IoT Based Automatic Greenhouse

Lakshay Sharma, Rodrigo Pessoa, Radhe Shyam Yadav - *17321163, 17337516, 17318855 – Students, Trinity College Dublin*

Abstract—The project presented a manual system to control sensors used to get information about plants' surrounding environment and devices which are used to adjust the environment to provide plants a condition in which they can grow.

Index Terms—IoT, Internet of Things, Smart devices, Smart farming, Automatic freenhouse, nRF52, ThingsBoard

1 Introduction

INTERNET of Things has seen an explosion of interest in recent years due to the ever-increasing connectivity of electronic devices, both embedded and otherwise, with actuators, sensors and networks. The creation of cloud based data storage and logic services enabled a decentralized approach to network design, resulting in a series of industrial and commercial applications, notably, smart home services like Google Home and Amazon's Echo product line (often called Alexa).

The possibility of exchanging data with various devices has been proven to be extremely beneficial to various areas, such as farming. Essentially, the benefit of internet of things applications can be reduced to a key point which is optimization of processes. These systems reduce production costs and resource waste using sensoring data. In addition, accurate data collected improves the decision-making of farmers (and, potentially, any machine learning logic implemented). Examples of internet of things applications in agriculture include smart equipment [1], livestock monitoring, conservation monitoring [2] [3] [4], plant and soil monitoring for precision agriculture. [2] [3] [5] [4]

The application of IoT to the area of agriculture has been made through three main approaches: the collection, manipulation and learning from remote sensor data, remote actuation based on information processing and the use of drones for precision agriculture. Focusing on the integration of the first two approaches, an internet of things based 'smart' (automated) greenhouse project was designed.

2 OBJECTIVES

The objective of this project was, primarily, to develop an infrastructure capable of being able to provide sensor data in real time and the ability of performing remote actuations, both programmatically (through time scheduling, cloud-based computer logic, machine learning, etc) and manually. Another key concept behind the choice of an automated greenhouse was not only the relevance to the current trend that is the implementation of smart farming, but also scalability. A small greenhouse provides an adequate environment to develop a communication infrastructure that could be scaled to industrial size applications, such as large greenhouses or extensive crop fields.

3 Overview

From a high level, the application developed provides the user with a customizable, user-friendly interface to view sensor data and perform actuations remotely. The sensor data is transmitted from the sensors to a central unit through the use of Bluetooth low energy (BLE) technology in addition to the latest version of the IPV6 internet protocol to create a wireless personal area network. This communication infrastructure is referred to as 6LoWPAN, an acronym made from the amalgamation of IPV6, low power and wireless personal area network. The protocol used to transmit the data packets from the sensors to the gateway and, subsequently, to the cloud was the MQTT protocol.

The sensors were simulated with Nordic Semiconductor's NRF52DKs and Thingsboard was used as the customizable, user friendly interface, gateway and cloud based service. The actuators were also simulated using NRF52DKs and, therefore, such devices acted as both publishers and subscribers in the MQTT communication protocol context. In the application level, the communication between Thingsboard and the devices was performed through the use of the two-way remote procedure calls (RPC) capabilities.

4 Context and Motivation

One of the main problems of the smart greenhouse and agricultural context in general is that the infrastructure is installed in remote locations and the system typically covers an extensive area. The choice was made, therefore, to decentralize the logic, moving to a cloud-based one. This was done through a Javascript JSON file, which was responsible for the processing of the received data (by the

NRF52Dks) and the publishing of the actuation-related information sent, also, to the NRF52DKs. Two common and persistent malfunctions were also considered and implemented for simulation purposes: sensor interference (bad sensor) and actuation breakdown.

5 System Design

The idea was to control the above said through an internet browser using ThingsBoard but it could not be implemented because of some limitations in nRF52DK, explained in the improvement section of this report.

The data from virtual devices – A Heater, Sprinkler, and Wind Breaker is simulated using Math functions to have realistic values.

