water

MDPI

Article

Utilizing the Internet of Things (IoT) to Develop a Remotely Monitored Autonomous Floodgate for Water Management and Control

Sami Salama Hussen Hajjaj [1,*], Mohamed Thariq Hameed Sultan [2,3,4,*], Muhammad Hafizuddin Moktar [5] and Seng Hua Lee [6]

1 Centre for Advanced Mechatroics and Robotics (CaMaRo), Universiti Tenaga Nasional (UNITEN), Jalan IKRAM-UNITEN, Kajang 43000, Selangor Darul Ehsan, Malaysia
2 Laboratory of Biocomposite Technology, Institute of Tropical Forestry and Forest Products (INTROP), Universiti Putra Malaysia, UPM Serdang 43400, Selangor Darul Ehsan, Malaysia
3 Department of Aerospace Engineering, Faculty of Engineering, Universiti Putra Malaysia, UPM Serdang 43400, Selangor Darul Ehsan, Malaysia
4 Aerospace Malaysia Innovation Centre (944741-A), Prime Minister's Department, MIGHT Partnership Hub, Jalan Impact, Cyberjaya 63000, Selangor Darul Ehsan, Malaysia
5 Department of Mechanical Engineering, Universiti Tenaga Nasional (UNITEN), Jalan IKRAM-UNITEN, Kajang 43000, Selangor Darul Ehsan, Malaysia; hafizskai07@gmail.com
6 Laboratory of Biopolymer and Derivatives, Institute of Tropical Forestry and Forest Products (INTROP), Universiti Putra Malaysia, UPM Serdang 43400, Selangor Darul Ehsan, Malaysia; lee_seng@upm.edu.my
* Correspondence: ssalama@uniten.edu.my (S.S.H.H.); thariq@upm.edu.my (M.T.H.S.)

Received: 15 December 2019; Accepted: 10 January 2020; Published: 12 February 2020

Abstract: In recent years, floods have increased in frequency and intensity, causing tremendous hardship. In badly affected regions, mostly the rural areas, Weir-type floodgates are the only measure against floods. However, these manually operated gates are numerous and scattered over vast areas. This makes flood mitigation efforts very challenging, which causes severe devastation. Current solutions to automate the floodgates are expensive, black-boxed, and focused on individual gates. In this paper, we present a Centralized Flood Monitoring and Coordination System developed through the Internet of Things (IoT) and other open-source technologies. For this work, we developed a working prototype of an autonomous floodgate that opens/closes according to the level of water. We also developed the required program to allow the gate controller to publish its data through the IoT gateway to the cloud. The data was then captured and viewed on a number of IoT clients, both for individuals and groups of floodgates, in real time. The developed system proved successful as the autonomous gates were monitored remotely through the established IoT framework, with room for future development and improvement. This paper serves as a proof of concept and a preparation for real, on-site implementation of the IoT-floodgates.

Keywords: Internet of Things (IoT); floodgates; watergates; flood management; open-source development; network-ready controllers; Raspberry Pi systems; open-source IoT clients

1. Introduction

In recent years, floods have increased in frequency and intensity all over the world, causing tremendous hardship in affected areas. This has increased the importance of flood mitigation measures and technologies, such as floodgates [1,2]. Floodgates vary in size and complexity; large and sophisticated gates can be found in hydro-dams, while small and simple ones are found on rivers or canals in rural regions, which are usually the most badly hit regions by floods [3]. Crude and simple in

design, these manually operated gates prevent flash floods, protect paddy fields, and prevent damage to properties and local infrastructures [3]. However, these gates are numerous and often scattered over vast rural areas, making flood mitigation efforts and coordination very challenging. As a result, seasonal floods usually come with severe devastation in these regions [4,5].

In the current practice, there are a few main challenges that need to be addressed. The delivery of water gate information and other details from different locations to the management or the manager through the site workers is rather time-consuming. This leads to delayed decision-making from the manager, especially for the water gates that are significantly related and connected to each other. For example, the opening and closing of water gates that are installed in a line on the waterway will change the water level at the next area and the management of the next gates. Hence, quick information updates from each water gate is important in water management.

Furthermore, the manager needs an overall view from all the regions of the waterways to easily manage the water gates. In other words, all real-time information needs to be received by the management body at the same time to make the right responses with water gate controls. Currently, satellites provide the forecast data for water management and preparation, but the situation also needs real-time data for quick actions.

Hence, water management needs to be improved with quick information updates and full real-time monitoring. Apart from these aspects, the management also needs to be able to directly control or override the water gates from the management center. An automation system is not the only required function for management, but it also needs to be interactive so the manager can take action on the water gates based on the full picture from the monitoring system.

The proposed solution in this paper is able to manage the challenges above, and provide a quick information update of the water level and the gates, all information from multiple locations, and an action button function from the client center.

2. Previous Solutions

Some mechanisms to automate floodgates do exist [6–8], but these efforts are expensive and usually focused on automating individual gates. Furthermore, they do not allow centralized monitoring of a group of gates, something the flood mitigation authorities really need as they usually monitor tens or even hundreds of floodgates in a region.

Some previous systems only applied monitoring of water level or other water parameters. These systems have no automation function for water gate opening and closure. Through different methods and focuses including GPRS communication [9–11], with their own developed system [12–14], and water quality monitoring system [15–17], these solutions were only applied for water monitoring. However, these systems still need human labor to control the water gate and take the necessary action at the locations. Although some solutions managed to connect multiple nodes from rivers and reservoirs [18,19], the centralized function is focused only on monitoring water with no online gate control.

Other solutions combined the water gate automation and monitoring function [20–22]. The gate is automatically opened and closed by the system based on the warning level of water; then, the particular information is visualized in the wireless system. However, the override button from the client interface is not available, which is the most important feature or function required by the manager to decide when to control the water gates.

In this paper, those functions from different solutions are combined in one system in order to provide a complete function for the user, especially in these aspects: quick data update, full picture of IoT clients, and interactive command button. None of the aforementioned solutions evaluated the performance through these aspects, but only provided the online interface of the system with other types of performance evaluation.

3. Methodology

This section presents the steps needed to develop, implement, and verify the proposed system. The main steps needed to develop the proposed system include the following elements:

1. Infrastructural requirements: Setting up the electrical power and connectivity at target floodgates.
2. Floodgate automation: Installing the needed sensors, actuators, and network-ready controllers.
3. Broadcasting gate data: Programming the controllers to publish their gate data to the cloud.
4. Viewing gate data: Capturing, processing, and viewing the data through the IoT framework.
5. Data interpretation: Evaluating the system performance with the data output.

