Web-enabled Software for Real-time Autonomous Wireless Sensors Data Visualization

Felix G. Hamza-Lup
Computer Science
Georgia Southern University
Savannah GA, USA
fhamzalup@georgiasouthern.edu

Hongjun Su
Electrical & Computer Engineering
Georgia Southern University
Savannah, GA, USA
hsu@georgiasouthern.edu

Coby Wissing
Computer Science
Georgia Southern University
Savannah GA, USA
cw09467@georgiasouthern.edu

ABSTRACT

Real-time autonomous wireless sensors must provide information that is both accurate and timely. With the widespread of IoT, a myriad of sensors are producing a large quantity of numeric data. Still, as users, we are inclined to consume data that has been processed and displayed as visual information in order to facilitate decision processes.

Web standards have evolved significantly over the past decade with X3D being adopted as an open ISO standard for Web3D. X3D and associated extensions like WebGL allows web-enabled visualization and facilitates remote collaboration.

In this paper, we present a software architecture that allows data collection from IoT sensors, data processing and an interactive Web-enabled 3D visualization using the X3D standard for soft real-time systems. We also discuss the potential applications of such a system in several domains.

CCS CONCEPTS

• Computer systems organization ~ Real-time system architecture • Hardware ~ Emerging tools and methodologies • Software and its engineering ~ Software prototyping

KEYWORDS

Real-time sensors, data visualization, Web3D, real-time software

ACM Reference format:

Felix G. Hamza-Lup, Hongjun Su and Coby Wissing 2019. Web-enabled Software for Real-time Autonomous Wireless Sensors Data Visualization. In *Proceedings of the ACM Conference ICNSER (ICNSER'19) the ACM, Shenyang, China, 5 Pages*. https://doi.org/10.1145/*********

1 Introduction

The current level of sensor deployment has elevated IoT to new levels through *sensorization*. As the micro-sensing technology takes new dimensions, we expect to see more of the things around to be sensor–tagged. Furthermore, the increasing availability and use of Real-time Autonomous Wireless Sensors (RAWS) in a variety of environments is the driving force behind home automation, driverless cars, as well as many automated industrial processes. However, when it comes to presenting data at least two more software components have to be involved: a data processing module able to extract information from a large amount of data (e.g. big data), and a presentation module able to present the data in a user-friendly format. Big data visualization allows us to take complex findings and present them in a way that is informative and engaging to all stakeholders. Moreover, visualization enables humans to understand complex analytics [1].

Focusing on the user interface component, we are proposing a soft real-time module (i.e. interactive) that is capable of displaying and converting RAWS data into a color spectrum for data visualization and interpretation. The interface provides web-based interactivity as well as playback capabilities, allowing sensor data review over large periods (i.e., up to 1 year, when coupled with long time/range autonomous sensors).

Real-time technologies generate real-time data streams and have the following characteristics [2]: in memory data storage for high-speed ingest, distributed architecture for horizontal system scalability, and real-time query-able for interactive data exploration. To satisfy all these constraints, a buffering system is usually maintained between the data producer (i.e., the RAWS) and the data consumer (i.e., the processing and display software module) not only to control the data flow but also to assure real-time display behavior.

We propose a simple yet effective, web-enabled thermal simulation system coupled with a cost-effective set of RAWS (e.g.

temperature, relative-humidity) that can provide valuable insights into sensor data interpretation. The paper is structured as follows: in Section 2 presents a brief overview on autonomous sensors, communication protocols, and the applications, Section 3 describes the implementation, Section 4 illustrates the real-time data visualization, and Section 5 discusses the experimental results.

2 Sensors Network, Protocols, and Applications

RAWS networks are deployed in many indoor/outdoor environments where they can collect data for long periods (i.e., 6 months to one year). Such sensor networks are used in many domains e.g., military, industrial, environmental, residential, and healthcare. Depending on their requirements and sensor capabilities one can define RAWS networks in terms of size (small to very large scale), sensors' capacity (homogeneous to heterogeneous), topology and mobility (static, mobile and hybrid) [3]. Agricultural applications can benefit from new, low-power network standards and platforms, as illustrated in [4]. Although LoRa and ZigBee [5] are perceived as more suitable, most implementations do not consider in their analysis IoT-based requirements such as connectivity and cost. Different RAWS deployment strategies can be adapted in this sense to solve coverage, network connectivity, deployment cost, energy efficiency, life span, data fidelity, and load balancing issues.

