Developing A Parallel Simulator For Distributed Online Gas Sensor Systems

Deniz Dural and Ahmet Özmen
Faculty of Computer and Information Science, Department of Computer Engineering, Sakarya University, Turkey.

Abstract

The design of distributed gas sensor systems becomes very complex problem due to large number of sensor counts. Distributed structure creates delays which results incorrect outputs when online measurements are intended. Testing such gas measurement or monitoring systems is time consuming and costly process. In this study, a simulator that can be used for optimization of designing distributed gas sensor systems is presented in order to simplify these problems. In the simulator, mostly used gas sensors are modeled and implemented virtually. So, users can form sensor cells by selecting known sensors from a toolbox and connect them to a central server like designing a real system by means of provided graphical interface. Each sensor is implemented as one thread in the simulator and communications between sensor cells and the central server are realized by MPI routines. In this way, both data processing and sample transfer operations have improved significantly.

Key words: gas sensors, simulator, MPI, threads.

1. Introduction

Electronic gas sensors are widely used in many applications from simple home safety to large building complexes to create energy efficient comfortable environments. Advances in electronic gas sensor systems create possibilities for distributed online gas monitoring systems. For example, air quality can be monitored in a wide area for detection of toxic or explosive gases using smart sensor networks. Also, these systems can be used for product process monitoring in the industry. In addition, these systems are quite useful in biomedical such as some diagnosis certain diseases or specific analyzes on patients [1-3].

Recently, there are many investigations about distributed gas sensor systems [4,5,6,7]. Collecting and processing of sensors data, deciding responses according to processed data immediately are very important subjects. During system design, any error can make the system unusable. Therefore, each of the design parameters that are sensors count, data collection types, data packaging formats, location of data processing and data presentation way, must be considered exhaustively. Thus, developing such systems is much more complicated than implementing single unit instrumentation. Generally, ad hoc methods have been used to build such complex systems up to now, and usually sensor developers have to do the tasks many times in order to get desired results. Building a distributed online gas sensor system and conducting tests on it becomes a daunting task. Therefore, a software simulator tool that reduces the complexity of developing such systems will be quite helpful. Through the simulator, a model can be reconfigured and tested iteratively but these steps can be done quickly [8]. In literature, there are many developed

*Corresponding author: Address: Faculty of Computer and Information Science, Department of Computer Engineering, Sakarya University, Sakarya, TURKEY. E-mail address: ddural@sakarya.edu.tr, Phone: +902642955895

software simulator for sensor networks such as Ns-2, Tossim, Emstar, Omnet++, Opnet etc. However, these tools only simulate sensor networks, networks traffics and their performance. A new protocol can't be developed. Sometimes you need to modify code for improving existing protocols. But most of them don't allow this [9,10,11].

In this study, the simulator is designed for gas sensors. Mathematical modeling of a few sensor types and the environment are developed and incorporated to the simulator. Then, some scenarios are devised to test the simulator system. In these scenarios, it is aimed to generate real like sensor responses. The system is distributed in nature and there is frequent data transfer from the sensor sources to the central server. So a new set of distributed simulator tools have been developed due to lack of such software simulator for distributed online gas systems.

The simulation study has started with modeling required components of such system. First the behavior of volatile organic components (VOC) is specified to imitate a desired target environment. Then, existing sensors are modeled and coded as a library. The other components of system were also modeled such as sensor cell and data collector. Since the rest of the system is also same with the real system, they are developed so that they can also be used for real systems. Hence, once a simulation study has completed by obtaining optimal parameters of the system, real sensor cells can be mounted to the end points instead of software counterparts.
There are interesting similarities between distributed gas sensor systems and performance data collection of distributed parallel monitoring systems. For both systems, the collected sensor data must be gathered at a central place without delay, processed and presented to the user with visual tools. The experiences obtained for parallel monitoring are carefully used in this study for developing a distributed gas monitoring simulator.

2. Distributed Gas Monitoring Simulator

Gas sensor systems consist of many distributed detectors. The detectors collect data, then collected data is processed and the results are produced as knowledge by software tools. Considering these situations, it is clear that gas sensor systems aren't different from distributed parallel systems. So the data is gathered at the center in both systems, and are processed without delay. The results are presented to users with graphical interfaces.