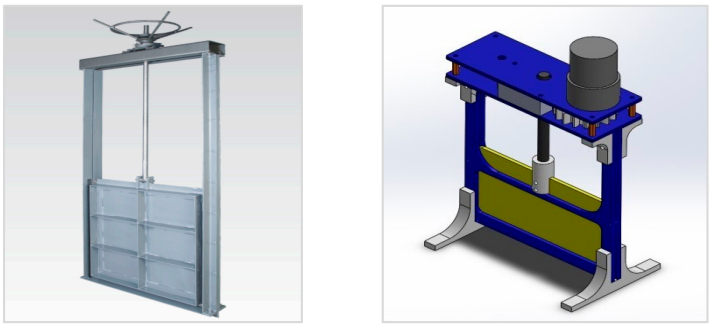
3.1. Infrastructural Requirements

Power requirements depend on the gate size and the torque needed to lift it up, as well as the frequency of fluctuation of the water levels and, therefore, the need to raise and lower the gate [23,24]. A solar cell can be installed near the gates, or it can be connected to the grid. Connectivity can be established through wireless hotspots, which have become more capable and cost-effective in recent years. Wired connection could also be established by extending the LAN cables along with the power cables discussed above.

This step could be the most challenging of all due to the expected costs. However, the authorities and city officials could consider this as a long-term investment for their region. Aside from enabling the proposed IoT-floodgates, the established infrastructure (extended power and connectivity) would greatly benefit the local communities. It could help boost commerce, enhance social interaction, and further improve development in these regions.

3.2. Floodgate Automation

There are different types of floodgates and mechanisms; the most common type is the Weir gate mechanism, shown in Figure 1. In this mechanism, a power screw is used to lift the gate door up or down. The screw is turned using a wheel mounted on top of the gate. For the purpose of this paper, a working prototype of this mechanism was developed and built by the authors [25,26]. This weir-type gate has been commonly used in other works [27–29].

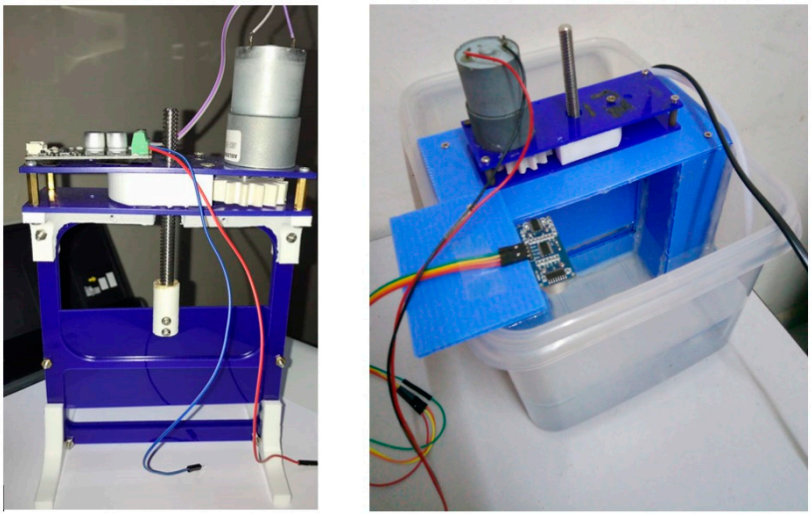


(**a**) (**b**)

Figure 1. Weir-type floodgate, (**a**) actual [25] vs. (**b**) the prototype developed for this work [26].

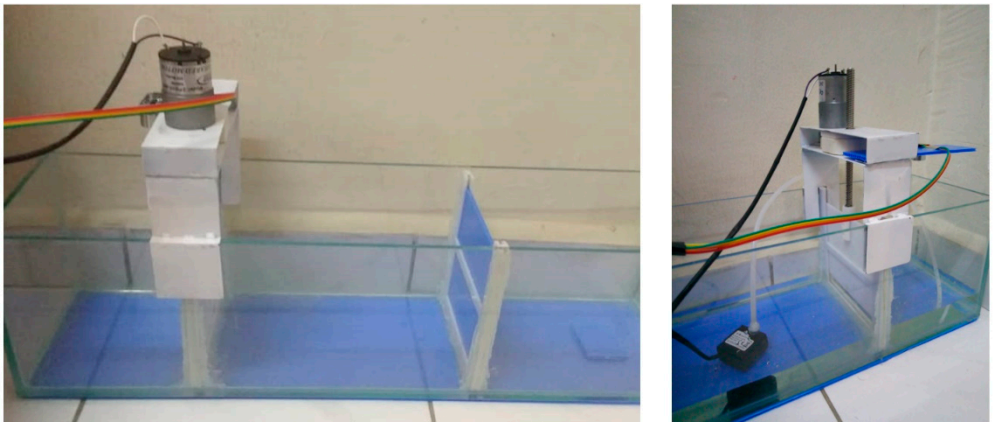
The developed prototype shows how this mechanism could be automated. The screw is connected to an electrical motor through a gearing system. The motor is controlled by a network-ready controller, which is also connected to the sensors measuring the water level and facilitating the gate automation [30].

Three water gate prototypes were developed in this work to demonstrate the centralized system with the full information. The two water gate models are in-lab models that were basically modified from an early stage to improve the gate stability after frequent usage and to make it able to control the water flow. Both gates were named Gate V1 and Gate V2 as shown in Figures 2 and 3.



(**a**) (**b**)

Figure 2. Actual model of gate: (**a**) at first stage vs. (**b**) the modified model, Gate V1.



(**a**) (**b**)

Figure 3. (**a**) Gate V2 vs. (**b**) the water test on Gate V2 with the water pump.

The third prototype, Gate V3, is a large-scale prototype that was built on the site at the Malaysian Agricultural Research and Development Institute (MARDI) to test the prototype with actual water flow (Figure 4). This prototype consisted of a 5 kg sliding gate, steel platform, and linear actuator with 150 N payload. The three water gate prototypes demonstrate multiple gate nodes at different locations. The data from those locations will be transferred to the cloud through a wireless network-ready controller. The hardware components of these three prototypes are shown in Figures 5–7.

(**a**) (**b**)

Figure 4. (**a**) In-field prototype, Gate V3 vs. (**b**) the water flow during Gate V3 opening.