Two applications that we are focusing on are indoors temperature maps for energy efficiency audits for large commercial buildings and Nitrogen cycle assessment for aquaponics systems.

There are four main components in a typical sensor system, which do the following: sensor data measurement, data transmission, data storage/analyzation, and data consumer. In our system, they are the sensor devices, the central storage/X3D enabled web server and the 3D visualization client. The prototype is directly channeled to the server through the USB connection; therefore, it is not shown in Figure 1, which illustrates how these components are connected in our implementation.

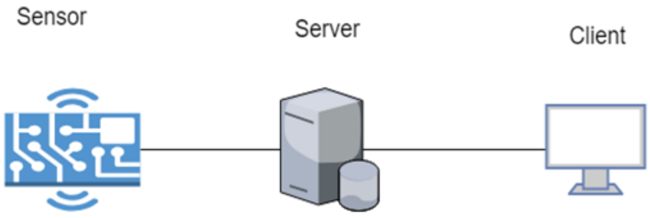

Figure 1. Typical Sensor Data Collection System

2.1 Wireless Sensor Networks

A wide variety of sensors technologies are available on the market today. Wireless Sensor Networks (WSNs) provide the "cells" for data collection and distribution within the IoT enabling the development of smart, context-aware applications. By sharing multiple types of power sources and maintaining power autonomy for large periods of time, these devices are the real enablers of the IoT applications, in terms of lifetime, energy efficiency, low costs, and connectivity.

2.2 Communication

Energy consumption, latency and throughput for different Medium Access Control (MAC) protocols for WSNs may have a significant impact on the sensor's performance [6]. A significant reduction in energy consumption (i.e., 18-45%) was possible for MAC protocols based on Bluetooth (BT) nodes with increased throughput and lower latency. Experimental data proves that Bluetooth Low Energy (BLE) is more energy efficient when compared to the ZigBee protocol. Translated in power consumption, an improvement from 35-40mW to 12-16mW can be gained. The possibility to develop smart applications with BLE is reviewed in [7]. Solutions based on BLE are more efficient than the Wi-Fi-based implementations due to lower power consumption and will be needed for sampling data at a higher frequency than done in this implementation.

2.3 Data Storage/X3D Server

Data storage solutions are required to store the RAWS data electronically and making it machine readable for offline processing. The current solution is to provide a MySQL database that would allow storage from each sensor based on the sensor ID. For each sensor, the generated data and corresponding time stamp are stored, generating an array that we record in conjunction with a sensor ID. The sensor data and time stamp are stored as integer values in the database. Upon connection, the sensor values are pushed to the server socket concurrently with being written to the database.

2.4 3D Visualization – Client Interface

Web standards have progressed considerably over the past decade with X3D being adopted as an open ISO standard for Web3D [8]. X3D and associated extensions like WebGL allows web-enabled visualization and facilitates remote collaboration. In this study, we employed X3D and ISO international standard to implement an early prototype that would visually represent temperature maps in a 3D web-based environment for commercial or residential buildings.

Each sphere in this model represents data from one sensor as an RGB value. The location of the sensors is visually communicated by their position in conjunction with the X3D geometry of the environment (as illustrated in Figure 2). The sensor readings are converted into a color map as discussed in Section 3.4.

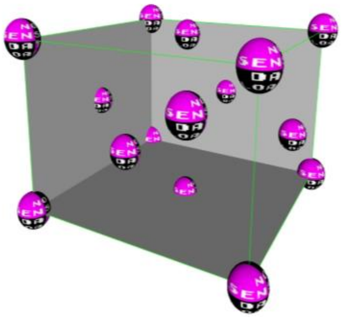

Figure 2. Visualization X3D Client. Location of wireless sensors in conjunction with the room geometry.

3 Implementation and Data Mapping

For this implementation, an Arduino Mega2560 with a DHT11 temperature and humidity sensor for the collection of real-world data was employed (as illustrated in Figure 3). We set the sensor node to provide temperature data at a rate of 1Hz for these experimental sessions.

