Post Emergency Power Management for IoT Based Precision Agriculture Irrigation System Using cost-effective Algorithm

Jiho Park
Computer Science and Engineering
Dongguk University
Seoul, Korea, South
jiho8345@dgu.ac.kr

Yujung Gil
Computer Science and Engineering
Dongguk University
Seoul, Korea, South
gilyujung@gmail.com

Minjeong Kim
Computer Science and Engineering
Dongguk University
Seoul, Korea, South
kimmin9642@dgu.ac.kr

Bryan Supinski
Computer and Information Technology
Purdue University
West Lafayette, IN, USA
bryansupinski@gmail.com

Damien Pham
Computer and Information Technology
Purdue University
West Lafayette, IN, USA
minhduypham0210@gmail.com

Max Li
Computer and Information Technology
Purdue University
West Lafayette, IN, USA
maxli32145@gmail.com

Parker Alexander
Computer and Information Technology
Purdue University
West Lafayette, IN, USA
carmelo15andonly@gmail.com

***Abstract*—The Internet of Things (IoT) in agriculture continues to be a key area that enables farms to efficiently reduce labor and increase crop production and harvest yield by remote management utilizing data from interactive sensors. Earlier research focused on smart farms primarily assuming that the power supply is sufficient. In a situation where the power supply is limited due to the occurrence of a natural disaster, the remaining power should be properly distributed to the crops until the power supply is normalized. This research approaches both the software and hardware aspects to design efficient irrigation systems. In terms of software, this project develops a cost-effective algorithm that uses less power. In terms of hardware, using Long Range Wide Area Networks (LoRaWAN) will allow changes to the system configurations so that the system itself can reduce power consumption. This research aims to experiment with how much power the developed system can save compared to a normal system that irrigates a regular amount of water and show that this project can become a milestone for systems in power-bounded situations.**

***Index Terms*—LoRa, LoRaWAN, IoT farm, smart irrigation system, limited power**

I. Introduction

Agriculture is the foundation of any national economy and irreplaceable to the growth of any civilization. In recent years, markets related to smart farming combined with IoT are predicted to grow on a large scale as new technology transitions from experience-based agriculture to data-based agriculture. Smart irrigation systems are an example of these smart farming technologies [1]. A smart irrigation system manages resources efficiently by automating the irrigation system through real-time monitoring, analysis, and prediction based on Information and Communication Technology (ICT), hence reducing resource related costs and increasing crop yields. As climate change continues, natural disasters have had a heavier influence further inland than before. As a result, the production of crops is heavily affected and can cause a huge economic loss to a country's economy. It is imperative that this research addresses the problem with a useful solution. Algorithms running in power-bounded situations were not designed in existing research papers because most of them assumed that there is unlimited electricity. In a disastrous situation, given that the power supply is compromised, a specifically designed algorithm could be devised and released into the system to minimize damage by allocating power to the areas that need it the most. This paper aims to prove that the system uses power more efficiently compared to existing systems by exploring ways in which power can be allocated to each crop area and suggesting an algorithm that can handle the power-bounded scenarios. In the system, the LoRa module attached to the edge node and the central controller enables long-distance communication with minimal power and makes the system work properly in power-bounded situations. The system is configured to allow users to monitor the soil moisture value and the state of the battery by processing data through Node-RED to the cloud. The central controller inspects the soil moisture value received from the edge node connected to the sensor and sends the irrigation command to the edge node by applying the devised algorithm in a situation where the crop needs irrigation. This research devises an algorithm that uses power consumption according to distance and the possibility of crops recovery according to the time change as weights by conducting three preemptive experiments related to the time interval for receiving crop data, water, and electricity. An algorithm proposed in this paper can be a milestone to facilitate crops to grow properly with only a limited amount of power, thereby minimizing resource wastage. Furthermore, by adding

various sensors, efficient power management is possible not only in automatic irrigation systems but also in various types of smart farm automation technologies. This project is notable for operating more efficiently in an emergency situation by calling the irrigation function only when the irrigation needs, and reducing the data transmission period.