In this study, generated system is called as parallel online gas monitoring simulator, because each sensor is considered as a data source and they are created as a thread. Many sensors (threads) come together to generate a sensor cell. It may also be called as a node in the system. In the real life, there will be many types of sensors in the each room of a building for air quality monitoring. In a similar way, a sensor cell contains many types of sensors. So a sensor cell can be thought like a room of a building. Sensor data is collected at the center that is a server generally, for converting graphic by presentation tools. Forwarding data and sensors management from the center are provided with MPI (Message Passing Interface) routines that are MPI_Send, MPI_Recv, MPI_Barrier etc. The simulator is coded in C# language. Figure 1 depicts the architecture of the simulator.

In the system, there are no direct paths among sensors. Therefore, when increasing sensor counts, synchronization and centralized management become harder. But using MPI functions provide an advantage about system management and performance and then save us from the complexity of communication protocols. Thus, the number of sensor in the system can be increased as much as desired.

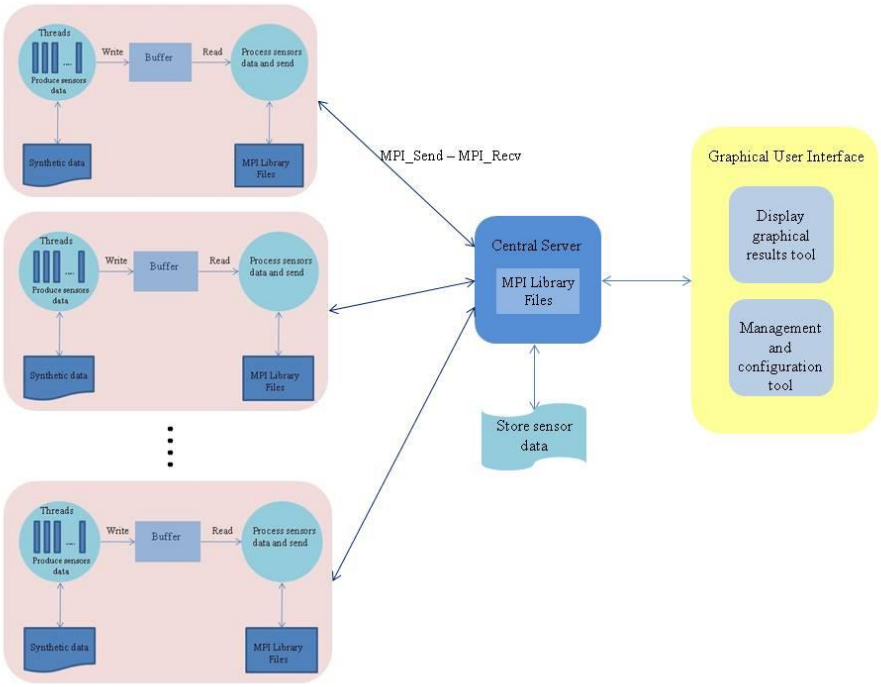


Figure 1: The architecture of the simulator system

3. Components of the System

This study is the first version of the distributed online gas sensor simulator. Firstly, characteristics of two sensor types that are sensitive to CO and CO_2 gasses are investigated and modeled as analytically. These are tin oxide gas sensors that are called Figaro TGS4161 and Figaro TGS5042 [12,13]. Their responses change based on temperature, humidity and cross effects with other gasses. Every modeling sensor types are added as a library to the software.

Secondly, the sensor cell is modeled as a geometrical shape. Commonly used geometrics as rectangular prism and cylinder are appended to the library. Also, different sensor types and models can be added to a sensor cell optionally. As mentioned above, two metal-oxide sensor types are modeled. Users can add any number and type of these sensors. In addition, different types of sensor (SAW, QCM etc.) will be modeled in the future work. So cell modeling will be more variety and reality.

Node computers and their processes represent sensor cells that are formed by users. While creating a sensor cell, each added sensors generate a thread. Namely, each sensor is thought like a thread. The purpose here is to benefit from advantages of thread's usage. The transition between

the processes is performed easier thanks to the threads. Also, obtained data are observed to be more useful because of thread synchronization.

There are one main thread and worker threads that are added by users, in the node computers. Worker threads interpret to existing synthetic data according to running principles of modeled sensors. They write obtained results to the buffer at certain time intervals. Also, the main thread works as a collector. The collector gathers data from sensors (reads data from the buffer) by way of sampling periodically. After the time and identifier are added to the collected data and it is packed. These packets are sent to central server from local nodes with MPI methods (MPI_Send, MPI_Recv, MPI_Pack etc.). The packet structure is as follows.