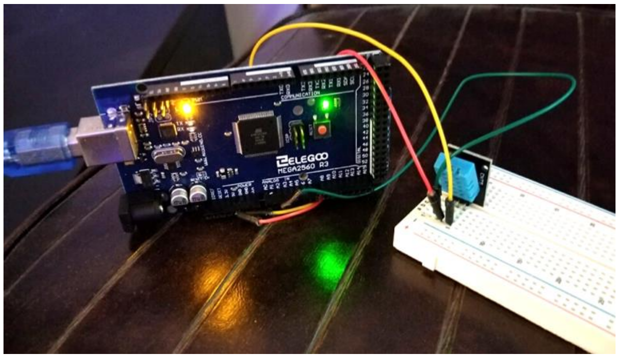

Figure 3: Prototype Sensor Implementation

This sampling rate is based on the research that near room temperature (23° C) it takes the body around 1 second to determine a 0.1° C difference in temperature.[9] The Arduino software module reads the sensor data, compresses it, and sends it to the server.

3.1 Data Sampling and Compression Algorithm.

To lower energy consumption, we can compress the sensor data sent to the server at the sensor end by sending temperature values of significant changes only. Figure 4 illustrates such an algorithm. In this simple algorithm, the sensor node will only send data to the server if the change of measuring data is beyond a given threshold, in our example, if the difference between the current measured temperature and the previously saved temperature passing a certain threshold. Depending on the application, applying a moving average algorithm can filter out the false alarm noise signals. The *timeStamp* reports the time of the event. The parameter *t* controls the sampling rate.

```
while (True) {
     if(abs(CurTemp-PreTemp)>Threshold){
              Send(CurTemp, timeStamp)
              PreTemp = CurTemp
     }
     Sleep(t);
}
```

Figure 4: Data Compression Algorithm

3.2 Data Collection and Analysis

Once the data is received at the server, the server will store the data and associated time stamp into the database for a complete history. Depending on the application, various smart AI algorithms can be applied to analyze the data and trigger an automatic warning signal if needed based on the freshly received data, history of data, time of events, as well as the location of the sensors. The server is also responsible for sending data to the clients.

For this prototype implementation, we used Node.js developed by Barry Van Dam [10, 11] to handle data receiving and the client requesting. Node.js allows automatic client update to be triggered by the arrival of new data.

3.3 Color Interpolation for X3D

On request of a client/automatic data update, the server will send data to the client. The JavaScript code running inside the client's web browser will manipulate the data received from the server using a form of quantizing that was originated from the Pulse Code Modulation (PCM) technique [10]. In this process, you will take an integer value and find the color zone corresponding to that integer value. For example, an integer value of 98 would correspond to the HTML color red because it is within the zone of 91-101 which was assigned red.

Using this technique, the color values are zoned to specific temperatures and the shade is then based on another zoning of the color zone. Depending on which sensor the data came from the color is mapped to that sensor probe's sphere on the X3D object. A cold to hot interpolation is used currently as illustrated in Figure 5, however, more complex models may accurately and better represent relative humidity/temperature values in the future.

Once the color mapping is done, the X3D engine running in the web browser will present the information visually in a 3D format. A user then can intuitively interpret the information presented.

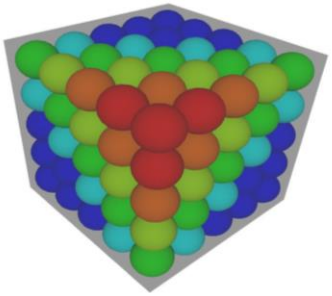

Figure 5. Color pallet for Temperature representation in the RGB colorimetric space

4 Soft Real-time Data Visualization

The sensor data is written, read, and visualized in Soft Real-Time, using event-driven programming and node.js [11]. The event-driven approach ensures the data will only be sent over the socket when there is a value change event and/or a request from the client. Node.js is perfect for event-driven applications because it enables objects to fire events.

To establish the average latency, the sensor sends fifty values in succession at different times throughout the day. Then the average latency is computed using the following equation.

$$Avg\ latency = \left(\sum_{1}^{t=50} T_{client\ time} - T_{Server\ time}\right) \div 50$$

Figure 6 illustrates the corresponding latency averages throughout different times of the day. As the results show jitter levels vary throughout the day, the system operates in soft Real-Time conditions [12].

