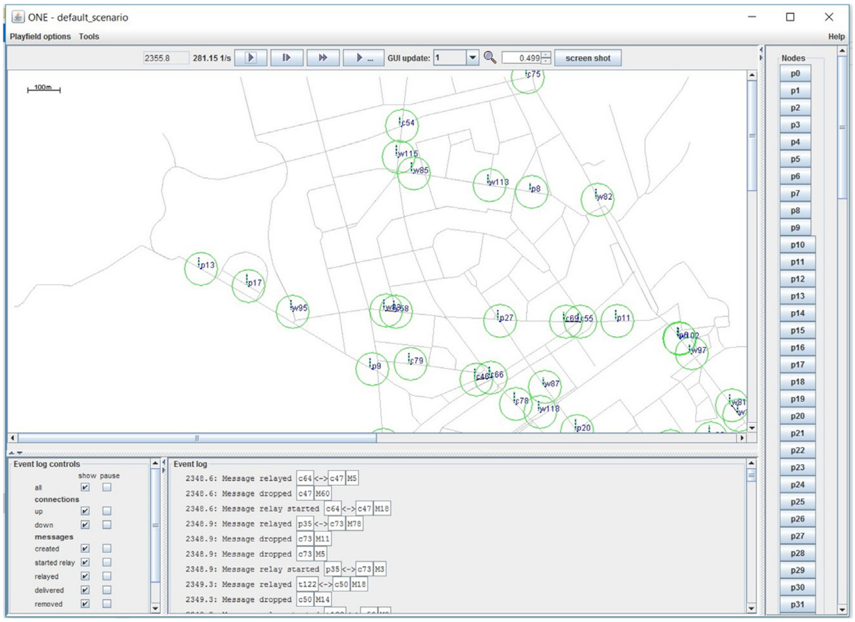
- *Spray and wait routing*: It is another routing scheme that performs in two different phases:
  1. In spray phase, a number of L messages gets forwarded and a node receives the copy of L distinct relays.
  2. If the routing unable to find the destination in the spray phase, each node makes a direct contact with the destination to send the copy of the message.

The mobility models, on the other hand, also take a vital role. In most of the situations, nodes are either move in a random direction or maybe in a specified path. In the case of random direction, most of the nodes are supposed to be a move by following the random waypoint or random walk mobility model. These classes of mobility models are the simplest mobility pattern ever for the nodes. Another mobility model, perhaps the most realistic, is the shortest path map-based mobility where node performs the movement by finding the shortest possible path among all possible paths.

The simulation for such opportunistic networks can be performed by various simulators. One of the most common among them is Opportunistic Network Environment (ONE) simulator. This is a free open-source simulation platform that is based on Java. The simulator package is available in Github (https://github.com/akeranen/the-one). The configurable parameters can be set using a configuration settings file with option = value method. We can even generate our own routing and the mobility models and deploy it in ONE and evaluate the performance of the same.

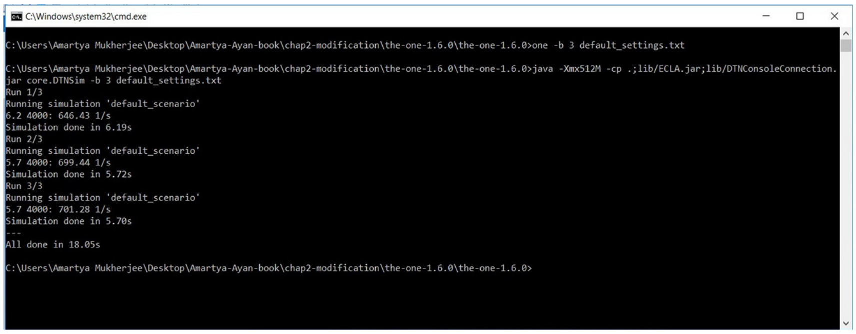
ONE is a discrete event-based simulation environment whose main objective is to model node movement, routing, internode contact, and message handling. One of the core advantages of this simulator is that it is based on the Java platform; thereby a cross-platform advantage will be obtained by default. The routing and mobility models can be designed and implemented in a pretty efficient way. The movement models are the synthetic models or some time existing movement traces. The node connectivity can be generated either by the location of the node or bit rate and communication range. Based on the mobility models, the routing strategies decide in which path should the message be forwarded. Further, the message generation has been controlled using an event generator. The messages are unicast in general in the simulation area.

Fig. 2.14 depicts the simulation environment for the default scenario. The execution of the simulation involves some simple steps. After downloading the simulation software package from GitHub just extract the.zip



```
2348.6: Message relayed c64<->c47 M5
2348.6: Message dropped c47 M60
2348.6: Message relay started c64<->c47 M18
2348.9: Message relayed p35<->c73 M78
2348.9: Message dropped c73 M11
2348.9: Message dropped c73 M5
2348.9: Message relay started p35<->c73 M3
2349.3: Message relayed t122<->c50 M18
2349.3: Message dropped c50 M14
```

Figure 2.14 One simulator graphical user interface environment.



```
C:\Users\Amartya Mukherjee\Desktop\Amartya-Ayan-book\chap2-modification\the-one-1.6.0\the-one-1.6.0>one -b 3 default_settings.txt

C:\Users\Amartya Mukherjee\Desktop\Amartya-Ayan-book\chap2-modification\the-one-1.6.0\the-one-1.6.0>java -Xmx512M -cp .;lib/ECLA.jar;lib/DTNConsoleConnection.
jar core.DTNSim -b 3 default_settings.txt
Run 1/3
Running simulation 'default_scenario'
6.2 4000: 646.43 1/s
Simulation done in 6.19s
Run 2/3
Running simulation 'default_scenario'
5.7 4000: 699.44 1/s
Simulation done in 5.72s
Run 3/3
Running simulation 'default_scenario'
5.7 4000: 701.28 1/s
Simulation done in 5.70s
---
All done in 18.05s

C:\Users\Amartya Mukherjee\Desktop\Amartya-Ayan-book\chap2-modification\the-one-1.6.0\the-one-1.6.0>
```

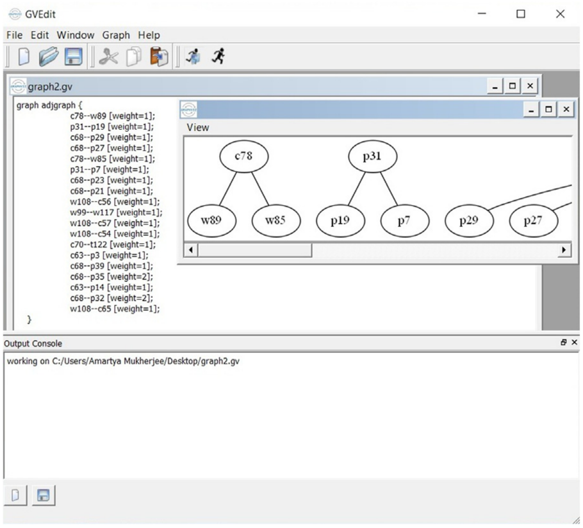
Figure 2.15 One simulator batch mode execution (no. of simulation is 3).

file and inside it, we can see a batch file compile.bat. Inside the folder, we may create a local.bat file in which we should put the command "cmd" and save the local.bat file. After that, just double click it to open the command line terminal. Now in command line terminal, we can write "compile" to compile the source code of the simulator. After successful compilation, we can now use the simulator for simulation purpose. We can run the simulator in two different modes. In the first mode, the GUI will appear, and it can be done by using the command "one" in command prompt. In the second case, we can run the simulation in batch mode. For that, we have to write "one −b number_of _simulation configuration_file_name." Fig. 2.15 shows the batch mode execution of one simulator. The simulator itself takes the configurable parameter in the form of a text file. By default the name of that file is default_settings.txt. Here we can set the mobility models, routing strategies, number of reports, and types of reports. Not only that we can also configure a number of available nodes, transmission range, node speed, and the initial energy of each node as well.

2.5.3 Simulation result and analysis

One simulator may generate various types of the report file. The most common of them are message status report, buffer occupancy report, encounter versus unique encounter report, adjacency graph, namely report, energy report, and so on. The message status report is mainly useful to check the number of the message created, delivered, dropped, and final delivery probability achieved by a particular scenario under a predefined number of nodes. Buffer occupancy report, on the other hand, produces the total amount of buffer memory occupied by message for a particular node along with the variance. The adjacency Graphviz report produces the weighted graph representation of the message transfer path from source to destination. The report produces a Graphviz format.gv file, which is then executed by Graphviz edit software. Fig. 2.16 is the Graphviz editor software to generate the node message transfer and encounter. And Fig. 2.17 illustrates the message transfer graph produced by Graphviz.

The message status report of the different scenario is also helpful to analyze the message delivery performance of the different strategies. Along



```
graph adjgraph {
            c78--w89 [weight=1];
            p31--p19 [weight=1];
            c68--p29 [weight=1];
            c68--p27 [weight=1];
            c78--w85 [weight=1];
            p31--p7 [weight=1];
            c68--p23 [weight=1];
            c68--p21 [weight=1];
            w108--c56 [weight=1];
            w99--w117 [weight=1];
            w108--c57 [weight=1];
            w108--c54 [weight=1];
            c70--t122 [weight=1];
            c63--p3 [weight=1];
            c68--p39 [weight=1];
            c68--p35 [weight=2];
            c63--p14 [weight=1];
            c68--p32 [weight=2];
            w108--c65 [weight=1];
 }
```

```
working on C:/Users/Amartya Mukherjee/Desktop/graph2.gv
```

Figure 2.16 GVedit editor console.

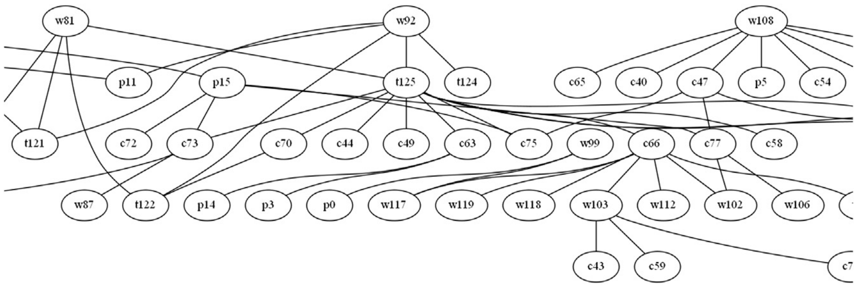


Figure 2.17 Message transfer graph generated by using the adjacency graph viz report.

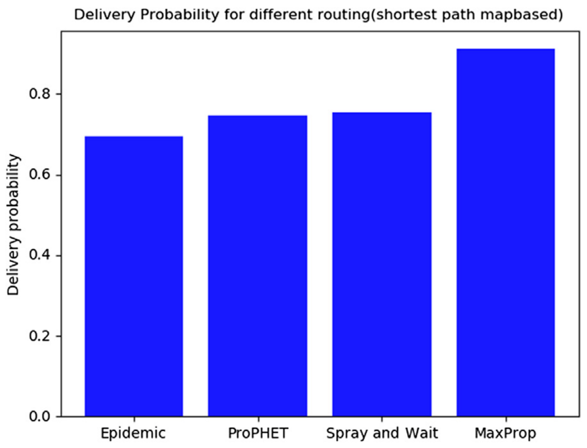


Figure 2.18 Delivery Prob. Comparison of the popular routing strategy under the shortest path map-based mobility model.

with that message overhead, latency and hop count are also a pretty important property that is to be incorporated.

Fig. 2.18 shows the comparison between the delivery probabilities of the different routing mechanisms in delay tolerant network (DTN) systems [16]. The figure shows MaxProp that performs best in the context of delivery ratio. This is because of the very important phenomenon of MaxProp that is the receiving of the acknowledgments of each message. Also for organized shortest path mobility model, it is natural that choosing the best path is significantly effortless. The epidemic, on the other hand, produces a benchmark of near about 0.69 because of flooding of the messages towards the nodes in a specific path. However, this leads to a buffer overflow for the relay nodes. The ProPHET algorithm, on the other hand, produces a better result in comparison to an epidemic because of its probabilistic nature which is mainly based on the restricted flooding based on delivery predictability metric. Spray and wait algorithm, on the other hand, produces better performance in terms of the delivery prob.

On the other hand, in average latency point of view spray and wait for strategy performs best because in the waiting phase a direct contact happens with the nodes as shown in Fig. 2.19. Second, the number of messages flooded in a particular moment of time is also limited. As a result, the buffer will not overflow and congestion will not happen in any circumstances.

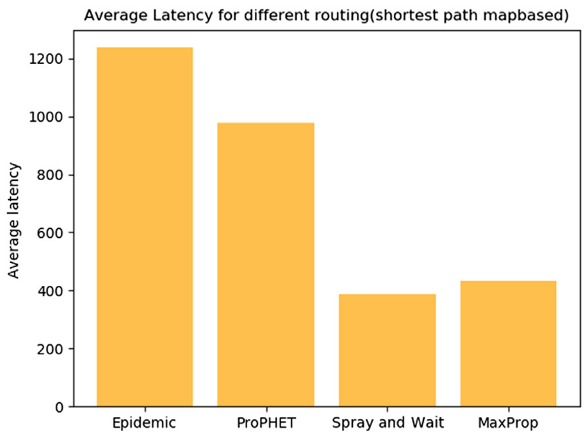


Figure 2.19 Average latency for different DTN routings in shortest path map-based model.

2.5.4 Localization of sensor nodes in simulation perspective

This particular simulation platform mimics the identification and the location of the sensor nodes in a WSN using unmanned aircraft systems. The simulator has been developed in Python environment. The main goal here is to locate the randomly deployed sensors in a specific area using unmanned aircraft system (aerial system) [17]. In the scenario, the drones are deployed in such a way that they are they perform optimal coverage over the whole area. The mobility of the drone is predefined, and they perform a raster scan over the whole coverage area. In some special situation, depending on the scenario, the mobility models like cluster mobility [18] are also applicable. Unmanned aerial vehicles (UAVs) are mainly used as data mule or ferry nodes whose sole job is to collect the data through a TCP-based network channel.

The simulation engine, in this case, comprises several modules. The main module associated with sensor module, network propagation module, message passing, figure generation, and finally a statistical analysis module. In the sensor module, the sensor has been defined with the feature of both the client and the server. It starts the operation to save the first point of contact. After that, it shares the list of voices to the UAV in communication. There is another signal threshold has been considered that indicates whether the UAV manages to receive a sensor node that is inside the radius of the coverage area. In order to calculate the threshold thereafter, initialize the radius of stationary sensor (RSS) threshold and communicate with another node ssi, when RSS $>=$ s_t then this threshold finally listen to the beacons sent by the UAV. In the second stage, the network initialization module works, where the first operation is to compute the distance between the drone and the sensor node Ɗ in the 3d plane as follows:

$$Ɗ = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2} \tag{2.4}$$

where (x_1, y_1, z_1) is the initial position of drone and (x_2, y_2, z_2) is the position of the stationary sensor node. Then start the UAV object and initialize the RSS threshold for each sensor node and start listening. As the UAV starts to listen to neighbor, the neighborhood discovery starts throughout the simulation field. In order to achieve that, a neighborhood list has been built, and it is to be assumed that two sensors or sensor and UAV are in neighbor if two are included in the communication range of each other means, distance $Ɗ = rs$, where rs is the radius of sensor. Further, the count of the number of scan line has been done in order to get the optimal execution time where execution time is nothing but the time contact of UAV with the sensor in a single passage. Here the execution time T_E can be expressed as follows:

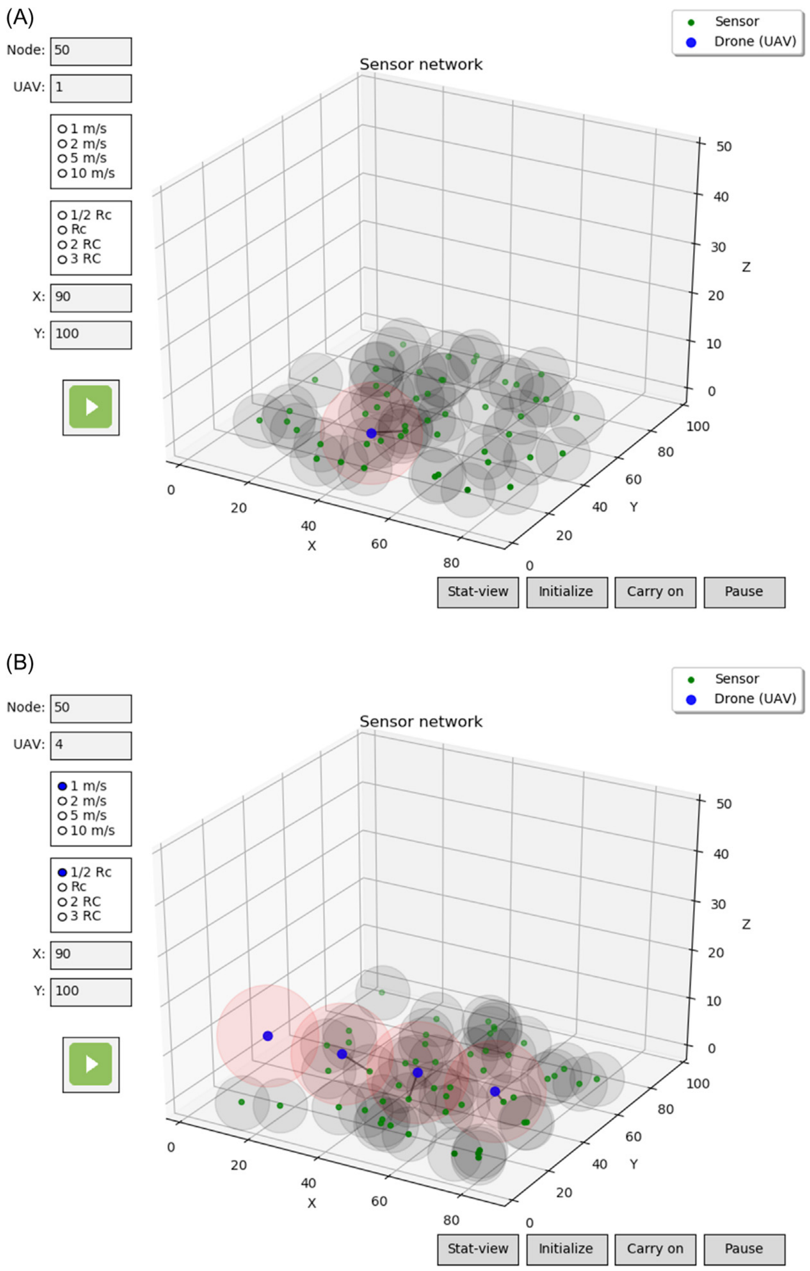
$$T_E = S_t(\text{single passage line}) \times N \tag{2.5}$$

Where S_t scanning time for single line and N is the number of the line.

Fig. 2.20A depicts the arrangement of the ground sensor node with the UAV trajectory within the simulation boundary and Fig. 2.20B shows four UAVs scanning whole-sensor deployment region within 3D plane in a raster scan pattern.

2.5.5 Performance analysis

The performance of the scanning of UAVs in localized sensor node deployment zone can be analyzed based on several performance metrics. The scanning rate of the UAV node can be controlled by the rate control parameter. In this case, we can choose the value of rate control (RC) in between 0.5 and 3. In case of increasing RC value, UAV will set a dense raster path from source to destination and hence the scanning of the sensor nodes happens in a repeated fashion. Depending on the speed and the RC value, the sensing precision has been judged. Fig. 2.21A–D represents computation time estimated for increasing speed of scanning, average error rate against scanning speed, message exchange rate within a certain time interval, and percentage of localized sensor scan covered for a predefined rate of scan control values. Fig. 2.21A shows that the computation time is decreasing with the increasing speed. This is because for an increasing speed, the number of ground sensor


(A)

Node: 50

UAV: 1

○ 1 m/s
○ 2 m/s
○ 5 m/s
○ 10 m/s

○ 1/2 Rc
○ Rc
○ 2 RC
○ 3 RC

X: 90

Y: 100

(B)

Node: 50

UAV: 4

[✓ 1 m/s]
○ 2 m/s
○ 5 m/s
○ 10 m/s

[✓ 1/2 Rc]
○ Rc
○ 2 RC
○ 3 RC

X: 90

Y: 100

Figure 2.20 (A) UAV trajectory and the scanned scattered ground sensor nodes. (B) Deployment of four UAVs in a raster scan mobility pattern.

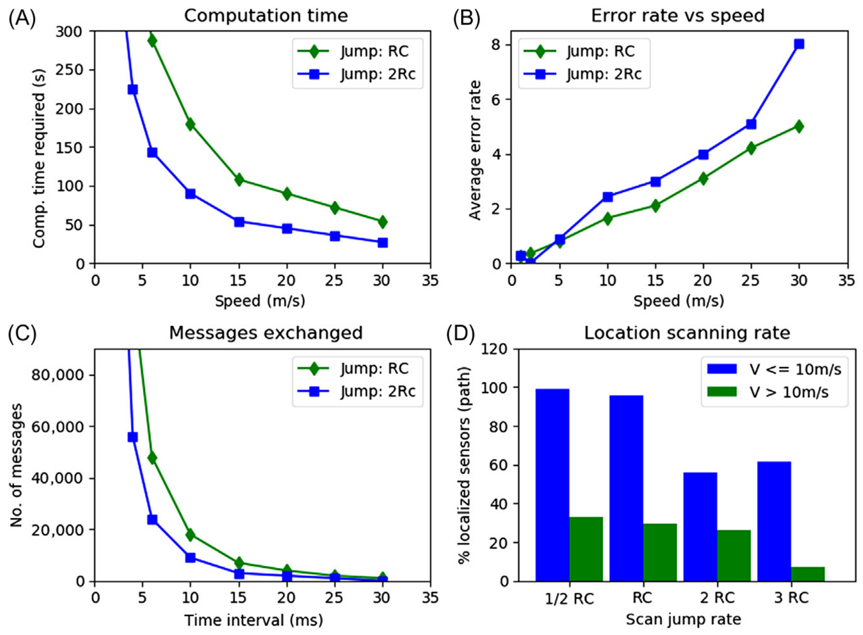


Figure 2.21 (A) Computation time estimated for increasing speed of scanning, (B) average error rate against scanning speed, (C) message exchange rate within a certain time interval and percentage of localized sensor scan covered, and (D) location scanning rate for all RC parameters.

scanned by the UAV is decreased, and as the number of sensors gets covered is less so the computation time involved with them is also decreased. The figure shows the pattern of computation time versus speed for RC and 2RC scan rate. Higher the scan rate, even more, decrease the computation time, as a result, the sensing error will increase as shown in Fig. 2.21B. Hence the increasing error may cause a chance of nonprecise scan output. In this scenario, the same situation appears for 2RC. The error rate has made the highest benchmark for the 2RC jump value.

The third figure in this case (Fig. 2.21C) shows no of message transfer through the beacon within a stipulated time interval. The result shows a decay of message with an increasing amount of time. This is mostly happening due to the message buffer overflow during the message acquisition and the unicast message transfer. Fig. 2.21D illustrates the location scanning rate for all RC parameters. From the diagram, it is clear that in case of the velocity of UAV less than equal to 10 m/s the sensor scanning rate is relatively high for all RC values. However, if the velocity value increases more than that, the scanning rate decreases drastically results in the lowest benchmark for 3RC.

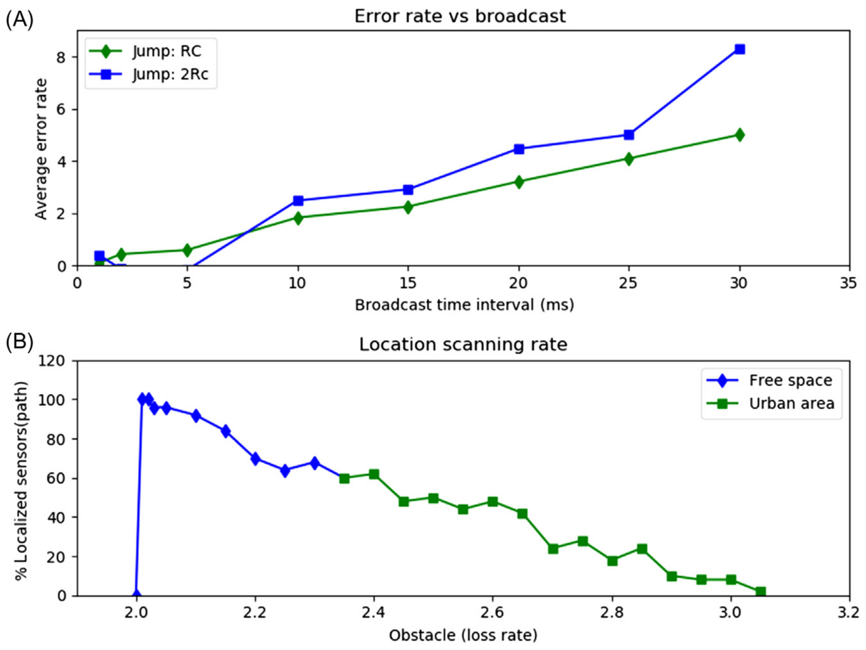


Figure 2.22 (A) Comparison of broadcast time versus error rate (B) reveals the loss rate for localized sensors.

Fig. 2.22A shows a comparison of broadcast time versus error rate. From the graph, it is clear that the increasing broadcast time leads to high average error rate. This will increases drastically for 2RC that doubles the scan jump rate value. It is quite a natural phenomenon because in an increasing scan jump rate, the number of sensors covered by the UAV will decrease, and therefore scan error will also increase. Second, Fig. 2.22B reveals the loss rate for localized sensors. Here we can easily understand that the loss rate for an urban area is very high due to the presence of dense obstacles. In case of free space, the loss rate is not much high, but the decreasing number of localized sensors may sometimes affect the loss rate in this case.

2.6 Summary

In this chapter, we have discussed several real-life applications of cutting edge application of sensors in the field of IoT and other related platforms. The technologies that are relevant to the application of sensor-oriented IoT architecture and the protocols that are widely used have been emphasized in this chapter. Different real-life applications using the existing technology have also been addressed. The simulation platforms reveal the actual development, and deployment challenges are mostly observed during the implementation of the WSN-based applications.

References

[1] N. Dey, A.E. Hassanien, C. Bhatt, A. Ashour, S.C. Satapathy (Eds.), Internet of Things and Big Data Analytics Toward Next-generation Intelligence, Springer, Berlin, 2018.
[2] A. Schaller, K. Mueller, Motorola's experiences in designing the internet of things, Int. J. Ambient. Comput. Intell. (IJACI) 1 (1) (2009) 75–85.
[3] E. Nasr, E. Kfoury, D. Khoury, A pervasive IoT scheme to vehicle overspeed detection and reporting using MQTT protocol, ICT for a Better Life and a Better World, Springer, Cham, 2019, pp. 19–34.
[4] A. Kurniawan, Learning AWS IoT: Effectively Manage Connected Devices on the AWS Cloud Using Services such as AWS Greengrass, AWS Button, Predictive Analytics and Machine Learning, Packt Publishing Ltd, 2018.
[5] B. Langmead, A. Nellore, Cloud computing for genomic data analysis and collaboration, Nat. Rev. Genet. 19 (4) (2018) 208.
[6] N. Dey, A. Mukherjee, Embedded Systems and Robotics with Open Source Tools., CRC Press, 2018.
[7] S. Fong, J. Li, W. Song, Y. Tian, R.K. Wong, N. Dey, Predicting unusual energy consumption events from smart home sensor network by data stream mining with misclassified recall, J. Ambient. Intell. Humanized Comput. (2018) 1–25.
[8] G. Elhayatmy, N. Dey, A.S. Ashour, Internet of things based wireless body area network in healthcare, Internet of Things and Big Data Analytics Toward Next-generation Intelligence, Springer, Cham, 2018, pp. 3–20.
[9] A. Mukherjee, N. Dey (Eds.), Smart Computing with Open Source Platforms, CRC Press, 2019. Available from: https://doi.org/10.1201/9781351120340.
[10] G. Elhayatmy, N. Dey, A.S. Ashour, Internet of things based wireless body area network in healthcare, Internet of Things and Big Data Analytics Toward Next-Generation Intelligence, Springer, Cham, 2018, pp. 3–20.
[11] N. Dey, S.A. Amira, F. Shi, S.J. Fong, J.M.R.S. Tavares, Medical cyber-physical systems: A survey, J. Med. Syst. 42 (4) (2018) 74.
[12] W. Arellano, I. Mahgoub, Traffic modeler extensions: A case for rapid VANET simulation using, OMNET++, SUMO, and VEINS, 2013 High Capacity Optical Networks and Emerging/Enabling Technologies, IEEE, 2013, pp. 109–115.
[13] A. Bhattacherjee, S. Roy, S. Chakraborty, A. Mukherjee, S.K. Bhattacharya, Study of wireless communication through coal using dielectric constant, Information Systems Design and Intelligent Applications, Springer, New Delhi, 2015, pp. 831–841.
[14] A. Mukherjee, N. Dey, R. Kumar, B.K. Panigrahi, A.E. Hassanien, J.M.R.S. Tavares, Delay tolerant network assisted flying Ad-Hoc network scenario: modeling and analytical perspective, Wirel. Netw. 25 (5) (2019) 1–21.
[15] S. Patra, A. Balaji, S. Saha, A. Mukherjee, S. Nandi, A qualitative survey on unicast routing algorithms in delay tolerant networks, International Conference on Advances

in Information Technology and Mobile Communication, Springer, Berlin, Heidelberg, 2011, pp. 291–296.
[16] S. Saha, A. Sheldekar, A. Mukherjee, S. Nandi, Post disaster management using delay tolerant network, Recent Trends in Wireless and Mobile Networks, Springer, Berlin, Heidelberg, 2011, pp. 170–184.
[17] A. Mukherjee, S. Chakraborty, A.T. Azar, S.K. Bhattacharyay, B. Chatterjee, N. Dey, Unmanned aerial system for post disaster identification, International Conference on Circuits, Communication, Control and Computing, IEEE, 2014, pp. 247–252.
[18] A. Mukherjee, N. Dey, N. Kausar, A.S. Ashour, R. Taiar, A.E. Hassanien, A disaster management specific mobility model for flying ad-hoc network, Emergency and Disaster Management: Concepts, Methodologies, Tools, and Applications, IGI Global, 2019, pp. 279–311.

CHAPTER THREE

Wireless sensor network: principle and application

Contents

The term “wireless system” is very common today. The concept involves sending and receiving signals in the form of radiofrequency. The fundamental idea in such case implies the methodology of sending voice, audio, video and any other form of data. The entire operation of the wireless communication perhaps can be visualized as a layered approach of the open systems interconnection (OSI) protocol stack. As the data in the wireless transmission can be considered as in the form of signal, the conversion of the signals in the data from is obvious. In this chapter, we are going to discuss and analyze the fundamental features and the application of wireless communication in the form of wireless sensor networks (WSNs) applications.