II. Literature Review

As IoT develops, a smart farm system that remotely monitors and manages farms is currently being studied. M. Rohith *et al.* designed a system that sends sensor values such as soil moisture, humidity, and temperature to the main system using a WiFi module for wireless communication and automatically irrigates by adjusting the irrigate motor according to the value of the sensors [2] and an automatic irrigation system communicating sensors and a master node using Zigbee, and operating fan, pump, and buzzer according to the sensor values was described by P. Chikankar *et al.* [3]. However, H Muthukrishnan *et al.* asserted that WiFi is not suitable for the agricultural environment because there is a limited communication distance of 200m and WiFi is designed to send data at a short distance in a short time, which consumes a lot of power. On the other hand,"LoRa is 2.85% better in terms of power consumption and 6.96% better in delay when compared with WiFi [4, p. 19]". ZigBee is short-range communication designed for low-power and is not suitable for long-range communication in outdoor areas [5]. LoRa is used in this research because the distance between the farm and central controller (Raspberry Pi) is too far and only soil moisture value is delivered. LoRa, a Low Power Wide Area Network (LPWAN), is a long-range communicating technology for IoT that is used for data communication in outdoor agriculture systems due to low-power and long-range (up to 15 kilometers) [6]. G. Codeluppi *et al.* use LoRaWAN as middleware to send sensor value installed on the farms to the central module, showing it to farmers using web service [7]. A system that changes the level of the motor by adjusting relays when the value from the sensor with the LoRa module is out of a threshold range is explored by T. Mohammed *et al.* [8]. M. Lekić and G. Gardašević described that the MQ Telemetry Transport (MQTT) is the optimal solution for IoT applications and it should be based on a more robust algorithm [9].

Due to the automatic irrigation system used in this research, Algorithms for making a decision are quite significant. A system that calculates the priority level according to the sensor value, the water level in the tank storage, and availability of electricity of each area of the farm using the fuzzy algorithm and irrigates, in turn, was proposed by J. Cruz *et al.* [10], and a cost optimizing system to determine which areas of crops to irrigate using a genetic algorithm and sensors related to irrigation to minimize multi-objective optimization problems is described by Ocampo and Dadios [11]. However, available power is not considered in most of the research about smart irrigation because they are used in the general situation where electricity is stably supplied and irrigation efficiency is reduced. The distance between irrigation facilities and crops is also not considered. Therefore, the irrigate algorithm that can be applied even in a power-bounded situation considering the amount of remaining power and variable of weight according to the distance between crops and irrigation facility is used in this study.

The value of the soil moisture sensor increases when it is dried and decreases when it is wet. The maximum value of soil moisture was 566 in the air, and the minimum value was 264 in the water. The soil moisture percentage of 70-80%, in which tomatoes grow best in fertilized soil, is observed by Jie Liu *et al.* [12]. When the maximum value of 566 in our experiment is set to 0% of soil moisture percentage, and the minimum value of 264 is set to 100%, the value of 70% is 354.6, and the value of 80% is 324.4. Therefore, The range of threshold values is set 324.4 to 354.6, which are values that tomatoes can grow well. It has also been investigated that the soil moisture percentage must be at least 55% for growing seeds by Jacob *et al.* [13]. Thus, the result of converting 55% to our soil moisture sensor value in the way we calculated the percentage above, which is 420, will be used in the preliminary experiment.

III. Methodology

A. System Overview

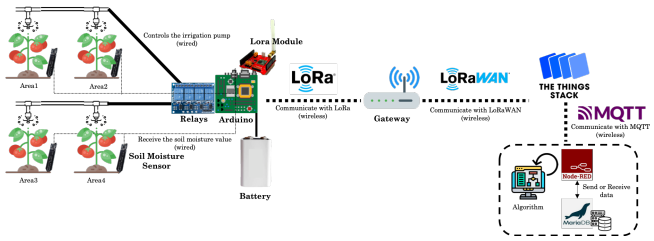

Fig. 1. System Architecture.

This research proposes an automated irrigation system using IoT devices to respond when farms no longer have a stable supply of power for irrigation due to a natural disaster.

The environment of this system consists of 4 areas which contain planted tomatoes as shown in Fig. 1. A soil moisture sensor, a sprinkler and water tube to supply water for irrigation are installed in each crop area. The water tubes are connected to the relay shield which controls the power of the pump, and the sprinkler irrigates crops on the surface of the area for reduced electricity consumption [14].

The soil moisture sensors are connected to the Arduino which sends the data of an individual crop area to a central controller which decides whether the chosen area of crops should be watered or not based on the devised algorithm. The crop data transmitted from the Arduino is transferred to the cloud through the gateway. LoRa communication is used when the crop data is transmitted from the Arduino to the gateway and LoRaWAN communication is used when the data is moved from the gateway to the cloud. In order to further save power, the cloud checks the input soil moisture sensor values, determines whether the area needs to be irrigated,

and sends an irrigation command to the edge nodes only when the irrigation is required. This project assumes that the use of WiFi or cellular service is limited in rural locations. Therefore, by using LoRa communication between edge nodes and the central node, it allows for communication between nodes without relying on a larger network.