Sensor Cell Id	Sensor ID	Time	Temperature	Humidity	Data

In this paper, when reading data from the buffer, the collector (main thread) reads data as the number of threads at a time. It packs these data and send to the central server. When considering sampling frequency of sensors (threads) and multitude of produced data, this usage prevents the growth of the buffer and provides to catch up threads' speed. Thus, instant measurements and results can be obtained.

Packets are collected into the central server continuously. So, there is a sensor cell id at the beginning of the packets for determining where they come from. In addition, sensor id is added to the packet for finding that the data belongs to which sensor. Also, sending time of packet, temperature and humidity of the environment are added to the packet. So, more reliable results are obtained for graphical representation.

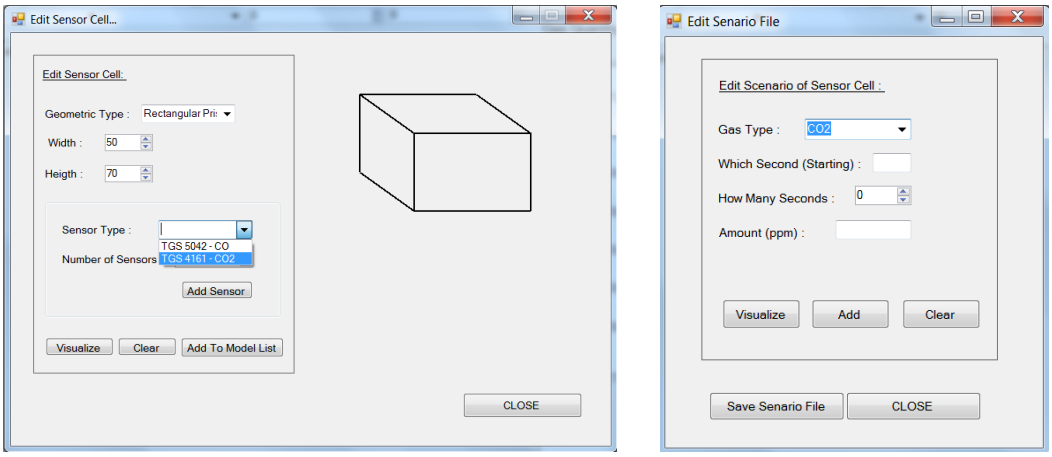
4. Configuration and Management of the Simulator

In this paper, it is assumed that many sensor cells are placed in a building. Each room of the building can be thought as a sensor cell. Sensors responses (sensor data) are sent to data collector via MPI commands and are processed in here for GUI. So, application is thought as online monitoring indoor air quality of a building.

In the simulator, users can model a sensor cell by placing any number and types of sensors in it. A sensor cell's contents must be determined by considering needs. Firstly, user can decide sensor cell's shape in the simulator. The choice of the shape is used for providing an image closer to reality. And then defined sensor types and counts are determined for placing into the cell. Finally, generated sensor cell is renamed and added to the model list. All steps are shown in Figure 2.a.

Scenarios files can be edited for generated sensor cells. Users must edit scenario file for each sensor cell (for each room). Scenarios contain circulating gas amount, gas types and time span in the environment. For example, the amount of CO (carbon monoxide) is 300 ppm between 70th second and 120th second and then dry air should be given between 121th second and 130th second. Through these file, user can do several tests on the generated sensor cells. We call these scenarios as synthetic data files that are shown in Figure 1. In the next study, as well as synthetic