3.1 Wireless communication and sensor networks

A sensor network can be considered as wireless as there is no physical wire connection among different nodes [1]. In such case, the only communication medium is the wireless protocols. Several kinds of standards can be addressed in this aspect. Most of the common protocols that can be used re the Bluetooth, the Zigbee, which complies with IEEE802.15 standard, and the Wi-Fi radio of IEEE802.11 b/g/n standard. A sensor network is basically a special kind of wireless network. The main difference can be made in the constraint of bandwidth, buffer memory, computation power, and battery life. For example, a Zigbee device that is widely used for data transmission in WSN takes 100 mW of power. Another widely used WSN processing module is Raspberry Pi 3's Wi-Fi unit, which takes a maximum of 20-mW power. The memory, in this case, will be of 512 MB as random access memory with a Broadcom BCM2837 having processing frequency of up to 1 GHz. If we design such a network with Arduino-based system, the processing speed is of certain MHz in that case. In another way, it can be said that the sensor network is basically different than the traditional network in the usage pattern. In a traditional network architecture, the users may be connected with a gateway node or hotspot and expect a service from it. It seems to be something similar to a client—server approach. The job of the network, in this case, is to bring the server and the client together. The principle of the sensor network is, on the other hand, something similar to a distributed architecture. Each node can grab the data from the ambience and process it within the node. A single node generates data that may not be useful in general, but the information generated by the system as a whole gives a meaningful information entity.

3.1.1 Network architecture

To design efficient sensor network architecture we can consider a multihop ad hoc network that consists of short-range, low power communication devices [2]. Depending upon the application deployment perspective the architecture of the sensor network may vary but the fundamental architecture of the network remains the same. From the bottom to the top we can view this component of the sensor network as a layered architecture where the bottom layer is the physical entity of the sensor components. After that the raw data can be stored in local storage and local

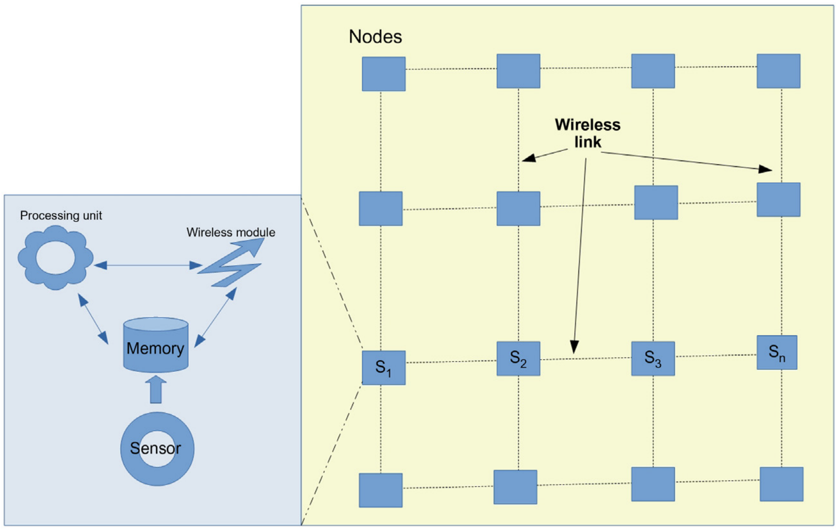


Figure 3.1 Sensor node and fixed grid architecture.

processing units. A parallel component of this layer is the radio unit that performs the wireless communication between the other sensor nodes and the base station. In the sensor layer, several diversified usages of the sensors have taken place. Such a multisensor system is widely used for ubiquitous sensing applications. This concept is perhaps the most well-known concept for the IoT applications [3,4] where the sensing task has to be performed through the heterogeneous classes of sensors. The figure illustrates a fundamental architectural building block of the sensor network (Fig. 3.1).

3.1.2 Network topology

The nodes in sensor networks mostly have a limited amount of power source. The design issue for the topology should consider this [5,6]. In general, energy-efficient routing models have been proposed in order to perform optimum message transfer method. There are two types of topological aspect that can be considered in this scenario. In the first case, the network nodes are stationary and they may form some fixed formation of sensor nodes. The most common topologies used are ring, mesh, and star. The other kind of systems such as grid and star-connected topology is also used. In order to achieve the performance, we have to select proper topological structures. In the case of bus topology, there is a single path available to communicate among the different sensor nodes. Chances of failure, in this case, are more as there is only one path. The congestion is also high due to a single path; proper load balancing cannot be done in such a scenario. Therefore, the topology is less reliable. On the other hand star and tree topologies have more load balancing capability in comparison to the bus. One of the drawbacks in ring structure is that the number of paths may be double but, it is in the same direction. In the case of the tree topology, the network might be fault-tolerant, but the energy consumption is high and the packet reception ratio is less. The more efficient topology that can be addressed in case of WSN is the mesh and grid. Here the data will route through multiple paths. The mesh and grid structures offer the most stable load balancing factor. The congestion in the path is also much less. The topologies also offer more amount of network lifetime and high packet delivery ratio.

There are some issues in the WSN topology design that should be considered. In the case of static nodes and dynamic node different class of design issues might affect the performance of the sensor networks. The studies said that the mobility of the node never directly affects the performance of the network. The major design concern is to find out the best topology such that the relative position and the orientation of the network node are almost fixed with each other. On feature that can be chosen as an important factor is the number of network usage by the different nodes. The message pattern and the message length for nodes may not be similar for all in most of the cases. The grid topology in this context showing up the best mechanism for transmission and receiving of data as vertices of the grid is considered as nodes that transmit the message through the edges. The edge connects the neighbors with single or multiple paths where data packet gets forwarded from source to destination. By assuming this criterion the n neighbor sensor network can be presented where n can vary from 1 to 8 in a two- and three-dimensional space.

In order to achieve that we can consider $\omega\ (x,\ y)$ is a sensor network of having an $x \times y$ grid which represents the $x \times y$ number of nodes at a time each node $n(a,b)$ such that $0 \leq a \leq y - 1$ and $0 \leq b \leq x - 1$. If a packet gets transmitted from source to destination node the following models must hold for a specified sensor node n for a given source (S) and destination (D).

$$S = (a_s,\ b_s) \tag{3.1}$$

$$D = (a_d,\ b_d) \tag{3.2}$$

$\triangle a = \left\| a_s - a_d \right\|$ The transition of the node in direction *a*.
$\triangle b = \left\| b_s - b_d \right\|$ The transition of the node in direction *b*.

The network is establishing in such a way that two adjacent nodes can be identified by the neighborhood of the nodes and presents an optimal number of hops from source to destination. One particular node might be composed of a minimum of two neighbors and up to eight neighbors. The neighbor configuration can be distributed in an entire grid through an aggregated manner. Fig. 3.2 depicts the different neighborhood discovery methods for WSN.

In the case of four neighbors, the WSN optimal hop count can be derived as

$$H(s, d) = \triangle a + \triangle b \tag{3.3}$$

In the case of six neighbors, the WSN value will become computed through the following criteria:

$$H(s, d) = \begin{cases} \triangle \mathrm{a} + \triangle \mathrm{b}, & \mathrm{a_s} > \mathrm{a_d} \text{ and } \mathrm{b_s} < \mathrm{b_d} \\ (\triangle \mathrm{a}, \triangle \mathrm{b}) & \mathrm{a_s} < \mathrm{a_d} \text{ and } \mathrm{b_s} > \mathrm{b_d} \end{cases} \tag{3.4}$$

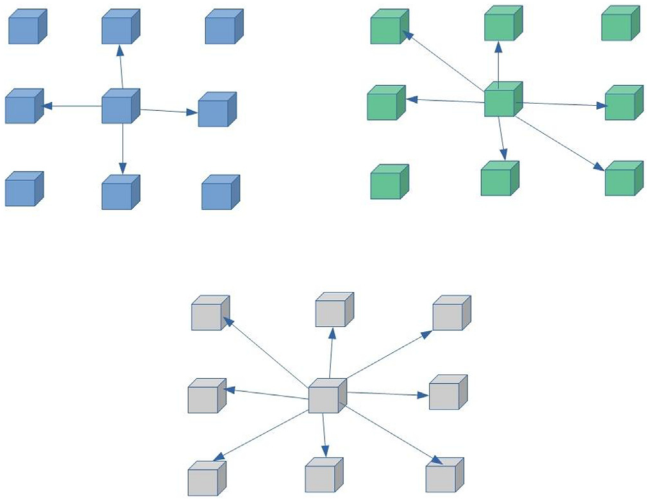


Figure 3.2 Neighborhood discovery of wireless sensor network (four neighbors, six neighbors, and eight neighbors respectively).

If it is the case of eight neighbors, then the total WSN hop count can be computed as

$$\mathrm{By}H(s,d) = (\triangle a, \triangle b) \tag{3.5}$$

3.2 Sensor components and technology

The core component and the technology that is evolving for the sensors are mostly evolved decades ago. In those days the technology was classified for the military applications. Thanks to the open-source revolution, nowadays various sensors are available in the markets, which are often used in the military application in a year ago. One of the paramount technologies that involve in the sensor application is micro electro-mechanical systems (MEMSs). In the medical domain also a significant improvement of the miniaturized network node notices in the form of capsules [7].

3.2.1 Micro electro-mechanical devices

MEMS is the most sophisticated sensor technology that is created by a human being. This is a technique of miniaturization of electrical and the electro-mechanical systems. Such sensors get fabricated in a fraction of time (even in micron) scale, but the features remain intact. The internal sensors of the smartphone, autopilot units, and automobile devices are often used in MEMS device [8]. The most popular devices are an accelerometer, gyroscope, magnetometer, Hall-effect sensors, sonar, barometric pressure, and many more. Typically those sensors cost around \$10–\$500.

Most of the MEMS-type sensors have inertial property. In this case, the sensors are made to transduces and convert the physical character of the body into a measurable signal quantity. The system is shown in Fig. 3.3.

Often the force that applied to the sensor gets transduced into a linear voltage output level that is scaled. A specific sensitivity value has to be imposed in to the linearly scaled voltage. The inherent properties of such sensors are the tiny package size, low cost, and precise performance. As these are the micromachined category sensors it can be fabricated on a mass scale. And it reduces the overall processing cost. As the device

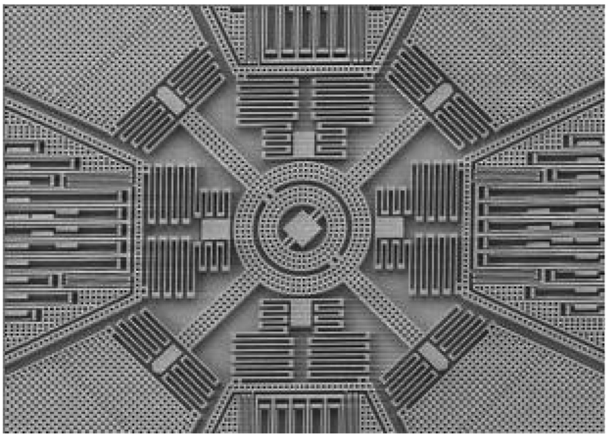


Figure 3.3 Micro electro-mechanical system sensor (courtesy: community.arm.com).

performance is one of the key factors, the micromachined sensors use successive disposition on structural and surface layer, which further produce a freestanding device. To obtain structure thickness of an order of a micron, polycrystalline silicon is preferably used. Although this type of micromachined devices sometimes suffers from sensitivity issue in comparison to full-scale sensors, this is negligible because of the trade-off between the cost and the volume of the sensing device. In the next section, we are going to discuss typical MEMS accelerometer.

3.2.2 MEMS accelerometer

Fig. 3.4 depicts the MEMS accelerometer that is fabricated as a package. The impulse generated by the accelerometer sends as an electrical signal to an adjacent signal processing unit.

The fundamental architecture of accelerometer consists of a central proof mass that is suspended between an anchor and a dashpot. The spring differs from the position of the inertial mass under the applied acceleration of the object on which the sensor is mounted. Mostly a capacitive type of accelerometer can be built on which mass is suspended inside of the two plates of the capacitor. A small change in the charge density is, therefore, being considered as acceleration, which is further processed and amplified by using a special electronic circuitry. In another method, a piezo-resistor is used to detect the stress on the spring of the mass, hence convert the stress-energy to an electrical signal.

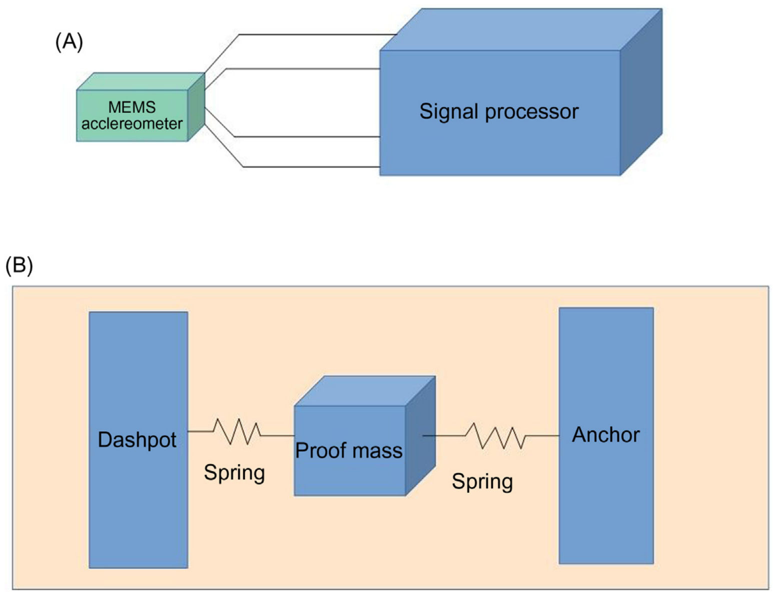


Figure 3.4 (A) Fabricated micro electro-mechanical system accelerometer and (B) working principle.

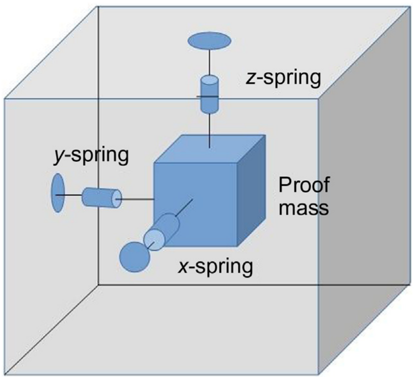


Figure 3.5 Angular rate gyroscope sensor.

3.2.3 MEMS gyroscope

Micromachined angular rate gyroscope is fabricated with a vibrating mass inside the core of the moving frame (Fig. 3.5). Within a fixed reference plane, the proof mass oscillates with a certain velocity vector.

If the reference frame rotates at a rate ω, it results in a Coriolis force. As a result, the Coriolis acceleration happens that results in the displacement of the mass. An energy transfer is the obvious effect of that from the primary mode of oscillation to a secondary mode of oscillation. There are two distinct modes of frequency generation that are similar to the oscillation frequency, which causes a temporal excitation signal.

The primary use of such angular rate gyroscope is to use in the absolute braking systems for automobile, autopilot units for controlling micro aerial vehicles where the stability of the frame is a crucial issue. Some of the sports utility vehicles, surveillance, and the unmanned aerial vehicle (UAV) for aerial photography [9] are also using this sensor for better yaw control.

3.3 Sensor network protocols

There are different approaches that are possible in concern with the routing protocols of the sensor networks [10]. A sensor network comprises plenty of nodes. Each node has computing, communication, and ambient sensing abilities. A sensor network protocol is mainly designed keeping in mind that the system is fault-tolerant, scalable, and dynamic. As in the sensor networks, the conservation of the energy is the primary concern, most of the routing strategy has been developed by considering the network lifetime, which has a direct effect on energy conservation.

The most conventional routing protocols that are widely used in the last decade are flat-based routing, adaptive-based, and hierarchical-based routing. In the next sections, we are going to discuss the different routing strategy for WSN.

3.3.1 Flat-based routing

In this method, there are three parameters that have to be considered: (1) priority levels of packets, (2) quality of service of each path, and (3) the energy resources of the network.

Direct diffusion method is one of the significant methodologies in this category (Fig. 3.6). This is a data-centric approach. Sensor data, in this case, can be identified by attribute-value pair. To make the network efficient, the in-network aggregation has been done. As a result, the packet redundancy and the repeated transmission have been reduced significantly that causes minimum energy consumption. There are three major types of

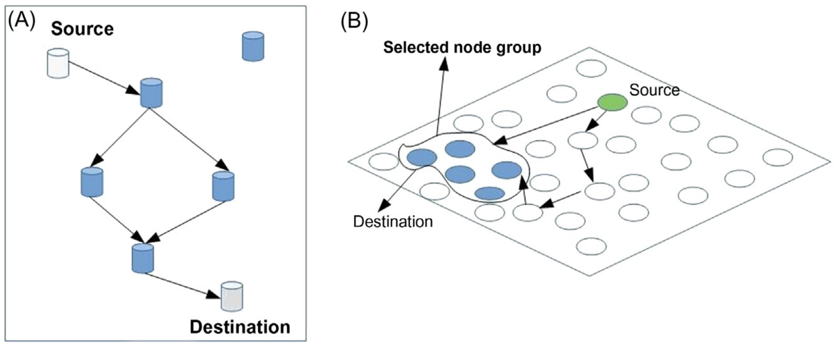


Figure 3.6 Direct diffusion method: (A) gradient based and (B) information forwarding toward a group of nodes.

direct diffusion can be addressed in this context. In the first technique, a gradient-based diffusion has been performed where a query is flooded and the gradient which is set up to satisfy the query has been checked. To restrict the flooding of the query in this method a reinforced method is to select the number of paths. In another concept, the valuable information has to be forwarded spontaneously in a certain region of the sensor network.

In sequential assignment based routing a tree-structured network, the hierarchy has to be designed. The path from one node to another node in that tree is built by avoiding the nodes with low energy and QoS. The factors like delay, QoS, and the energy dissipation during packet traversal in the specified path has been considered in this scenario.

Minimum cost forwarding involves the concept that all nodes know the direction of the routing of data packets. In that case, the sensor node does not identify by the unique ID value. Cost estimation of packet forwarding happens between the sensor sources and the base station. In this mechanism, each node forwarded the message to its neighbor and the node checks whether the packet has been received through the least-cost path based on hop count. If it is so then the data get forwarded to the neighbor until it finds the destination.

3.3.2 Adaptive routing mechanism

Adaptive routing assumes that the node of the sensor networks has the potential to acts as an independent base station [11]. The protocol mainly exploits the phenomena called information negotiation. The node, in this case, expects the immediate information exchange as the query has been generated by a node. Second, cluster-based route optimization and the load balancing protocol (ROL) is also new research domains in the current era. In order to achieve the requirement of the application, the protocol uses the QoS metrics such as (1) back off: this technique minimizes the setup traffic toward the cluster. Each node must join to the nearest cluster and send the data packet through the minimum number of hops. As a result of the bandwidth utilization and propagation, the delay due to queuing is reduced significantly and hence, it guarantees the low power consumption. (2i) Hop number: this plays a significant role to increase the transmission range. In most cases the number of hops increases, and the retransmission becomes cumulative. But to avoid interference during long-range transmission the multihop communication is the best choice. The ultimate effect of this increases throughput. (3) Backup channel count: the backup channels are essentially needed for switching off from one to another in case of channel failure. A channel healing procedure is widely introduced to reduce data loss and network availability time. (4) Multiple paths: as the number of the paths for routing is increased, the chances of data delivery are also increased. Second, bandwidth and network robustness will also increase significantly.

The algorithm acts based on the following phases

1. Setup phase: in this phase the cluster is formed, and as nodes receive the discovery message from the sink the state of the node changes from waiting to discover. After that, it checks whether it has been selected as a channel to form the cluster or not. If it is so, the cluster is ready to broadcast the advertisement messages. In another case, it broadcast the discovery message to the neighbors. While the channel broadcasts the message it takes the hop count as 0. As a node receives the message, then it can do either of the two:
   a. If it already belongs to the same cluster, it ignores the message.
   b. If the message has smaller hop count than the stored message, the latter is deleted and the former is retained.
2. Transmission phase: in this phase the nodes transmit the data toward the channel and before it, and the data will be aggregated. After that, the data will be immediately multiplex by the time division multiple access (TDMA) slot assignment. The communication between the cluster and the sink has been done by means of code division multiple access mechanisms.

3.3.3 Hierarchy-based routing

The primary advantage of hierarchical routing is the scalability and efficient communication paradigms. In this case, the nodes perform a hierarchical process. The nodes having a higher energy level will act as a processing node that processes the data coming from the different sensors. The low energy node on the other hand used for the sensing activity. Low energy adaptive clustering (LEACH) is one of the classical hierarchical clustering protocols. This protocol supports distributed cluster support. A randomized rotation of the cluster head is to be made in this case. LEACH protocol uses a reclustering after a given interval of time. The protocol performs two distinct operations. In setup phase where the cluster has been set up and the head of the cluster get selected. This can be done by a predefined parameter τ, and the cluster head elects as follows. A node chose a random number R where $0 < R < 1$. If the number is lesser than the threshold value, then that node is the cluster for the current iteration. The set of the node that is not selected as the head can be considered as γ and the threshold value $\check{D}$ is denoted by

$$\check{D} = \frac{\tau}{1-\tau\left(R\mathrm{mod}\left(\frac{1}{\tau}\right)\right)} \mathrm{if}\, n \in \check{D} \tag{3.6}$$

Fig. 3.7 depicts the operation of the LEACH protocol [12]. To perform a comparative study between adaptive and hierarchical routing, we can consider the data delivery ratio as a crucial parameter. This service level parameter shows the effectiveness in the transmission of the data in a

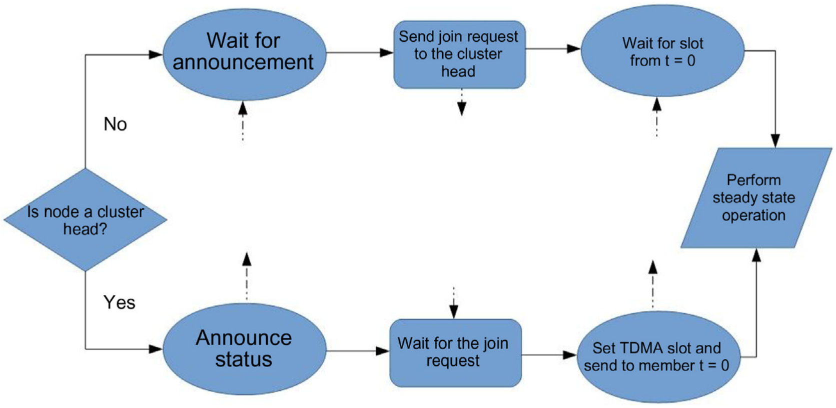

Wait for announcement
Send join request to the cluster head
Wait for slot from t = 0
No
Perform steady state operation
Is node a cluster head?
Yes
Announce status
Wait for the join request
Set TDMA slot and send to member t = 0

Figure 3.7 Operation of cluster head selection for low energy adaptive clustering protocol.

single direction. The parameter data delivery ratio also reflects the robustness of the model because the packet that arrives late has to be discarded and considered as insignificant.

The term data delivery ratio can be expressed by giving the following model:

$$\mathrm{DDR}(n_i) = \frac{\sum \text{data delivered to sink}}{\sum \text{data generated by } n_i} \times 100 \tag{3.7}$$

The study reveals that the ROL significantly outperforms with respect to LEACH in a timeless constraints scenario in the presence of the communication failure. The evaluation criterion simulates the case of communication failure due to hardware and battery failure. In the general case, the data delivery ratio data delivery ratio (DDR) decreases as the number of node increases. In such a case, the ROL protocol holds better delivery ratio in comparison to the LEACH. The end-to-end delay in case of LEACH is also higher in comparison to the ROL protocol because in ROL the data has multiplexed over multiple paths but in case of LEACH, the node can transmit the data based on the TDMA schedule.

A single hop communication is obvious in such a scenario. As well as the majority of the time delay happens due to the recovery of the lost path. In the energy point of view, the ROL methodology outperforms than LEACH. This is due to the robustness and low power consumption. Fig. 3.8(A) and (B) depicts the performance comparison of LEACH and ROL in the context of data delivery ratio versus node density and end-to-end delay versus node density respectively.

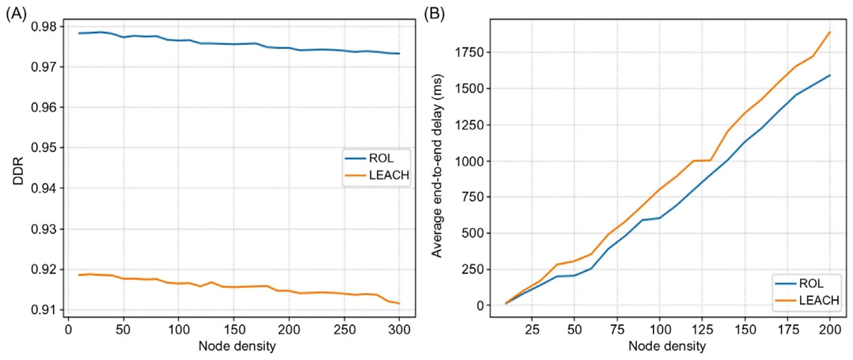


Figure 3.8 Performance comparison of route optimization and the load balancing protocol and low energy adaptive clustering algorithms: (A) data delivery ratio versus node density and (B) average end-to-end delay versus node density.

3.4 Sensor networks application scenario

There are numerous sensor network application can be observed in the current age. In the field of precision agriculture, there is wide variety of the sensor network applications. Another crucial application that is highly tangible is the establishment of the underwater sensor networks using acoustic sensors. Further, the most emerging application that has been observed nowadays is the unmanned aircraft systems (UASs)-based sensor networks often called aerial sensor networks. In the field of environmental monitoring, such networks are popularly used. In this section, our objective is to illustrate the diversified applications of the sensor networks.

3.4.1 UAS-based sensor network

UASs are the best choice for the creation of a commercial sensor network application because of its low maintenance and establishment cost. The drones like multicopter are most significant in this context due to its efficient maneuverability and hovering phenomenon. Multiple UAV in this context makes swarms and sometimes controlled by the base station. It is being observed that the aerial network is having some different characteristics than a traditional WSN. This is not only due to the link phenomenon but also due to the ease of use of such kind of network that is basically a mission-specific. This characteristic imposes some constrains in the network density, delay tolerance, communication, mobility design, and topology changes. The primary ambition of such network design involves the design of the movable UAV nodes. In recent days, the multi-rotors are gaining primary attention in this context. Effectively a multirotor is a UAV that comprises of multiple numbers of a rotating arm, which controls the dynamics of the UAV. Typically the controls are in x, y, and z directions and known as roll, pitch and yaw. There are various sensors that are involved with the movement of the node are in the form of inertial measurement unit (IMU) and other sensors like ultrasonic, air pressure, airflow, magnetic compass, global positioning system (GPS), and camera module.

The concept of multi-UAV system, on the other hand, is more game-changing when compared to a single UAV system because of cooperative service of UAV, where multiple numbers of UAVs search for a specific target and share the updated information. The primary objective of such a system is to achieve some sort of distributed processing.

In the second case, a multi-UAV sensor network can be viewed as an ad hoc network model (often called flying ad hoc network). The flying ad-hoc networks (FANET) is basically an extended concept of mobile ad-hoc networks (MANET) where the node faced a frequent topology change [13,14]. The research issue in this domain encompasses to design and develop a simulation model for the performance analysis. Redefining the routing and mobility model for the multi-UAV environment is another big concern. The concept of heterogeneous UAS is another challenging model that illustrates the multi-UAV as well as the multivehicle cooperative control. The communication involves, in this case, is the air to air as well as air to ground. The introduction to the micro aerial vehicle link (MAVlink) communication protocol is most relevant in this context.

3.4.2 MAVlink protocol background

MAVlink is an extremely lightweight messaging protocol. It uses a point-to-point and hybrid-published subscribed design pattern. Datastream is sent as topics. The subprotocol is to be classified as mission protocol and parameter protocol. The point-to-point retransmission phenomenon has been implemented by the protocol with binary telemetry support-source constrained systems and bandwidth-constrained links. The data through the telemetry are sent in a multicast design. The mission protocol is the subprotocol is allowing the ground control station (GCS) to manage the flight plan. The MAVlink packet format has been described in Fig. 3.9.

The packet format shown in the diagram is optimized for the source constrained system. Most of the elements in the packet are is the uint8_t (8-bit unsigned integer). The largest field comprises 16 bit (the checksum). The field protocol magic marker is a special digit sequence that has to be sent to the receiver before the establishment of the communication between UAV and GCS. The second parameter is the payload length that describes the protocol payload length. The third parameter is an incompatibility flag that must be understood by the receiver; otherwise,

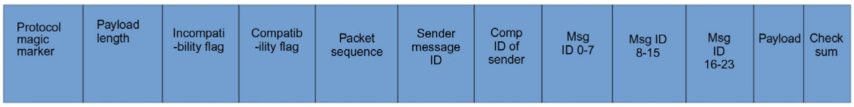


Figure 3.9 Micro aerial vehicle link packet format.

the communication will never be established. The compatibility flag can be ignored if it is not understood. The fifth parameter is the packet sequence number. Next is the sender message identification number followed by the message sender component ID. The 8th, 9th, and 10th parameters show the ID numbers of 0–7th, 8th–15th, and 16th–23rd bit, respectively. The second last parameter is the actual payload. This consists of 255 bit at maximum. Finally, the checksum is added as a last parameter of the packet.

MAVlink is specifically designed for the hybrid network and it generally uses a data transfer with a high data rate from source to the destination (namely the UAS to UGV and GCS). The fundamental insight of the protocol ensures the configuration of an onboard mission or the system parameter changes by using a point-to-point communication methodology. Further, the MAVlink protocol uses two different types of integrity checks. In the first case, the check is on the integrity of the packet during transmission time. This has been done by using CRC-18-CCITT checksum. In the second case, the data definition integrity check has been performed. This is to ensure that two messages of having the same message ID must contain the same information. To get data definition it is also run through the CRC-18-CCITT module and the ultimate result has been seeded to the CRC packet.

In recent ages, the multiple UAV control station architecture has also been considered as a crucial architecture. Several open-source platforms have been used to design such architectures. Ardupilot mega (APM) planner and Droid planner are the softwares that are based on model view controller (MVC) architecture and can be used as a key component of the ground station. The model component is solely responsible for the physical realization of the drone system itself. The status of the unit has to be changed based on the message available through the MAVlink protocol. Additionally, the storage of the status data of the UAV is being performed as a log file (often called tlogs or telemetry logs). In order to realize the multiple drones model component has to be replicated based on the number of the allowable drones to be controlled. Each instance of the model, in this case, has the necessary information about the drones. Message received by the GCS is then redirected to the process for each different drone to represent the activity of the UAV on board. Fig. 3.10 represents the multi-UAV architecture using the MVC paradigm.