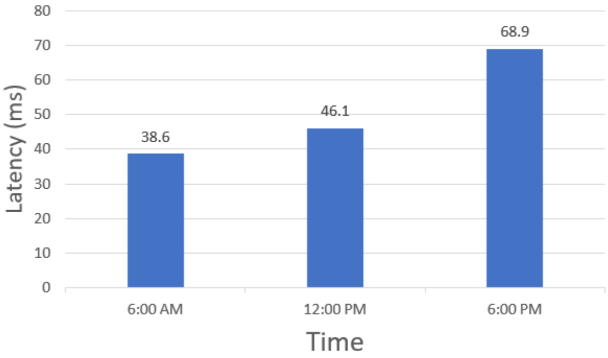

Figure 6. Transfer Latency throughout the Day

5 Results

The initial prototype proved the efficiency of our design therefore; we moved to start improving the design with further implementations. For the next phase, we will focus on the following improvements: implementing an multiple socket system to handle various events, the addition of an automatic sensor/channel identification system to distinguish various sensors/sources and subscribers/viewers, providing a complete sensor data history storage component on the server for each subscriber, to allow the user to “play back” the sensor data for any time interval, provide an improved user interface and a smooth X3D rendering experience for the users using AJAX technology [13] reduce the energy cost by adding preprocess nodes at both the sensor-side and the server-side to reduce data transmission, add smart AI components to help the decision making process.

5.1 User Interface

The UI started from an induvial implementation and then moved into the full room implementation. Figure 7 shows an individual sensor interpolated onto a cube, which was the basis for how we wanted all the sensor node to be visualized.

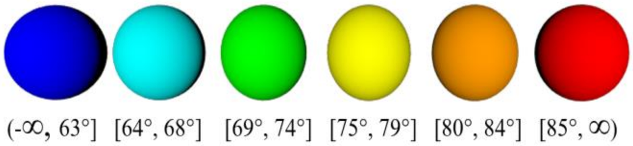

Figure 7: Sensor data representation using X3D primitives. Data is mapped into the RGB color space.

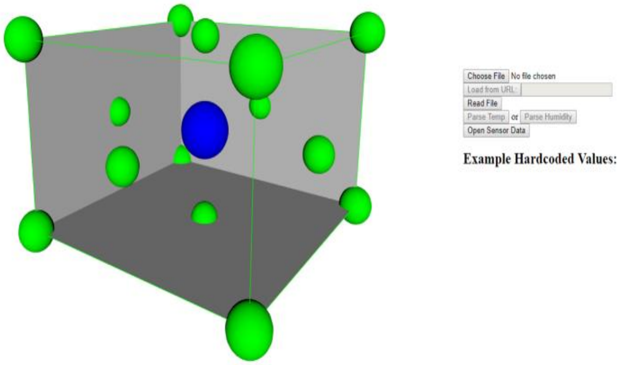

Figure 8: Prototype User Interface – Sphere Colors represent Temperature Values Dynamically.

Figure 8 illustrates how the sensor nodes (green) are geometrically distributed on an X3D object (room) to represent a WSN for visualizing temperature for a square room. Modifications to the UI will be necessary in subsequent implementations and will be based on the shape and scale required by the application in order to build an accurate X3D sensor data representation.

Conclusions

In this paper, we present a system that enables soft real-time sensor data visualization using the X3D programming standard in a web-based application. The system can be adapted to various applications such as indoor thermal map generation for energy efficiency audits and could be targeted at large commercial buildings and refrigeration systems for food safety. The system can also be customized with proper sensors to provide Nitrogen cycle assessment for aquaponics systems and water level mapping of agricultural sites. The Web3D indoor thermal mapping could facilitate decisions making for the architect and the contractor for building sites, or decision processes between the owner and the leaseholder of a large-scale commercial building [14]. On the agricultural side, the system would provide the user with the knowledge necessary to make a more informed decision about watering and fertilization cycles. Additionally, it can provide exceptional decision-making data for crop storage (e.g., temperature/humidity control for potatoes storage [15]). The system can be improved with additional sensors for measuring wind speed, temperature, humidity, and pesticide/herbicide levels.