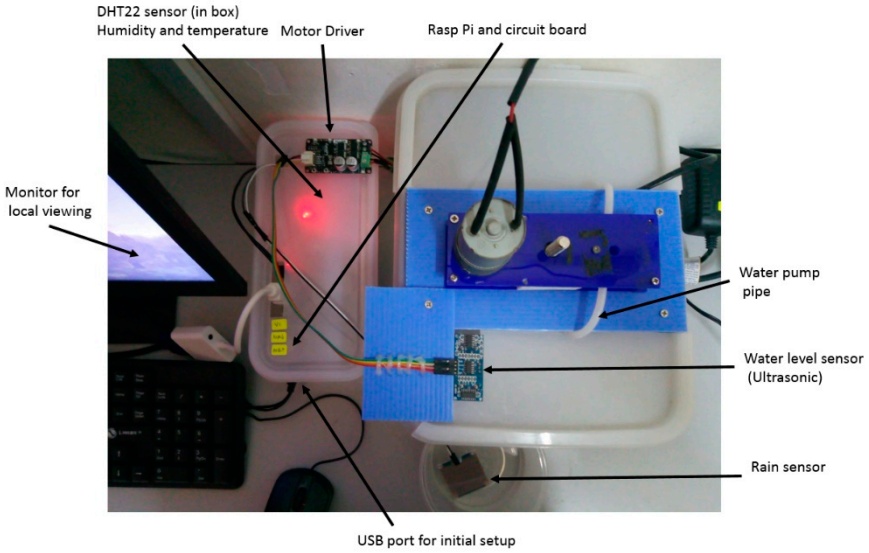


Figure 5. Gate V1 hardware components.

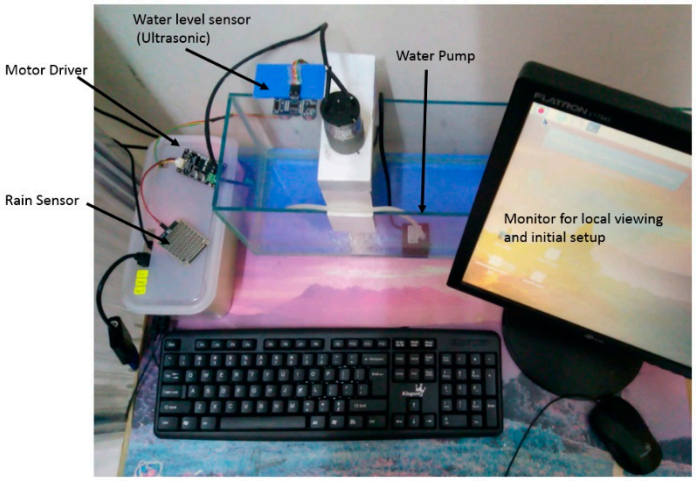


Figure 6. Gate V2 hardware components.

Figure 7. Gate V3 hardware components.

3.2.1. Capturing Water Level

To measure the water level of the river/canal, the HC-SR04 ultrasonic sensor was used. This sensor shoots an ultrasound ping in the direction of an object, and by measuring the time it takes for its echo to return, the distance between the sensor and the object can be found, as shown in Figure 8.

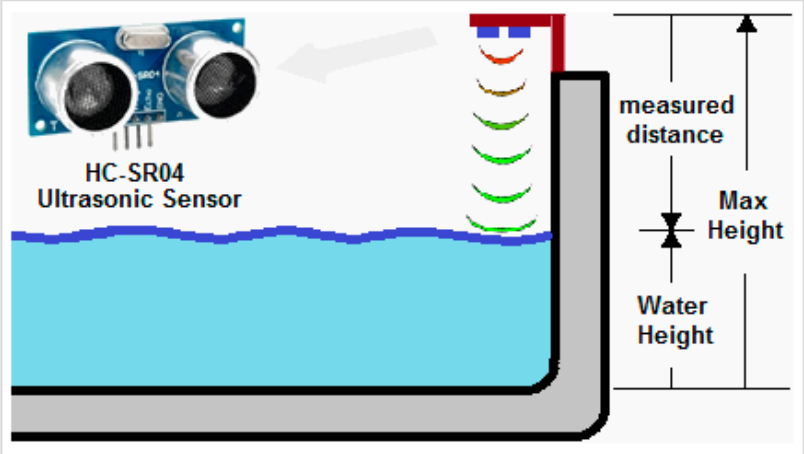


Figure 8. The HC-SR04 sensor used to measure the water level.

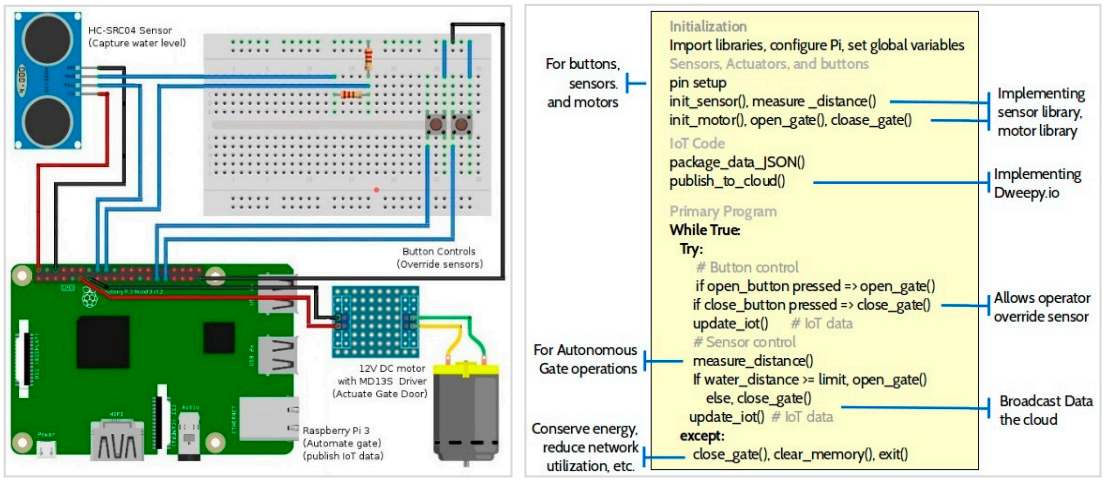
As successfully demonstrated by Texas Instruments [31], when the sensor is positioned to face the water (by being placed on a pole or a similar fixture), the water height can be found. The frequency of the ultrasonic ping can be set programmatically, as discussed below. Several works were done by using the ultrasonic sensor to measure the water level [14,18,32].