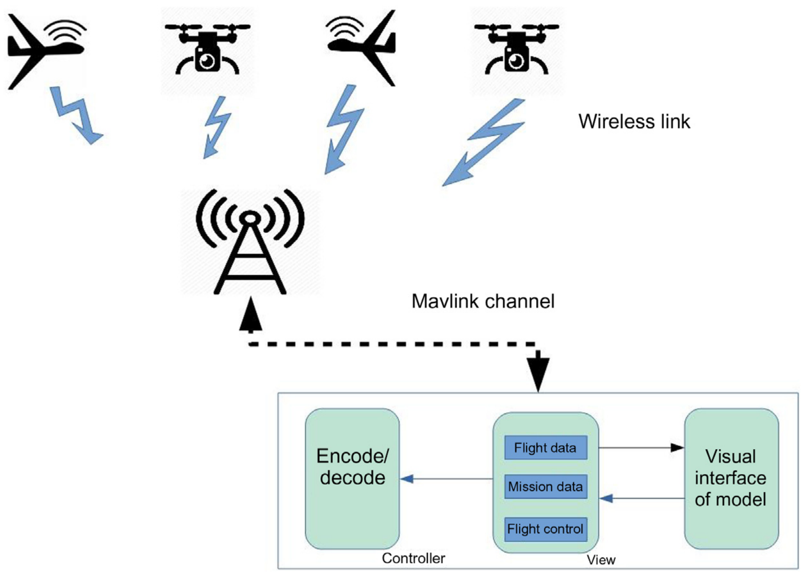


Figure 3.10 Conceptual model of MVC architecture for multiple UAV control station.

The view layer represents user interaction with the real-life data entity. It shows the changes in the model that are received by the user actions that are necessary to control the UAV. The controller tire, on the other hand, manages the incoming messages that are coming through the MAVlink channel and redirect them to the corresponding processes related to the message for the UAV model entity. Also, this component is responsible for sending the control message to the physical UAV through the MAVlink channel. It is the sole responsibility of the control layer to classify the message so that the proper message has been to the proper UAV instance into the visualized model.

The communication infrastructure for such network involves MAVlink protocol in the upper layer. The primary requirement for the network infrastructure should support the transfer of the mission semantics information, precise the area coverage, unicast, multicast, and broadcast support. In order to achieve the goal, the modification of the MAVlink message is required. In standard MAVlink protocol, the inherent security feature does not exist. To add this additional feature the MAVlink messages can be encapsulated into another protocol that offers more security. In order to achieve this, a Zigbee-based mesh network is highly suitable.

3.4.3 A case study on Ardupilot and mission planner for sensing and communication

Ardupilot is an open-source hardware-based flight control system that highly popular experimental and emulation platforms nowadays to deploy Airborne sensor networks and flying ad hoc networks. The platform offers software and the hardware bundle consists of an autopilot unit with an inbuilt IMU. And the ground station software to monitor and analyze the performance of drones that are useful as sensor nodes. The major sensor that involves, in this case, is the barometric pressure sensor, accelerometer, gyroscope, magnetic compass, and GPS receiver. In order to control a sensor node or UAV, in this case, the knowledge of the orientation in the three-dimensional space of the node is an important factor. The orientation of the airborne node determines the attitude of the node on the fly. The attitude heading reference system (AHRS) is the methodology that determines the attitude and the orientation of the system for fly nodes like UAV. In most of the cases, micro UAVs consist of small MEMS sensors that are not so much costly but the drawback is the unwanted noise generated due to the vibration of the aircraft frame. Thereby filtering, calibration, validation, and the correction of the sensor data are highly required in order to achieve the precise behavior of the nodes. The data in this system are mainly measured by the sensor associated with the IMU. AHRS algorithm thus provides the orientation information of the sensors with respect to the body frame where an orientation has been measured by Euler angles. The fundamental objective of the AHRS system is to optimize the sensor information collected from gyroscope, accelerometer, and the magnetometer.

- *AHRS measurement approach*: The classical approach that has followed by AHRS component is to fuse the sensor data in a weighted method. This involves the estimation of the relative orientation from the initial orientation by measuring the angle offered by gyroscope data. This can be expressed by the following equation:

$$Ɗ_{g}(n)^{\sim} = Ɗ_{g}(n-1)^{*} \times \delta_{g}(n) \quad (3.8)$$

Where $Ɗ_{g}(n)^{\sim}$ is the orientation offered by gyroscope at a sample time n. $Ɗ_{g}(n-1)^{*}$ here represents previous fused orientation estimation. The value of $\delta_{g}(n)$ is considered as a small amount of rotation measured in nth sample interval by the gyroscope. The value of $\delta_{g}(n)$ can be defined by the following expression:

$$\delta_g = \left[\left(\frac{|\omega| dt}{2} \right), \sin\left(\frac{|\omega| dt}{2} \right) \frac{\omega(1)}{|\omega|} \ldots, \sin\left(\frac{|\omega| dt}{2} \right) \frac{\omega(n)}{|\omega|} \right] \tag{3.9}$$

Where the value of $n = 3$ for a three-axis gyroscope. And ω is the gyroscope angular rate in radians per second. The value $|\omega|$ signifies the norm of the gyroscope reading and in the nth sample time $\omega(1), \omega(1) \ldots \omega(n)$ represents the angular rate.

- *GPS application*: The global positioning-based navigation is another important aspect that primarily deals with the auto navigation of a drone. In this scenario, the location of the drone computed is based on a number of satellite fix, and the dilution of precision. The dilution of precision can be characterized by geographical and position dilution of precision, which is nothing but the error caused by the relative position of the GPS satellite. From the ground observer perspective, if satellites are spread apart from each other, it results in better visibility of the ground receiver, hence the location is more precise, but if the satellites are close together the dilution of precision (DOP) value becomes high. As a result, the position error will increase. Second, the atmosphere itself also affects the signal strength and the propagation speed. Another crucial error that needs to be computed and rectified is the multipath effects. This causes due to the reflection of the signal and repeatedly receiving the same signal more than one time that generates an ambiguity. Further the GPS receiver can improve its accuracy by associating with a ground-based receiver. As the stationary GPS receiver detects the same satellite as the GPS receiver, the correction is, therefore, to be done based on the previously set precisely surveyed locations. This type of augmented system is highly precise and used for mission-critical applications where accuracy is required in the order of a centimeter.

In the next part, a case study of the attitude control of the UAS has been analyzed based on the performance of the gyroscope, accelerometer, magnetometer, and the GPS. The standard Ardupilot Mega 2.6 has been used with UBlox GPS of 4 Hz refresh rate. The mission has been designed in such a way that the drone navigates by the ground control and in an autonomous mode with a flight time of 23 minutes. The flight path of the UAV is shown in Fig. 3.11.

All flight data can be downloaded from the autopilot itself in the form of telemetry and data flash logs. The log files here are nothing but the MAVLink messages sent by the autopilot to the ground station. The MAVLink message, in this case, has been received by the base with a baud rate of 115,200.

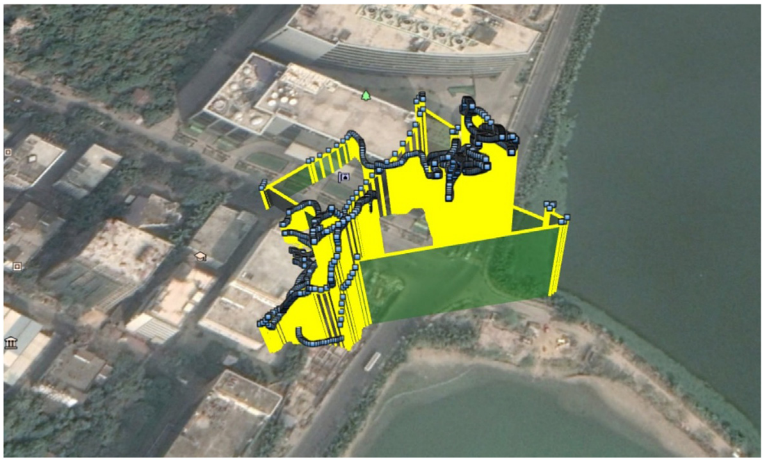


Figure 3.11 Flight path of the UAV with base station control and autonomous mode.

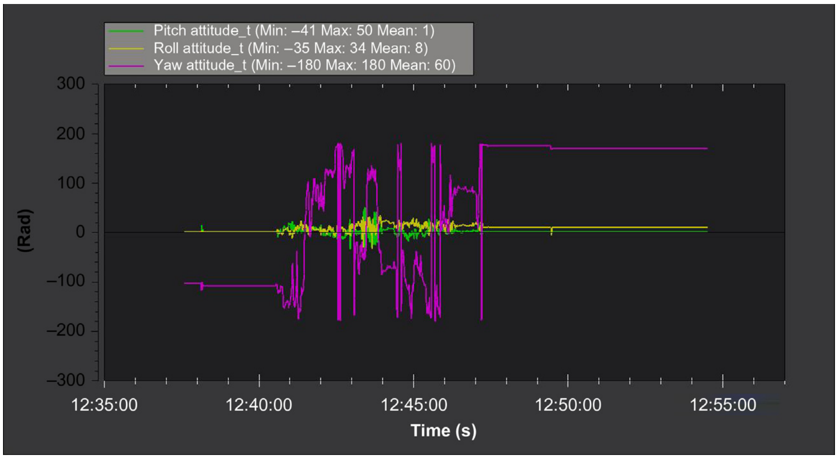


Figure 3.12 Pitch, roll, and yaw attitude of the UAV on the fly.

The following part gives an idea about the sensor log analysis of the APM autopilot IMU system. Fig. 3.12 describes the roll, pitch, and yaw attitude of the UAV within the flight time. From the figure, it is clear that to navigate into a proper flight path a number of yaw is required is very high. The pitch and roll change is moderate that is desirable for automated flight.

The Fig. 3.13(A, B, C) represents the absolute orientation of the UAV in *x*, *y*, and *z* plane. The major component of IMU involves

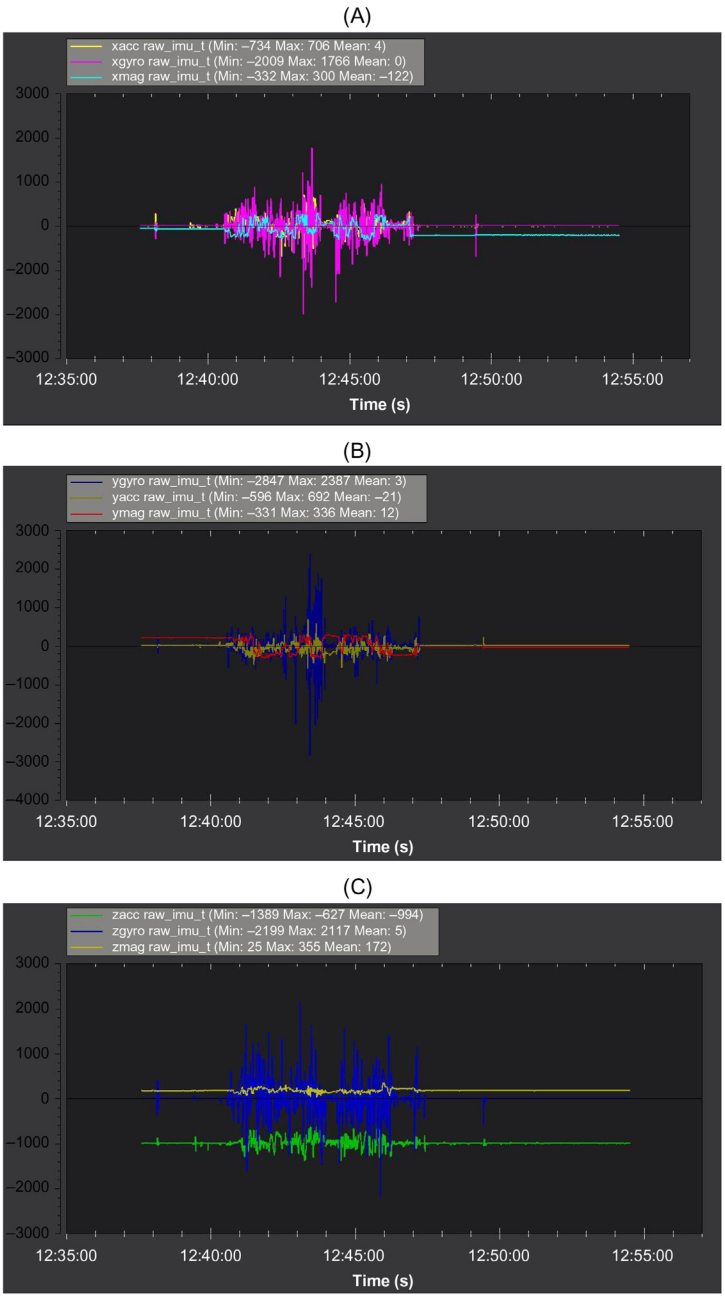


Figure 3.13 Raw inertial measurement unit data for accelerometer, gyroscope, and magnetometer: (A) *x*-axis, (B) *y*-axis, and (C) *z*-axis.

accelerometer, gyroscope, and magnetometer. The figure shows the raw IMU data that shows the *x*, *y*, and *z* positional changes of the UAV under a given interval of time.

The graph clearly shows that the gyroscope data give more noise when compared to magnetometer and accelerometer. The AHRS system should ensure that the sensor data should be filtered and optimized in order to get a more accurate result. Fig. 3.13(B) shows the *y*-axis orientation of the drone by taking the data from the same sensor.

Fig. 3.13(C) illustrates the *z*-axis orientation value of IMU data. In this case, the accelerometer produces more errors in reading in comparison to the value offered by gyroscope and magnetometer.

The change of angular rate (ω) has been illustrated in Fig. 3.14. The AHRS system itself tuned the raw data in such a way that the final output produces a smooth and precised controlled movement of the flight controller. From the graph, an extreme change in the *z*-axis has been observed due to the making of the rapid yaw by the autopilot to navigate precisely through the predefined path.

GPS takes a vital role to navigate the airborne nodes in a precise way. In order to achieve the proper and precise path, the visibility of the GPS is a primary need. As a general rule, as the number of visible satellites increases, the accuracy of the location information also increases. Fig. 3.15 shows the satellite visibility of raw data within the flight time. From the visibility data, it is clear that the UAV gets on an average of eight satellites constantly. At the initial stages, the visibility is nearly 4 but as time

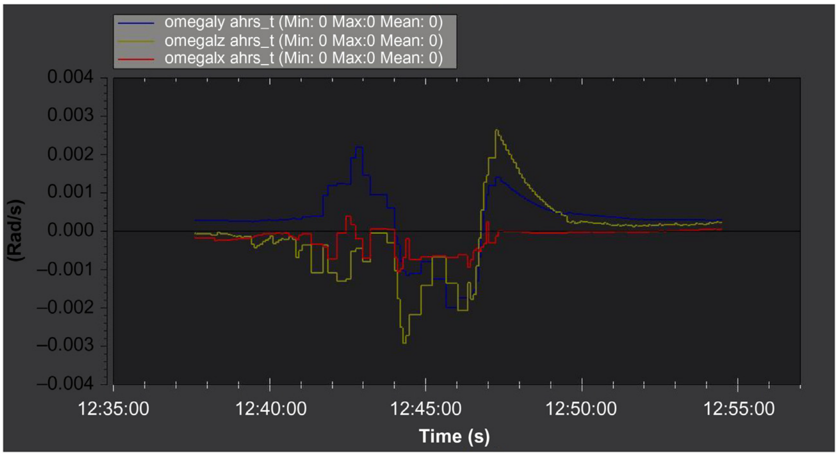


Figure 3.14 ω-value change for *x*-, *y*-, and *z*-axis.

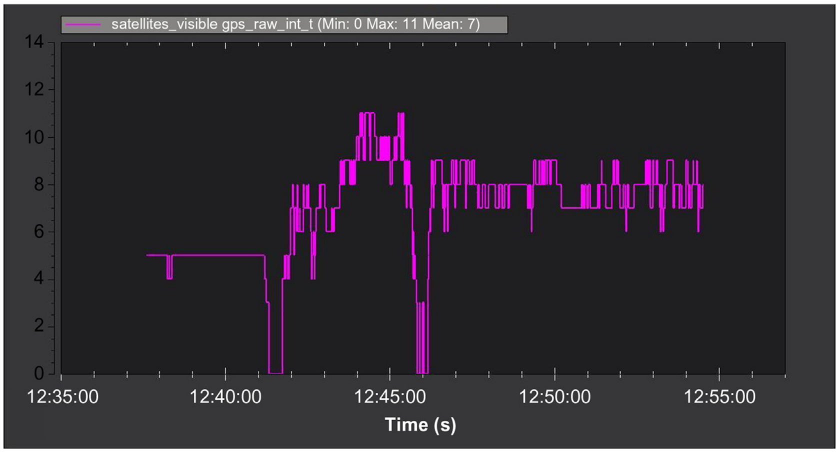


Figure 3.15 Satellite visibility raw data under observed flight time.

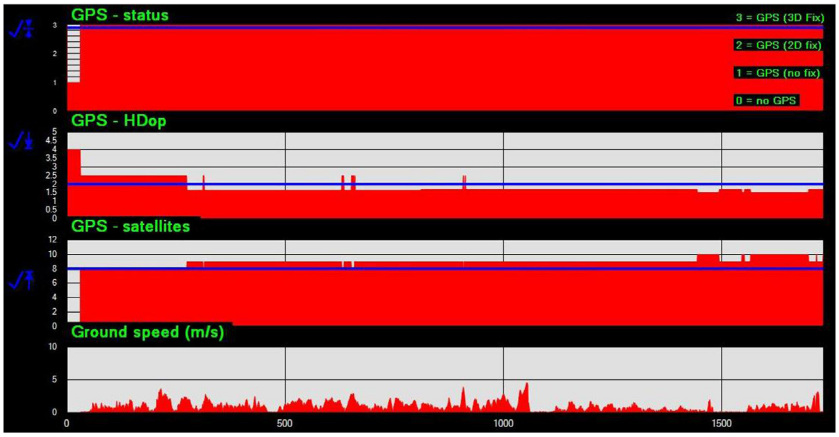


Figure 3.16 Global positioning system status report for the UAV node.

increases the number of visible satellites also increases. Fig. 3.16, on the other hand, gives essences of the various parameters related to GPS. The parameter includes GPS receiver status, horizontal dilution op precision (HDOP) values, number of visible satellites, and the ground speed of the node [15]. From the graph, it is easily understandable that the total number of satellite visible is almost 6–8 initially and it increases as time passes. Among them three satellites are in 3D fix mode, two satellites in 2D fix mode and satellite has no fix. The HDOP value on the other handset from 4 and it gradually decreases down to 1.5. This is because as the receiver is visible to the satellite, it tries to maintain a link between those satellites that are far from each other. As a result, the position becomes more precise and the decreasing Hdop value reflects the same fact. Finally, the ground speed represents the speed of the node that is not beyond the maximum of 5 m/s.

3.4.4 Renewable energy integration using WSN

In a smart grid system, wireless data transfer is a crucial approach. The system involves several components, such as theft monitoring, protection, and operation between interrelated units. There are numerous communication infrastructures that can be considered in this context based on bandwidth, network coverage, and cost of installation.

The most common suitable communication, in this case, is the IEEE802.11 Wi-Fi for its point-to-point and multipoint communication capability, robustness, and scalability. WiMAX (IEEE802.16 standard) is another popular type of communication infrastructure with 2.3, 2.5, 3.5, and 5.8 GHz frequency band. For the fixed infrastructure communication 3.5 and 5.8 GHz is to be allocated. On the other hand the 2.3 and 2.5 GHz is primarily used for the mobile node communication. The 3G and 4G networks are also usable nowadays to create a green WSN. 824—894 MHz spectrum is mainly used with a 60—240 kbps transmission rate. One of the major advantages of using a cellular network is that there is no additional cost involved in the setting up of the network infrastructure.

Zigbee/IEEE802.15.4 protocol is an extremely vital methodology that integrates the distributed renewable generation (DRG) and the smart grid. Zigbee creates a different topology structure for the convenience of the application. The fundamental structure that has been implemented includes a star, peer to peer, and a cluster-based network. Star topology not enough suitable for such application infrastructure for inefficient power management. Cluster tree topology is supposed to be most suitable to create a Zigbee-based personal area network. The network, in this case, is split into two major components, namely the cluster head and the children node. Nodes in the Zigbee consist of transducers and the radio transceiver units. In most cases, the nodes have some minimum processing capability with minimum processing power. Reduced instruction set computer (RISC) architecture is preferred in such a case. The node also has an autonomous power unit to power up the node. A hybrid structure of the mesh and the star is the optimal configuration of such piconet configuration.

3.4.5 Vehicular sensor network for agriculture and food management

Agriculture and food management is another significant domain where the utility of sensor network is of paramount use. The sensor provides risk management. For example, the sensor network provides precise microclimate information, frost damage, and many more. The scenario is likely a very extensive paddy field, where the nodes in the network are significantly far away from each other. The mobile nodes can be deployed in the form of an aerial vehicle or ground vehicle as a mule. Some of the conditions in such case are involves. First, the vehicle must cover at least a 10-km distance with uninterrupted communication. Second, it must detect the diseases of crops and grab high-quality images. Also, the live video transmission feature must be incorporated. If diseases are detected the vehicle must able to carry the substances and must launch them in the proper position.

In order to achieve the requirement, an absolutely unmanned aerial vehicle is the best choice. UAV in this scenario is primarily a fixed-wing drone that consists of a propeller, autopilot unit, and the power source. It consists of a radio-transmitter-receiver unit so that the control is sent to the UAV from the ground station. The data, on the other hand, will be sent through a separate channel, which consist of a Wi-Fi or 5.8-GHz transmitter-receiver module. The autopilot units that are mainly used are Ardupilot Mega or Pixhawk PX4. The Ardupilot mega is basically an Arduino-based autopilot that operates in 16-MHz clock frequency. The inbuilt MPU6000 is present over there, and it accommodates eight input and output channels. The Pixhawk autopilot, on the other hand, has ARM Cortex M4 core. It is a 32-bit processor with 168-MHz clock frequency. It also comprises 2 MB flash memory and 256 KB RAM.

The food quality management is another crucial approach that can be handled using a sensor network (Fig. 3.17). The concept of an intelligent container is a major implementation that can be used for logistics transportation for perishable goods, such as foods, vegetables, and fruits. In this case, the temperature of the goods logged in persistent storage, and if a certain temperature is superseded the condition is classified as okay. The object of an intelligent container is not only to measure the temperature but also to analyze the result and generate a decision locally. The system is designed in such a way that the ambient temperature and the humidity have been sent to the local base station from time to time. Moreover, an artificial intelligence mechanism [16] has also been incorporated to analyze and measure the condition of the goods by using statistical data. Finally,

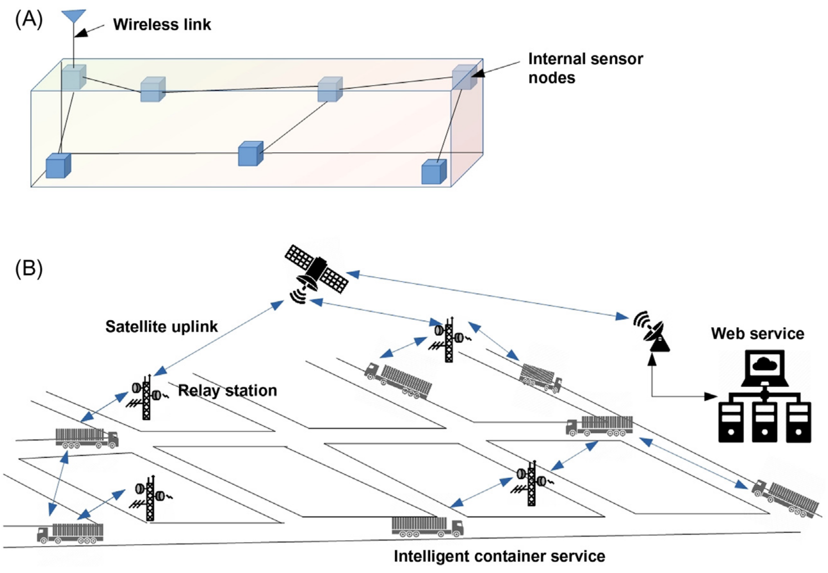


Figure 3.17 (A) Internal organization of intelligent container and (B) abstract architecture of vehicular network-based food management system.

the temperature and the humidity get controlled by using proper actuation methodology to make the food fresh enough.

Vehicular DTN-based IoT mechanism [17] is also considered to be a state-of-the art mechanism to implement an opportunistic network approach that can be used for food supply chain monitoring. The concept exploits the phenomena of the vehicular ad hoc network in the form of opportunistic network. Each vehicle, in this case, acts as a sensor node and performs storing and forwarding of the data in an opportunistic mode. The data are finally sent to the base station for further analysis. The main challenge in this scenario is to minimize the time of the delivery of the status information of the food. In order to do that the message delivery, end-to-end delay, buffer usage, and the QoS are of major concern.

3.5 Summary

In this chapter, we have discussed various aspects of the WSNs and their applications. The main emphasis has been done on the different sensors and their working principle, sensor network topologies, and mobility issues. Various routing methodologies in wireless networks have been addressed. We are also focused on the various real-life application scenario and the different research issues that may produce future direction of the broader research aspect in this field.

References

[1] N. Dey, A.S. Ashour, F. Shi, S.J. Fong, R.S. Sherratt, Developing residential wireless sensor networks for ECG healthcare monitoring, IEEE Trans. Consum. Electron. 63 (4) (2017) 442−449.
[2] S. Fong, J. Li, W. Song, Y. Tian, R.K. Wong, N. Dey, Predicting unusual energy consumption events from smart home sensor network by data stream mining with misclassified recall, J. Amb. Intel. Hum. Comput. (2018) 1−25.
[3] N. Dey, A.S. Ashour, C. Bhatt, Internet of things driven connected healthcare, Internet of Things and Big Data Technologies for Next Generation Healthcare, Springer, Cham, 2017, pp. 3−12.
[4] A. Schaller, K. Mueller, Motorola's experiences in designing the Internet of things, Int. J. Amb. Comput. Intel. (IJACI) 1 (1) (2009) 75−85.
[5] S. Kumar, N. Nagarajan, Integrated network topological control and key management for securing wireless sensor networks, Int. J. Amb. Comput. Intel. (IJACI) 5 (4) (2013) 12−24.
[6] W. Wei, H. Song, W. Li, P. Shen, A. Vasilakos, Gradient-driven parking navigation using a continuous information potential field based on wireless sensor network, Inf. Sci. 408 (2017) 100−114.
[7] N. Dey, A.S. Ashour, F. Shi, R.S. Sherratt, Wireless capsule gastrointestinal endoscopy: Direction-of-arrival estimation based localization survey, IEEE Rev. Biomed. Eng. 10 (2017) 2−11.
[8] N. Dey, A. Mukherjee, Embedded Systems and Robotics with Open Source Tools., CRC Press, 2018.
[9] S. Samanta, A. Mukherjee, A.S. Ashour, N. Dey, J.M.R.S. Tavares, W.B.A. Karâa, et al., Log transform based optimal image enhancement using firefly algorithm for autonomous mini unmanned aerial vehicle: An application of aerial photography, Int. J. Image Graph. 18 (04) (2018) 1850019.
[10] M. Faheem, V.C. Gungor, Energy efficient and QoS-aware routing protocol for wireless sensor network-based smart grid applications in the context of industry 4.0, Appl. Soft Comput. 68 (2018) 910−922.
[11] Z. Zheng, A.K. Sangaiah, T. Wang, Adaptive communication protocols in flying ad hoc network, IEEE Commun. Mag. 56 (1) (2018) 136−142.
[12] S.K. Singh, P. Kumar, J.P. Singh, A survey on successors of LEACH protocol, IEEE Access 5 (2017) 4298−4328.
[13] A. Mukherjee, N. Dey, R. Kumar, B.K. Panigrahi, A.E. Hassanien, J.M.R.S. Tavares, Delay tolerant network assisted flying ad-hoc network scenario: modeling and analytical perspective, Wirel. Netw. 25 (5) (2019) 1−21.
[14] A. Mukherjee, V. Keshary, K. Pandya, N. Dey, S.C. Satapathy, Flying ad hoc networks: A comprehensive survey, Information and Decision Sciences, Springer, Singapore, 2018, pp. 569−580.
[15] A. Mukherjee, N. Dey, N. Kausar, A.S. Ashour, R. Taiar, A.E. Hassanien, A disaster management specific mobility model for flying ad-hoc network, Emergency and Disaster Management: Concepts, Methodologies, Tools, and Applications, IGI Global, 2019, pp. 279−311.

[16] A. Mukherjee, N. Dey, Smart Computing with Open Source Platforms, 1st Edition, CRC Press, 2019.
[17] S. Saha, A. Sheldekar, A. Mukherjee, S. Nandi, Post disaster management using delay tolerant network, Recent Trends in Wireless and Mobile Networks, Springer, Berlin, Heidelberg, 2011, pp. 170−184.

CHAPTER FOUR

Overview of sensor cloud

Contents

4.1 Basics of cloud computing

Data are generated in every instance in and around our environment. The facts extracted from the raw data are called information which is then processed, stored, and analyzed each and every second. Operations are performed on the information locally, that is, in one's own system or globally in a remote server. When we are to define the word cloud computing the only word over which a stress is imposed is the word "sharing." Cloud computing paradigm [1] can be defined as:

"Cloud Computing is a processing paradigm where a large number of devices are connected over the network over which data and

resources are shared to provide – storage, databases, data analytics, and intelligence over the internet in a dynamically scalable manner."

Cloud computing provides a model for enabling ubiquitous, convenient, and on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort.

4.1.1 Key features of cloud computing

- *Cost*: cloud computing eliminates the cost of setting up of data processing center. Data and resources are shared over the network and processed online.
- *Speed*: a large number of computing resources is provisioned in minutes as the devices are connected over the network.
- *Maintenance*: all the maintenance is provided by the service provider, and the user using the cloud platform need not worry about this aspect.
- *Multitenancy*: As the resources are sharable among the various devices, multiple users can use the same pool of resources.
- *Global scale*: Cloud computing platform is vastly scalable, that is, for a particular application if more computing power or storage is required it can be easily provisioned in minutes.
- *Reliability*: Multiple resources of the same data are kept in the cloud platform across multiple servers, hence it is inherently distributed. So if one of the servers fails then the data and resources can be accessed from another server.
- *Security*: The cloud service providers offer a broad set of policies and technologies to power up the security in the network to help protect data.

4.1.2 Cloud computing architecture

The cloud computing architecture [2,3] is divided into numerous layers. The whole model is created in an abstracted way where a client-side program such as a browser or any mobile application is used for the very purpose of using the system. The working of the whole cloud computing network is hidden from the users. Fig. 4.1 shows a simple cloud architecture.