This system has a User Interface (UI) that displays the current moisture levels of each of the areas, the battery level, and the last time an area was irrigated using Node-RED on the Cloud. The system stores the date-time, unique number of the area and soil moisture values using a Maria database (Maria DB). The system is composed of three main parts in this research as shown in Fig. 2.

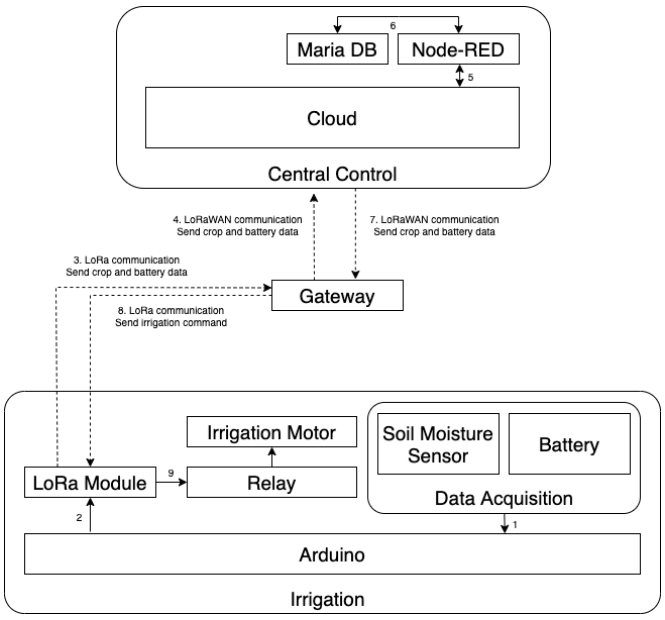

Fig. 2. The overview of the auto irrigation system in this research.

- *Data Acquisition*: In this part, soil moisture sensors acquire soil moisture values from the tomato fields. The sensors are connected to the Arduino by wires. The soil moisture value is delivered as an integer type and 4 values are delivered as an array. Data on the remaining amount of battery to operate the automatic irrigation system and the acquired crop data are collected on an Arduino connected to the sensor. The sensor data collected by the Arduino is then transferred to the gateway through LoRa communication.
- *Central Control*: This part is the main part of the project. The collected data from the Data Acquisition Unit arrives at the cloud from the gateway. It stores the crop data in the database at the time that the data arrives and applies the devised algorithm. The result of this algorithm, which is an irrigation command, is transmitted to the Arduino. By using Node-RED in the cloud, the farmer can monitor the data through the Node-RED User Interface (UI) remotely.
- *Irrigation*: This part receives and executes the irrigation command sent from the cloud. In this part, the motor connected to the water pipe for irrigation is connected to the relay. The relay connected to the Arduino receives the "turn the motor on and off" command sent from the LoRa module to execute automatic irrigation.

B. Algorithm

Algorithm 1: Irrigation Automation System in Power-Bounded Scenario Algorithm Operating in the Arduino

Input : $moisture\_value \leftarrow$ The value received from the each moisture sensor
$volt\_value \leftarrow$ The analog value received from the battery
Output : $result\_value \leftarrow$ The soil moisture value after applying this algorithm
$power\_value \leftarrow$ The amount of power that this irrigation system can use

1. $moisture\_value_i \in moisture\_value,\ result\_value_i \in result\_value,$
2. **for** each $moisture\_value_i \in moisture\_value\ (i = 0, 1, \cdots,$ number of $moisture\_value)$
3. **if** $moiture\_value_i < min(threshold)$:
4. $result\_value_i = 0$
5. **else**
6. $result\_value_i = moisture\_value_i$
7. **if** $\sum_{i=0}^{\text{number of } moisture\_value} result\_value_i == 0$:
8. **end**
9. **else**
10. $power\_value = 9.2 \cdot volt\_value * 9.2V / 1023.0$
11. **send** $power\_value, result\_value_0, result\_value_1, \cdots,$ number of $result\_value$ **to Algorithm 2**
12. **end**

Fig. 3. Pseudo code for algorithm operating in Arduino.