For the sensor part, the type of the sensor is not the highlight of this work; it can be modified and improved with other types of sensors that are more suitable and accurate based on the condition of the location. In addition to that, the component itself is available and is not the unique method in this work. However, the utilization of the components with an IoT framework are the main focus to achieve the centralized and interactive system.

3.2.2. Controller Selection and Programming

For the controller, the authors opted to use Raspberry Pi 3 (RPi). The RPi is a network-ready, cost-effective computer-on-chip. This allowed the authors to program it to perform the tasks of the proposed system: capturing and processing sensor data, actuating the gate motor, and publishing the

gate data to the cloud. The Arduino does not have the network capability without a physical network adaptor [33,34]. Figure 9a shows a schematic diagram of the developed system.



(**a**) (**b**)

Figure 9. Setup for the Raspberry Pi controller, (**a**) setup of the hardware components, (**b**) Controller program algorithm.

Aside from connecting all the input/output devices, the RPi has a built-in network adaptor, which allows it to connect to the Internet via Wi-Fi, a cable, or Bluetooth; the authors opted for a Wi-Fi connection. Figure 9b shows the algorithm of the RPi controller program developed for the purpose of this work. The figure shows only the pseudo code, methods, and libraries used. The actual program can be found in [35]. Excerpts from the developed program are shown in Figures 10–13. The figures show the key elements of the developed code, namely the sensor data processing, motor control, and IoT.

```
def measure_distance():
    # The actual echo
    GPIO.output(trig, True)
    sleep(0.00001)
    GPIO.output(trig, False)

    global echo_start, echo_end
    while GPIO.input(echo)==0:  echo_start = time()
    while GPIO.input(echo)==1:  echo_end = time()

    echo_duration = echo_end - echo_start
    distance = round((echo_duration * 17150),2)
    init_sensor()

    return distance
```

Figure 10. Code Excerpts: using sensor data to measure water level.

$$34300 = \frac{Distance}{Time/2}$$

$$17150 = \frac{Distance}{Time}$$

$$17150 \times Time = Distance$$

Figure 11. Distance calculation for ultrasonic sensor [36].

```
def open_gate(gate_speed, duration):
    global gate_openned
    if not gate_openned: # if gate is already closed
        gate_dir = GPIO.output(motor_dir, GPIO.HIGH)
        GPIO.output(motor_dir, GPIO.HIGH)
        pulse_rate.start(gate_speed)
        sleep(duration)
        pulse_rate.start(0)
        gate_openned = True
        print("Gate Openned")
    else: print("Gate already Openned")
```

Figure 12. Code Excerpts: controlling motor to open the gate.

```
# IoT code
def update_iot(gate_id, water_height, gate_status):
    # package data in a JSON object
    iot_data = {
        'Water Level': round(water_height,2)
        ,'Gate Status': gate_status
        ,'Gate ID': gate_id
        }
    dweepy.dweet_for('iotfloodgate_'+str(gate_id), iot_data)
```

Figure 13. Code Excerpts: using open-source IoT libraries and services, such as Dweet.io.

As seen in Figure 10, the sensor shoots the ping for just 0.00001th of a second and then, it measures the time for the echo to return. Next, knowing that the speed of the ping is 343 s (speed of sound at sea level), as also provided by the sensor manufacturer, the distance can then be calculated. The time needs to be divided by two because the echo_duration is the time of the ultrasonic pulse travel in one trip to the water and back again. Hence, the distance calculation in the python script follows the method in Figure 11. This, in turn, facilitates the opening and closing of the water gate.

Figure 12 shows a snippet of the program that controls the opening of the gate, namely the open_gate() function. As seen in the figure, the open_gate() function first checks if the gate is not already opened; only then, it would proceed by turning the motor in the direction needed to lift the gate up, clockwise or counter-clockwise, based on its mechanical configuration.

As for the gate data, which includes gate id, status, water level, and other related data, it is broadcasted through the IoT framework using the update_iot() function, as shown in Figure 13.

The gate_id is a unique number assigned to each gate to differentiate it among other gates in the target region. After packaging the gate data in a JSON (JavaScript Object Notation) object, which is a data format understood by all cloud services, browsers, and databases, the data is then published to the cloud through Dweet.io, an open-source IoT service, as discussed in detail in the next section.

3.3. Broadcasting Gate Data

In order to allow the stakeholders to remotely monitor the floodgates through the IoT framework, the controller needs to publish its data to the cloud. The process is shown symbolically in Figure 14. As shown in the figure, publishing gate data to the cloud involves the following processes:

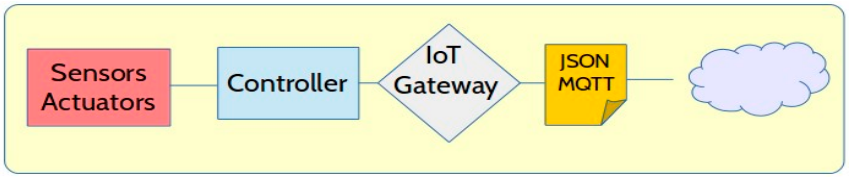


Figure 14. Publishing gate data to the cloud.

Restructuring the data: As the data is to be transferred over webservers and clients, it needs to be packaged in one of the two standard IoT messaging formats: the JSON format, as shown in Figure 13, or the MQTT (message queuing telemetry transport) protocol [37,38]. There are operational differences between the two formats, but most IoT clients/services can handle both.

IoT gateway: It is a network-ready device that connects the controller to IoT services and applications. For the Raspberry Pi, which is network-ready, there is no need for a separate device, but rather a configuration patch, as discussed in [38]. For non-network-ready controllers, such as the popular Arduino controllers, a physical gateway is needed.

Publishing web-ready data to the cloud: Once the networking setup is completed and the IoT protocols are implemented, the JSON or MQTT formatted data are published to the cloud by hard-coding the JSON or MQTT publish commands into the controller program. This can be further automated using a number of IoT data publishing services, such as Dweet.io. As seen in Figure 13, with Dweet.io, all the programmer needs to do is declare the structure of the data. Be it JSON, MQTT, or others, Dweet.io (through its Python client, Dweepy), will do the rest; it would Dweet, or Data Tweet the needed information.