The users access the cloud using a client-side application program. The data are stored into the server in an efficient distributed manner

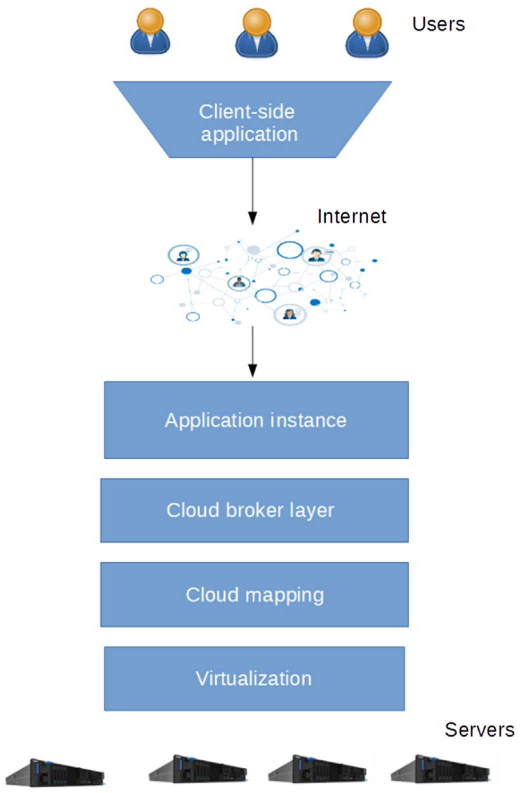


Figure 4.1 Abstraction in the cloud platform.

where the client's resources are shared among each other. The users are not concerned about the complexity of the system; their main motive lies in the service provided by the cloud service providers. The cloud service providers [4] are the important organizations who provide the service. Some of the top cloud service providers are Amazon Web Service, Microsoft Azure, and Google Cloud. The cloud broker layer provides a middle layer between the one using the cloud and the one providing the service, that is, the users and service providers. The job classification of a cloud broker is as follows:

- A cloud broker can be a software application that can be used for the distribution of workload over the different cloud service provider.
- A cloud broker can provide a service over the services provided by the cloud provider such as encryption and duplication of data.
- A cloud broker may partner with one or more cloud service and can select cloud service on behalf of the user.

Virtualization is the way of creating a virtual version of the resources, that is, the servers and services. Virtualization process ensures that the group of servers works as a single pool of computing resources, for example, a firmware name Hypervisor acts as a virtualization manager. The duty of the virtualization manager is to simulate the underlying hardware, that is, the servers where raw data is stored by using the software.

4.2 Types of clouds

The cloud computing paradigm is classified into various types to serve various purposes according to the need of the users. There are many types of models and services that have evolved, and accordingly it is manipulated. The manipulation means the way in which the cloud is deployed, that is, servers and its content. Following are the three ways in which cloud computing is deployed:

4.2.1 Public cloud

Public cloud [5] is the most popular cloud computing (Fig. 4.2) as it is what most people refer to as the cloud. It is a cloud service offered by the third-party provider over the Internet. Anyone be a person or an organization who wants to set up the cloud service can get the service from these service providers that are open to all. One does not need to purchase or maintain the cloud infrastructure. All the services are provided by the cloud service providers; hence, the resource and services are shared by all the customers.

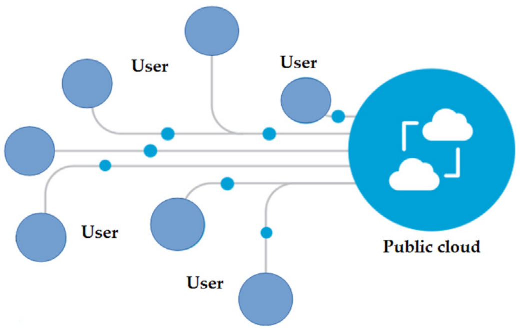


Figure 4.2 Public cloud.

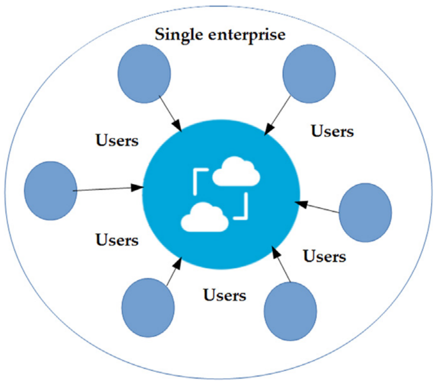


Figure 4.3 Private cloud.

4.2.2 Private cloud

Private cloud [6] is also called an internal cloud service. The word internal is used as it is private to a selected set of users of a single enterprise. Its main focus is security as the name suggests. The data are shared and computation is done locally within the enterprise. The private cloud paradigm can be externally hosted or can reside within the organization's data center. In externally hosted private clouds a cloud service provider facilitates and maintains the private cloud for an organization. Fig. 4.3 depicts a private cloud paradigm. If the cloud service is internally hosted then it is called on-premise private cloud; the model provides a more standardized process and protection.

4.2.3 Hybrid cloud

Hybrid cloud [7] combines the power of on-premise private cloud [6] and public cloud into one. The overall sharing of data and computational load is shared among the private and public cloud. The services provided are hybrid services that provide greater flexibility and scalability. The hybrid cloud can provide on-demand resources and computation power from its public infrastructure to its private infrastructure. Fig. 4.4 shows a hybrid cloud network.

4.3 Cloud computing models

The sole objective of the cloud computing paradigm is to provide ubiquitously and on-demand network access that can be rapidly provisioned and released. Cloud computing is divided into models with essential characteristics.

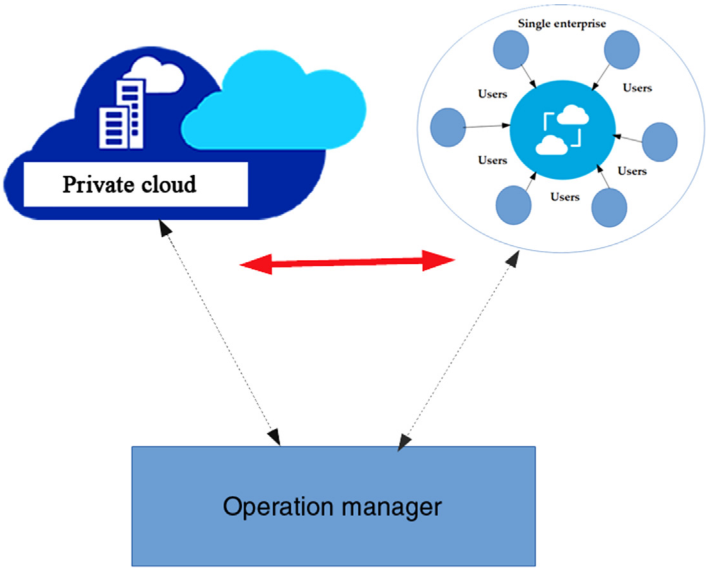


Figure 4.4 Hybrid cloud.

4.3.1 Software as a service

One of the most popular on-demand software computing platform SaaS [8,9] stands for software as a service (SaaS). It is closely related to application service provider. Instead of companies running the software in one of their own servers, they provide their service through various companies who own and operate the cloud computing platforms and provide the environment to run the SaaS. In the SaaS model, the provider gives customers network-based access to a single copy of an application that the provider created specifically for SaaS distribution. The companies pay a monthly fee to those service providers to continue the service. More specifically, they pay for amount of time the software runs on their system. The application's source code is the same for all customers and when new features or functionalities are added, they are given to all customers.

An overview of the platform can be understood from Fig. 4.5(A) and (B) where it can be observed that in Fig. 4.5(A) the clients are connected to a server. The web server provides services such as emails, application program, and file resources. If one wants to use the services it has to be accessed from the server. In Fig. 4.5(B), it can be observed that the server is connected to the high-speed Internet. The web server has offshored the files, application program on the Internet to a SaaS-based platform. As all

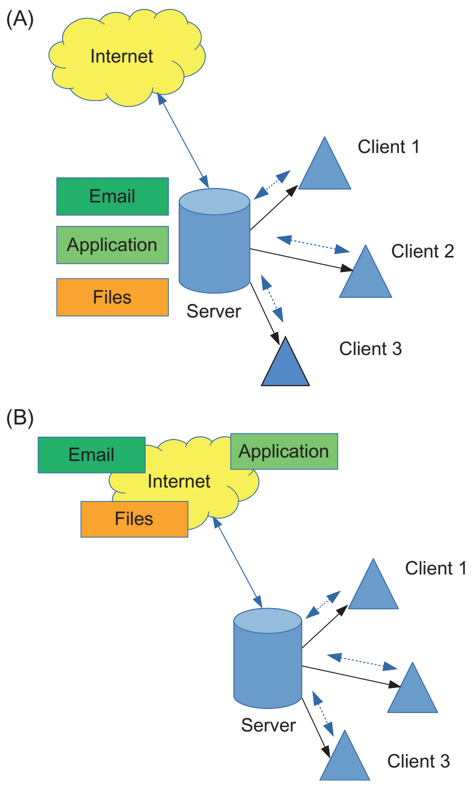


Figure 4.5 (A) Resource hosting without the cloud. (B) Cloud-based resource hosting.

the clients have direct access to the Internet they can directly access the application and files without accessing the company's server.

4.3.2 Platform as a service

Platform as a service (PaaS) [10] is a class of cloud computing services that allows its users to develop, run, and manage applications without thinking much about the underlying infrastructure. The focus of the user solely lies in building applications rather than the complexities of the internal hardware. There are various PaaS services offered by various computing providers, such as Amazon, Google, and Azure. PaaS can be deployed on top of IaaS, or, independently on any virtual machine or containers.

One of the open-source cloud platforms [11] is cloud foundry that provides a choice of clouds, developer frameworks, and application services. The deployment procedure is quite easy, and it can be deployed on IaaS, like Amazon Web Service or OpenStack.

4.3.3 Infrastructure as a service

Infrastructure as a service (IaaS) is one of the main cloud computing paradigms along with SaaS and PaaS. IaaS [12,13] provides an abstracted usage of the resources to the users by providing resource virtualization. IaaS provides on-demand physical and virtual computing resources, storage, network, firewall, load balancers, and so on. To provide virtual computing resources, IaaS uses some form of the hypervisor, such as Xen, KVM, VMware ESX/ESXi, and Hyper-V. Hypervisors are procedures for hardware virtualization that allows various guest operating systems to run on a single multiple host operating system. A popular hypervisor used in IaaS platform ESX creates a logical pool of system resources so that the same physical resources can be shared among many virtual machines. ESX server is an operating system that functions like a hypervisor and runs directly on the system hardware. ESX server inserts a virtualization layer between the virtual machine and hardware, turning the system hardware into a pool of logical computing resources that ESX server can dynamically allocate to any operating system or application. The guest operating systems running in virtual machines communicate with the virtual resources as if communicating with physical resources. Fig. 4.6 shows an ESX server running virtual machines. Each of the virtual machines is running an operating system and application independent of the other virtual machine. The ESX server supports Linux, Windows, BSD, Netwear, and Solaris guest operating system.

In the IaaS infrastructure, the top-level users are not concerned with the underlying architecture, hence, the whole paradigm of using the cloud and its services is in the application level. Some of the popular IaaS providers are Amazon EC2, Rackspace, Google Cloud Computing, and Go Grid. IaaS is the backbone of all cloud services, providing the compute resources.

After getting the compute resources, they provide other services on top of that. An IaaS platforms can be depicted as follows: let us say that you want to have a set of ten Linux systems with 4 GB RAM each, and two Windows systems with 8 GB each to deploy your software. You can go to any of the IaaS providers and request these systems. Generally, an IaaS provider creates the respective virtual machines in the background, puts them in the same internal network, and shares the credentials with

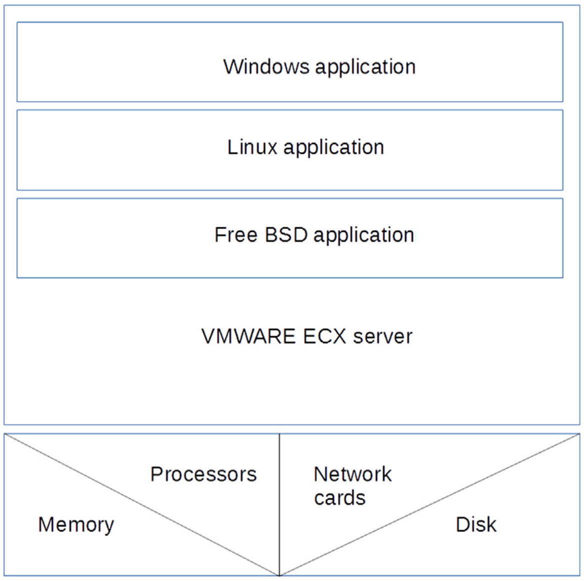


Figure 4.6 Infrastructure as a service.

you, thus allowing you to access them. Other than VMs, some IaaS providers offer bare-metal machines for provisioning.

Some of the popular IaaS frameworks are as follows:

1. *OpenStack*: The OpenStack framework infrastructure [14] is provided with public as well as a private cloud. Openstack is an open-source, scalable, and flexible infrastructure. OpenStack helps users in deploying virtual machines and other instances that do different tasks for maintaining a cloud environment. It makes horizontal scaling easy, which means that the tasks that benefit from running concurrently can easily serve more or fewer users on the fly by just spinning up more instances. For example, a mobile app that needs to communicate with a server that is remotely kept might be able to divide the work of communicating with each user across many different instances, all communicating with one another but scaling quickly and easily as the application gains more users.
2. *Nimbus*: Nimbus [15] provides IaaS and it is divided into two parts; one is nimbus infrastructure and the other is nimbus platform. Nimbus infrastructure provides IaaS implementation that is compatible with Amazon EC2/S3. Again Nimbus platform provides additional tools for simplifying the infrastructure management. It is mostly considered for scientific cloud computing solutions.

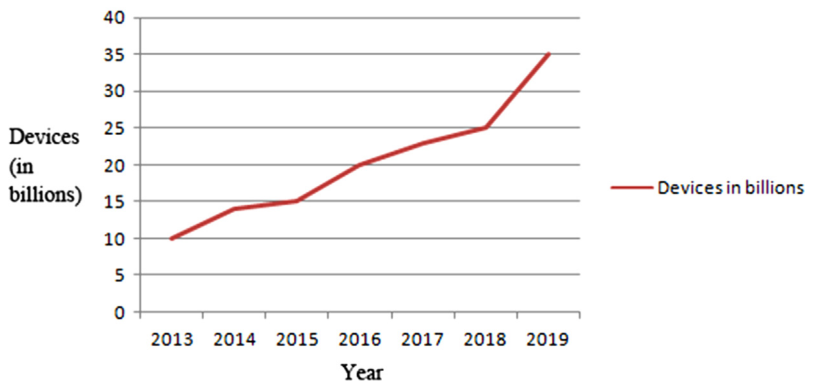


Figure 4.7 Increase in number of devices per year.

4.4 Sensor cloud platform

It is a challenging affair to share information in sensor communication over the Internet and it makes a lot of sense to connect sensors over the Internet. The number of physical sensors is increasing day by day as the technology is advancing. Fig. 4.7 shows a graph where a number of devices (in billions) is plotted against year. We can see that the number of sensors is increasing in billions every year, and the sensors not only necessarily mean devices that are only used for dedicated sensing purpose but also for the electronics devices that are used every day, for example, a smartphone. In order to manage all the sensors and effectively share data among each other, an infrastructure is needed where the user can share different kinds of physical sensors, and various new services can be provided if collaboration takes place between wireless sensor network (WSN) service providers.

The WSN service providers deploy their sensors for a dedicated sensing purpose. The customer gets the benefit of the service from these individual service providers for their application. But such types of infrastructure have a limitation in scalability, performance, and data sharing. Modifying the original definition of SaaS to sensing as a service we lead our way to a whole new infrastructure where the limitation in performance barrier is broken by means of collaboration between these service providers (Fig. 4.8).

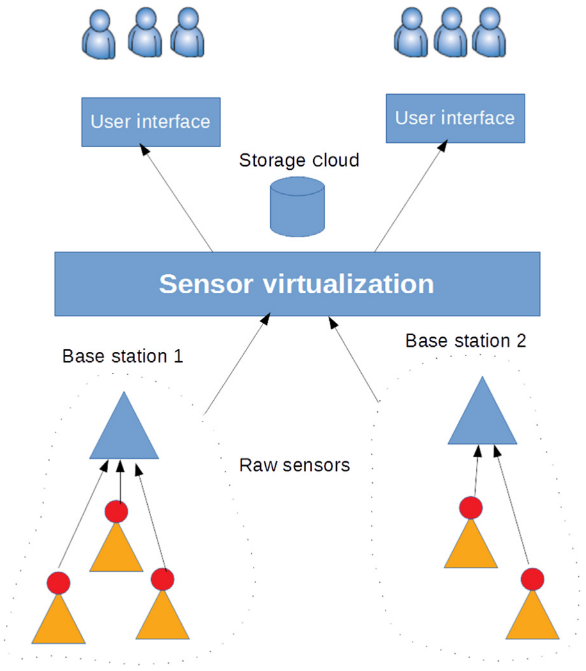


Figure 4.8 Cloud platform overview.

Sensor cloud [16–19] is a very new concept; the whole idea revolves around the word "sharing." Companies can maximize their usage of their existing hardware infrastructure by applying cloud computing in the sensor environment. Sensor cloud can be defined as:

> *"**A sensor cloud is unique sensor data storage and management platform composed of virtual sensors built on top physical wireless sensors which provide powerful cloud computing platform and can be very easily provisioned and de-provisioned according to the user's application need**."*

Sharing the sensors among the WSN service providers paves the way toward building cloud infrastructure in the sensing environment. A set of virtual sensors (VSs) work on top of the physical sensors over which the user interface provides an abstraction to the users using the system through their respective application user interface. The users can use and control its view of the WSN with a set of functions for a variety of parameters such as data sampling frequencies and security. As the WSN is shared the overall price for data collection reduces.

4.5 Sensor cloud architecture

In a sensor cloud infrastructure, various WSN service providers register to offer sensors-as-a service to the users. The word cloud along with sensor implies that the resources are sharable. The service providers can register or de-register their physical sensors from the cloud infrastructure with ease. The whole sensor cloud architecture is abstracted from the user; hence they are not concerned about the intricate complexities of the cloud. The registered sensors are virtualized into VSs. The user requesting sensors are allotted physical sensors from the template of VSs. Fig. 4.9 shows the architecture of typical sensor cloud architecture.

In sensor cloud infrastructure, sensor virtualization is very important. The physical sensors deployed are from various service providers, and

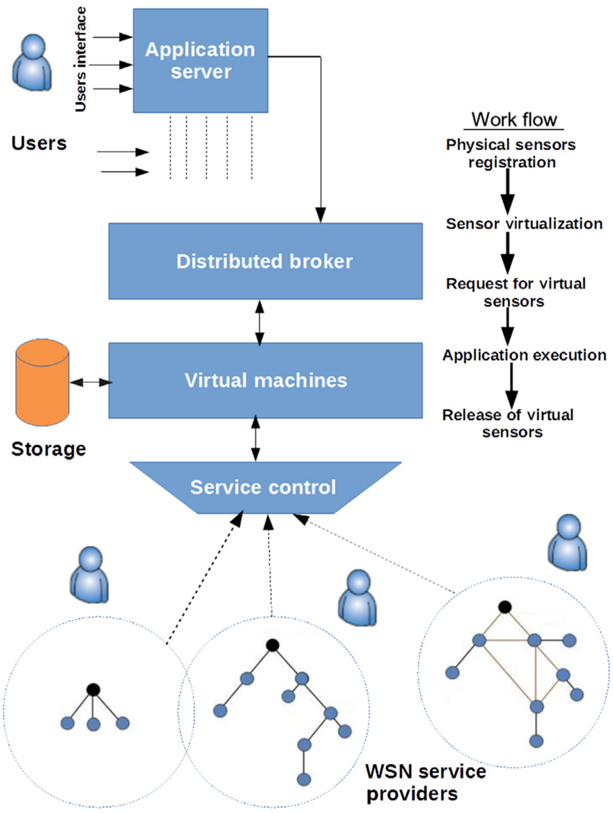


Figure 4.9 Sensor cloud architecture.

virtualization provides separation of architecture from the services provided by the sensor cloud infrastructure.

The *application server* is a user-centric server whose sole duty is to provide abstraction and user interface of the sensor cloud model and project it as one single PaaS to the users using the system. The application layer/server facilitates and manages the communication between the user and the physical sensors. *Virtualization* provides an interface to the users to use the physical sensors in an effective abstracted manner without knowing the location or the specification of the physical sensors. VSs are mapped with the physical sensors to obtain data and information from the underlying physical sensors. It might be the case that multiple physical sensors are mapped into one VS or each individual physical sensor can be mapped to a single VS. A group of VSs can be created from one or more physical sensors. Users can specify the frequency at which the VSs will collect data; they can also activate and deactivate any sensors according to their need. Fig. 4.10 shows a diagram to depict the physical sensors and how they are virtualized. The set of VSs are given to users to operate.

It is difficult to run the different application on the WSN without the use of a virtual server or virtual machines as used in cloud computing platforms. Each virtual machine operates on the VSs to perform the respective

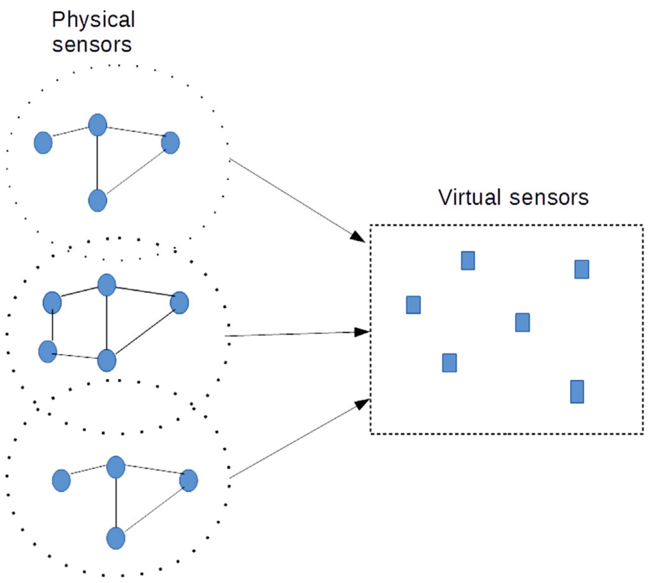


Figure 4.10 Virtualization of sensors.

operation. The VSs hold the information about the physical sensor it is mapped to and the information about the user to whom it is currently allocated to.

A *distributed broker* is a third-party broker whom partners with the WSN service provider to act as middle man in between the users and the service provider. The physical sensors are deployed at various geographical locations and all the information about the sensors are hidden from the users using the cloud platform. The *service control* is a phase where control is passed onto the VS servers that switch the WSN associated with the selected region for data gathering.

4.6 Sensor cloud workflow

The sensor cloud environment consists of four important servers that uphold the working of the cloud platform. The four servers are:

- *Portal server*: This server is an application-level server. The portal server provides services and interaction of users with the underlying structure in the sensor cloud. It provides users to register and de-register sensors at ease. The portal server also passes information to the other servers in the network as required.
- *Grant server*: Grant server contains the necessary protocols that are triggered to manage the workflow. The Grant server provisions the VSs from the template of VSs. It reserves the various resources required by the user's application.
- *Monitor server*: The information about the various VSs is collected by the monitor server. The data collected can be viewed by the users through the application portal they are using, and it can be a simple web browser. The received data about the VSs are stored into a database.
- *Virtual server*: Virtual servers contain the VS group that is provisioned on a virtual machine by the Grant server. The VS groups can be controlled by the end-users to whom they are allocated.

4.6.1 Workflow procedure

1. An end-user who wants to use the cloud platform registers and logs into the sensor cloud platform through the portal server.

2. The user specifies its requirement and selects the template of VSs provided by the platform through the portal server.
3. The portal server sends the request to the Grant server.
4. The Grant server reserves the resources needed by the user application and allocates the virtual machine by reserving the IT resources needed by the application first. The template for the VS group is selected from the repository. Each of the virtual machines has a monitoring agent that monitors the VSs and gathers essential information. Fig. 4.11 shows the workflow of the sensor cloud in a pictorial manner.

The physical sensors are deployed by various governmental as well as private organizations to provide sensing as service to the various other users/company. For example, sensors are deployed by multiple service providers in a forest area for various monitoring purposes.

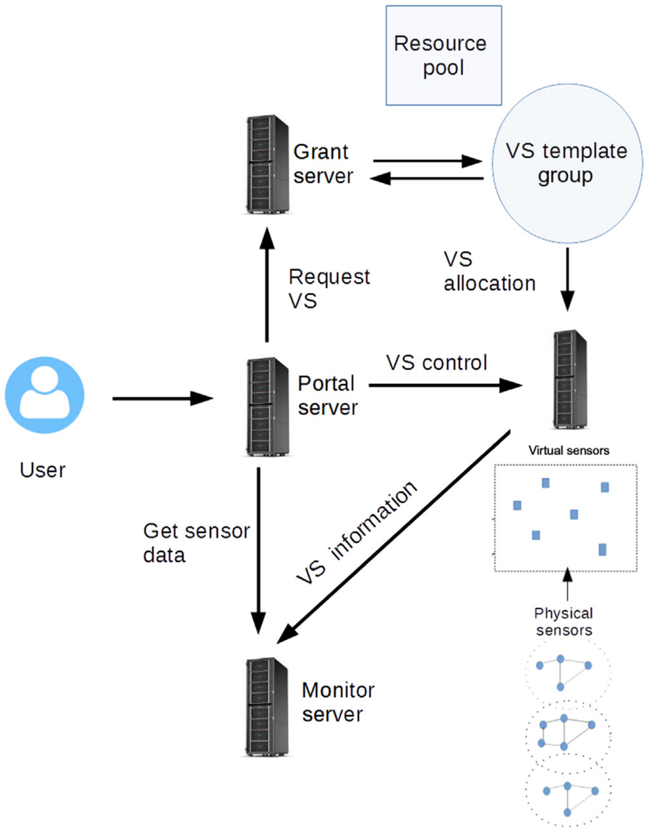


Figure 4.11 Sensor cloud workflow.

Applications can range from surveillance, animals monitoring, forest fire, and so on. If some organizations wants to monitor a forest fire, the sensors although they are deployed by various service providers, have different specifications, the whole platform of sensor cloud works collectively to abstract the internal complicacies and shows the whole system as one system. The request is given to the sensor cloud platform through the portal server. The template of a VS group is provisioned for the monitoring of forest fire in an area. The resources are reserved by the Grant/Provision Server. If more sensors are needed for the sensing purpose then more sensors are provisioned automatically. When the application procedure ends the VSs are deleted and physical sensors which were reserved are released.

Hang et al. [19] have proposed a design and implementation of the sensor cloud platform where the VSs are dynamically deployed to monitor and control the physical sensors. The design simplifies the process of generating multiuser environment. An overview of the layers involved in a sensor cloud design is depicted in Fig. 4.12. The topmost layer is the application layer that is the same as the transmission control protocol/internet protocol (TCP/IP) model, where the whole objective lies in abstracting the internal complexity of the design from the user and provides a user interface to use the underlying structure. Session and service layer organizes the services for provided various features such as user authentication, service provisioning, load balancing, and user authentication. The virtualization layer is the layer responsible for the mapping of the physical sensors with VSs. The VSs hold

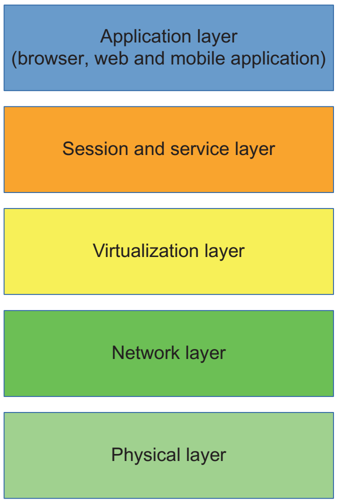


Figure 4.12 Layers in sensor cloud.

the metadata about the physical sensors such as ID, type, interface, and location. The job of the network layer is routing, that is relaying the information from one hop to another toward its base station and finally to the respective user. The physical layer is the raw hardware, that is, the sensors connected together with the ability of sensing, storage, and processing.

4.6.2 Difficulties in sensor cloud platform

With the increasing number of devices, the number of sensors is increasing exponentially. The infrastructure for sensing is unnecessarily replicated for the same type or different types of application purpose. Sensor cloud is the best solution to utilize the sensors and the sensing service in a systematic manner. In the sensor cloud platform, the sensors are treated as a sharable resource that allows rapid provisioning and scalability. Sensor cloud brings about a plethora of advantages such that the users can control the sensors by means of VSs freely, the end-user can monitor the status of VSs, the users can use the system freely without knowing about the complexities of the system, but there are certain difficulties that a sensor cloud platform faces. A number of issues arise when detecting the physical sensors, last but not least when selecting suitable diagnostic systems and encountering difficulties in physical node placement in a place for a particular application. Although sensor virtualization removes the constraint imposed on scaling the network when only dealing with a physical WSN, there are still some difficulties that are faced by sensor cloud platforms.

Few of the difficulties in sensor cloud platforms are:

- The specification and status of the sensors are not known by the user using the system.
- The allocation of sensors is dynamic so the system development complexity is very high.
- A monitor server is required for each application that monitors the status of the sensors.
- A proper software and hardware resource allocation has to be done for each user request in the sensor cloud platform.

4.7 Application scenario

Sensor cloud platform is one of the most sophisticated platforms for providing an abstracted platform of WSN services in a collaborative manner. The application range of sensor cloud is very wide, and some of them are:

i. *Weather services*: Weather monitoring and forecasting are one of the most crucial day-to-day activities. Various individuals or organization can use the services of the sensor cloud platform to gather weather-related data and perform prediction on them. The end-user has a weather service request to the platform to construct a VS group that is indirectly used to guide and control the physical sensors. Let us consider an example of an occurring catastrophic event. The weather service provided will use the VSs, analyze multiple sensor data timely, and give rapid control and monitoring effectively. When the catastrophic event goes away the sensors are released. The whole internal hardware and its working are abstracted. The WSNs work in a collaborative platform and act as a single unit.

ii. *Ubiquitous healthcare*: In order to improve the patient's condition both physical and mental, ubiquitous healthcare is one of the most emerging fields in this generation. Sensors are used to gather various types of data such as heart rate and oxygen saturation. The sensors are used for supporting medical cares by the hospital services. Various emergency situations are also monitored by the sensors. During any emergency event, the VSs can be deployed to monitor every patient and take immediate actions accordingly.

iii. *Monitoring transportation*: The transport monitoring system includes basic management systems, such as traffic signal control, navigation, automatic number plate recognition, toll collection, emergency vehicle notification, and dynamic traffic light. A practical example of transportation monitoring is done in peer-to-peer transportation companies such as UBER. The data pertaining to each ride are gathered from every individual vehicle to determine the blocked road, shortest route, and so on.