Algorithm 2: Irrigation Automation System in Power-Bounded Scenario Algorithm Operating in the Cloud

Input : $moisture\_value \leftarrow$ The value received from the each moisture sensor
$power\_value \leftarrow$ The amount of power that this irrigation system can use
Output : $final\_level \leftarrow$ The level represents the amount of water using the current $power\_value$
calculate_level(moisture_value) : The function which is returning the level(0,1,2,3) using moisture sensor value

1. $moisture\_value_i \in moisture\_value,\ result\_value_i \in result\_value$
2. **calculate** $level$ using $calculate\_level(moisture\_value)$
3. **def** $irrigate(powe\_value, level)$:
4. $expected\_power = [\ ]$
5. **for** each $level_i \in level\ (i = 0, 1, \cdots,$ number of $level)$:
6. **if** $level_i$ is not 0 :
7. **append** $power\_consumption_j\ [level_i - 1]$ to the end of the $expected\_power$
8. **else :**
9. **append** 0 to the end of the $expected\_power$
10. **calculate** $total\_power$: $total\_power$ is sum of the $expected\_power$
11. **if** $total\_power < power\_value$:
12. $power\_value = power\_value - total\_power$
13. $final\_level = level$
14. **return** $final\_level$
15. **else :**
16. **for each** $expected\_power_i \in expected\_power\ (i = 0, 1, \cdots,$ number of $expected\_power)$:
17. **if** $power\_value <$ **=** $expected\_power_i$:
18. $level_i$ **= 0 ,** $expected\_power_i = 0$
19. **if** $\sum_{i=0}^{\text{number of } level} level_i$ is zero:
20. **return** [-1]
21. $index = [0, 1, 2, 3]$
22. **while** $power\_value$ **> 0:**
23. $indice$ **= calculate** max value of the level
24. **for each** $indice_i \in indice\ (i = 0, 1, \cdots,$ number of $indice)$:
25. **if** $power\_value$ **>=** $expected\_power_i$:
26. $power\_value = power\_value - expected\_power_i$
27. $final\_level_i = level_i$
28. $level_i$ **= 0**
29. **else:**
30. $power\_value$ **= 0**
31. $remain$ **= calculate** difference set of $index$ and $indices$
32. **return** $final\_level$

Fig. 4. Pseudo code for algorithm operating in the cloud.

The algorithm devised in this research aims to minimize damage to crops until the power supply is recovered through more efficient irrigation power operations in power-bounded emergency situations.

The Arduino receives the soil moisture value in an integer list from the soil moisture sensors. As seen in Fig. 3, the algorithm of step 1 operates in Arduino.

Step 1: If each soil moisture value is less than the *threshold* value (355), the value is changed to zero owing to having no reason to irrigate. This algorithm sends the list to the main algorithm if any values are not zero. If the specified conditions that can be moved from step 1 to step 2 are met, the main algorithm operating in the cloud is called. As seen in Fig. 4, this system assumes that the distance between the crop areas and the irrigation pump is stored in advance. The algorithm uses a new list of levels using the soil moisture values and the results of preliminary experiment 2. The power consumption is predicted based on the distance and irrigation level which is the result of preliminary experiment 3.

Step 2: The system will send the *level_array* for irrigation to all requested areas when the sum of predicted power consumption is less than the remaining power. It is impossible to irrigate when the predicted power in an area is more than the remaining power.

Step 3: The system determines the *final_level* list by considering the initial level values. The algorithm then sets the *final_level* value again to make any matching values different using the predicted power consumption when there are identical *final_level* values.

Step 4: The system gives priority to areas with the shortest distance when the predicted power consumptions are the same.

As a result, the final output is a list consisting of integers between zero to three. This system calls appropriate functions in the Arduino for irrigation by using this output.

C. Node-RED

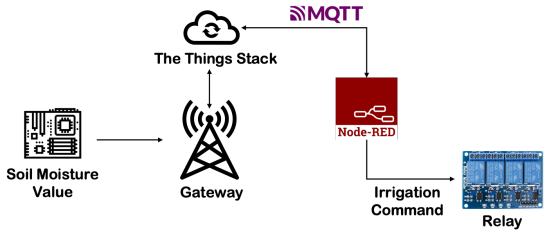

Fig. 5. Wireless Communication Flow chart for getting sensor value from the cloud using Node-RED.

As seen in Fig. 5 this research uses Node-RED for getting sensor values from the gateway and The Things Stack (TTS) to the Node-RED by using a protocol named MQTT. Node-RED is running on the local environment. The above algorithm as shown in Fig. 4 is running on Node-RED, and the output of the algorithm is sent to the Arduino containing the irrigation command for the relay.

IV. Experiment

A. Preliminary Experiment

Before proceeding with the experiment for our research, some preliminary experiments were needed to obtain the necessary setup information for the experiment.

1) Determining the upper bound: An experiment was used to determine the upper bound of the soil moisture value for irrigation by verifying the minimum soil moisture value that the crop can survive. In an emergency situation, it is necessary to set the upper bound for the reason that there are situations where only the crops that can survive should be saved. This experiment set the upper bound by checking the variation of the moisture value using a hundred thousand data points extracted for 3 weeks.