3.4. Viewing Gate Data

Once converted to a web-ready format (JSON, MQTT, or others), the gate data can then be viewed in any client that supports and understands these formats. This includes most operating systems, web-applications, as well as android and iOS apps, as shown in Figure 10. Once again, a web developer or an app developer could create a custom-client to view the IoT data or utilize any of the available open-source IoT solutions for viewing data, such as ThingsBoard.io, FreeBoard.io, or others; both approaches have advantages and disadvantages. For this work, especially in the early testing stages, the authors opted to use FreeBoard.io, as shown in Figure 15.

Using Freeboard.io is similar to using social media such as Facebook. The developer logs into Freeboard, adds the data source, and selects the viewing options. Referring to Figure 9, the multiple gates are shown with different names: Gate V1, V2, and V3. From the cloud, the information including water level, gate status, temperature, humidity, and location of those water gate prototypes are published in Freeboard.io.

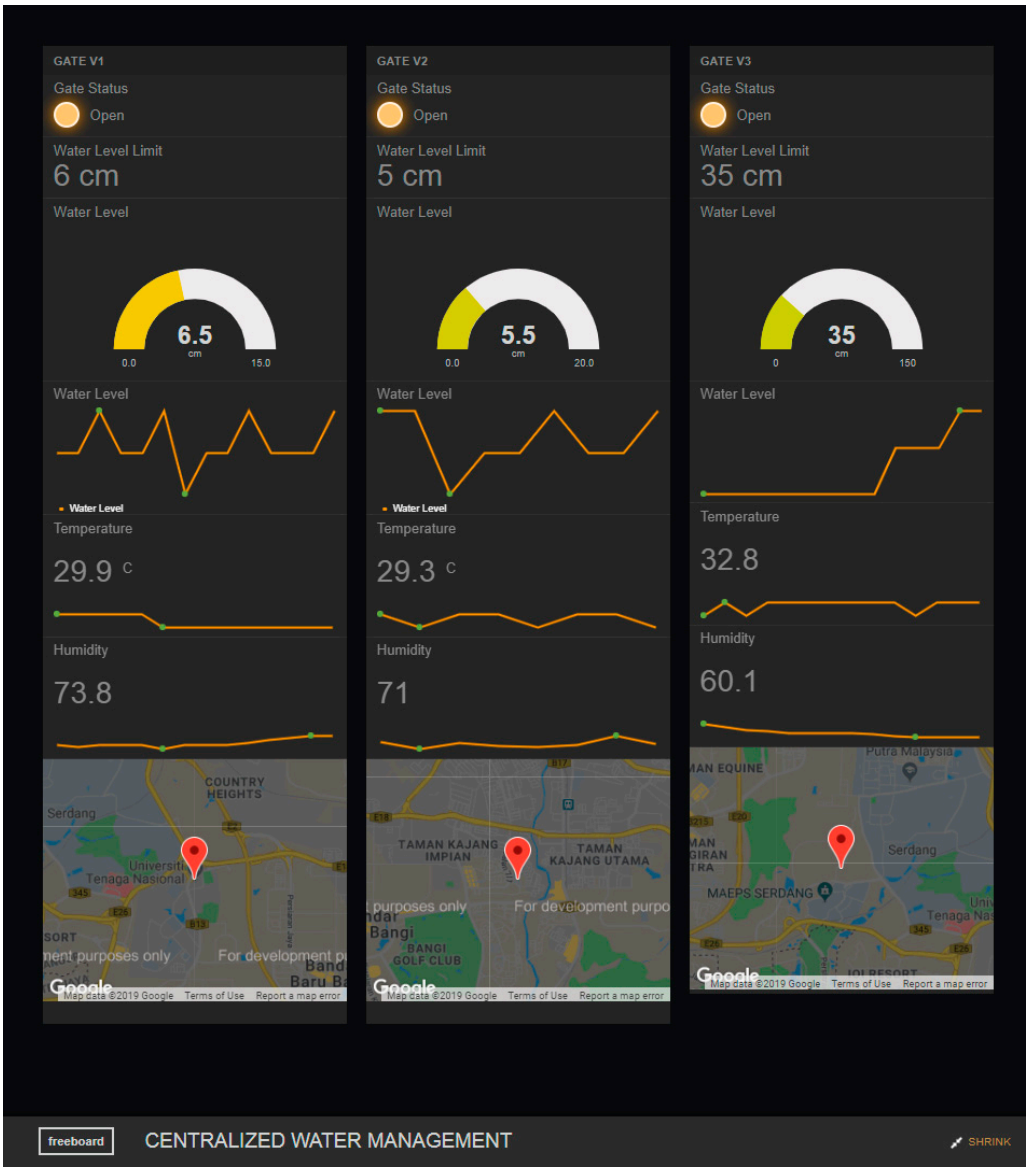


Figure 15. Publishing gate data to the cloud.

3.5. Data Interpretation

To prove that the centralized water management system is effective, the data must be collected, and the developed system needs to be evaluated from the data. There are three main parameters that need to be tested on the water gates and the system, which are the data update rate, percentage of data completion, and response time (Figure 16).

Data update rate is the delay time of the system when the data from the controller terminal is updated to the IoT Client, Freeboard.io. The average data update rate of each water gate is calculated from the collected data of the delay time of the water level update. From that result, the authors can evaluate how fast the developed centralized water monitoring system functions compared to the current practice as well as the previous solutions.

The percentage of the data completion is evaluated by how many features/data in the IoT Client are updated compared to the number of the total aspect features in one system. The IoT Client should demonstrate 100% of data completion to show the full picture of the monitored data.

The response time is the opposite of the data update rate, which is the time of signal update from the developed open and close buttons from the IoT Client to the controller of each water gate. In other words, it is the time between clicking the override button and the motor response.