We also included 2 malfunction simulations:

1. Adds noise to the temperature data giving an effect of temperature sensor being broken.
2. Simulates the Heater heating more than a maximum value causing overheat resulting in abortion of the heater.

For adding the noise to the temperature data, we used Marsaglia polar method which is a pseudo-random number sampling method for generating a pair of independent standard normal random variables. [6]

The calculations for simulating the data are done on the nRF52DK and are sent to the ThingsBoard.
The nRF52DK runs Zephyr as operating system which enables it to connect to the MQTT broker (ThingsBoard).

6 Performance of the system

The performance of the system was measured with relation to three areas: time response for the retrieval and processing of real time sensor data, time response of the logic processing and remote actuation based on the data received, reliability and stability of overall wireless connection. A study to validate how realistic the malfunctions implementations were… was also conducted.

The acquisition time of data was varied from 15000 milliseconds (15 seconds) to 1000 milliseconds (1 seconds). The time response for acquisition and processing of data was in the order of milliseconds and did not seem to vary considerably with the changes in the acquisition time. The sending of remote actuation data demonstrated a temporal lag. This lag was more noticeable for smaller acquisition times being, effectively, unnoticeable for acquisition times greater than 10 seconds.

The reliability of the exchange of data between the NRF52DK devices and Thingsboard showed satisfying results, with 0% loss of packets regardless of changes in the acquisition time, provided there was an active connection. The stability of the connection, however, deteriorated with decreases in the acquisition time, making the overall connection increasingly unstable, resulting often in loss of communication between the devices. This was extremely frequent for acquisition times lower than 5 seconds.

7 Experiments and Results

The source of the connection instability was traced to the implementation of the MQTT code: during the MQTT initialization, the NRF52DK device waited for an acknowledgement message from Thingsboard, confirming and starting the connection and the data exchange. Sensor information was still being published during the initialization however, leading to an accumulation of messages in the Zephyr's operating system internal message queue. This often resulted in a message queue overflow, causing a fatal crash on the software, rendering the NRF52DK broken until a hard reset was performed.

The connection stability problem was improved by the addition of a semaphore during the code's initialization stage. The semaphore blocked all publish requests, preventing them from being added to the message, until the acknowledgement message was received, and the data exchange was established. The semaphore implementation improved the connection instability problem for lower acquisition times and had the added benefit of removing 'data burst'. Data burst was the rapid publishing of accumulated MQTT message requests when the system could initialize before the Zephyr's message queue overflowed.

The actuation breakdown implementation presented no problems and worked as expected: whenever there was a simulated 'breakdown' signal, the user was unable to change the actuation state, which was set to 'off' regardless of user input. The sensor data information was contaminated with a zero-mean normal random noise of predetermined standard deviation. This required the implementation of a normal random variable generator function (called 'randn'), which was done by implementing the Marsaglia polar method. This method required uniform random variables, which were obtained utilizing the inbuilt 'rand' function (available in the standard library stdlib.h) to generate them.

The normal random variable generator provided an interesting problem: the algorithm had iteration times of over 10 seconds, which was unacceptable and lead to timing issues with the communication protocols. The problem was not with the implementation of the Marsaglia polar method itself, but rather with the inbuild uniform random variable generator. The rand function utilized a variable, called RAND_MAX whose value depends on implementation but is typically around 2^{14} or 2^{15}. A change in the value of this variable to 2^{20} resulted in drastic decreases in computational time for the implemented algorithm, from 10 seconds to the order of milliseconds. It is conjectured that the reason for this is that the rand function is a pseudo-random function with an

implementation that contains some sort of bias. Such bias has an adverse effect for values of RAND_MAX which are not sufficiently high, failing repeatedly one of the conditions of the Marsaglia polar method, subsequently leading to excessively high computational times.