2) Determining the amount of water at one time: Based on the results of preliminary experiment 1, an experiment is needed to determine the amount of water to be supplied at once. Therefore, preliminary experiment 2 was conducted to determine the required amount of water. The experimental method is as follows: First, the soil moisture sensors are connected with the Arduino. Secondly, the Arduino sends the soil moisture values to the Node-Red. Lastly, the Node-RED connects with the MariaDB through the database node on the Node-RED and stores the soil moisture values in the MariaDB by using an insert query statement. The soil moisture values were stored in the MariaDB once every minute. While changing the amount of water for irrigation according to the current soil moisture value, this experiment is repeatedly conducted to determine how much water should be given so that the sensor value can enter a stable condition after irrigation.

3) Measuring the power consumption: Preliminary experiment 3 was carried out to measure the power consumption and operation time of the water pump while irrigating according to the soil moisture value and the distance from the water pump to the crops. As the distance increases, it is expected that both power consumption and operating time of the water pump will be greater because the water pump operates for a longer time when first sending water than a short distance, and this experiment was conducted to prove this. In preliminary experiment 2, the soil moisture value was divided into 3 levels, and the amount of water to be supplied was 50ml at level 1, 100ml at level 2, and 150ml at level 3. The distances to be used in this experiment were configured to increase twofold. The output of this experiment is the power consumption and elapsed time according to the distance.

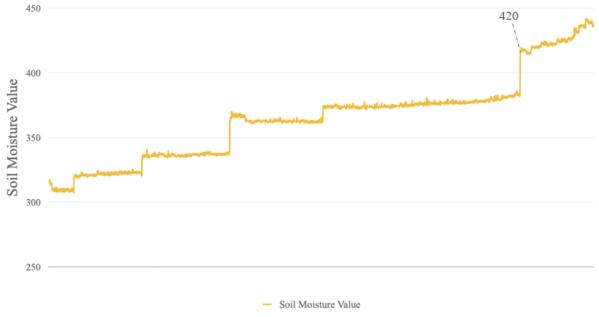

Fig. 6. The variation of the soil moisture value.

As seen in Fig. 6, the value is maintained for a certain period

of time before the sensor value exceeds 420. At the time that the sensor value exceeds 420, the range of irregular vibrations increases. Therefore, it can be expected that the upper bound of the soil moisture value is 420. To ensure the reliability of this experiment, it is necessary to reference previous research related to this experiment. The minimum soil moisture value that crops can survive is 55% [12], and that value is 420 when converted to the soil moisture value for the purpose of this research as calculated in the previous literature review. As a result, the results of preliminary experiment 1 can be generalized.

TABLE I
MINIMUM AND MAXIMUM VALUES OF THE SOIL MOISTURE VALUES DIVIDED INTO FOUR LEVELS

	soil moisture value	
	min	**max**
Wet	264	-
Level 0	324	354
Level 1	355	370
Level 2	371	395
Level 3	396	420
Dry	-	566

Preliminary experiment 2 was repeated for about 2 weeks using a total of 4 tomato pots, and about 16,000 data values were stored from each tomato pot for a total of about 64,000. Based on repeated experiments, it was possible to divide the minimum and maximum values of the soil moisture values into four levels as shown in the Table I. Level 0 is a stable condition, which is to mean that the moisture of the soil is appropriate. As the value of the level increases from level 1 to level 3, the soil moisture value of the soil increases. Therefore, the appropriate amount of water for irrigation is 150ml, 100ml, or 50ml depending on the level through the data obtained through this experiment. The result of the experiment is shown in the Fig. 7 below.

Fig. 7. Variations in the soil moisture value according to the amount of water.

Level 1 is a section in which the sensor value is 356 to 370, and when 50 ml of water is supplied, the sensor value enters the stable section.

Level 2 is a section in which the sensor value is 371 to 395, and when 100 ml of water is supplied, the sensor value enters the stable section.

Level 3 is a section in which the sensor value is 396 to 420, and when 150 ml of water is supplied, the sensor value enters the stable section.

The amount of water required to enter the stable section per level is shown in the Table II below.

TABLE II
AMOUNT OF WATER REQUIRED FOR EACH LEVEL

	Level 1	**Level 2**	**Level 3**
Water	50ml	100ml	150ml

Table III and Table IV are the results of preliminary experiment 3 and Fig. 8 and Fig. 9 are graphs respectively showing the results of Table III and Table IV.