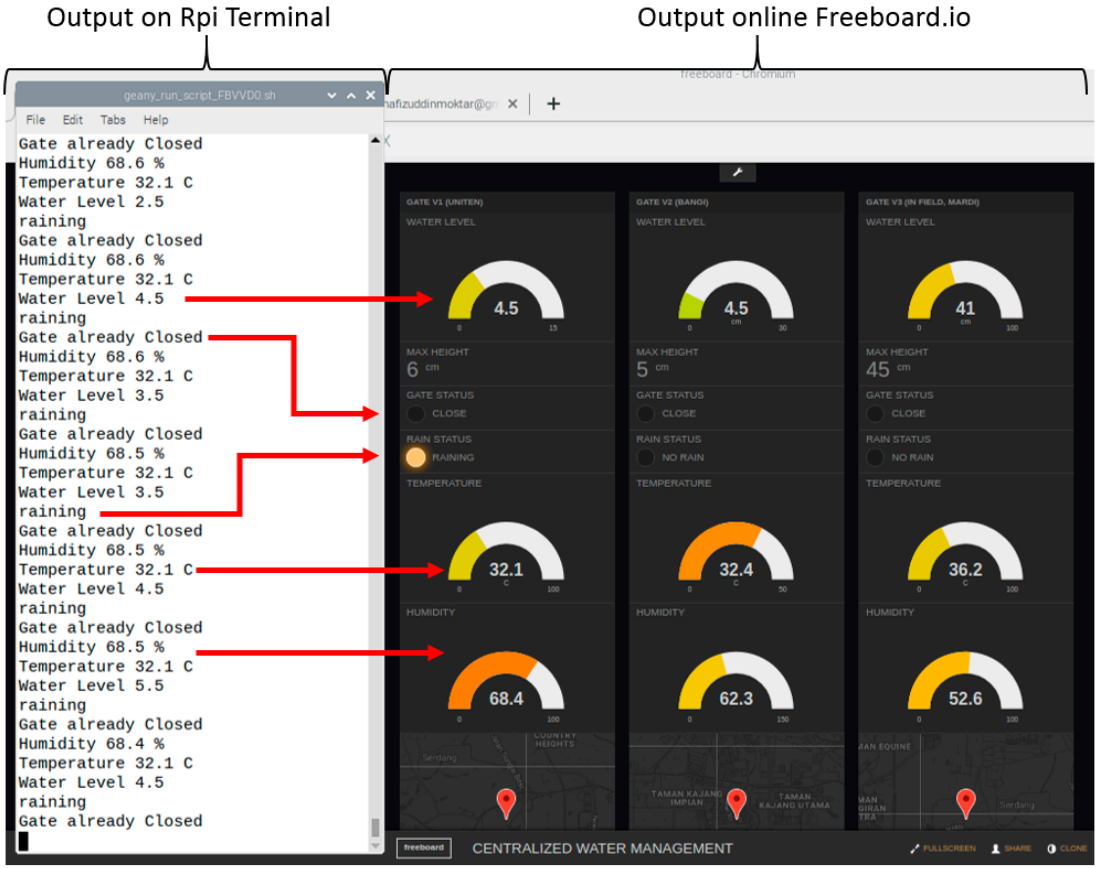


Figure 16. Time interval would be calculated for the data transmission between the local Raspberry Pi terminal and the Freeboard.io.

4. Results and Discussion

4.1. Data Update Rate

The data below were collected from the three different water gates: Gates V1, V2, and V3. Gates V1 and V2 were from the lab location and Gate V3 was from the in-field location. For every water level update and gate status update, the update rates were collected during a few cycles of gate opening and closure.

In Figure 17, Gate V1 opened four times with a longer update rate of over 7 s compared to the data update rate when it is closed. The opening of the gate takes a few seconds before updating the data to the IoT Cloud. In the IoT framework, the open command was set with 5 s of the motor rotation. The average data update rate was calculated as 2.57 s for every update.

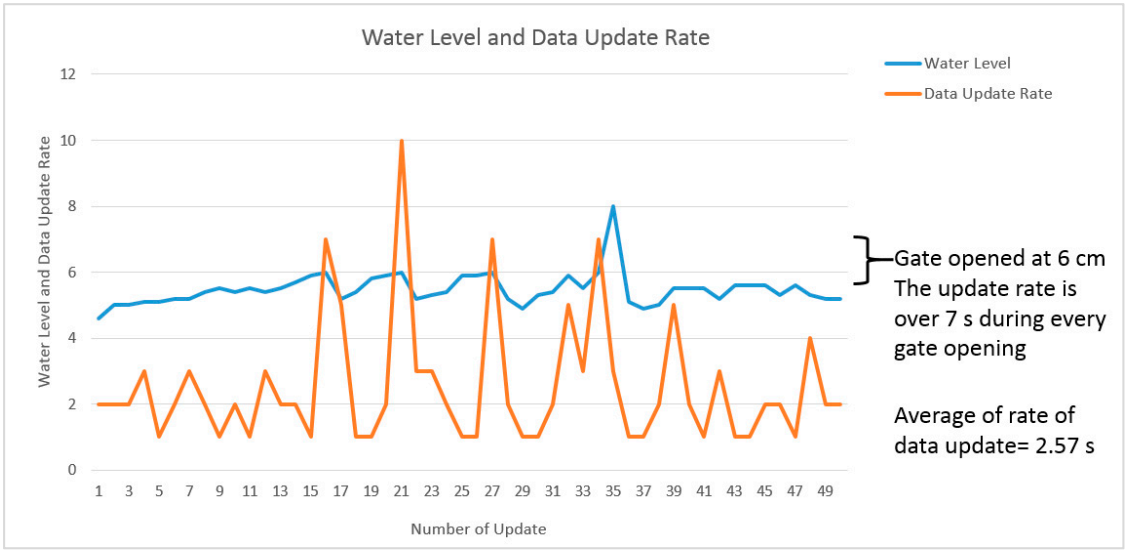


Figure 17. Gate V1 water level and data update rate.

Similarly, as shown in Figure 18, some data update rates of the in-lab Gate V2 were higher during the gate-opening phase because the motor rate was set for a few seconds to ensure the gate was fully opened. As the water level was higher than 5 cm, the water gate opened to reduce the water level. The average update rate for this gate was 2.93 s per update.