In conclusion, this paper focused on the implementation of a soft real-time sensor visualization system through experimental software architecture. We also discussed some improvements for future implementations along with possible applications of this architecture.

ACKNOWLEDGMENTS

We would like to acknowledge the Erasmus+ grant for International Collaboration at Georgia Southern University. We would also like to thank Parker Towery for his contributions to the X3D user interface implementation.

REFERENCES

[1] Big Data for Humans: The Importance of Data Visualization https://dataconomy.com/2017/05/big-data-data-visualization/.

[2] Ousmane Diallo, Joel J.P.C. Rodrigues Mbaye Sene "Real-time data management on wireless sensor networks: A survey", Journal of Network and Computer Applications. Volume 35, Issue 3, May 2012, Pages 1013-1021

[3] Senouci, M. R.; Mellouk, A. Deploying Wireless Sensor Networks Theory and Practice, Elsevier, ScienceDirect, 2016

[4] Jawad, H.; Nordin, R.; Gharghan, S.; Jawad, A.; Ismail, M. Energy-Efficient Wireless Sensor Networks for Precision Agriculture: A Review. Sensors 2017, 17(8), https://doi.org/10.3390/s17081781

[5] Elkhodr, M.; Shahrestani, S.; Cheung, H. Emerging Wireless Technologies in the Internet of Things : A Comparative Study. International Journal of Wireless & Mobile Networks, (IJWMN) Vol. 8, No. 5, October 2016, https://doi.org/10.5121/ijwmn.2016.8505.

[6] Razaque, A.; Elleithy, K. Energy-Efficient Boarder Node Medium Access Control Protocol for Wireless Sensor Networks Sensors, 2014, 14(3), 5074–5117, https://doi.org/10.3390/s140305074.

[7] Garcia C. G.; Ruiz, I. L.; Gómez-Nieto, M. A, State of the Art, Trends and Future of Bluetooth Low Energy, Near Field Communication and Visible Light Communication in the Development of Smart Cities, Sensors 2016, 16(11), 1968; https://doi.org/10.3390/s16111968.

[8] Polys, N. F.; Brutzman, D.; Steed, A.; Behr, J. Future Standards for Immersive VR: Report on the IEEE Virtual Reality 2007 Workshop. IEEE Computer Graphics and Applications, 28 (2):94-99, 2008, Extensible 3D (X3D), Web3D Consortium. Available online: www.web3d.org (accessed on 02. 09. 2019).

[9] Dan R. Kenshalo, Charles E. Holmes, and Patrick B. Wood, 1968. Warm and cool thresholds as a function of rate of stimulus temperature change. Perception & Psychophysics 3, 2 (March 1968), pp. 81 - 84. DOI: https://doi.org/10.3758/BF03212769FNM Surname (2018). Article Title. Journal Title, 10(3), 1–10.

[10] A. Tanenbaum and D. Wetherall, Computer Networks, --5th ed. Upper Saddle River, NJ: Prentice Hall, 2010. pp. 153

[11] Van Dam, B.; Node.js communicating with Arduino, Jan 10, 2013; http://www.barryvandam.com/node-js-communicating-with-arduino/ (accessed on 01. 25. 2019).

[12] Buttazzo, G.; Lipari, G.; Abeni, L and Caccamo M. (2005) " Synchronization Protocols for Hard and Soft Real-Time Systems" in Soft Real-Time Systems, Series in Computer Science book series, pp. 133-174, https://doi.org/10.1007/0-387-28147-9.

[13] Thomas Powell. 2008. Ajax: The Complete Reference (1 ed.). McGraw-Hill, Inc., New York, NY, USA.

[14] Hamza-Lup, F. G.; Maghiar, M. Web3D Graphics enabled through Sensor Networks for Cost-Effective Assessment and Management of Energy Efficiency in Buildings, Graphical Models Journal, Elsevier, 2016.

[15] R. E. Voss, K. G. Baghott, Modoc-Siskiyou and H. Timm, Specialist (2015) "Proper Environment for Potato Storage", U.C. Davis.