TABLE III
POWER CONSUMPTION DEPENDING ON THE DISTANCE AND AMOUNT OF SUPPLIED WATER

	50inches	**100inches**	**200inches**	**400inches**
50ml(Lv.1)	1mAh	2mAh	3mAh	5mAh
100ml(Lv.2)	2mAh	3mAh	4mAh	6mAh
150ml(Lv.3)	3mAh	4mAh	5mAh	7mAh

TABLE IV
OPERATING TIME OF WATER PUMP DEPENDING ON THE DISTANCE AND AMOUNT OF SUPPLIED WATER

	50inches	**100inches**	**200inches**	**400inches**
50ml(Lv.1)	4.41s	5.81s	7.72s	16.3s
100ml(Lv.2)	6.31s	10.34s	12.08s	18.29s
150ml(Lv.3)	10.87s	14.27s	16.43s	22.82s

As shown in Fig. 8 below, the power consumption increases in proportion to the distance and the amount of supplied water, respectively. As seen in Fig. 9 below, the operating time of the water pump also increases proportionally with distance and the amount of supplied water. Among the results of this experiment, the operation time of the water pump is used as a variable that determines how long the relay will operate for each level, and the amount of power consumed is used as a variable in the algorithm that determines the priority according to the remaining power, levels, and distances.

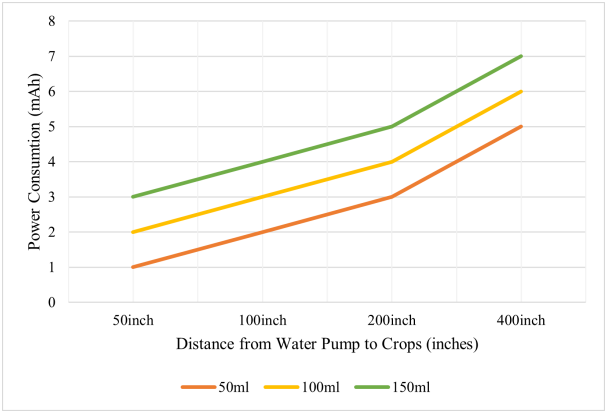

Fig. 8. Power consumption according to the distance and amount of supplied water.

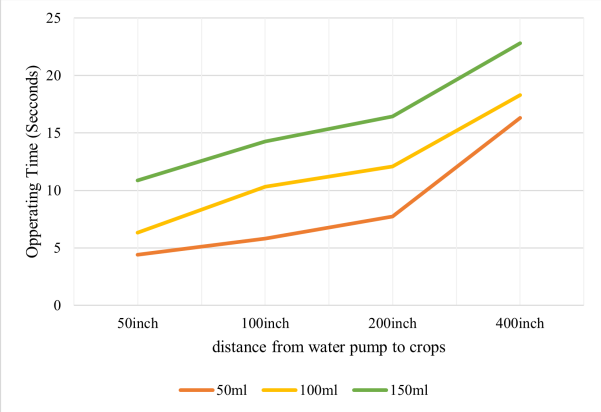

Fig. 9. Operating time of water pump according to the distance and amount of supplied water.

B. Main Experiment

1) Experimental environment configuration: The purpose of this experiment is to verify the efficiency of the devised algorithm. In order to compare with the existing irrigation system on real farms, 150ml was irrigated each time the soil moisture value rose up to 420 as a control group. In each of the experimental group and the control group, four pots were placed centered on one LED light for plant cultivation and the hoses connected to the pots were 50, 100, 200, and 400 inches, respectively as shown in Fig. 10. The soil moisture levels of the pots located in the same place based on the LED light in each group were matched. After the experiment was conducted for 120hours, the moisture value and power consumption of each group were compared.

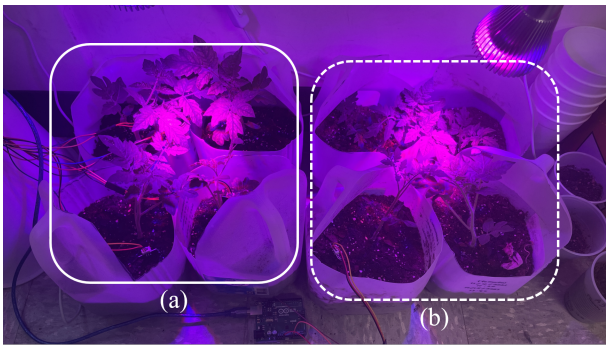

Fig. 10. The figure of the experimental group and control group. (a) Control Group. (b) Experimental Group. 50, 100, 200 and 400 inches clockwise from top left.

2) Data flow in the experimental group: As seen in Fig. 11, the experimental group is divided into three parts.

- *(a):* The Arduino is registered as an end device in the TTS and issued the device address, network session key, and application session key. The sensor data is read and preprocessed in Arduino, and the LoRa code that generates the LoRa packet and sends the data to the gateway is uploaded on the Arduino.