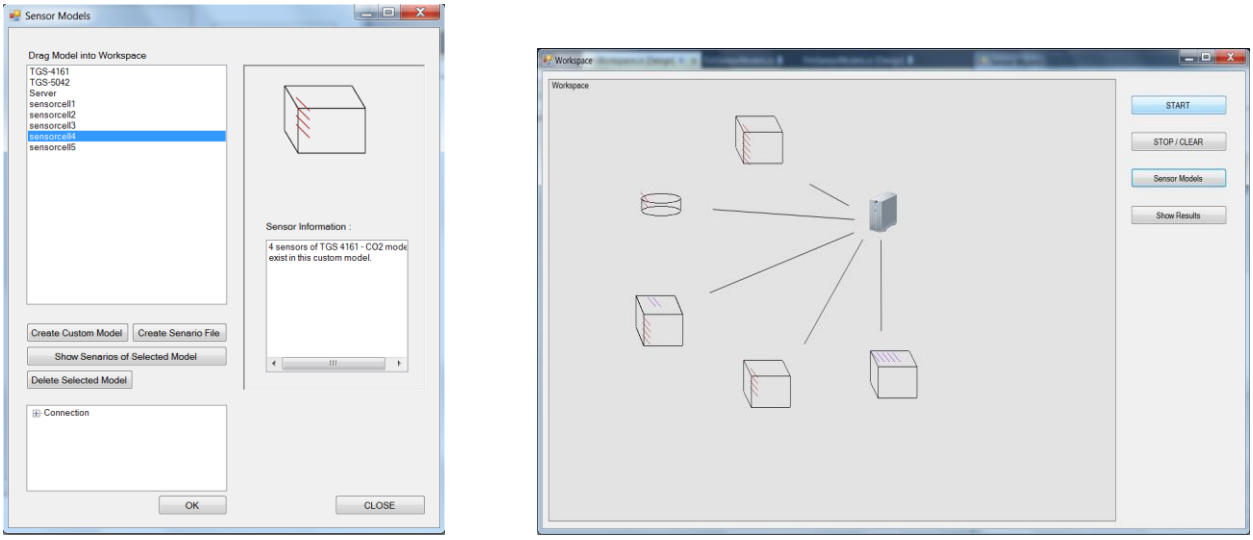
data, the simulator will make dynamic measurement. Figure 2.b. shows that these scenario files can be edited easily and realistic.



2.(a) 2.(b)

Figure 2. 2.a. Editing sensor cell in the simulator. 2.b. Editing scenario files for sensor cells.

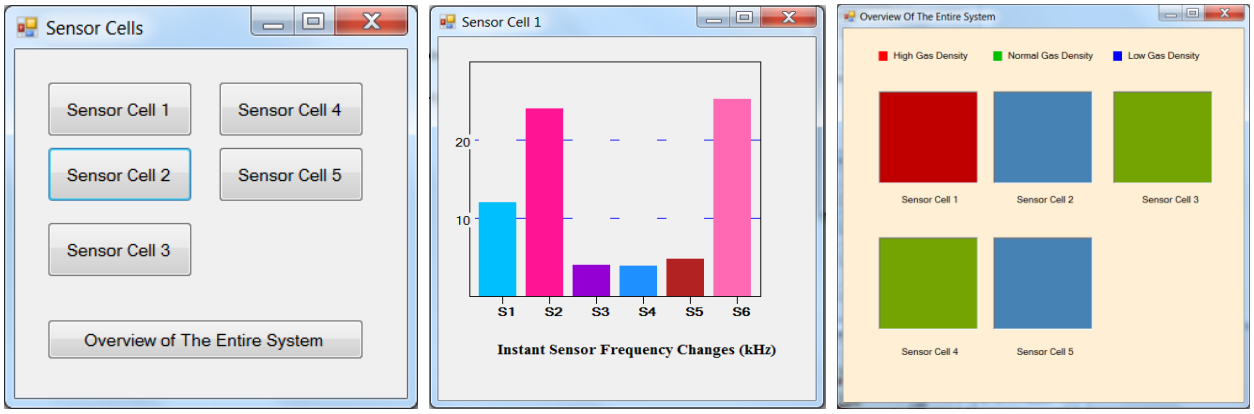
After the sensor cell is added to the model list, general information and each scenario of cells are listed that are shown in Figure 3.a. Now, modeled sensor cells and other environment elements can be dragged to workspace. Thus, realistic visualization can be created. When the simulator runs, sensors responses are calculated using scenario files and graphical outputs of each sensor cells are generated. So, users can inform about environment air quality. For example, 5 sensor cells and their scenario files are configured and dragged to the workspace. So, their distributed topology of sensor cells and central server can be viewed. Screenshot of the workspace is shown in Figure 3.b.



3.(a) 3.(b)

Figure 3. 3.a. All sensor cell models information and scenarios lists. 3.b. Topology of the configured sensor cells in the workspace

When the simulator is started, each sensor cell is produced results according to their scenario files. The result buttons are generated as the number of the sensor cells in the workspace. This is shown in Figure 4.a. Then, two different graphics are obtained. First graphic is shown in Figure 4.b and shows responses of the each sensor that are in the sensor cells. Instant sensor frequency changes are shown in there. In second graphic, gas concentrations of the rooms (sensor cells) are expressed in color by taking the average of the sensor responses. For example, in Figure 4.c., gas density is over in the sensor cell1. In sensor cells 2 and 5, the density is lower than average but in sensor cells 3 and 4, it is observed that the average level. Thus, general information can be obtained about the entire system.