4.8 Summary

- Cloud computing provides an environment for ubiquitous, shared, and on-demand processing and storage.
- Cloud service providers are an important organization that provides on-demand cloud service.

- There are three types of cloud, namely public, private, and hybrid.
- Sensor cloud is a collection of VSs that are used to manipulate communicate, and perform processing using the various physical sensors deployed by multiple WSN service providers
- The sensor cloud platform consists of four major servers, namely portal, grant, virtual, and monitor that uphold the platform.
- Sensor cloud platform reduces the unnecessary replication of the same type of resources.

References

[1] B. Hayes, Cloud computing, Commun. ACM 51 (7) (2008) 9–11.

[2] Y. Jadeja, K. Modi, Cloud computing-concepts, architecture and challenges, 2012 International Conference on Computing, Electronics and Electrical Technologies (ICCEET), IEEE, 2012, pp. 877–880.

[3] P. Rajeswari, S. Viswanadha Raju, A.S. Ashour, N. Dey, Multi-fingerprint unimodel-based biometric authentication supporting cloud computing, Intelligent Techniques in Signal Processing for Multimedia Security, Springer, Cham, 2017, pp. 469–485.

[4] L. Wu, S.K. Garg, R. Buyya, SLA-based resource allocation for software as a service provider (SaaS) in cloud computing environments, Proceedings of the 2011 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, IEEE Computer Society, 2011, pp. 195–204.

[5] P. Hofmann, D. Woods, Cloud computing: the limits of public clouds for business applications, IEEE Internet Comput. 14 (6) (2010) 90–93.

[6] A.D. JoSEP, R. KAtz, A. KonWinSKi, L.E.E. Gunho, D. PAttERSon, A. RABKin, A view of cloud computing, Commun. ACM 53 (4) (2010).

[7] T. Dillon, C. Wu, E. Chang, Cloud computing: issues and challenges, 2010 24th IEEE International Conference on Advanced Information Networking and Applications, Ieee, 2010, pp. 27–33.

[8] A. Dubey, D. Wagle, Delivering software as a service, McKinsey Q. 6 (2007) (2007) 2007.

[9] S. Goswami, P. Roy, N. Dey, S. Chakraborty, Wireless body area networks combined with mobile cloud computing in healthcare: a survey, *Classification and Clustering in Biomedical Signal Processing*, IGI Global, 2016, pp. 388–402.

[10] G. Lawton, Developing software online with platform-as-a-service technology, Computer 41 (6) (2008) 13–15.

[11] A. Mukherjee, N. Dey, Smart Computing with Open Source Platforms, vol 1, CRC Press, 2019, ISBN: 9781351120340, (1).

[12] S. Bhardwaj, L. Jain, S. Jain, Cloud computing: a study of infrastructure as a service (IAAS), Int. J. Eng. Inf. Technol. 2 (1) (2010) 60–63.

[13] A. Mondal, M. Sarkar, S. Sanyal, Chattapadhyay, K.C. Mondal, Performance analysis of structured, un-structured, and cloud storage systems, Int. J. Amb. Comput. Intel. (IJACI) 10 (1) (2019) 1–29.

[14] O. Sefraoui, M. Aissaoui, M. Eleuldj, OpenStack: toward an open-source solution for cloud computing, Int. J. Comput. Appl. 55 (3) (2012) 38–42.

[15] J. Peng, X. Zhang, Z. Lei, B. Zhang, W. Zhang, Q. Li, Comparison of several cloud computing platforms, 2009 Second International Symposium on Information Science and Engineering, IEEE, 2009, pp. 23–27.

[16] M. Yuriyama, T. Kushida, Sensor-cloud infrastructure-physical sensor management with virtualized sensors on cloud computing, NBiS 10 (2010) 1–8.
[17] A. Alamri, W.S. Ansari, M.M. Hassan, M.S. Hossain, A. Alelaiwi, M.A. Hossain, A survey on sensor-cloud: architecture, applications, and approaches, Int. J. Distrib. Sens. Netw. 9 (2) (2013) 917923.
[18] S. Madria, V. Kumar, R. Dalvi, Sensor cloud: a cloud of virtual sensors, IEEE Softw. 31 (2) (2013) 70–77.
[19] L. Hang, W. Jin, H. Yoon, Y. Hong, D. Kim, Design and implementation of a sensor-cloud platform for physical sensor management on CoT environments, Electronics 7 (8) (2018) 140.

CHAPTER FIVE

Sensor data accumulation methodologies

Contents

The fundamental task of a sensor network is to deploy the sensor node that grabs the physical data in the form of electrical signals. The data are collected, and some means of aggregation are performed on them. Based on different types of data and their collection mechanisms, several classifications can be made. The data are collected by the sensors that may be available in a different way. Sometimes, data are available in a continuous manner, and sometimes they are available within a certain interval. Depending on the data delivery, various prediction analyses can also be performed on them. One of the fundamental challenges of data manipulation is the data aggregation techniques that we apply on data. The fundamental objective of aggregation is to apply rule based on the sensor data to get a knowledge base, which is further redirected to sink by choosing a path that has a minimal cost. Several algorithmic approaches such as centralized, low latency, and tiny are very popular in this context.

5.1 Sensor data classification

Information is everywhere; classifying the types of data with respect to various applications is of prime importance. The sensors are used to detect physical environmental events; they are developed according to the need of the application and type of data that is sensed or collected. Some of the examples of sensors with respect to their applications are as follows:

- Photosensors detecting the presence of visible light.
- Charged coupled device where an image's pixel is converted into electrical charge.
- Gravitational acceleration is detected using the accelerometer.
- Smart grid providing real-time data for grid conditions.

Sensor data are one of the integral components in the Internet of things (IoTs) [1] environment. The generalized classification of the sensed data is as follows:

- Periodic data
- Nonperiodic data
- One-shot data

Periodic data are periodic information that is sensed by sensors, and the transmission of that information takes place periodically that is after every certain amount of time. Periodic sensors play a major part in a real-time system. Example of periodic data monitoring is monitoring forest fire using sensors deployed in the jungle and traffic data monitoring. Nonperiodic information is simple data sensing using sensors where there is no time bound. One-shot data are such data that need to collect and transmitted immediately from all the sensors that are deployed in the sensor network.

While designing a sensor network, the types of data that we obtain from it are a major concern. Primarily, the data that have been produced by sensor nodes are in the form of integer or decimal values. Sometimes, data may come in the form of audio, text, or image/video. Most of the data, in this case, are unlabeled. An unlabeled data have no meaning; therefore one of the major tasks in this context is to label the data.

- *Labeled data*: These data are basically the groups of samples. The samples are tagged with one or more labels. The task labeling involves grouping of the set of unlabeled data and assigning some meaningful tags over them. For example, if we consider a picture, the labeling may be done based on the feature or certain object present in that picture. For raw temperature data, it might be the upper and the low bound of the temperature that has been grabbed by the device. For audio data, it may be a particular sound or sounds that are used to uniquely identify the data samples. The advantage of labeling of the data involves asking a human to make a decision about the given set of unlabeled data. Making data labeled may leads to huge cost with respect to the representation of the unlabeled data. After obtaining labeled data, now any machine learning model can be applied over it so that it can generate another set of predicted labeled data.
- *Time-series data*: These are a series of data elements or points that are listed or plotted with respect to the order of time. Mostly such data have been captured based on the specific snapshot of time with an equally distributed time space. Thus it produces a discrete time-series interval. Time-series data may be in the form of values or some images or videos. The example such as power grid load data for each day (Fig. 5.1) is a time-series data. Sample image of the position of the sunspot with respect to time is also be considered as time-series data. Time-series data are widely used in various analyses and mainly plotted in the form of line plotting. There are wider domains where we can

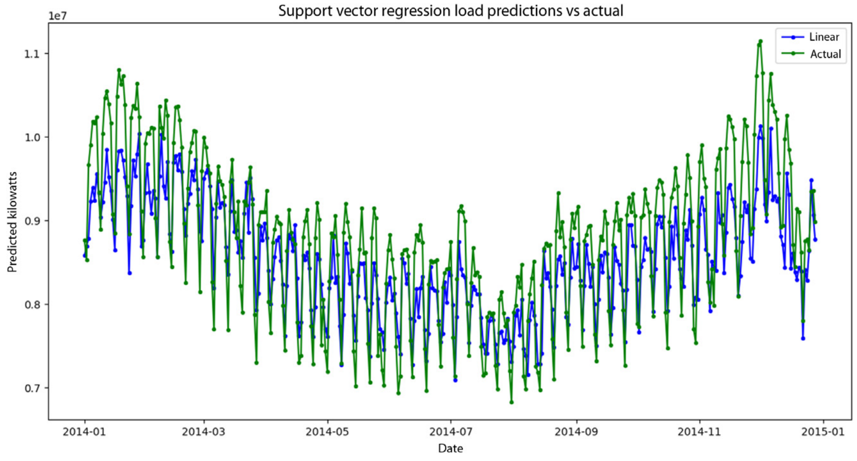


Figure 5.1 Time-series data.

use the time-series data such as signal processing, stock trading, forecasting, critical analysis, and many more.

There is a wide variety of analysis paradigm that can be addressed in the context of the time-series data. One of the salient methodologies is the time-series analysis. The technique comprises the analysis of time-series data to obtain meaningful statistical information. Another key terminology is the time-series forecasting where some model may predict the result based on a sequence of previously observed time-series data value. One of the major advantages of such data is that it has inherent characteristics of temporal ordering. One of the major facts is that time-series analysis can be applied to continuous, numeric, real-valued data.

Another major aspect that has to be taken care of, while grabbing the sensor data, is the deviation of the data. As no sensor mimics the ideal transfer function, so some sort of deviation may be observed that limits the sensor accuracy. First of all, sensitivity of the sensor may differ based on the value specified, which causes the sensitivity error. Value of constant and offset bias may cause errors, which is due to y-intercept of the linear transfer function. The range of output signal is also limited so the maximum scale of input and output may not be achieved. Noise is another issue that varies with respect to time. In case of digital sensors, the output is digitized and hence it converts the actual pulse in the form of an approximate discrete-time pulse. Consequently, a quantization error is obvious in such a case. A hysteresis error is another problem that may be experienced by sensors that cause the output depending on the previous input values. In the case of digital monitoring of the sensor signal, the sampling frequency causes a dynamic error. If the input and the noise change the frequency of the signal in a multiple of the sample rate, the aliasing effect is obvious.

5.2 Data transmission methodology

The fundamental phenomena of the sensor network are the scarcity of the node. Each independent node is responsible for collecting data and routes them to the base station often called a sink as well as to the end users. This is to be done through a multihop transmission using an infrastructure network architecture that comprises Internet and satellite communication system.

The communication protocol architecture in this scenario can be thought of as a three-dimensional protocol stack that consists of the standard layer of transmission control protocol/internet protocol (TCP/IP) suite along with the capability of task, mobility, and power management. This ensures the efficient communication energy and power management, strategic routing awareness, and promotion of cooperative effort of sensor nodes.

The application layer in this scenario runs several application software platforms for data handling and management. The transport layer controls the data flow that the application layer required. The network layer, on the other hand, performs the routing of the data packets. Several routing procedures can be addressed in this context.

5.2.1 Unicast

The unicast transmission [2] mechanism primarily works on one-to-one delivery of the message from one sensor node to another (Fig. 5.2). The expected capability of the node of the sensor network is having a limitation. Mostly it comprises a small battery, a processor of certain MHz clock frequency with the maximum memory capacity in the order of megabyte. The media access control (MAC) protocols may support time-division multiple access (TDMA) [3] mechanism, and the basic and lightweight version of operating can be supported by the node.

In this scenario, the direct diffusion transmission is suitable. In this transmission, data packets are named based on the attribute-value pair.

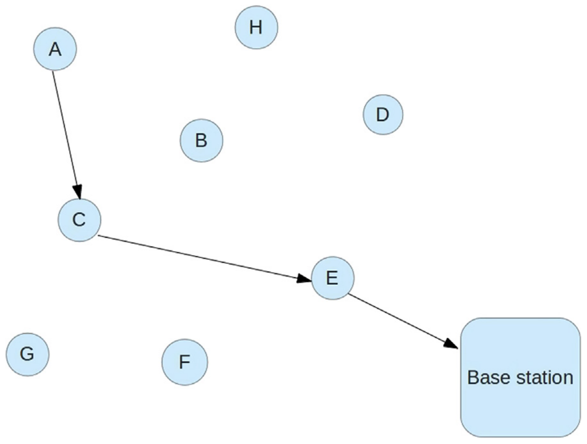


Figure 5.2 Unicast transmission.

The sensing task has been disseminated throughout the network. This process produces a draw event, and the event starts flowing along multiple paths. The network chooses the smallest path(s) among them. The task that performs is to be given in the form of attribute value, and a specified region has to be chosen for the intercommunication based on the task allocated. The named task description consists of interest as well. This interest has to be injected into any arbitrary node as known as a sink. The packet transmission solely depends on the priority of interest in this case.

In case of flooding-based strategy, there are several other subcategories that are reported, such as direct contact-based routing where source and sink are mobile and come into contact directly. It does not have the universal knowledge of the network. In tree-based approach, the nodes relay the message with the number of copy required, which is quite similar with diffusion computation mechanism. In the exchange-based models, the principle focus is on information exchange. The mechanism is highly relaid on summary vectors that keep tracks on all packet ids. Delivery predictability is also a key phenomenon for packet delivery in this case. As flooding leads to a reduction of network throughput, we can also think for some restricted flooding scenario where a restricted copy of message gets transferred from source to sink. The challenge, in that case, is the message delivery from source to sink because of the sparse nature of the network.

5.2.2 Multicasting

Multicast model (Fig. 5.3) [4] is fundamentally based on the packet delivery to a group of host sink simultaneously. Direct diffusion routing is mostly focused on those cases where there is a limitation of geographical routing. As multicast routing focuses on delivering the data packet to a group of node, it also emphasizes certain metrics such as bandwidth and the latency of the data packet in the network. One of the advantages of the geographical multicast can be observed by exploiting location-based multicast using global positioning system (GPS) that can be effectively used by the stationary or moving sensor nodes. As the location information is obtained by the sensor nodes, the tree-based approach can be utilized by implementing a multicast tree. In this case, multicast root node can locally construct the complete physical tree; second, it constructs the complete overlay tree by choosing limited group members. Here each overlay member hops should use a common unicast routing mechanism.

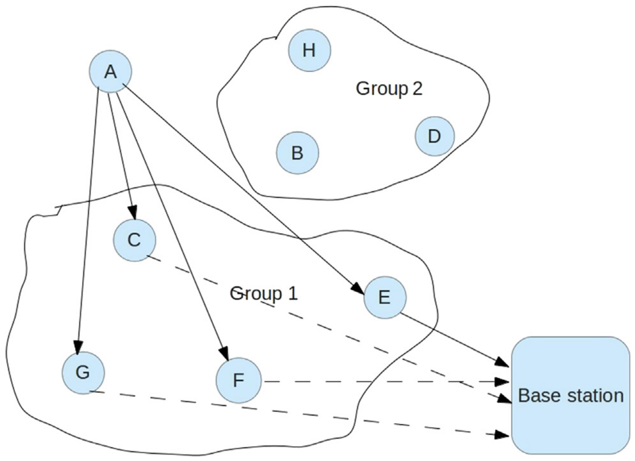


Figure 5.3 Multicast mechanism for message transmission.

Another mechanism involves with multicasting is hierarchical multicast routing. A hierarchical geographical multicast routing is a protocol that improves forwarding efficiency in comparison to the location-based multicasting. Each node that forwards data, in this case, selects a subset of its neighbor node and relay message toward the destination. This selection can be done using a greedy heuristic method that optimizes cost over progress ratio. The cost, in this case, is proportional to the number of selected neighbors. The ultimate goal of the cost function here is to minimize the amount of bandwidth consumption.

5.2.3 Broadcasting

Broadcasting involves data to be forwarded to all possible nodes in the network. Consequently, the requirement of the bandwidth will increase drastically. A broadcast scenario for sensor network is depicted in Fig. 5.4. Several data broadcasting mechanisms have been addressed in this context. In the broadcasting technique, the sensor nodes transmit the data toward multiple nodes or multiple sinks at a time. A chain-based protocol is very much useful in this context. Chain-based protocol produces a transmission chain of all nodes in order to save energy. Mainly the linear chain concept is used in this context. It connects all the nodes in a network, and the data transmit from one end to another end. Every node in the chaining process broadcasts the data to its neighbor node with an information header. As the neighbor node receives, it also adds a header that increases

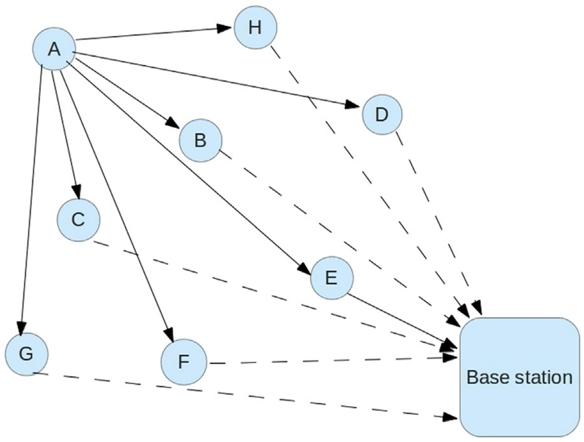


Figure 5.4 Broadcast scenario in sensor network.

the total size of the data packet. This process continues until the packet reaches the destination. This process again repeats while delivering the message from the destination to the source again.

5.3 Convergecast: inverse of broadcasting

Broadcasting as we have studied is the concurrent transmission of the same message to all the nodes in the same network. Now the objective of deploying sensor nodes is to collect data and perform processing on the data. Collecting and relaying the sensed data from multiple deployed sensor toward a sink node are called convergecasting [5]; hence it can also be defined as the inverse of broadcasting technique. In convergecasting, data flow from many different source nodes to a single sink node or the base station.

5.3.1 Important issues

Wireless sensor networks (WSNs) are composed of a large number of low-power, small-size, and low-cost sensor nodes. A sensor node is an electronic device with the capability of detecting physical conditions, computation, and communication. Those sensor nodes can be scattered to perform a variety of applications such as wildlife monitoring, habitat monitoring, and fire surveillance.

In many applications, it is crucial to provide a guarantee on the delivery time as well as to increase the rate of such data collection. For instance, in safety and mission-critical applications where sensor nodes are deployed to detect oil/gas leak or structural damage, the actuators and controllers need to receive data from all the sensors within a specific deadline, failure of which might lead to unpredictable and catastrophic events. This falls under the category of one-shot data collection. In contrast, applications such as permafrost monitoring require periodic and fast data delivery over long periods, which falls under the category of continuous data collection.

5.4 Data aggregation

5.4.1 Fundamentals of aggregation

Aggregation can be defined as *agglomeration of data and information toward a single sink node with the idea of reducing energy and increasing the lifetime of the sensor network.*

In simple term "aggregation" means "collection" that in turn means gathering of data toward a sink. A WSN is formed by deploying various sensors that communicate with each other to perform their assigned task. Data aggregation is employed with the objective of relaying information that is collected by various sensors toward a single base station. With the advancement of technology, sensor networks are composed of small and effective sensing devices equipped with wireless radio transceivers for the monitoring of various environmental events. The key objective is to make certain decisions with respect to the data monitored. For example, sensors are deployed by a governmental organization in the forest area to monitor for any terrorist intrusion, and the sensing is performed by the help of camera or infrared sensors. The sensors channel the information toward a single sink that is its respective base station.

Data aggregation algorithms are developed for the very purpose of data gathering in an energy-efficient manner. In the era of the data age, information is everything and information are gathered ubiquitously. For example, if one goes into a shopping mall, one will observe that the door opens automatically; hence there is a sensor whose sole duty is to sense people approaching and make the decision of opening the gate. In real-time traffic

Table 5.1 Classification of wireless sensor networks.

Increasing the network capacity	Reducing energy consumption	Data accuracy
On performing data aggregation, less number of packets are sent toward the sink; by sending less number of packets, the network capacity is saved	The redundant data packet transmission gets decreased, which in turn reduces the energy consumption for the sensor network	Accuracy in wireless sensor means the distinction between the received and sent data
Hence the network also becomes scalable, that is, it can be increased		Aggregation procedure reduces the number of retransmission of redundant data from several nodes by aggregating the received data at each level into a digest

communications and high data rate sensors, the real-time constraints lie in meeting the deadlines and minimizing the communication delay. Hence to meet these constraints, data aggregation has to be achieved.

5.4.2 Need for aggregation

Convergecasting is the fundamental data gathering operation from a collection of sensors toward a sink node. In performing convergecasting, the main motive lies in performing such data gathering in an energy-efficient manner. Data aggregation summarizes data collection in an efficient manner. The need for aggregation in WSN can be classified as (depicted in Table 5.1):

1. Increasing network capacity.
2. Reducing energy consumption.
3. Reducing retransmission and packet loss.

5.5 Choice of MAC layer

The MAC layer is one of the two sublayers that make up the data link layer. The main job of the MAC layer is to move the data from the

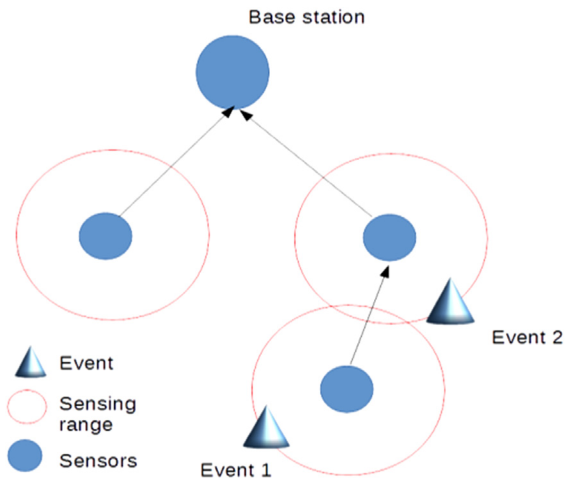


Figure 5.5 Collision-free sensor network.

network interface card of a network to a shared channel. The MAC layer along with data link control is responsible for the complete physical addressing of the data link layer. The communication in WSN occurs from various data sources for the aggregation information, which is relayed toward a single sink node. With the advent in technology, MAC protocols have come a long way, long before we used the additive links on-line hawaii area (ALOHA) protocols then slotted ALOHA and so on. Then carrier sense multiple access protocols came to light. As the battery in the wireless sensor cannot be replaced, efficient power saving communication is very important.

As information is relayed from multiple sensors, there is a chance that collision might occur. As traffic is event triggered, the traffic of two nodes closing together will be dependent in nature, and as traffic is many to one, there is high probability of collision. Fig. 5.5 shows how collision might occur during data aggregation.

In Fig. 5.6, we can see that event 1 and event 2 are sensed by all the underlying leaf sensor nodes. The same information against event 1 is relayed to the sensors above it. It might be the case that both try to send the data in the very same time frame, due to which collision occurs. Collision avoidance is one of the basic tasks in any MAC protocols. MAC protocol developed in a way so that they adopt a contention-based scheme. Redundant data transmission makes contention-based protocols less efficient in energy than TDMA protocols. So it has to be avoided.

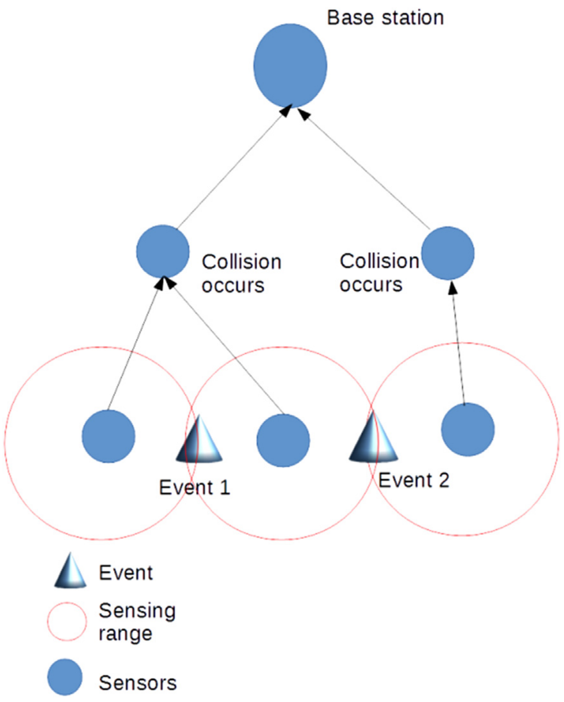


Figure 5.6 Sensor deployment.

Collision leads to:

- Data packet loss.
- Decreases reliability and increase the retransmission.
- Energy consumption increases and latency also increases.

MAC protocol such as contention-free or contention-based MAC protocols can be used to avoid a collision. In contention-based MAC protocols, the acquired channel mostly uses control packets. Mostly used for transmission in the independent traffic pattern. Contention-free MAC is used for the dependent network. It can be used if the network traffic is static or changes rarely. Some of the popular MAC protocols are as follows:

1. MACA, MACAW—contention based
2. TDMA, code division multiple access mechanism, frequency-division multiple access (FDMA)—contention free

5.6 Energy analysis

WSNs have spatially or densely distributed sensors to monitor environmental conditions such as temperature, pressure, sound, and so on.

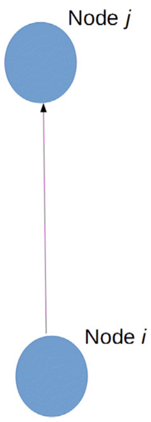


Figure 5.7 Wireless node configurations.

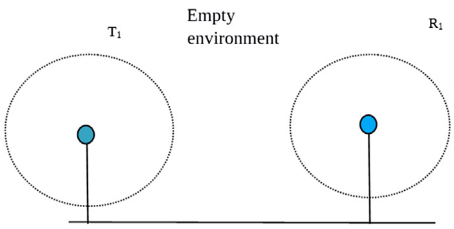


Figure 5.8 Transmitter and receiver in empty environment.

The information has to be collected, and the information has to relay from one node to another. In Fig. 5.7, a wireless node configuration is shown, and a successful reception of packets from a node *i* to node *j* depends on the ratio between the received signal strength at *j* and the cumulative interference caused by other transmitting nodes and the ambient noise level.

Packet received successfully at *j* if signal-to-noise ratio *(SINR)* $>\beta$ *(threshold)*.

In the free space propagation model, the receiver and transmitter are located in an empty environment. The environment does not have any obstacle or any reflecting surface. Hence the total influence of the earth's surface on the propagating signal is practically nil. The free space propagation model is an ideal case. Fig. 5.8 shows a transmitter T_1 and a receiver R_1 in an empty environment, which is a classic example of the free space propagation model.

For transmission and receiving to be successful, the sensor nodes should be within the range of each other. In Figs. 5.8 and 5.9, we have

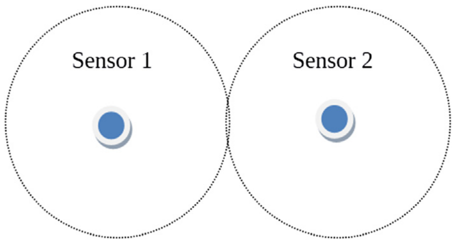


Figure 5.9 Sensors with omnidirectional antenna.

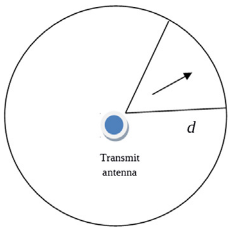


Figure 5.10 Sensor modeled as a pin point source.

considered omnidirectional antenna for sensors 1 and 2. The energy radiated for transmitting is spread over the surface of the sphere. Hence the effective distance on the received signal power can be analyzed. The sensor node is modeled as a point source as shown in Fig. 5.10. The radius of the range of the antenna is "*d*." Hence the surface area of the sphere is $4\pi d^2$. Here data are transferred from sensor 1 to sensor 2. Hence the received power density can be calculated at sensor 2, with transmitting power P_t and transmitting power gain G_t.

$$w = \frac{P_t G_t}{4\pi d^2} \tag{5.1}$$

$$P_r = \frac{P_t G_t}{4\pi d^2} \times A \tag{5.2}$$

"A" is area of aperture.
if

$$G_r = \frac{4\pi A}{\lambda^2} \tag{5.3}$$

then

$$A = \frac{G_r \times \lambda^2}{4\pi} \tag{5.4}$$

"λ" can be written as c^2/f^2.

Hence Eq. (5.4) can be written as $A = \frac{G_r \times \boldsymbol{f^2}}{4\pi \boldsymbol{c^2}}$
Putting Eq. (5.4) in Eq. (5.2), we get

$$P_r = \frac{P_t G_t}{4\pi d^2} \times \frac{G_r \times \boldsymbol{f^2}}{4\pi \boldsymbol{c^2}}$$

$$= \frac{P_t G_t G_r \times \boldsymbol{f^2}}{(4\pi d)^2} \tag{5.5}$$

$$P_r = \frac{P_t G_t G_r \times f^2}{(4\pi d)^2 c^2}$$

In free space propagation model, it is clearly visible from Eq. (5.5) that the received signal power is inversely proportional to the square of the distance; that is, the power decays with the negative square root of distance. The distance d is the separation distance from sensor 1 to sensor 2 as shown in Fig. 5.10. The overall gain of the receiving sensor is directly proportional to the aperture "A" as given in Eq. (5.4).

The free space propagation model acts a basis over which other propagation models are considered or formulated. The free space propagation model is an ideal case where there is no obstacle in the environment between the sensor transceivers. Again the type of antenna should also be taken into account while formulating the transmitter and receiver formula. In Figs. 5.9 and 5.10, we have considered an omnidirectional antenna and hence the range of the transceiver is a sphere shape. The Eq. (5.5) can further be simplified as follows:

$$P_{r(d)} = \frac{P_t}{\frac{d^2}{d_0^2}} \tag{5.5a}$$

P_t is the reference power from the transmitter.

There are various other models that are considered by modifying the free space propagation model. In order to model the path loss in wireless transmission, one has to consider the basic propagation mechanism.

The basic propagation mechanism [6] in wireless transmissions is (Fig. 5.11) as follows:

a. *Reflection*: The transmission of data due to the impingement of electromagnetic wave on objects in the environment.

b. *Diffraction*: It occurs when the electromagnetic transmission is obstructed by sharp edges in the environment.

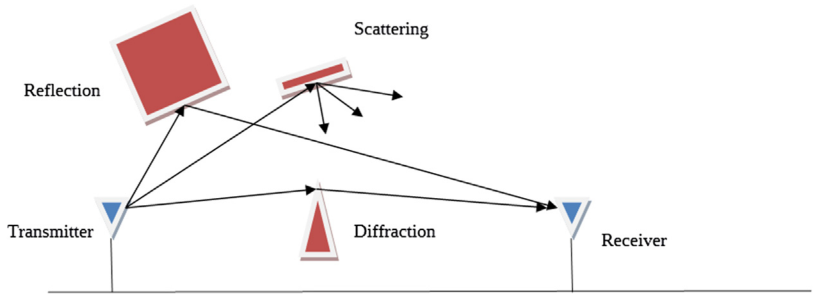


Figure 5.11 Basic propagation mechanisms.