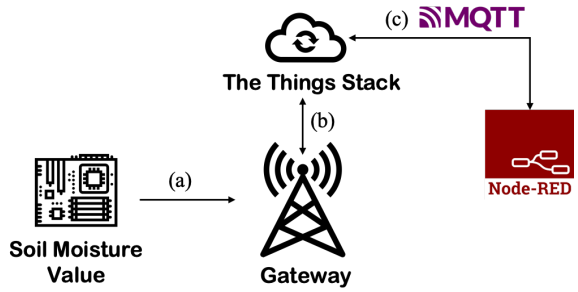

Fig. 11. The figure of the data flow of the experimental group.

- *(b):* The gateway operates in packet forwarder mode and serves to transmit data arriving at the gateway to the cloud.
- *(c):* The MQTT protocol is used to apply algorithms by transmitting data from the cloud to Node-RED. API key for MQTT communication is issued and connected with the MQTT IN node of Node-RED.

3) Flow construction in Node-RED: Fig. 12 is a flow of node-red configured for the progress of this experiment.

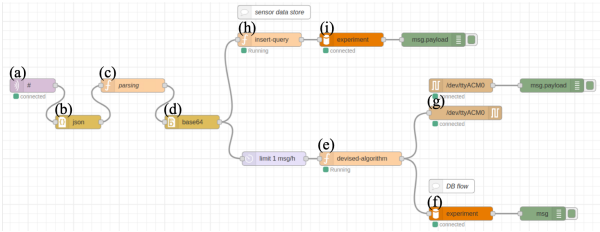

Fig. 12. Flow on Node-RED.

- *(a):* MQTT IN node. This project used TTS and MQTT which is a communication protocol [11] to get sensor values from the gateway to Node-RED.
- *(b):* It is used to convert the value received as JSON into an object suitable for msg.payload.
- *(c):* JSON.parse() is a function which is used to convert msg.payload object value into javascript data. The variable named pay is returned in the form of payload using uplink_message.frm_payload to get the necessary part for this research from the data above.
- *(d):* It is used to convert from base64 format to string type.
- *(e):* It is a part where the algorithm as seen in Fig. 12 works, and for pre-processing, the payload is converted to UTF-8 format and the soil moisture value is parsed. The output of this node is the String type irrigation command. It only operates like FaaS when the irrigation needed data occurs.
- *(f):* This is a database node. It stores the result of the devised algorithm, irrigation area, and irrigation time.
- *(g):* This is a Serial out node. It sends the result of the devised algorithm from the Node-RED to the Arduino

which is connected to the irrigation tube to operate the irrigation system.

- *(h):* This is a python-function node to insert the crop data to the database.
- *(i):* This is a database node. It stores the crop data before applying the devised algorithm.

4) Sending irrigation command to the relay:

Step 1: Read the Irrigation command sent from Serial out node in Node-RED to the Arduino using Serial.readStringUntil().

Step 2: Cast and parse the irrigation command from String type to int type and operate the irrigation code for the appropriate time according to the preliminary experiment 3.

5) Result: The amount of electricity was reduced by about 12% as shown in Fig. 13 compared to the existing system that provided a predetermined amount of water when it exceeded a certain threshold.

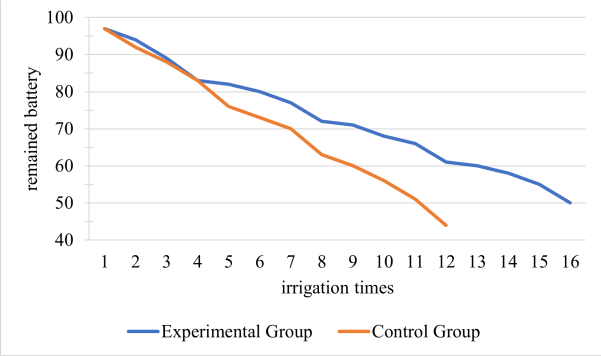

Fig. 13. The result of main experiment.

V. Conclusion

In this paper, the system is designed to use power efficiently in power-bounded situations by developing the existing irrigation algorithm and using LoRaWAN. By using this system, the amount of electricity was reduced by about 12% compared to the existing system that provided a predetermined amount of water when it exceeded a certain threshold. This research can be suggested as one of the methods to slow global warming by making crops grow properly with a small amount of power and minimizing resource wastage. In addition, there is a possibility that it can develop into a system that is applicable from various types of farms if various sensor values are used. If this system is developed by using Function-as-a-Service (FaaS) instead of calling functions using Node-RED in the cloud, it can improve the accessibility for users.