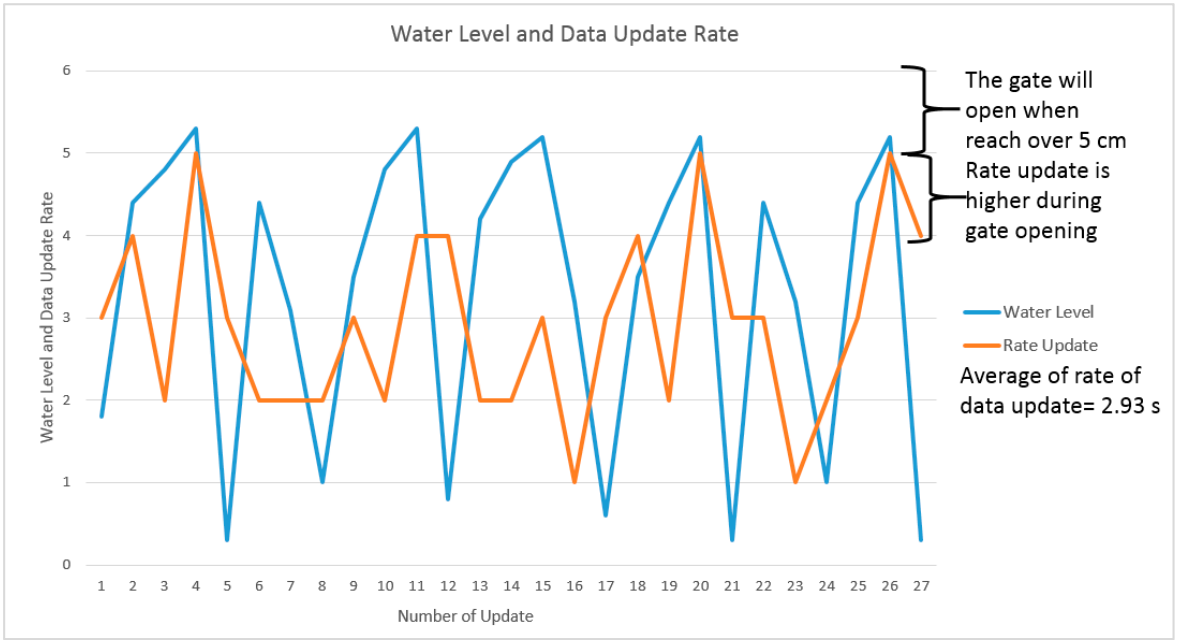


Figure 18. Gate V2 water level and data update rate.

Figure 19 shows the three long cycles of the water gate opening and closure. This in-field water gate updated the water level data together with other information with an average of 2.14 s of update rate, from the controller terminal to the IoT Client. The maximum water level was set at 45 cm, which caused the water level to increase slowly. During the closed phase, the water level increased over time and reached the maximum limit, and then the gate opened with 10 s of the motor rate.

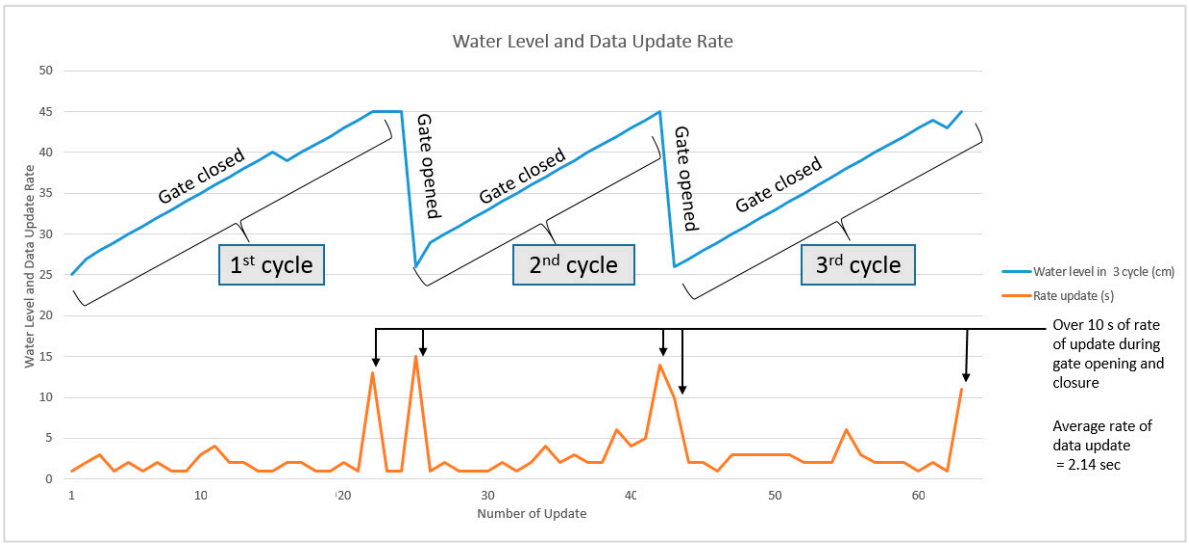


Figure 19. Gate V3 water level and data update rate.

From all of the data update rates of those three water gates, the authors can conclude that the data transmission from each location to the Freeboard.io only takes less than 3 s, which proves that the developed water management system can monitor the information faster than the current practice, which is through human updates to the management body.

4.2. Percentage of Data Completion

In this subsection, the results are only the images from the IoT client that show the updated features in one time. The images show whether the data completion is full or partial.

Figure 20 shows the update was only from one active node at Gate V3, including the data of water level (gauge and level trend), temperature, and humidity of air. The other two nodes only showed the previous update of the water level, implying that the status of both gates were not active at that time. The IoT Client in Figure 14 only demonstrates that 1/3 of the required data were updated from the controllers. Thus, the data completion was only 33.33%.

Figure 20. 1/3 of data completion from the IoT Client.

Compared to Figure 21, only two gates updated the particular data, which were Gate V2 and V3. Gate V1 stopped updating after publishing the information to the cloud several times. Although the water pump was switched on, the water level did not increase over time because of the disconnected system. The IoT Client from Freeboard at that time showed 2/3 of data completion.

Figure 21. 2/3 of data completion from the IoT Client (Gate V1 stop updating).

Figure 22 demonstrates the full picture of the IoT Client, which showed all parameter updates from different nodes of the water gates including water level, gate status, temperature, humidity, and location of the individual gate. Hence, the IoT Client basically showed 100% of data completion when each node actively updated the information through wireless connection. From that, the authors can state that this developed system is suitable for the user from the water management to get a full picture of all the water properties.

Figure 22. 100% of data completion from the IoT Client.

4.3. Response Time

Response time is the rate of the signal from the IoT Client to the controller at the water gate location. The override button is under development by using Netpie.io (Figure 23), another ready-made IoT Client that can provide the button feature compared to Freeboard.io. The response time is the same as the rate of data update because both conduct a two-way data transmission. Hence, the response time also averages between 1 to 3 s in order to send the open and close command from the IoT Client to the controllers.