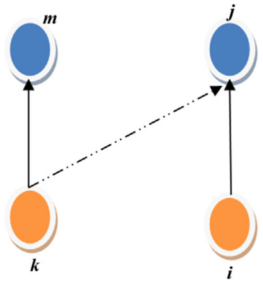


Figure 5.12 Snapshot of four adjacent sensors in WSN.

c. *Scattering*: It occurs when the transmitted wave is obstructed by small objects of size smaller than the wavelength of the signal.

The propagation model is mainly focused on estimating the average signal strength drop at different transceiver (T–R) separation. The variation of signal strength is due to changes in the propagation path between a transmitter and a receiver. The propagation mechanism, the other wireless devices in the communication flow, and the frequency and distance are the major things that impact the energy and path loss model.

During convergecasting, information is relayed wirelessly from one sensor node to another. The nodes that gather information or senses and generate data are called coverage nodes, and the nodes that transfer the data toward its respective base station are called relay nodes. A coverage node can act as a relay node. Fig. 5.12 shows four sensor nodes, and node *i* channels the data to its parent node *j*, and similarly, *node k* relays the information *node m*. Successful reception of a packet from node *i* to *j* occurs when the SINR is greater than a certain threshold. The signal to noise can be given as follows:

$$\mathrm{SINR}ij = \frac{P_i \times g_{ij}}{\sum_{k \neq i}^{n} P_k \times g_{kj} + N} \tag{5.6}$$

$$g_{ij} = \frac{1}{d_{ij^{\propto}}} \tag{5.7}$$

P_i is the transmission power, and g_{ij} is gain at $\{i,j\}$. The multiplied value of gain and power is divided with all the external wireless signals $(P_k \times g_{kj})$ that affect the transmission and receiving of packets along with the external noise N.

The typical path loss exponent $\propto$ varies according to various environmental changes. The received power at the receiver is inversely proportional to $d^{\propto}$. In case of free space as we have seen from the Eq. (5.5), the path loss exponent is 2; for the urban area, the path loss is 2.7−3.5; for the suburban area, the path loss is 3−5; and so on.

5.7 Data collection methodologies

As we know in convergecasting data are collected from various coverage nodes and relayed toward their respective sink node or the base station, the procedure used in such collection always remains one of the major concerns. With respect to the MAC protocol used in the data transmission, the scheduling for the convergecast is determined. For example, in the case of time division multiplexing, the sensor data are sent from sensors in time slots. The average slot size is determined with respect to the type of convergecasting we are using; here the schedule length is the slot size.

Convergecasting can be broadly classified as follows:

i. *Raw data convergecast*: In raw data convergecasting [7], the schedule length increases as the sensor network gets sparser. This is the case because when the network gets sparser, the reuse of the slots will be higher which in turn will reduce the schedule length. As the network gets sparser, the number of nodes that can directly reach the base station decreases, and hence the relay nodes will have to relay the information or the collected data toward their respective sink. Hence a number of packets have to schedule in a single transfer.

ii. *Periodic aggregate convergecast* [8]:

Data aggregation's main objective is to eliminate redundancy and to minimize transmission, thus saving energy and improving the lifetime of the network. Aggregation can be done in many ways, such as applying data compression, suppressing duplicate data or information, and so on. The overall size of the aggregated data transmitted by each node is constant and does not depend on the size of the sensor reading.

We have briefly discussed aggregation and why it is needed. Aggregation in convergecast is done when we want to summarize the overall data that are sensed by a sensor network. Raw data convergecast is done when every data from every sensor is equally important. In the case of periodic data sensing, the data collection is usually summarized. Hence in case of continuous data, aggregate convergecast is used. Periodic aggregation and raw data are usually the two extreme cases of data collection. There are various collection procedures whose method of the collection lies in between these two extremes.

Data aggregation can be classified into:

i. No aggregation
ii. Partial aggregation
iii. Full aggregation

No aggregation contributes to transmission delays, and full aggregation contributes to energy saving. However, in some applications, we require a short delay and low energy consumption. For that very purpose, we propose a partial data aggregation method wireless routing protocol (WRP), where some data are aggregated and some are not. Some of the difficulties faced by full data aggregation are overcome by partial data aggregation.

In order to work with sensor network and modeling, the sensor network can be modeled as graph problem, where graph *S(V, E)* contains "*V*" number of sensor nodes among which there is one base station "*b*" and list of coverage nodes "*c*" and relay nodes "*r*." *Nodes 1−5* are sensor nodes that relay their information toward a single sink B_1, which is the base station. The sole objective of data aggregation is to reduce the number of redundant data transmission and decrease the energy required. From the perspective of energy saving, data aggregation can collect much more significant data. However, the delay in transmission is equally important to application such as disaster monitoring. Hence to achieve proper convergecasting in WSN, a tradeoff between energy and delay is very necessary.

As depicted in Fig. 5.13, the length of each slot is 4 (i.e every slot carries 4 packets) and hence payload is 4. The number of TDMA slots required for transmission is calculated using a simple assignment

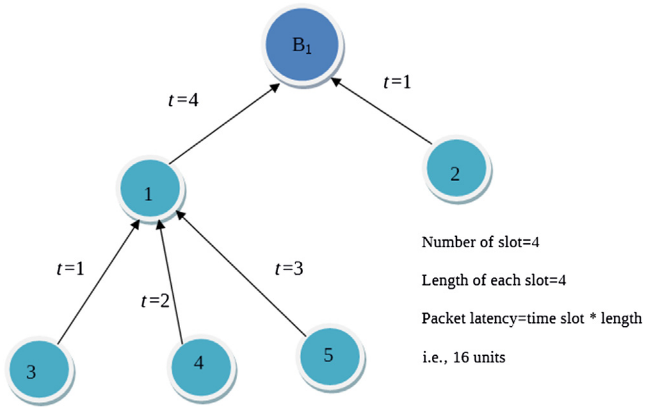


Figure 5.13 Slot allocation examples.

mechanism. Nodes 3, 4, and 5 are sending their data to their parent node 1 using time slots 1, 2, and 3, respectively. Node 1 cannot use the slots (1,2,3) for its transmission purpose to the base station B_1 as it is already occupied by its adjacent sensors and hence it uses slot 4 for its transmission. Node 2, on the other hand, can utilize from slots 1—4 for its transmission to B_1 (in this case, it is allocated to slot 1).

5.8 Types of aggregation

5.8.1 Network aggregation

In this methodology, mainly centralized mechanism is addressed, where each node sends the information to a central node through the shortest path through multihop transfer. In case of in-network aggregation [9], the data packet sent through the multihop network and process information within the node; therefore the explicit energy consumption will reduce significantly. This method can be categorized with size reduction where the packet length scales down as it moves to the neighbor node. Fig. 5.14 shows the basic principle of in-network aggregation.

5.8.2 Tree-based approach

In this case, the data aggregation can be done based on the spanning tree approach (Fig. 5.15). Every node must incorporate with its parent node

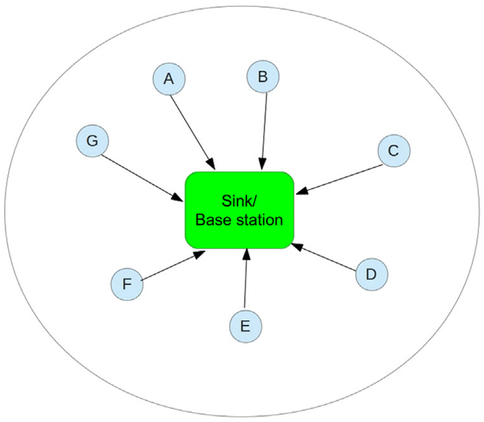


Figure 5.14 **A** Network approach.

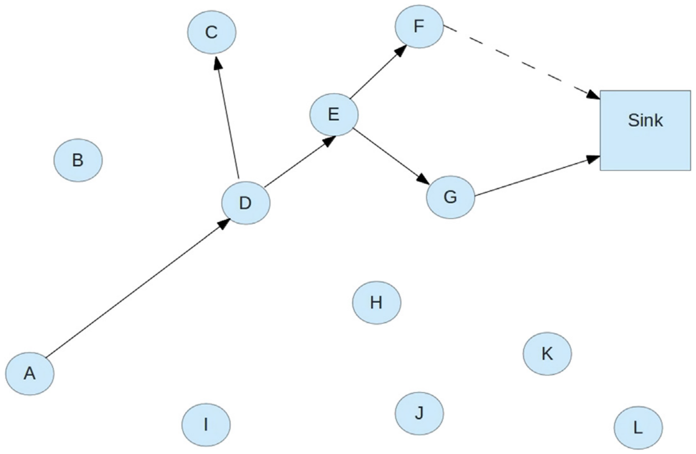


Figure 5.15 Tree-based approach.

and the data forwarded from the source to sink by relaying it through the parent node of a node [10]. Therefore the data are collected by leaf nodes, and periodically they are aggregated by parent node at each level.

5.8.3 Cluster-based aggregation

In this technique, the deployment region of the sensor geographically segregated as a set of clusters illustrated by Fig. 5.16. One of the nodes (that can be thought as leader node) can be used as a cluster head [11]. The cluster head may be chosen arbitrarily, and it collects the data from all

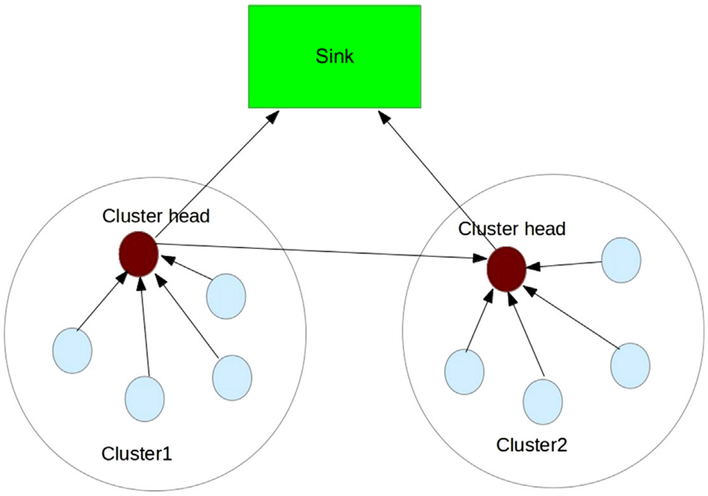


Figure 5.16 Cluster-based approach.

other slave nodes. Some interconnections between the clusters are possible by performing message transfer between cluster heads and sink.

5.9 Summary

- Sensor data are the most important components in the IoTs. The data are mostly classified as periodic, nonperiodic, and one-shot data.
- Unicast, multicast, and broadcast are the fundamental data transmission methodology used.
- Convergecasting is the inverse of broadcasting. In convergecast, the sensor data are relayed toward their respective sink node.
- Data aggregation is the summarization of the sensed data from various sensors. The main objective of aggregation is to increase the lifetime of the network and decrease retransmission.
- Convergecasting is broadly classified into raw data convergecast and periodic aggregated convergecast.

References

[1] J. Gubbi, R. Buyya, S. Marusic, M. Palaniswami, Internet of things (IoT): a vision, architectural elements, and future directions, Future Gener. Comput. Syst. 29 (7) (2013) 1645–1660.

[2] J. Bernsen, D. Manivannan, Unicast routing protocols for vehicular ad hoc networks: a critical comparison and classification, Pervasive Mob. Comput. 5 (1) (2009) 1−18.
[3] L.F. van Hoesel, T. Nieberg, H.J. Kip, P.J. Havinga, Advantages of a TDMA based, energy-efficient, self-organizing MAC protocol for WSNs, 2004 IEEE 59th Vehicular Technology Conference. VTC 2004-Spring (IEEE Cat. No. 04CH37514), Vol. 3, IEEE, 2004, pp. 1598−1602.
[4] D.G. Zhang, K. Zheng, T. Zhang, X. Wang, A novel multicast routing method with minimum transmission for WSN of cloud computing service, Soft Comput. 19 (7) (2015) 1817−1827.
[5] B. Malhotra, I. Nikolaidis, M.A. Nascimento, Aggregation convergecast scheduling in wireless sensor networks, Wirel. Netw. 17 (2) (2011) 319−335.
[6] J. Xu, W. Liu, F. Lang, Y. Zhang, C. Wang, Distance measurement model based on RSSI in WSN, Wirel. Sens. Netw. 2 (08) (2010) 606.
[7] O.D. Incel, A. Ghosh, B. Krishnamachari, K. Chintalapudi, Fast data collection in tree-based wireless sensor networks, IEEE Trans. Mob. Comput. 11 (1) (2011) 86−99.
[8] X. Xu, X.Y. Li, P.J. Wan, S. Tang, Efficient scheduling for periodic aggregation queries in multihop sensor networks, IEEE/ACM Trans. Netw. (TON) 20 (3) (2012) 690−698.
[9] B. Krishnamachari, D. Estrin, S.B. Wicker, The impact of data aggregation in wireless sensor networks, In: ICDCS Workshops, vol. 578, 2002.
[10] K. Dasgupta, K. Kalpakis, P. Namjoshi, An efficient clustering-based heuristic for data gathering and aggregation in sensor networks, 2003 IEEE Wireless Communications and Networking, 2003. WCNC 2003, Vol. 3, IEEE, 2003, pp. 1948−1953.
[11] F. Yuan, Y. Zhan, Y. Wang, Data density correlation degree clustering method for data aggregation in WSN, IEEE Sens. J. 14 (4) (2013) 1089−1098.

Further reading

Y. Chu, P.D. Mitchell, D. Grace, ALOHA and Q-learning based medium access control for wireless sensor networks, 2012 International Symposium on Wireless Communication Systems (ISWCS), IEEE, 2012, pp. 511−515.

CHAPTER SIX

Intelligent sensor network

Contents

6.1 Introduction

If we search for the word “intelligent” related to computer science we will come across various terms related to automation, machine learning, artificial intelligence (AI), embedded system, and many more. An intelligent system can be classified as an agglomeration of all the stated words that is a system which is able to perform certain computation without any human intervention and is able to take decisions accordingly.

Researchers have developed algorithms created systems and researched in the domain of machine intelligence for the past many years. In our day-to-day environment, we can observe sensors are practically incorporated everywhere, be it the devices that we daily use, such as a fridge, washing machine, microwave, devices used for weather monitoring, or simply a sensor to open the door in a shopping mall. Sensors are deployed for the very purpose of creating a pervasive environment, which is the basic foundation of the Internet of things (IoT) [1] we know of today. Intelligence related to sensors or sensor networks [2] does not simply cover the intelligent computing aspect, that is, the inferences and decisions drawn from the sensed data; intelligence related to sensor network also covers the routing or relaying of information in an intelligent manner. A regional wireless sensor network (WSN) [3] is a sensor network where a region of the area has to be sensed via sensors, while the target covering WSN [4] is a WSN design where a target area has to be sensed and the sensed data has to be relayed to the particular sinks. In both the design, gathering the data in a timely manner is of prime importance. The chapter discusses various domains related to intelligent sensing such as classification and clustering techniques and gives an insight into the sensor data mining with practical examples. The chapter also discusses the aspect of intelligent node deployment and relaying of information and hence, in general, making the sensor network as a whole intelligent.

6.2 Intelligence hierarchy

The IoT as we know of today is much more than connecting devices over the Internet, it is widely used in modern intelligent services around the world. Cognitive IoT [5] is a new paradigm where the focus lies in implementing intelligent IoT systems and services with the use of AI in the working of a system as a whole. The flow of any system can be categorized into its constituent subsystems. When we talk about WSNs or sensor cloud in general, the whole model can be broken up into small subsystems where the data is gathered, routed, analyzed, and accordingly predictions are made and decisions are taken. In the lower level of services, we have the sensors; the sensors are the most important components in any WSN. As the sensors are usually battery-operated devices, it is very essential for the WSN designers to take into consideration the energy-efficient algorithm in order to increase the lifetime of the sensors. Computer intelligence for making intelligent decisions during every phase of the operation is very essential. Applied intelligence in WSN model can be generally classified as:

1. Node deployment and gathering level intelligence.
2. Routing and relaying level intelligence.
3. Analysis and prediction level intelligence.

6.3 Preliminary concepts of AI and Machine Learning

AI is the concept of performing intelligent decisions in the flow of an algorithm with respect to some knowledge base. AI creates a way of improving our existing algorithms through previous experience. Some of the popular tools for AI used in various fields are optimization, bio-inspired algorithms such as genetic algorithm and particle swarm optimization (PSO), and classifiers and clustering techniques.

Machine learning is one of the key branches of AI. Making a computational machine to learn and infer information and to make predictions accordingly is of prime importance in today's modern world. As technology is advancing, the volume of data is multiplying; over the past decade, the volume of data generated by various applications online and offline is huge, 2.5 quintillion bytes of data are generated every day and the pace is increasing with the growth of the IoT.

A machine learns by performing predictive modeling. Predictive modeling can be broadly classified into:

1. Regression classification
2. Pattern classification

Regression models [6] are based on the analysis of relationships between variables in order to make predictions upon continuous variables. Pattern classification, on the other hand, applies discrete class labels to particular observations as outcomes of a prediction. A simple overview can be explained using an example of weather forecasting. Predicting the highest temperature for the upcoming day is a problem of regression classification, while the task of predicting whether the forecasted weather will be sunny, windy, or rainy, is a problem for pattern classification.

A class is much like a name given to common property of attributes for objects, for example, a car is a class, and Mercedes, BMW, and so on are objects of that "car" class. The label is the procedure of putting the data into their respective class or category. We specifically use the word classified instead of the word labeled.

A machine learns by putting data into their respective classes and identifies or predicts what needs to be done; this is the case when the classes for the data are known. For example, a room is full of various objects, and if we specify three buckets and name them as electronic gadgets, books, and stationary objects then we are explicitly specifying the classes for the objects. The objects have to be kept in the respective buckets to which they belong. It might also be the case that four buckets are kept but they have no label. Now one might cluster similar types of objects into one bucket, for example, laptops, mobiles, and so on might be clustered into one bucket as they are electronic gadgets. Hence, machine learning algorithm can be broadly classified into supervised and unsupervised learning that is one having a class label and the other that does not have any class label as depicted in Fig. 6.1. The prediction and analysis depend on the type of data we want to predict. For example, if we try to predict a simple true/false or yes/no then the number of classes in which the data will be classified is two. Hence we call it binary classification. Similarly, if there are numerous classes more than two we call it multiclass classification; it might also be the case that we are trying to predict a real value, for example, stock values of the next day using previous historical values [7], then it is called regression classification. For performing machine learning algorithms there are various tools and packages provided namely Weka, Tensorflow, scikit learn, and R-caret, etc. The field of machine learning amalgamates the field of statistics, computer science and probability into one. In order to understand the interdisciplinary field of machine learning, it is better to have some preliminary knowledge of

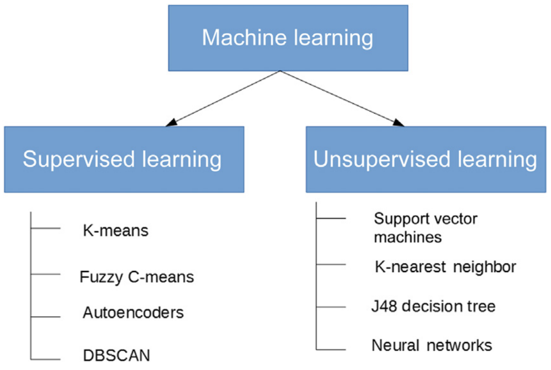


Figure 6.1 Some popular machine learning algorithms.

some of the mathematical fields such as linear algebra (Eigen vectors and values, LU decomposition), probability and statistics, and calculus.

6.3.1 Unsupervised learning method

Unsupervised learning is the method of creating the labels for each of the data when the class labels are not known. The classes are formed accordingly with the nature and values of each of the data, that is, the same type of data are clustered into a single label. Unsupervised learning can be defined using a single term called clustering. The clustering procedure can be visualized using Fig. 6.2. Random data points are taken over some dependent and independent variables. The same type of data is clustered into a single cluster, here clusters are formed over a set of data points considering the simple distance as the criteria between the points.

Clustering method can be classified as

1. *Density-based methods*: These techniques are usually used for nonlinear structural shapes. These methods consider the clusters as the dense region having some similarities and are different from the lower dense region of the space. These methods have good accuracy and the ability to merge two clusters. One of the popular density-based clustering approaches is density-based spatial clustering of applications with noise (DBSCAN) [8].
2. *Hierarchical-based methods*: The clusters formed in this method forms a tree-type structure based on the hierarchy. New clusters are formed using the previously formed one. Some of the examples are clustering

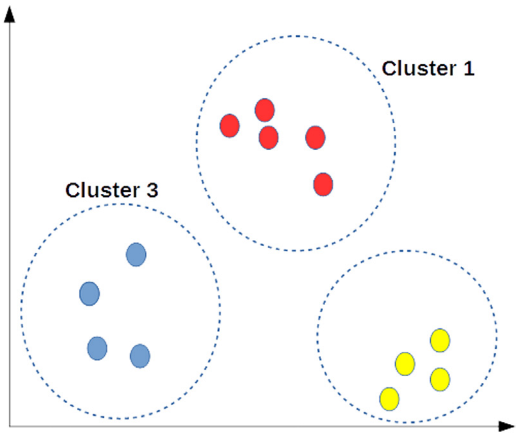


Figure 6.2 Clustering example.

using representatives, balanced iterative reducing clustering and using hierarchies and so on [9].

3. *Partitioning methods*: These methods partition the objects into K clusters and every partition is considered as a cluster. The method is mostly used to satisfy an objective criterion of a function by considering various parameters such as distance to form the clusters.
4. *Probabilistic methods*: In probabilistic methods, the degree of belongingness is calculated for each of the data points. That is a probability factor determines that in which cluster the data will be kept. One of the popular examples of a probabilistic clustering is Fuzzy C-means clustering [10].

Every cluster has a central vector, which does not necessarily have to be part of the cluster dataset. When the number of clusters is fixed to K, say we are considering the K-means clustering, it gives a formal definition as an optimization problem: find the K cluster centers and assign the objects to the nearest cluster center such that the squared distances from the cluster are minimized. In K-mean clustering [11] the data are partitioned into K cluster. The clustering is performed according to the nearest mean value of the cluster.

6.3.2 Supervised learning method

Supervised learning is a broader domain for the branch of machine learning algorithms where the class to which the data will belong is known. Supervised learning is the machine learning task of inferring a function from labeled training data. The training data consist of a set of training examples. When we consider the supervised learning model the dataset are broken up in training and testing examples. A supervised learning algorithm analyzes the training data and produces an inferred function, which can be used for mapping the test data. The best scenario is when an algorithm correctly determine the class labels for unseen instances. This requires the learning algorithm to generalize from the training data to unseen situations in a "reasonable" way. For any classification operation, the feature is the prime concept used for prediction operation. A good feature selection helps in better classification.

The training set and the testing set can be divided in many ways. One can put the whole of the gathered data into the training set and give a new set of gathered data into the testing set. The percentage split is another option where the whole of the dataset is divided into training and testing set with respect to some percentage. For example, 60% of the feature data are kept in the training set and the rest 40% are kept in the testing set. Another way of partitioning the training and testing data is *K*-fold cross-validation, where *K* can be varied accordingly. If *K* is 10, then the whole dataset is broken up into 10 folds, the first nine are allotted into the training model and the last one is kept for testing.

Let us consider a simple medical database;the training set would have relevant patient information recorded previously, where the prediction attributes are whether or not the patient had a heart problem. Tables 6.1 and 6.2 below illustrate the training and prediction sets of such database.

The feature set considered in Table 6.1 is age, heart rate and blood pressure with which a model is created accordingly to serve the prediction purpose of whether the individual in Table 6.2 will have a heart problem or not. The accuracy of the prediction can be calculated from a truth table also called the confusion matrix than contains the actual predicted values of the individual. A comparison between the actual and predicted value gives us the percentage of accuracy for the created model.

Some of the popular approaches in supervised learning methods are:

1. Naïve Bayes classifier [12].
2. Neural network [13].
3. Decision tree learning [14].
4. Inductive logic programming [15].
5. Support vector machines (SVMs) [16].

Table 6.1 Training dataset.

Age (years)	Heart rate (bpm)	Blood pressure (mmHg)	Heart problem
65	78	150/70	Yes
37	63	120/76	No
61	87	108/65	No

Table 6.2 Prediction set.

Age (years)	Heart rate (bpm)	Blood pressure (mmHg)	Heart problem
43	62	128/70	?
65	80	135/76	?
84	73	108/65	?

? indicates class label is not known.

6.3.3 Real-life example of prediction

- *Face recognition using OpenCV*

  Identifying a person with his or her face in real world is one of the best ways to distinguish one individual from another or just to identify the very person in general. Face recognition in computers is one the most sought out research domain in artificial intelligence and machine learning. One should not confuse face recognition with face detection, as face recognition is the way of identifying "who the person is," while face detection is the procedure of detecting "faces" from images or videos. One of the popular libraries for python programming is the open computer vision library popularly known as OpenCV [17]. It was created by Intel and it is a cross-platform open source library providing numerous functions for image analysis and prediction analysis. One of the popular ways of identifying faces is the Eigenface method, which is based on the reduction of face-dimensional space using principal component analysis (PCA).

  PCA is a dimensionality-reduction method that is used to reduce the dimensionality of very big datasets by modeling the large set of variables into a smaller one that still contains most of the information in the large set. The main motive here is to find the set of features that are most important for prediction. For example, let us consider a problem of identifying a person X in the real world; the set of features that have been considered for the prediction purpose can be the face, height, or attire. Now out of these features, the principal component over here is the face, then the height, and the attire feature can be removed from the feature set as it has no significant contribution toward the prediction analysis. Thus the dimension of the feature set is reduced.

  In this very example of image classification, Haar cascade classifier in OpenCV library is used for training the model. A Haar cascade classifier is basically used for detecting objects from the source. The classifier works by gathering the negative images for the training set. The training set for the data is gathered by the cascade classifier. A gathered database of face images is depicted in Fig. 6.3(A) and (B) both taken from two different candidates. First, the image is scanned from a graphical user interface (GUI) interface through a webcam or any camera device. From the gathered image training repository is created and a name for that particular face is assigned. There are various ways of

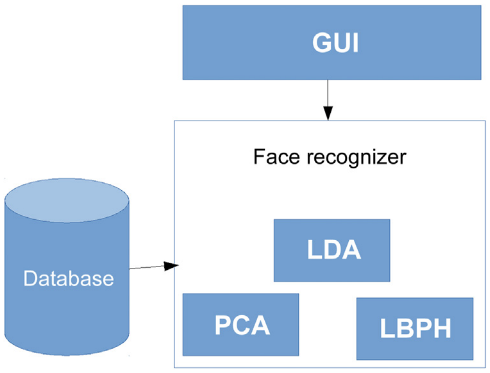


Figure 6.3 **(A)** Training set candidate 1. **(B)** Training set candidate 2.

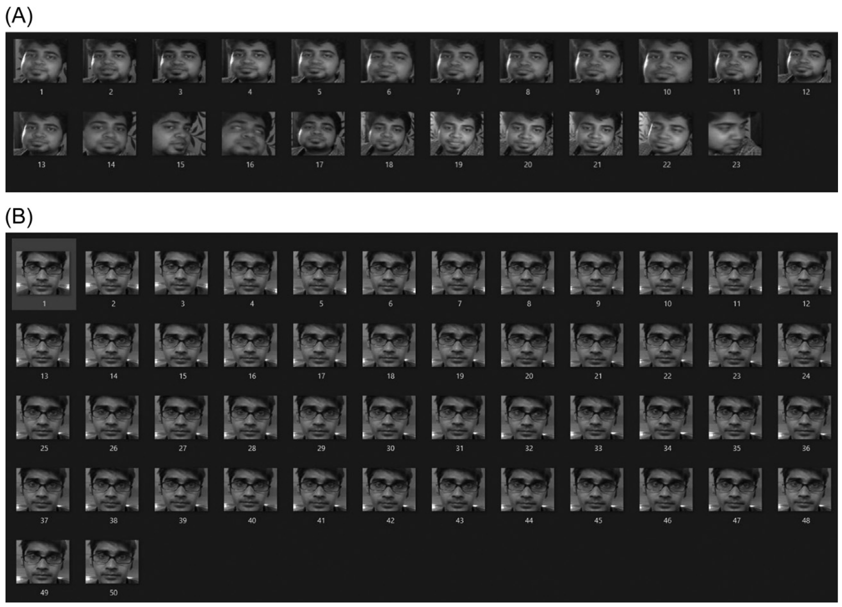


Figure 6.4 Outline procedure.

classifying facial data; some of the popular ways are local binary pattern histogram [18], Fishersfaces, Eigenfaces, and so on. The overview of the procedure is depicted in Fig. 6.4. The classifier is created from a prebuilt face recognizing the model. Finally, the recognizer identifies who the person is in real time.