References

[1] B. Citoni, F. Fioranelli, M. A. Imran, and Q. H. Abbasi, "Internet of Things and LoRaWAN-Enabled Future Smart Farming," in *IEEE Internet of Things Mag.*, vol. 2, no. 4, Dec. 2019, pp. 14–19, doi: 10.1109/IOTM.0001.1900043.

[2] M. Rohith, R. Sainivedhana, and N. Sabiyath Fatima, "IoT Enabled Smart Farming and Irrigation System," in *2021 5th Int. Conf. on Intelligent Computing and Control Systems*, 2021, pp. 434–439, doi: 10.1109/ICICCS51141.2021.9432085.

[3] P. B. Chikankar, D. Mehetre, and S. Das, "An automatic irrigation system using ZigBee in wireless sensor network," in *2015 Int. Conf. on Pervasive Computing*, 2015, pp. 1–5. doi: 10.1109/PERVASIVE.2015.7086997.

[4] H Muthukrishnan, A Jeevanantham, B Sunita, S Najeerabanu, and V Yasuvanth, "Performance Analysis of Wi-Fi and LoRa Technology and its Implementation in Farm Monitoring System," in *IOP Conf. Series. Materials Science and Engineering*, vol. 1055, no. 1, Feb. 2021, doi:10.1088/1757-899X/1055/1/012051.

[5] A. I. Ali, S. Z. Partal, S. Kepke, and H. P. Partal, "ZigBee and LoRa based Wireless Sensors for Smart Environment and IoT Applications," in *2019 1st Global Power, Energy and Communication Conf.*, 2019, pp. 19–23, doi: 10.1109/GPECOM.2019.8778505.

[6] S. Ko *et al.*, "LoRa network performance comparison between open area and tree farm based on PHY factors," in *2018 IEEE Sensors Applications Symp.*, 2018, pp. 1–6, doi: 10.1109/SAS.2018.8336763.

[7] G. Codeluppi, A. Cilfone, L. Davoli, and G. Ferrari, "LoRaFarM: A LoRaWAN-Based Smart Farming Modular IoT Architecture," in *Sensors*, vol. 20, no. 7, Apr. 2020, pp. 2028, doi: 10.3390/s20072028.

[8] T. S. Mohammed, O. F. Khan, and A. Al-Bazi, "A Novel Algorithm Based on LoRa Technology for Open-Field and Protected Agriculture Smart Irrigation System," in *2019 2nd IEEE Middle East and North Africa COMMunications Conf.*, 2019, pp. 1–6, doi: 10.1109/MENACOMM46666.2019.8988583.

[9] M. Lekić and G. Gardašević, "IoT sensor integration to Node-RED platform," in *2018 17th Int. Symp. INFOTEH-JAHORINA*, 2018, pp. 1–5, doi: 10.1109/INFOTEH.2018.8345544.

[10] J. R. dela Cruz, R. G. Baldovino, F. B. Culibrina, A. A. Bandala, and E. P. Dadios, "Fuzzy-based Decision Support System for Smart Farm Water Tank Monitoring and Control," in *2017 5th Int. Conf. on Information and Communication Technology*, 2017, pp. 1–4, doi: 10.1109/ICoICT.2017.8074669.

[11] A. L. P. de Ocampo and E. P. Dadios, "Energy cost optimization in irrigation system of smart farm by using genetic algorithm," in *2017IEEE 9th Int. Conf. on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management*, 2017, pp. 1–7, doi: 10.1109/HNICEM.2017.8269497.

[12] J. Liu, T. Hu, P. Feng, L. Wang, and S. Yang, "Tomato yield and water use efficiency change with various soil moisture and potassium levels during different growth stages," in *PLOS ONE*, vol. 14, no. 3, Mar. 2019, pp. 1–14, doi : 10.1371/journal.pone.0213643.

[13] Jacob N. Barney, J. Jeremiah Mann, Guy B. Kyser, Eduardo Blumwald, Allen Van Deynze, and Joseph M. DiTomaso, "Tolerance of switchgrass to extreme soil moisture stress: Ecological implications," in *Plant Science*, vol. 177, Dec. 2009, pp. 724–732, doi: 10.1016/j.plantsci.2009.09.003.

[14] Jongsoon Kim, Wonsik Choi, Kiyeol Jung, Sanghun Lee, JongMin Park, SoonGu Kwon, Donghyun Kim, and Soonhong Kwon, "Development of Soil Moisture Controlling System for Smart Irrigation System," in *Journal of The Korean Society of Industry Convergence*, vol. 21, no. 5, 2018, pp. 227–234.