4.(a) 4.(b) 4.(c)

Figure 4. 4.a. Result buttons of the all sensor cells for displaying their instant case. 4.b. Instant sensor frequency changes. 4.c. Overview of the entire system according to gas concentration.

4. Results and Discussion

The study is the first version of the developed simulator for modeling online distributed gas sensor systems. So, only TGS4161 and TGS5042 sensors that are known as tin oxide sensors are modeled and added as a library into the simulator. For future works, it is aimed that sensors which have changeable frequency ability like QCM, SAW will be modeled. In this study, different sensor types are added into sensor cells that are formed a variety of geometric shapes. And then, virtual test environments are created by writing scenarios that are very close to the actual ambient conditions. But in the next studies, the environment will be modeled close to reality and then dynamic input will be acquired besides synthetic data. In virtual ambient, different sensor types' responses are monitored. Collected sensor responses are packed and transferred to the central server. However, only a packet format is tried in this study. In next steps, effects of different packet formats onto systems working will be investigated. In study, there is data flow from sensor cells to central server continuously. If we consider increasing the number of sensor and sensor cells, some topics such as topological properties of the system, traffic congestion over the server and determining place of the data processing (in nodes or central server) must be examined. In this way, optimal designing parameters of the distributed gas measurement systems will be determined by the future versions of the simulator.

Conclusions

In this study, an online gas simulator system has been developed to improve distributed gas sensor system design and deploy process. Various existing gas sensors are implemented as software component in a toolbox. Using the simulator, firstly a user can form a sensor cell. And then different types of sensors can be added in it. Each inserted sensor works as a thread. Environment scenarios are created easily by a provided tool, so the sensors interact and generate responses accordingly as in real. And then sensor responses are sent to a collector server from the each sensor cell via MPI commands. Through the graphical tool, current status information of each sensor cell is obtained using synthetic sensors responses. So, using this simulator air quality of a building can be monitored as shown in Figure 4.

References

[1] Tran. V.H., Chan, H.P., Thurston, M., Jackson, P., Lewis, C., Yates, D., Bell, G., Thomas, P.S. Breath Analysis of Lung Cancer Patients Using an Electronic Nose Detection System. IEEE Sensors Journal, 2010; 10:1514-1518.
[2] Sieber, A., Enoksson, P., Kroze, A. Smart Electrochemical Oxygen Sensor for Personal Protective Equipment. IEEE Sensors Journal, 2012; 12:1846-1852.
[3] Postolache, O.A., Pereira, J.M., Girao, S.P. Smart sensor network for air quality monitoring applications. IEEE Transactions on Instrumentation and Measurement, 2009; 58:3253-3262.
[4] Postolache O., Pereira J.M., Girão P. S., Postolache G. Distributed Smart Sensing Systems for Indoor Monitoring of Respiratory Distress Triggering Factors. Chemistry, Emission Control, Radioactive Pollution and Indoor Air Quality, 2011; 311-330.
[5] Tsujita W.,Yoshino A., Ishida H.,Moriizumi T. Gas sensor network for air-pollution monitoring. Sensors and Actuators B, 2005; 304–311.
[6] Ivanov B., Zhelondz O., Borodulkin L., Ruser H. Distributed smart sensor system for indoor climate monitoring. 2002 KONNEX Scientific Conference, 2002.
[7] Pillai, M.A., Veerasingam, S., Yaswanth Sai D. Implementation of Sensor Network for Indoor Air Quality Monitoring Using CAN Interface. Advances in Computer Engineering (ACE) International Conference, 2010; 366 – 370.
[8] Maria, A., Introduction to Modelling and Simulation. Proceedings of the 1997 Winter Simulation Conference,1997; 7-13.
[9] Sundani H., Li H., Devabhaktuni V., Alam M., Bhattacharya P. Wireless sensor network simulators a survey and comparisons. Internatonal Journal of Computer Networks (IJCN), 2011; 2: 249-265.
[10] Lakshmanarao K., VinodKumar Ch.R., Kanakavardhini K. Survey on Simulation Tools for Wireless Networks. International Journal of Engineering Research and Technology (IJERT), 2013; 2:608-612.
[11] Chhimwal P., Rai R. S., Rawat D., Comparison between Different Wireless Sensor Simulation Tools, IOSR Journal of Electronics and Communication Engineering, 2013; 5:54-60.
[12] Datasheet website: http://www.figaro.co.jp/en/pdf/CO2GasSensorTGS4161.pdf
[13] Datasheet website: http://www.figaro.co.jp/en/pdf/5042ProductInfo0607.pdf