• *Database creation for face detection procedure*

```
import cv2, sys, numpy, os, time

count = 0

size = 4

fn_haar = 'haarcascade_frontalface_default.xml'

                #file taken from github

                #https://github.com/opencv/opencv/blob/master/data/haa
                rcascades/haarcascade_frontalface_default.xml

fn_dir = 'database'

fn_name = 'agniva' #name of the person

path = os.path.join(fn_dir, fn_name)

if not os.path.isdir(path):

os.mkdir(path)

(im_width, im_height) = (112, 92)

haar_cascade = cv2.CascadeClassifier(fn_haar)

webcam = cv2.VideoCapture()

webcam.open(0)

print("-----------------------Taking pictures----------------------")

print("--------------------Give  some  expressions---------------------
")

# The program loops until it has 20 images of the face.

while count < 50:

(rval, im) = webcam.read()

im = cv2.flip(im, 1, 0)

gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)

mini = cv2.resize(gray, (int(gray.shape[1] / size),int(gray.shape[0] /
size)))

faces = haar_cascade.detectMultiScale(mini)

faces = sorted(faces, key=lambda x: x[3])
```

```
if faces:

face_i = faces[0]

(x, y, w, h) = [v * size for v in face_i]

face = gray[y:y + h, x:x + w]

face_resize = cv2.resize(face, (im_width, im_height))

pin=sorted([int(n[:n.find('.')]) for n in os.listdir(path)

if n[0]!='.' ]+[0])[-1] + 1

cv2.imwrite('%s/%s.png' % (path, pin), face_resize)

cv2.rectangle(im, (x, y), (x + w, y + h), (0, 255, 0), 3)

cv2.putText(im, fn_name, (x - 10, y - 10), cv2.FONT_HERSHEY_PLAIN,

1,(0, 255, 0))

time.sleep(0.38)

count += 1

cv2.imshow('OpenCV', im)

key = cv2.waitKey(10)

if key == 27:

break

print(str(count) + " images taken and saved to " + fn_name +" folder
in database ")
```

- *Classification using PCA*

```
import cv2, sys, numpy, os,time

size = 4

fn_haar = 'haarcascade_frontalface_default.xml'

fn_dir = 'database'

print('Training...')

haar_cascade = cv2.CascadeClassifier(fn_haar)

# Create a list of images and a list of corresponding names

(images, lables, names, id) = ([], [], {}, 0)

for (subdirs, dirs, files) in os.walk(fn_dir):

forsubdir in dirs:

names[id] = subdir

subjectpath = os.path.join(fn_dir, subdir)

for filename in os.listdir(subjectpath):

path = subjectpath + '/' + filename

lable = id

images.append(cv2.imread(path, 0))

lables.append(int(lable))

id += 1

(im_width, im_height) = (112, 92)

# Create a Numpy array from the two lists above

(images, lables) = [numpy.array(lis) for lis in [images, lables]]

defdraw_text(img, text, x, y):

cv2.putText(img, text, (x, y), cv2.FONT_HERSHEY_PLAIN, 1.5, (0, 255,
0), 2)

defdetect_face(im):

im = cv2.flip(im, 1, 0)

gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
```

```
mini = cv2.resize(gray, (int(gray.shape[1] / size),int(gray.shape[0] /
size)))

faces = haar_cascade.detectMultiScale(mini)

faces = sorted(faces, key=lambda x: x[3])

if faces:

face_i = faces[0]

(x, y, w, h) = [v * size for v in face_i]

face = gray[y:y + h, x:x + w]

returnface,faces[0]

recognizer = cv2.face.LBPHFaceRecognizer_create()

recognizer.train(images, lables)

print("Training done")

'''

img=cv2.imread('face.jpg')

face, rect = detect_face(img)

label= recognizer.predict(face)

print(names[label[0]])

'''

cascadePath = "haarcascade_frontalface_default.xml"

# Create classifier from prebuilt model

faceCascade = cv2.CascadeClassifier(cascadePath);

# Set the font style

font = cv2.FONT_HERSHEY_SIMPLEX

# Initialize and start the video frame capture

cam = cv2.VideoCapture()

cam.open(0)

# Loop
```

```
while True:

# Read the video frame

ret, im =cam.read()

# Convert the captured frame into grayscale

gray = cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)

# Get all face from the video frame

faces = faceCascade.detectMultiScale(gray, 1.2,5)

# For each face in faces

for(x,y,w,h) in faces:

# Create rectangle around the face

cv2.rectangle(im, (x-20,y-20), (x+w+20,y+h+20), (0,255,0), 4)

# Recognize the face belongs to which ID

label= recognizer.predict(gray[y:y+h,x:x+w])

# Put text describe who is in the picture

cv2.rectangle(im, (x-22,y-90), (x+w+22, y-22), (0,255,0), -1)

cv2.putText(im,    str(names[label[0]]),    (x,y-40),    font,    2,
(255,255,255), 3)

# Display the video frame with the bounded rectangle

cv2.imshow('FRAME',im)

# If 'q' is pressed, close program

if cv2.waitKey(10) & 0xFF == ord('q'):

break

# Stop the camera

cam.release()

# Close all windows

cv2.destroyAllWindows()
```

6.4 Intelligent approaches in WSN node deployment

The WSN is a paramount domain of research. There are wide varieties of the sophisticated applications in environment monitoring, smart city, surveillance, and health care [19], cyber-physical systems are considered under the umbrella of WSN. The fundamental goal of WSN is to sense the environment and the physical system. The nodes are considered to be inexpensive low power devices with a minimum amount of processing capability. The inherent architecture of the WSN encompasses a centralized powerful node that agglomerates and aggregates the data thereby considered as centralized storage. The central node can also be used as an intelligent decision-making unit that processes the data by using an intelligent algorithm to generate a decision.

There are several issues of the WSN system that can be addressed like the routing of the data [20], node deployment [21], energy awareness, intelligent data aggregation, and many more. Intelligent approaches in a WSN are also a big research domain in the current era. There are several domains where the intelligent approach can be included like sensor deployment, classification of the ambient data, the information routing from one node to another node. Again in the design of mobility models for dynamic nodes, optimization of the number of sensor nodes in the field of task allocation is one of the major issues. The traditional optimization methodology requires an enormous amount of analytical and computational power. The sensor node on the other hand have low computation power and low memory; therefore, more sophisticated optimization algorithm is required to develop that gives a good result in such a challenged scenario.

One of the challenging problems of the sensor network is to deploy the node in a proper location. The parameter like coverage, connectivity, and energy efficiency, and the optimum number of nodes are the primary issues in order to achieve the best performance of the WSNs. Such an optimal sensor network guarantees significantly high network longevity, a better quality of service, and financially viable infrastructure.

6.4.1 Node deployment using PSO-Voronoi

PSO falls under the category of evolutionary computation or to be precise nature-inspired optimization usually found in flocking of a group of birds while flying from a source node to a destination node. It is considered as a multidimensional technique to get an optimized solution. The high-quality optimized solution, with better computational efficiency, is the phenomena of PSO algorithms. The behavior of PSO generally refers to flocking of the group of nodes like birds. The model comprises of k candidate often known as a particle in an n-dimensional hyperspace. The objective is to search a global optimal solution where n can be considered as a number of determinable parameters. Let a particle i occupy a position

X_{id} and the velocity of the same is considered as V_{id} in d-dimension space with a consideration of $1 \leq I \leq k$ and $1 \leq d \leq n$. an objective function can be derived for the particle as follows

$$F(m_1,\ m_2,\ \ldots m_n)\ \text{where}\ f\colon \check{R} \rightarrow R \tag{6.1}$$

The fundamental goal here is to minimize the cost function or maximize fitness. The velocity V and the position X of the particle can be updated using following equations:

$$V_{\text{id}}(m+1) = wV_{\text{id}}(m) + \varphi_1 r_1(m)(pb_{\text{id}} - X_{\text{id}}) + \varphi_2 r_2(m)(gb_{\text{id}} - X_{\text{id}}) \tag{6.2}$$

$$X_{\text{id}}(m+1) = X_{\text{id}}(m) + V_{\text{id}}(m+1) \tag{6.3}$$

Here m is the number of iteration. The position of the particle i with the lowest cost is signified by pb_{id} and the position of the best particle is $gb_{\text{id}}.\varphi_1\varphi_2$ are the constants. The value of $r1(m)$ and $r2(m)$ signifies the random numbers where $0 \leq r_1(m) \leq 1$ and $0 \leq r_2(m) \leq 1$.

The positioning of the node in the proper optimal region is a challenge for the sensor network. Voronoi partition-based algorithms [22] often called PSO-Voronoi is quite useful in such cases. In this scenario, each region of interest (ROI) point has to be covered by a sensor node. Thereby the whole area has been covered. PSO-Voronoi approach is mainly used to optimize the sensor coverage area. The length measurement technique is highly used in this WSN coverage problem. The point inside ROI has been covered by sensor S_i and if the distance point of ROI is less than the *radius* (s_i), thus all the point inside ROI has been covered. In general an ROI is basically a combination of an infinite number of points. The sampling method has been introduced so that among them a fixed number of points are considered to evaluate coverage.

The coverage evaluation algorithm, in this case, uses a PSO-based Voronoi technique in spite of grid technique. To obtain the position of the sensors in ROI, PSO is used, and the current position coverage is obtained by the Voronoi diagram. A Voronoi diagram of seven sites has been depicted in Fig. 6.5. Each sensor in the case act as the main component of the sites and a polygon can be generated against it. In this case, a bounded region has to be considered in order to deal with the positioning of the sensor nodes within a specific ROI. So the set of points, in this case, are randomly selected along each boundary. These points along with the vertices are considered as interest points. The coverage of the sensor

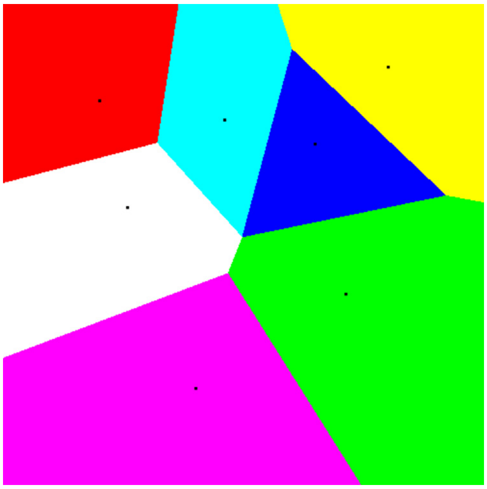


Figure 6.5 Voronoi partition for seven sites.

is, therefore, be obtained by computing the fitness function, which can be represented by the distance between the interest point and nearest sensor.

The algorithm for computing fitness function is illustrated as below

Algorithm 1 Fitness function

```
Begin: initialize: fitness
Input: sensing_radius
1. setinterest_point ⊛ [polygon vertices, randomly selected n point along
   boundary]
2. loop: point ⊛ each interest point
3. Dist ⊛ compute_distance(interest_point—nearest_sensor)
4.    If: $dist > sensing\_radius$ then
5.        Fitness ⊛ $fitness + (dist - sensing\_radius)$
6.        End if
7. End
```

The fundamental objective of the above-mentioned algorithm is to minimize fitness value. Ideally, the value must be ≤ 0, which signifies that the nearest sensor's interest point distance is within the sensing range of the sensor. The PSO algorithm here will be ended fewer than two following criterion: (1) As ideal fitness function has achieved and (2) the algorithm performs its maximum iteration. The sensing algorithm for randomly distributed sensors is illustrated below.

Algorithm 2 Sensing Algorithm

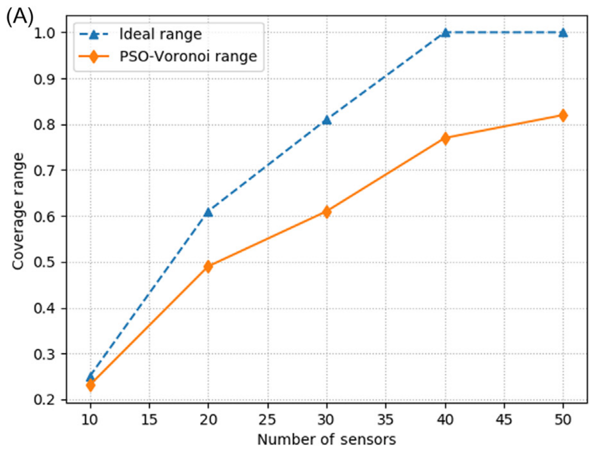
```
Begin:initialize:particles_population
1. Do: compute fitness ⊛ current_fitness
2.     If: $P_{id} >$ current_fitness then
3.         update ($P_{id}$)
4.     $P_{gd}$ ⊛ best_fitness(neighbor($P_{gd}$))
5.     Loop: for each particle
6.         Velocity $V_{id}$ ⊛ $V_{id}(m+1) = wV_{id}(m) + \varphi_1 r_1(m)(pb_{id} - X_{id})$
             $+ \varphi_2 r_2(m)(gb_{id} - X_{id})$
7.     Position$X_{id}$ ⊛ $X_{id}(m) + V_{id}(m+1)$
8. While: iteration = MAX_iteration or fitness = Ideal_fitness
9. End
```

The result of the test that was conducted on the given ROI is discussed below. In the present situation, the constant ROI value is considered and the numbers of sensors varies from 10 to 50. It is observed that a sensor can cover up to a coverage area of 50 × 50 efficiently, which can be evaluated by a minimum number of the circle ( ) needed to make a coverage area that is given as follows:

$$\acute{n} = \frac{\acute{\alpha}}{3 \times \sqrt{3r^2/2}} \tag{6.4}$$

Fig. 6.6(A) depicts the ideal versus. proposed coverage with respect to the number of sensors. The figure shows an incremented trend because of the introduction of more number of sensors. From the graph, it is also clear that in order to achieve full coverage a minimum number of sensors are required. And it is evident that the number of sensors provided is satisfactory to cover the proposed sensing area. Further, the PSO method should ensure that it must place the sensor in such a way that no sensor range gets overlapped, which guarantees full utilization of the sensor capability.

Fig. 6.6(B) here gives a comparison of the ROI and the cumulative sensing coverage of the sensors. From the figure, it is clear that increasing the size of ROI basically decreases the coverage area for a constant number of sensors. It is a fact that the sensors must have limited coverage, so if we increase the ROI amount for a constant sensor the decrement of the coverage is obvious. PSO-Voronoi mechanism, in this case, produces a result almost close to that of the ideal case.



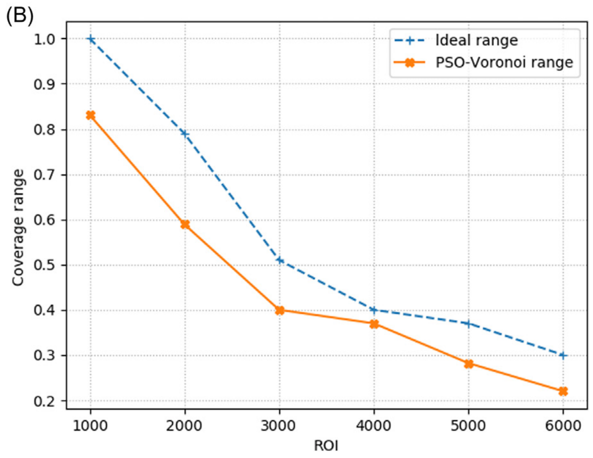


Figure 6.6 (A) Coverage range versus number of sensors for ideal and particle swarm optimization-Voronoi method. **(B)** Region of interest versus sensing coverage for sensors.

6.4.2 Node deployment using ACO

The foraging behavior of the ant species that is their way of collecting food can be mimicked into an optimization problem popularly known as ant colony optimization. It is a meta heuristic-based approach that is inspired from the way the ants search and collect food from an area; this process is called the foraging behavior of the ant species. The ants communicate with each other by pheromones. The ants use a probabilistic approach of whether to follow the path or not [6]. The path which has

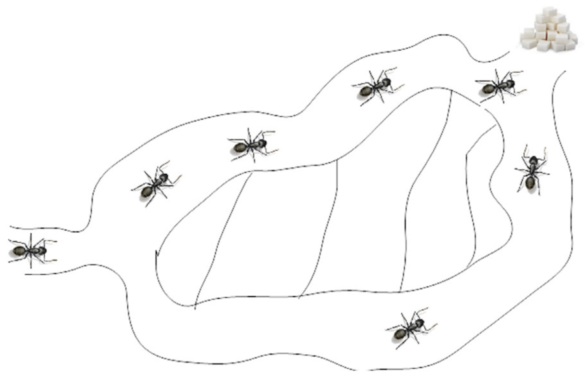


Figure 6.7 Foraging behavior.

more pheromones has a higher chance to be considered in processing. Deneuboroug et al. have investigated the pheromone behavior of ants by proposing and experimenting the double bridge as depicted in Fig. 6.7.

There are two bridges from the ants' nest to a food source of equal length as well as unequal length as depicted in Fig. 6.7. As the ants traverse, they release pheromones along the paths. Initially, we can consider the two bridges are of equal length. We will be able to observe that the ants choose the path randomly along the way as more and more ants traverse and release pheromones along the path. The path with most pheromones count is usually opted by the later ants. The optimal path is selected as the system progresses. The above Fig. 6.7 can be modeled into a graph as shown in Fig. 6.8. Let the probability of transition of ants from a particular node j to node i is p_{ij}, which is calculated using a heuristic information H_{ij} with respect to the trail of pheromones K_{ij} of the move, where $i,j = 1, 2, \ldots. n$.

$$pij = \frac{K_{ij}^{\propto} H_{ij}^{\beta}}{\sum_{k \in \text{allowed}} K_{ik}^{\alpha} H_{ik}^{\beta}} \tag{6.5}$$

The path with more pheromone value is the one best to be selected. Ant colony optimization (ACO) is widely used in various assignments and routing problems, but it can also be modeled to optimize the sensor node deployment and perform load balancing in a sensor network.

There are various types of ACO procedures, and one of the popular procedures is the Max—Min system where the pheromone trail amount is restricted ($K_{\min}$ to $K_{\max}$), $K_{\max}$ is the upper bound and $K_{\min}$ is the lower

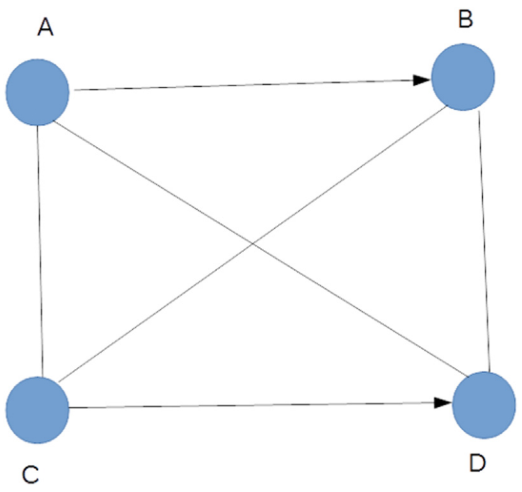


Figure 6.8 Graphical representation.

bound. Authors Findanova et al. have depicted the use of such procedure in sensor node deployment. A strong exploration of the search space for sensor node deployment is carried, which is done by allowing a single ant to add pheromone at each iteration level. After the very first iteration, the pheromone trail is initialized to $K_{\max}$, *for* the next iteration the transition performed for the best solution receives a pheromone. The pheromone update rule is given as follows:

$$K_{ij} = \rho K_{ij} + \partial K_{ij} \ \ (\mathrm{A}) \tag{6.6}$$

$$\partial K_{ij} = \frac{1}{C(V_{\text{best}})} if \ \ i,j \in \text{best solution} \ \ 0 \ \ \text{otherwise} \tag{6.7}$$

Here ∂K_{ij} is the small change in the pheromone level and V_{best} is the best solution result and $i,j = 1, 2\ldots, n$, $\rho \in$ [0, 1] models evaporated in nature. The WSN layout is considered as a graph $\{g_{ij}\}_{N\,x\,M}$ that is preinitialized. The location site is represented as $\mathrm{P} = \{P_{ij}\}_{NxM}$. The initial value of the pheromone is usually taken a very small value. The point where the base station is located and where the gathered data are transferred from each of the coverage sensors are included in the solution of the optimization process as the first point.

The ants create the rest of the solution from a random node that communicates with the base station. The target points that need to cover can be thought of as the ACO target point. The ant traverses with each of the node randomly at every iteration. A probabilistic next step calculation is performed by each of the ants with respect to the pheromone value. They have addressed the WSN deployment problem by ensuring full coverage and connectivity as constraints, while the objective function is the number of the sensors hence accordingly the ACO is used for optimization.

6.5 Intelligent routing overview

6.5.1 Routing basics

One of the most crucial factors in WSN is energy. Node deployment as we have studied plays a vital role in increasing the lifetime of the sensors nodes. Similar to node deployment, routing of data and information is another key aspect where WSN designers should look into. Routing in WSNs is very challenging as they are different from other wireless networks like mobile ad hoc networks or cellular networks. Furthermore, a WSN contains a large number of sensor nodes, and it is not possible to build a global addressing scheme for the deployment of a large number of sensor nodes as ID maintenance overhead is high. Thus, traditional IP-based protocols may not be applied to WSNs. The overview of routing protocols in WSN has been discussed in Chapter 3, Wireless sensor network: principle and the application. It has been observed that among the routing methods, hierarchical routing best addresses the energy issue in WSN properly with respect to the rest of the methods. A routing problem can be modeled into a graph *G (V, E)*, where each node can be considered as sensor nodes. The algorithms can be executed in a simulated environment to check for its feasibility before its actual use.

In a hierarchical-based approach, the network is divided into clusters. A novel hierarchical-based method has been discussed in [23] by Kiani et al. An overview of clustering achieved in a sensor network can be portrayed using Fig. 6.9. At the very beginning of the routing process, every node knows that to which cluster it belongs. Every cluster contains a main node or cluster head node (CH node) whose duty is to collect the information gathered by the clustered nodes from its respective cluster toward the main base station. An algorithm proposed for energy-efficient routing in [23] called the FTIEE algorithm. In the FTIEE algorithm, the

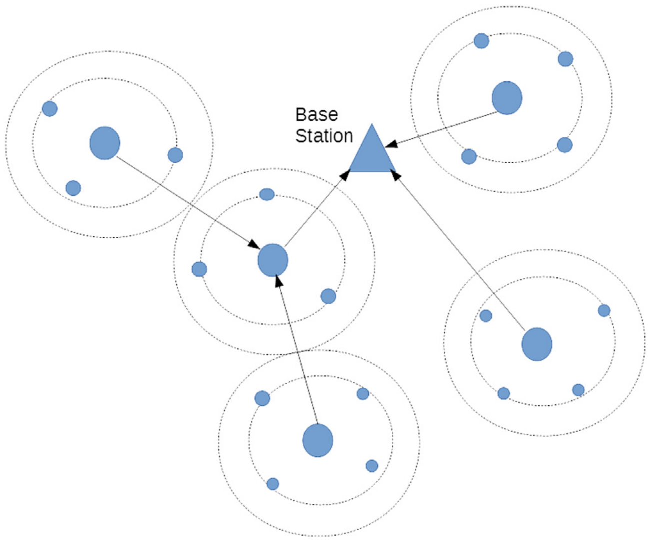


Figure 6.9 Clustering in wireless sensor network.

selection of CH node is done with the help of a machine procedure. The machine learning model used is a genetic algorithm [24] with reinforcement learning [20]. Genetic algorithm is a popular evolutionary search technique that simulates the process of natural selection. This means that the best genes from the fittest parent propagate through generations. One can have a good overview of the genetic algorithm and its application from [21]. Reinforcement learning, on the other hand, is an area in machine learning, which is quite different from the supervised learning methods. Each supervised learning methods has its own true values or confusion matrix that contains the set of correct predicted values that helps in identifying the accuracy of the classifier. Reinforcement learning, on the other hand, does not have any such table, and it just decides what to do in order to perform the given task. In the absence of the training dataset, it is bound to learn from experience. In [23] they have modeled the problem as a graph problem a new clustering technique is applied. Nodes that are closer to the base station form small clusters, while the cluster size increases as one progress away from the bases station. Each of the clusters is assigned an ID to uniquely identify the cluster. FTIEE uses a reinforcement learning approach that is called the Q-learning technique [25]. The algorithm learns the optimal CH node that has the shortest path and cost to the base station. Each of the sensors learns on their own and chooses its actions for data transmission to a neighbor or selects himself as CH node. A next-hop selection for transmission of packets is done in each cluster. If the next hop is selected within the same cluster, then it will buffer received packets for a determined period of time and send them to the base station. A reward system is indicated by the Q-value. It is calculated by two parameters. The first parameter is the distance of the respective nodes to its base station, which is used in reducing the overhead. The second is based on the residual energy of a node that is done for delay control in the whole network. These are the cost functions used in the FFIEE protocol. More insight into the FFIEE protocol can be learned from [23].

6.5.2 SDN overview

Traditional networking uses integrated hardware and software to direct traffic across a series of routers. The original usage of SDN was to virtualize the network by separating the control plane that manages the network from the data plane where traffic flows. There is a smart controller running the specialized software that manages all network traffics in data centers and a series of routers and switches that forward the packets. Today SDN [26] has so many other applications in the field of IoT.

The software-defined network provides network management in a centralized point called the controller. It provides a programmable behavior by removing the rigidity of the static protocols. The administrator has centralized and programmable control of network traffic without making modification in the hardware. Software-defined WSN or SDWSN is a very new topic of research where the relay of information during convergecasting operation in a network is guided by the controller inside the base station to take the best route for information relaying without making any modification in the hardware system. In order to achieve SDN services in WSN platform the first thing that needs to be done is to cluster the nodes. Each cluster will have a CH. The routing decisions are not made by the sensors in the network; instead, the packets are forwarded to the next CH. Each of the CHs has a controller that makes the decision from location information of the sensors when finding the best route. More insight into the SDWSN design can be taken from [27].

6.6 Sensor data mining

Data are generated every second in our environment. The universe itself is an infinite sea of data repository, the only thing that is important is the processing, interpretation, and organization of data in a structured manner when it finally becomes information. Mining data falls under the intersection of machine learning, statistics, and database systems. With the technological advancement sensors are becoming low power battery-operated devices and the sensor network deployment is done greatly according to various application needs, such as object tracking, habitat, environment monitoring, and forest management. Sensors are also greatly used in biomedical engineering as biosensors for monitoring heart rate, blood pressure temperature, and blood glucose level, and play a great role in patient monitoring and various medical diagnostics.

6.6.1 Accelerometer sensor data classification: a case study

Human activity sensing is a large research domain in the field of ubiquitous computing [28,29]. The classification of such activity has also been useful to study human activity and behavior and of course in the field of pervasive healthcare application development. In this section, a typical accelerometer data classification has been discussed as a case study. Nowadays, several body sensing devices are available in the market that comprises three-axis accelerometer. In this case, a movement of the body is detected using the chest-mounted accelerometer mounted in the wrist. The movement is classified into several categories by using random forest classifier. Testing and training accuracy has been measured. A three-axis accelerometer is basically a piezoelectric system. It comprises microscopic crystal structures that are stressed by the acceleration or the gravitational forces. The standard accelerometer device is a three-axis system and the working bandwidth is in the range of 50–200 Hz. Generally, two types of accelerometers can be used, either analog or digital. In the case of analog accelerometer, the measured value is in the form of an analog voltage level, which must be converted into a digital signal using A/D converter. A digital accelerometer, on the other hand, produces a pulse-width modulation (PWM) pulse. The pulse is then conditioned by using proper calibration methods and stored as a physical quantity.

The raw sensor data directly cannot be used for activity pattern reorganization. In order to purse the classification feature variable selection is a major task. It is important here to grab those features that are having high information. The popular and perhaps the best approach is to take a signal average of the DC components from the data samples for each axis data frame. These components are well suited and appropriate for the measurement of the orientation of the body within space. The coefficient of correlation is also a crucial feature component that can be obtained by the dot product of each frame vector. In the case of high-dimensional feature space, the classifiers' learning parameter is difficult to determine. This is most crucial when the size of the training set very small. It is also considered for a dataset that the ratio between the training and the test dataset in the feature space must be 10:1.

In this case, the future vector has been created with a sampling frequency of 52 Hz. The total number of participants in this case is 15, and the total number of activities is equal to 7. Based on participants the dataset has been segregated. There are four parameters corresponding to the accelerometer orientation, and has been recorded, namely, *x_orientation*, *y_orientation*, *z_orientation* and the status label for the activities where the label has been numbered in the following order.

1. Working on PC
2. Walking, going up and down the stair and standing up
3. Standing up
4. Walking
5. Going up/downstairs
6. Walking
7. Talking and standing.

6.6.2 Random forest classification

Random forest classifier [30] has been chosen in this case to classify the sensor data. Random forest (RF) is a statistical learning algorithm that produces an ensemble of the decision tree. The advantage of this technique is that the classification required a simple training and tuning steps. The set of classification tree builds is based on recursive binary split. A randomly chosen input variable subset is then considered in order to find out the best split. In this case, random forest classifier module of scikit learn has been used. The procedure for the classification involves the following steps.

Algorithm 3 Random forest classification on accelerometer data

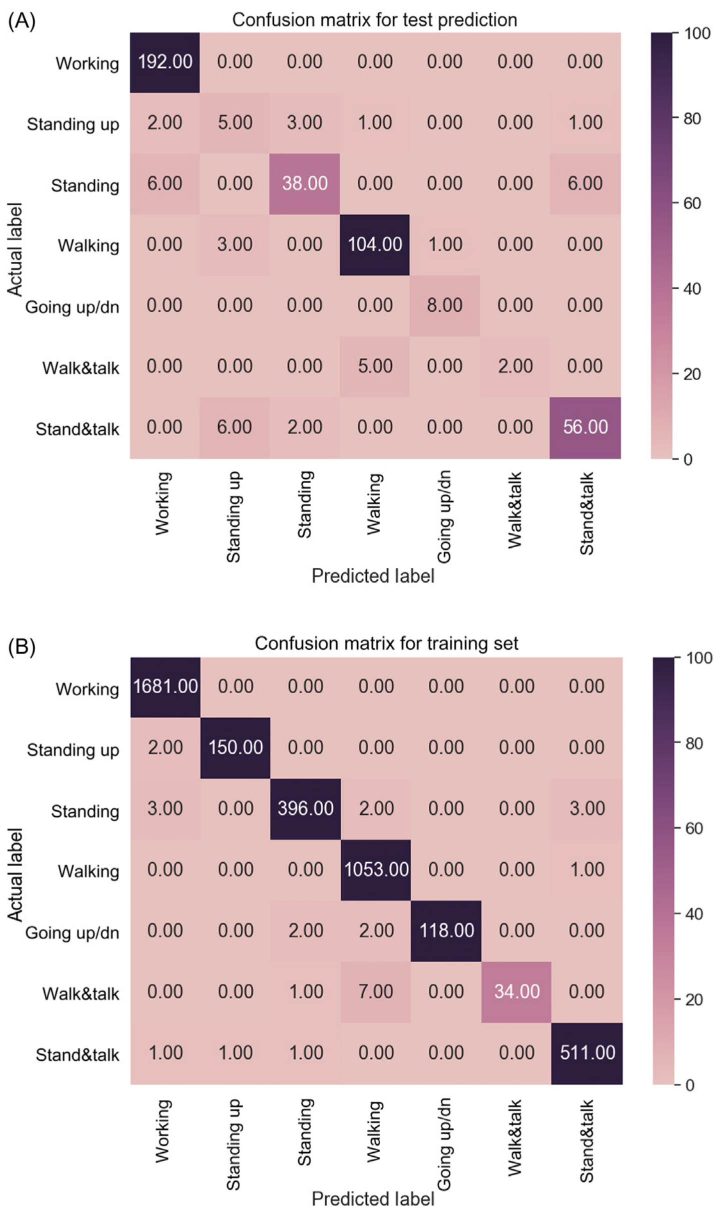
```
Import ← pandas, scikit learn, scipy, and numpy module
Window_size ← 52 (no. of subtrees for accuracy)
Input: Feature vector ("Serial," "x-axis," "y-axis," "z-axis," "activity")
Output: Test and train accuracy, actual and predicted outcome

Step1: dataset ← read Dataset and add header
Setp2: am ← $\sqrt{x} + \sqrt{y} + \sqrt{z}$
Step 3:avg ← mean (x,y,z)
Step 4:std ← std (x,y,z)
Step 5: max ← max (x,y,z)
Step 6: min ← min (x,y,z)
Step 7:cleaned_dataset ← vertical_stack(am, avg, std, max, min)
Step 8: datframe ← generate_dataframe(cleaned_dataset)
Step9:train_x, test_x, train_y, test_y ← train_test_split(dataset[feature_headers], dataset[target_header])
Step 10:trained_model ← random_forest_classifier(train_x, train_y)
Step 11:predictions ← trained_model.predict(test_x)
Step12: output1 ← print (accuracy_score(train_y))
Step13: output2 ← print (accuracy_score(test_y))
Step14: output3 ← print (accuracy_score(test_y))
Setp15: generate confusion matrix.
Step 16: End
```

The confusion matrix for the test and the training feature set has been depicted in Fig. 6.10(A,B). From the figure, it is clear that the working feature category having the highest recognition rate for both test and the train feature set and it is followed by the feature walking.