A problem was also encountered when multiple NRF52DK devices were connected to the same Thingsboard instance. The first device connected presented no problems, but subsequent devices had a connection but no exchange of data. Further investigation showed that the Thingsboard MQTT broker acknowledged the NRF52DK's request, but ignored it, effectively 'losing' the sent data. It is unclear if this problem lies with the Thingsboard processing of the NRF52DK requests or the Bluetooth connection. Repeated experiments to determine the source of the error resulted in extremely rare instances where multiple devices were successfully connected, without any change in code or connection method. This suggests that the problem is not within the code implementation, but rather connection issues with the Bluetooth adaptor or the Thingsboard instance. The extreme unreliability of connection of multiple devices persisted as a problem throughout the system implementation.

8 Improvements

The project if given more time could be improved in many ways out of which some are:

1. **Use of Cloud for execution**
   Code execution could be passed to the cloud instead of the nrf52 board. It would result in faster processing of the necessary function running on the board as the resources are very limited and when the simulation calculations run on the board, it makes it slower to push data to the cloud and sometimes it forces the board to stop working.
2. **Automation of devices**
   Devices – Heater, Wind breaker, Water sprinkler could be automated according to the sensor data. For example, if it's too cold then heater should turn on automatically and likewise turn off when temperature is normal.
3. **Updates from weather forecast**
   If the code execution would have been going on on the cloud (ThingsBoard) then a bot could be implemented which would take the readings from weather forecast and devices attached to the system would change their working pattern based on near-future weather conditions. For example, if it is going to rain then sprinklers do not need to be turned on.

9 Future Development

So much work is already being done in the direction of IoT based Farming. ThingsBoard provides ready-to-use interactive dashboards specially designed for smart farming.

In context of this project, the future implementations can be stated as:

1. **More control to user**
   User can be provided with more features like mobile phone notifications on weather changes, malfunctions, etc.
2. **Logging**
   Logging of the devices working to create normal conditions, energy consumed, etc can be logged and analyzed for better planning of when and what kind of crops/plants to grow.
3. **Plants' growth monitor**
   Monitoring the growth of the plants using computer vison, and rot warning alerts can be useful.

10 Conclusion

The final project consists of 3 versions of code which were tested on 3 different nrf52 DKs. The reason behind making 3 versions was to make it easily debuggable, and because the nrf52 cannot run a large piece of code smoothly.

Rodrigo's version of code has temperature simulation and heater actuation. Likewise, Lakshay's code has Wind speed simulation with Wind breaker actuation, and Radhe's code has Soil humidity simulation with water sprinklers actuation.

For simulating the sensor data, math functions are efficienty used giving very real-like values.

Although the goal which was set when started this project could not be reached – to connect all 3 nrf52 DKs together in a cloud and automate the devices through code executing in the cloud, we still see this project as a success because we were able to see the results as wanted and predicted. (See figure below)

The figure above is a screenshot of Lakshay's version of code output on ThingsBoard. It shows a temperature and wind speed gauge with Heater and Wind breaker running state. Along with gauges, there is also a table that shows the telemetry values that it receive from the board.

References

[1] "CLAAS", CLASS, [Onine]. Accessible at: http://www.classofamerica.com/

[2] M. Jhuria, A. Kumar, R. Borse, "Image processing for smart farming: Detection of disease and fruit grad-

ing", 2013 IEEE Scnd. Inter. Conf. on Img. Inf. Proc., Shimla, India, DOI: 10.1109/ICIIP.2013.6707647

[3] S. R. Rupanagudi, Ranjani B. S, P. Nagaraj, "A novel cloud computing based smart farming system for early detection of borer insects in tomatoes", 2015 Inter. Conf. on Comm., Inf. & Comp. Tech., Mumbai, India, DOI: 10.1109/ICCICT.2015.7045722

[4] M. C. O'Connor, "TempuTech Smartens Up Grain Management With GE Platform", IOT Journal, [Online]. Accessible at: http://www.iotjournal.com/articles/view?13479

[5] "The internet of Soil", Cropx, [Online]. Accessible at: https://www.cropx.com/

[6] "Marsaglia polar method", Wikipedia https://en.wikipedia.org/wiki/Marsaglia_polar_method