It is also noticeable that many erogenous classifications occur here, such as feature walking, going up and down, and walk and talk due to the high occurrence of misclassification. The categories such as working and standing, walking and walk and talk, with 3% and 7% confusion rate respectively. The features such as standing up and walking, going up/down and standing, and going up/down and standing here show a confusion rate of 2%.

The testing and training accuracy of the model has been depicted in Fig. 6.11.

From the figure it clear that the training has been done with an accuracy percentage of almost 99%. And the value is 0.9954648 in this case.



(A) Confusion matrix for test prediction

Actual label \ Predicted label	Working	Standing up	Standing	Walking	Going up/dn	Walk&talk	Stand&talk
Working	192.00	0.00	0.00	0.00	0.00	0.00	0.00
Standing up	2.00	5.00	3.00	1.00	0.00	0.00	1.00
Standing	6.00	0.00	38.00	0.00	0.00	0.00	6.00
Walking	0.00	3.00	0.00	104.00	1.00	0.00	0.00
Going up/dn	0.00	0.00	0.00	0.00	8.00	0.00	0.00
Walk&talk	0.00	0.00	0.00	5.00	0.00	2.00	0.00
Stand&talk	0.00	6.00	2.00	0.00	0.00	0.00	56.00

(B) Confusion matrix for training set

Actual label \ Predicted label	Working	Standing up	Standing	Walking	Going up/dn	Walk&talk	Stand&talk
Working	1681.00	0.00	0.00	0.00	0.00	0.00	0.00
Standing up	2.00	150.00	0.00	0.00	0.00	0.00	0.00
Standing	3.00	0.00	396.00	2.00	0.00	0.00	3.00
Walking	0.00	0.00	0.00	1053.00	0.00	0.00	1.00
Going up/dn	0.00	0.00	2.00	2.00	118.00	0.00	0.00
Walk&talk	0.00	0.00	1.00	7.00	0.00	34.00	0.00
Stand&talk	1.00	1.00	1.00	0.00	0.00	0.00	511.00

Figure 6.10 **(A)** Confusion matrix for test prediction. **(B)** Confusion matrix for the training feature set.

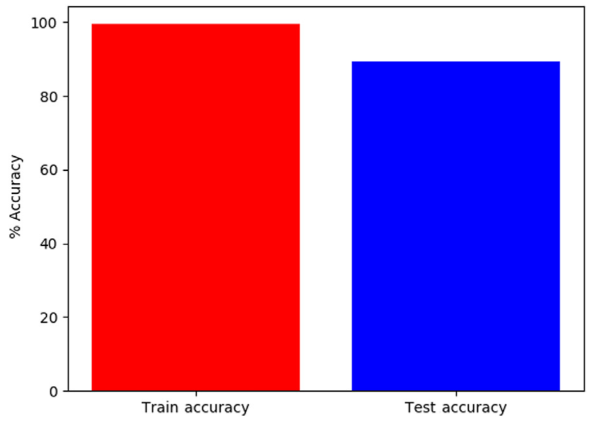


Figure 6.11 Train versus test accuracy for accelerometer data classification.

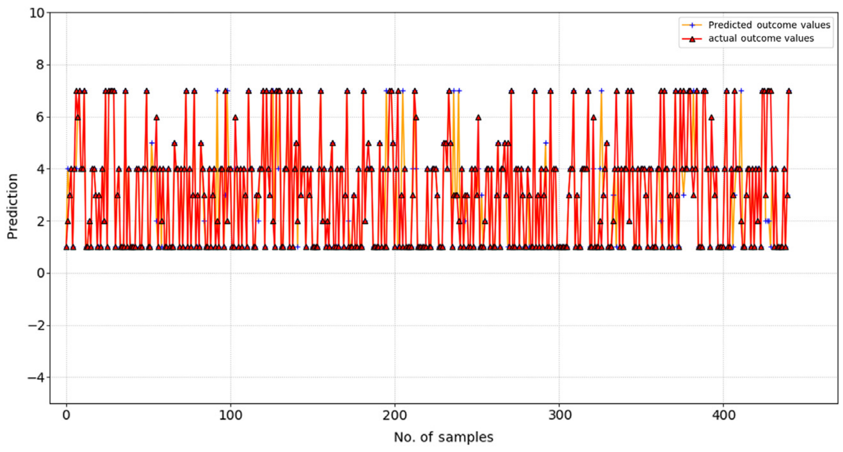


Figure 6.12 Actual versus predicted outcome values for the number of samples.

Whereas in the test case the accuracy reaches up to 90% and the value, in this case, is 0.907029478.

Actual versus predicted outcome values over the number of samples have been illustrated in Fig. 6.12. Here the comparison between the actual and the predicted outcome gives an essence of how accurately the model predicts the features. From the figure, it is quite clear that the predicted outcome values for the samples are almost the same in case of actual and the predicted, which therefore illustrates enough satisfactory prediction accuracy.

6.7 Intelligent sensor network applications

Sensor localization is the technique of figuring out the location of the sensors nodes in WSN platform. In a WSN, localization [31] of a moving target is a crucial issue. The fundamental objective of the localization method is to compute the positions of the moving node in a two or three-dimensional space. A wide range of application of localization can be considered, such as surveillance and tracking, smart city, home automation, health care, pollution control, and many more. The major component of the localization of the sensor node is to utilize the knowledge of the location from the small subset of the nodes often called anchor nodes. The localization is a one-time process for the stationary nodes for a sensor network. In case of a dynamic wireless network, the localization is more complicated and required precise tracking of the target points. There are various localization techniques that can be considered in this context such as assumption based, closest point-based, Monte Carlo, and machine learning approach of localization. Among them, machine learning-based approach has been considered as the primary topic of interest here.

The main advantage of the machine learning-based localization is the improved performance in an unfavorable weather condition too. There are several mathematical models that can be applied for this. Some of the most popular and perhaps effective techniques are:

1. SVM-based localization
2. Extended Kalman filter (EKF)-based localization.

SVM-based localization [32] in a two- and three-dimensional plane is supposed to be the most effective technique for target tracking in case of moving nodes. The principle of SVM is based on structural risk minimization philosophy. SVM technique also has superior classification performance. The fundamental concept involved is to the mapping of the data in a higher dimension Euclidian space using nonlinear mapping methodology. Consider x is an arbitrary test dataset. By using SVM we can say that x can be classified in the following way:

$$f(x) = sgn\left\{\sum_{i=1}^{l} \alpha_i y_i K(x_i.x) + \beta\right\} \tag{6.8}$$

Where x is the support vector and y is the class label for x whose value can be determined by $\in(-1,1)$. The value $K(x_i.x)$ is the kernel function and α_i is the Lagrange parameter and β is the classification threshold.

sgn{} here acts as symbol-function. The two classes linear SVM is consist of a hyperplane H and the parallel plane of the hyperplane are H1 and H2, respectively. The margin between two hyperplanes H1 and H2 is always 2/||w||.

In order to model the localization consider N nodes $\{T_1, T_2, T_3, \ldots T_k\}$ deployed in the 200×200 square meter area. We consider $\{F_1, F_2, \ldots F_a\}$ are the set of anchor nodes. In this case, typically the number is 16. The total number of stationary unknown node positions here is 196. Fig. 6.13 depicts the anchor nodes and stationary positions. Each node, in this case, can estimate the distance between other anchor nodes by maintaining a distance vector. Now consider a dynamic node that moves in the area with a specific trajectory and guided by the anchor nodes. The trajectory of the dynamic node follows the motion model, which is based on EKF of order 1.

In the above scenario, the fundamental problem definition is to predict the trajectory of the dynamic node by applying the knowledge of the predefined trajectories as the model learns from the set of trajectory datasets.

Fig. 6.14(A,B,C) depicts three different trajectories of the dynamic node. There are two different techniques that have to be applied in order to get the predicted path. In the first case, the SVM-based prediction is considered and in the second case SVM and the EKF-based trajectory prediction has been introduced. Before we go deeper into the application let us first generalize the concept of Kalman filter. Kalman filtering is an efficient way to measure the best estimate from the noisy data. It cleans

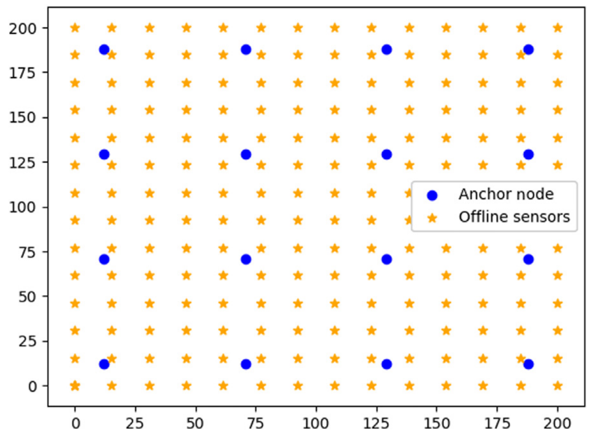


Figure 6.13 Network model in a 2D plane.

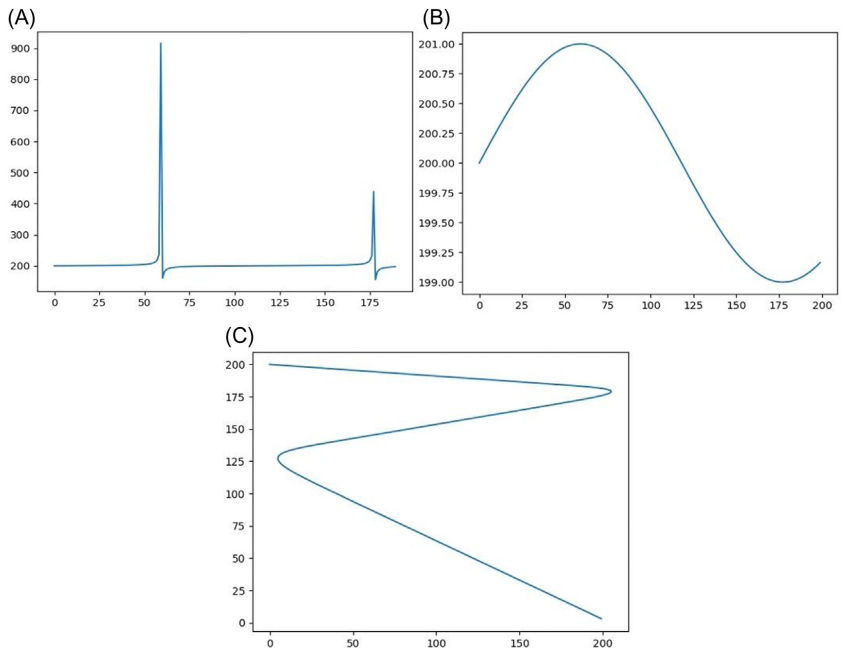


Figure 6.14 (A,B,C) Predefined trajectory set for the dynamic node.

up the data measurement and projects it to a state estimate. Kalman filter is also considered as an optimal mean square error filter. Kalman filter is highly used to estimate the systems state that cannot be measured directly. The Kalman filter works over the whole range of all possible combinations of position and velocity. Most of those samples having a high chance that they are true and some of them are more likely than others. Kalman filter takes the sample values as Gaussian distributed quantity and each value must have a mean and covariance. The main goal of the Kalman filer [33] is to compress the valuable information from the uncertain measurement as soon as possible. In our case, we have considered to follow the trajectory and find out the minimum error path from start to the end. In order to achieve that the following EKF algorithm will perform its task

Algorithm 4 EKF

```
1. Set initial_position_vector ⊛ 0
2. Accleration_noise ⊛ input_noise
3. Error_covarience ⊛ input
4. Settransition_matrix ⊛ [[1., 0], [0, 1]]
5. Setinput_matrix ⊛ [[dt, 0], [0, dt]]
6. SetMeasurement_Matrix ⊛ [[1, 0], [0, 1]]
7. Setinitial_estimate ⊛ [[init_pos[0]], [init_pos[1]]]
8. initial_estimation_covarience ⊛ Acceleration_noise$^2$ × ([[dt$^2$, 0], [0, dt$^2$]])
9. if:initial_position = 0 then
10. Initial_position ⊛ duration[0]
11. else:
12. position_change ⊛ Measurement_Matrix × initial_estimate
13. while(length(duration)):
14. recent state estimate to the present time
15. initial_estimate ⊛ transition_matrix × initial_estimate + input_matrix
16. inno_vector ⊛ Accleration_noise - initial_estimate
17. inno_covarience ⊛ Measurement_Matrix × initial_estimation_covarience ×
    Measurement_Matrix$^T$ × error_covarience
18. Kalman_Gain ⊛ transition_matrix × initial_estimation_covarience ×
    Measurement_Matrix$^T$ × inv(inno_covarience)

19. initial_estimate ⊛ initial_estimate + Kalman_Gain × inno_vector
20. initial_estimation_covarience ⊛ (transition_matrix
     × initial_estimation_covarience ×
    transition_matrix$^T$) − (transition_matrix
     × initial_estimation_covarience × Measurement_Matrix$^T$)
     × inv(inno_covarience) × Measurement_Matrix
     × initial_estimation_covarience × transition_matrix$^T$
21. returninitial_estimation_covarience
22. Ends
```

Fig. 6.14(A,B,C) illustrates the performance of the prediction for both SVM-based and SVM + EKF-based trajectory mapping techniques. From the figures, it is clear that trajectory mapping for the third trajectory is better than first and the second as shown in Fig. 6.15(C). In the case of SVM, the data points have been predicted in a scattered way along the side of the actual trajectory. Whereas the *SVM + EKF* technique tries to a virtual predicted path based on the optimal data point obtained from the SVM predictions. In the case of the second trajectory (Fig. 6.15(B)), the prediction is almost precise except the case of spikes.

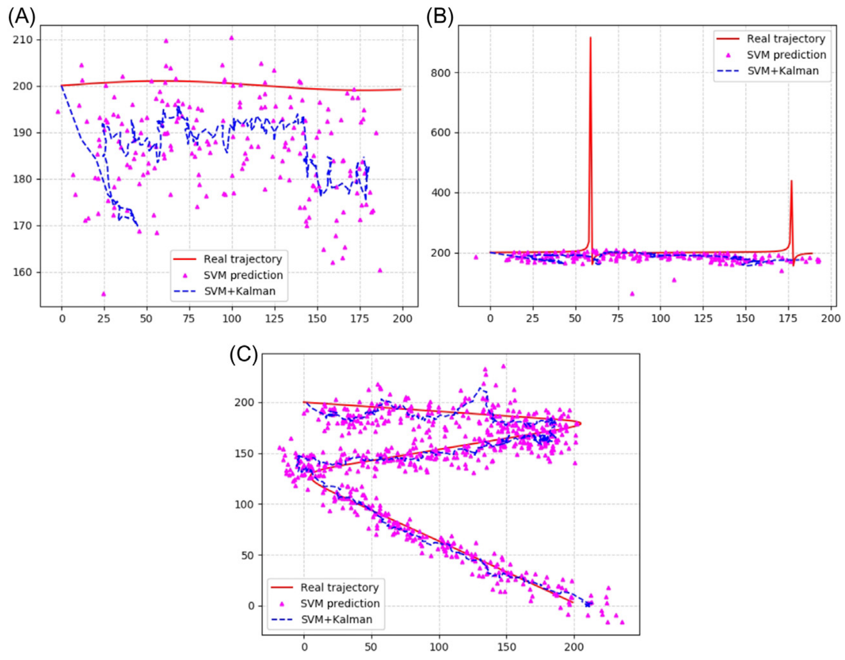


Figure 6.15 (A,B,C) Trajectory prediction comparison with actual, SVM-based, and SVM + extended Kalman filter based. *SVM*, support vector machine.

In the case of the third trajectory, both of the techniques have been performed as the SVM predicts the data that points almost near to the trajectory, and the SVM + EKF builds the path based on an optimal point related to the actual trajectory.

Fig. 6.16 shows the learning pattern of SVM with E-Kalman filter. From the graph, it is clear that the mean absolute error for learning is gradually decreasing as the progress of the trajectory because here as the time increases the number of samples supplied by the sensor to obtain the trajectory also increases. Therefore, this result shows a significant improvement of the learning pattern in the case of WSN node localization in the case of moving node scenario.

6.8 Summary

In this chapter, we have discussed several intelligent approaches to sensing. Further, some of the real-life applications such as accelerometer

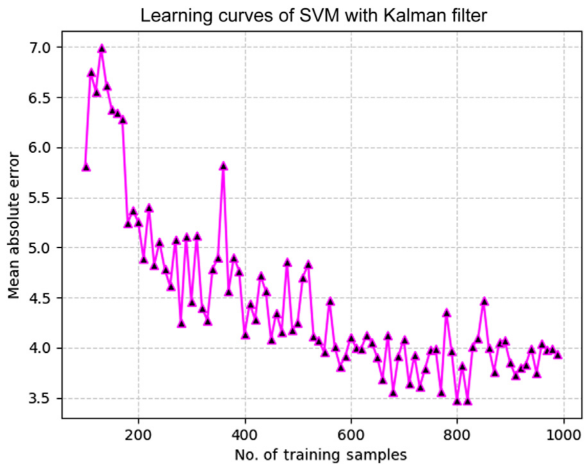


Figure 6.16 Trajectory learning curve for SVM with E-Kalman filter.

data acquisition and classification using a well-known classification model have been implemented. Nature-inspired sensing, which is an emerging topic in the field of intelligent sensing, has also been discussed. Further a classical sensor localization problem has been addressed in order to show the real-life application of the intelligent system to predict and identify the movement trajectory of the mobile sensor node in a particular target area. The topic will surely impact new researchers to get the insights of intelligent sensing and its applications.

References

[1] L. Atzori, A. Iera, G. Morabito, The internet of things: a survey, Comput. Netw. 54 (15) (2010) 2787–2805.
[2] J. Schmalzel, F. Figueroa, J. Morris, S. Mandayam, R. Polikar, An architecture for intelligent systems based on smart sensors, Proceedings of the 21st IEEE Instrumentation and Measurement Technology Conference (IEEE Cat. No. 04CH37510), Vol. 1, IEEE, 2004, pp. 71–75.
[3] F.J. Pierce, T.V. Elliott, Regional and on-farm wireless sensor networks for agricultural systems in Eastern Washington, Comput. Electron. Agric. 61 (1) (2008) 32–43.
[4] M. Naderan, M. Dehghan, H. Pedram, Sensing task assignment via sensor selection for maximum target coverage in WSNs, J. Netw. Comput. Appl. 36 (1) (2013) 262–273.
[5] A. Sheth, Internet of things to smart iot through semantic, cognitive, and perceptual computing, IEEE Intell. Syst. 31 (2) (2016) 108–112.
[6] D.G. Kleinbaum, L.L. Kupper, K.E. Muller, A. Nizam, Applied regression analysis and other multivariable methods, Vol. 601, Duxbury Press, Belmont, CA, 1988.
[7] E. Schöneburg, Stock price prediction using neural networks: a project report, Neurocomputing 2 (1) (1990) 17–27.

[8] D. Birant, A. Kut, ST-DBSCAN: an algorithm for clustering spatial–temporal data, Data Knowl. Eng. 60 (1) (2007) 208–221.
[9] R. Cheng, G.W. Milligan, Mapping influence regions in heirarchical clustering, Multivar. Behav. Res. 30 (4) (1995) 547–576.
[10] J.C. Bezdek, R. Ehrlich, W. Full, FCM: the fuzzy c-means clustering algorithm, Comput. Geosci. 10 (2-3) (1984) 191–203.
[11] Wagstaff, K., Cardie, C., Rogers, S., & Schrödl, S. (2001, June). Constrained k-means clustering with background knowledge. In Icml (Vol. 1, pp. 577–584).
[12] Rish, I. (2001, August). An empirical study of the naive Bayes classifier. In IJCAI 2001 workshop on empirical methods in artificial intelligence (Vol. 3, No. 22, pp. 41–46).
[13] Rowley, H.A. (1999). Neural network-based face detection (No. CMU-CS-99-117). Carnegie-Mellon Univ, Pittsburgh, PA, Dept of Computer Science.
[14] S.R. Safavian, D. Landgrebe, A survey of decision tree classifier methodology, IEEE Trans. Syst. Man, Cybern. 21 (3) (1991) 660–674.
[15] R. Camacho, R. King, A. Srinivasan (Eds.), Inductive logic programming: 14th International Conference, ILP 2004, Porto, Portugal, September 6–8, 2004, Proceedings, Vol. 3194, Springer Science & Business Media, 2004.
[16] C.C. Chang, C.J. Lin, LIBSVM: a library for support vector machines, ACM Trans. Intell. Syst. Technol. (TIST) 2 (3) (2011) 27.
[17] G. Bradski, A. Kaehler, Learning OpenCV: computer vision with the OpenCV library, O'Reilly Media, Inc, 2008.
[18] T. Ahonen, A. Hadid, M. Pietikainen, Face description with local binary patterns: application to face recognition, IEEE Trans. Pattern Anal. Mach. Intell. 12 (2006) 2037–2041.
[19] G. Acampora, D.J. Cook, P. Rashidi, A.V. Vasilakos, A survey on ambient intelligence in healthcare, Proc. IEEE 101 (12) (2013) 2470–2494.
[20] O. Younis, M. Krunz, S. Ramasubramanian, Node clustering in wireless sensor networks: recent developments and deployment challenges, IEEE Netw. 20 (3) (2006) 20–25.
[21] X. Yuan, M. Elhoseny, H.K. El-Minir, A.M. Riad, A genetic algorithm-based, dynamic clustering method towards improved WSN longevity, J. Netw. Syst. Manag. 25 (1) (2017) 21–46.
[22] C. Xiong, D. Chen, D. Lu, Z. Zeng, L. Lian, Path planning of multiple autonomous marine vehicles for adaptive sampling using Voronoi-based ant colony optimization, Robot. Auton. Syst. 115 (2019) 90–103.
[23] F. Kiani, E. Amiri, M. Zamani, T. Khodadadi, A. Abdul Manaf, Efficient intelligent energy routing protocol in wireless sensor networks, Int. J. Distrib. Sens. Netw. 11 (3) (2015) 618072.
[24] S.M. Zin, N.B. Anuar, M.L.M. Kiah, A.S.K. Pathan, Routing protocol design for secure WSN: review and open research issues, J. Netw. Comput. Appl. 41 (2014) 517–530.
[25] W. Guo, W. Zhang, A survey on intelligent routing protocols in wireless sensor networks, J. Netw. Comput. Appl. 38 (2014) 185–201.
[26] Q. Xu, J. Zhao, A WSN architecture based on SDN, 4th International Conference on Information Systems and Computing Technology, Atlantis Press, 2016.
[27] J. Puente Fernández, L. García Villalba, T.H. Kim, Software defined networks in wireless sensor architectures, Entropy 20 (4) (2018) 225.
[28] A. Varshavsky, S. Patel, Location in ubiquitous computing, Ubiquitous Computing Fundamentals, Chapman and Hall/CRC, 2018, pp. 299–334.

[29] G. Elhayatmy, N. Dey, A.S. Ashour, Internet of things based wireless body area network in healthcare, *Internet of Things and Big Data Analytics Toward Next-Generation Intelligence*, Springer, Cham, 2018, pp. 3–20.
[30] S.S. Chawathe, Recognizing human falls and routine activities using accelerometers, 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), IEEE, 2019, pp. 0120–0126.
[31] W. Lv, Z. Duan, X. Sun, Monte Carlo box node localization algorithm for mobile WSN based on support vector machine, *2018 13th World Congress on Intelligent Control and Automation (WCICA)*, IEEE, 2018, pp. 755–760.
[32] S. Chatterjee, S. Sarkar, N. Dey, A.S. Ashour, S. Sen, Hybrid non-dominated sorting genetic algorithm: II-neural network approach, Advancements in Applied Metaheuristic Computing, IGI Global, 2018, pp. 264–286.
[33] A. Mukherjee, N. Dey, N. Kausar, A.S. Ashour, R. Taiar, A.E. Hassanien, A disaster management specific mobility model for flying ad-hoc network, Emergency and Disaster Management: Concepts, Methodologies, Tools, and Applications, IGI Global, 2019, pp. 279–311.

Further reading

A. Mahmood, K. Shi, S. Khatoon, M. Xiao, Data mining techniques for wireless sensor networks: a survey, Int. J. Distrib. Sensor Netw. 9 (7) (2013) 406316.
T. Murata, H. Ishibuchi, H. Tanaka, Multi-objective genetic algorithm and its applications to flowshop scheduling, Comput. Ind. Eng. 30 (4) (1996) 957–968.
Sutton, R.S., & Barto, A.G. (1998). Introduction to Reinforcement Learning, vol. 2, no. 4, MIT Press, Cambridge.
K.S. Tang, K.F. Man, S. Kwong, Q. He, Genetic algorithms and their applications, IEEE Signal Process. Mag. 13 (6) (1996) 22–37.

CHAPTER SEVEN

Conclusion

Contents

The book *A beginner's approach to data agglomeration and intelligent sensing* practically serves as a guide not only to the researchers but also to the people who are new and who try to start their research in the field of sensor networks and intelligent systems. The fundamental approach of the book involves the illustrations of the basic principles of several sensor networks, intelligent sensor orientation, localization, sensor cloud, routing, and mobility modeling with some real-life examples and deployment scenarios. The flow of the chapters has been organized in such a way that from novice to professional everyone gets a good essence of the cutting edge approaches in the field of wireless sensor network (WSN) data aggregation, sensing, simulation, and deployment.

7.1 Chapters 1 and 2

Chapter 1, Introduction to sensors and system, mainly discusses the fundamental concept of the sensors. Different classification of the sensors has been addressed here. Not only various networking methodology related to sensors has been discussed, but the fundamental introduction and the working principles of several types of analog and digital sensors have also been addressed in this chapter.

Chapter 2, Real-life application of sensors and systems, on the other hand, mainly emphasizes the application level perspective of the sensors and the systems. Here the application of sensor network in the Internet of thing paradigm is a crucial subject of interest. The protocol like Message

Queuing Telemetry Transport and the related platforms such as Amazon Web Service and Google cloud have been discussed. Real-life applications and simulation platforms are also been addressed. Some of the well-known simulation environments such as omnet++ and delay tolerant network (DTN)-based opportunistic network simulator are also focused on the context of fixed and dynamic WSN design paradigms.

7.2 Chapters 3 and 4

The main goal of Chapter 3, Wireless sensor network: principle and the application, is to discuss sensor networking principles. Here the fundamental fixed architecture of the sensor networks has been addressed. Challenges in networking are also a vital issue that has been addressed here. Also, the fundamental implementation of the routing in WSN scenario is a crucial approach that has been added in this chapter. Some of the typical protocol such as MAVLink protocol and its applications in deployment, and the communication of the dynamic sensor node such as unmanned aerial vehicle (UAV) have been addressed.

Chapter 4, Overview of sensor cloud, fundamentally deals with the concept of sensor cloud. The detailed discussion of the cloud-computing models has been addressed here. Conceptualization and the deployment of the sensor cloud platform have also been discussed.

7.3 Chapters 5 and 6

The major component of Chapter 5, Sensor data accumulation methodologies, involves the classification of the data as well as the sensor data transmission methodology. Another crucial aspect of data aggregation and data collection has also been added in this chapter.

Chapter 6, Intelligent sensor network, mainly emphasizes the fundamental principle of the intelligent sensor networks where node positioning and the localization are supposed to be the crucial issues. Several case study-based approaches in the field of intelligent sensing and sensor data analysis have also been addressed in this chapter.

7.4 Scope for future enhancement

The book fundamentally serves as a beginner guide for intelligent WSN development and deployment. Various theoretical and algorithmic approaches of the WSN node deployment, communications, and packet routing have been discussed in this book. However, there are lots of enhancement scopes that are present in order to make an in-depth study on various aspects of routing protocol, node mobility, privacy, and the security of the WSN. We believe this book will surely give an essence of the intelligent approach of wireless sensor network modeling and design, and the beginner researcher would imbibe the fundamental knowledge of the deployment and data agglomeration in the sensor network scenario.

Index

Note: Page numbers followed by "*f*" and "*t*" refer to figures and tables, respectively.

A Beginner's Guide to Data Agglomeration and Intelligent Sensing

Amartya Mukherjee, Ayan Kumar Panja and Nilanjan Dey

A Beginner's Guide to Data Agglomeration and Intelligent Sensing provides an overview of the sensor data accumulation, sensor cloud platform, converge-casting, and data aggregation in support of intelligent sensing and relaying of information. The book begins with a brief introduction on sensors and transducers, giving readers insight into the various types of sensors and how one can work with them. In addition, it provides several real-life examples to help readers gain a thorough understanding of the concepts. An overview of topics such as wireless sensor networks, cloud platforms, and device-to-cloud and sensor cloud architecture are explained, as is data gathering in wireless sensor networks and aggregation procedures. Final sections explore how to process gathered data and relay the data in an intelligent way, including concepts such as supervised and unsupervised learning, sensor data mining, and smart systems. The book also discusses several simulation platforms in order to understand the fundamentals of routing and data exchange mechanisms within an intelligent sensor network.

Key Features

- Presents the latest advances in data agglomeration for intelligent sensing
- Discusses the basic concepts of sensors, real-life applications of sensors and systems, the protocols and applications of wireless sensor networks, the methodology of sensor data accumulation, and real-life applications of intelligent sensor networks
- Provides readers with an easy to learn and understand introduction to the concepts of the cloud platform, sensor cloud, and machine learning

A Beginner's Guide to Data Agglomeration and Intelligent Sensing is an ideal resource for computer and data scientists, biomedical engineers, researchers, and software engineers who are looking for a foundational overview of data agglomeration and intelligent sensing.

About the Authors

Amartya Mukherjee is an assistant professor at the Institute of Engineering and Management, Salt Lake, Kolkata, India. His primary research interest is in embedded application development, including mobile ad hoc networking, delay-tolerant networks, and Internet of things and machine learning.

Ayan Kumar Panja is an assistant professor of computer science at Institute of Engineering and Management, Salt Lake, Kolkata, India. His primary research interests include localization, sensor network, and sensor cloud architecture.

Nilanjan Dey is an assistant professor in the Department of Information Technology at Techno India College of Technology, Kolkata, India. His main research interests include medical imaging, machine learning, computer-aided diagnosis, data mining, and so on.

ISBN 978-0-12-820341-5

9 780128 203415

ELSEVIER

ACADEMIC PRESS
An imprint of Elsevier
elsevier.com/books-and-journals