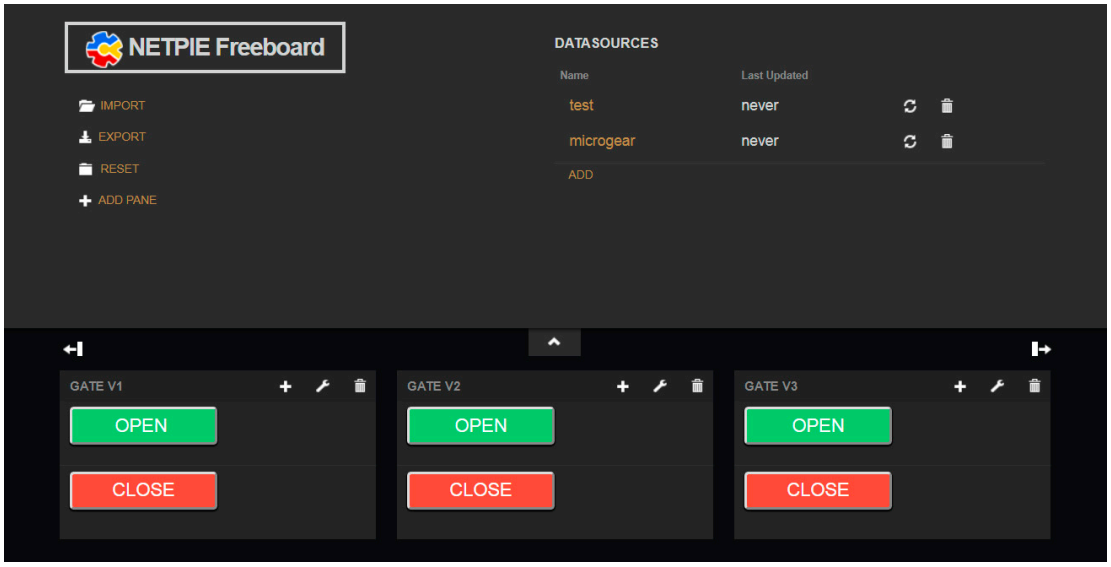


Figure 23. Button function in NETPIE.io.

From the results presented in Sections 4.1–4.3, the authors can conclude that the developed system manages water through multiple water gates. The developed system is centralized because it can monitor the related information of different water gates in real-time with a quick data update and a full picture from the IoT Client. Additionally, the system not only implemented an automation feature, but also provided a two-way communication that facilitates the interaction between the client and the water gates. The client can control the water gates after getting and evaluating the full picture from the IoT Client, through the override button that sends the command to each water gate. Thus, the developed water management system is not only centralized and automated, but also interactive.

5. Conclusions

This paper presents the development steps needed to utilize the IoT to develop a Centralized and Interactive Water Management system. Apart from the cost of equipment, the technology used is completely open-source, and thus, it is open for customization, extension, and future development and improvement. The HC-SR04 ultrasonic sensor was used to measure the level of water in real time. This information was captured by the Raspberry Pi, where it was packaged in JSON format, and published to the cloud.

Extensive use of IoT solutions and services was implemented in this work. However, a custom IoT client was developed for the centralized monitoring platform. This combination made it possible to develop parts of the proposed system.

The success of the working prototype provided a working ground for the on-site implementation and testing. Aside from the hardware implementation, more robust sensor choice and infrastructural setup, the developed software, IoT solutions, and networking protocols remain the same.

There is plenty of room for improvement in this work: more sensors could be added to capture more relevant data, and remote control of the gates could be developed once the required networking

setup is achieved. Other than that, the power source setup can be improved by adding a solar power system after the power usage research, battery consideration and transformer selection.

In the real application, the system should use a suitable sensor that can withstand the environmental effect, especially the rain and storm effect. Thus, the selected sensors need to be waterproof for deployment.

The available data transmission technologies can be selected based on the situation of the application/field, the advantages and shortcoming of the technologies. Some aspects need to be considered including the cost, range, and battery life. This project was started with Wi-Fi technology in the early stage for the water gate test. For future works, the utilization of LoRa technology is expected in order to manage multiple gates that are scattered over a large area.

Author Contributions: Conceptualization, S.S.H.H. and M.H.M.; Methodology, S.S.H.H. and M.H.M.; Validation, S.S.H.H. and M.T.H.S.; Formal analysis, S.S.H.H. and M.H.M.; Investigation, M.H.M.; Resources, S.S.H.H. and M.T.H.S.; Data curation, S.S.H.H. and M.H.M.; Writing—original draft preparation, S.S.H.H., M.T.H.S., M.H.M. and S.H.L.; Writing—review and editing, S.S.H.H., M.T.H.S. and S.H.L.; Visualization, S.S.H.H., & M.H.M.; Supervision, M.T.H.S.; Project administration, M.T.H.S.; Funding acquisition M.T.H.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Higher Education Center of Excellence (HICoE), Ministry of Higher Education, Malaysia.

Acknowledgments: This work is supported by UNITEN and UPM under GP-IPS grant (vote: 9647100) and Higher Institution Centre of Excellence (HICoE). The authors would like to express their gratitude and sincere appreciation to the Centre of Advanced Mechatronics and Robotics (CaMaRo), Universiti Tenaga Nasional, Department of Aerospace Engineering, Faculty of Engineering, Universiti Putra Malaysia and Laboratory of Biocomposite Technology, Institute of Tropical Forestry and Forest Products (INTROP), Universiti Putra Malaysia (HiCOE) for their close collaboration in this project.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Guha-Sapir, D.; Philippe, H.; Wallemacq, P.; Below, R. Annual Disaster Statistical Review 2016. Available online: http://emdat.be/sites/default/files/adsr_2016.pdf (accessed on 26 September 2019).
2. United Nations Office for Disaster Risk Reduction. Global Assessment Report on Disaster Risk Reduction 2019. Available online: https://gar.unisdr.org/sites/default/files/reports/2019-05/full_gar_report.pdf (accessed on 23 September 2019).
3. Eco-Business Research. Flood Controls in Southeast Asia. Available online: http://www.eco-business.com/media/uploads/magazine/flood_controls_in_southeast_asia_2017.pdf (accessed on 1 October 2019).
4. Sonali, S.S. Wastewater in Vientiane: Natural and Built Solutions. Available online: http://sea.iwmi.cgiar.org/2017/03/22/wastewater-vientiane-natural-built-solutions/ (accessed on 2 October 2019).
5. Masseroni, D.; Moller, P.; Tyrell, R.; Romani, M.; Lasagna, A.; Sali, G.; Facchi, A.; Gandolfi, C. Evaluating performances of the first automatic system for paddy irrigation in Europe. *Agric. Water Manag.* **2018**, *201*, 58–69. [CrossRef]
6. Saha, G.; Parua, A.; Sushmitha, R.; Bhat, S. Automatic floodgates control using PLC with added focus on human safety. In Proceedings of the 2015 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), Kumaracoil, India, 18–19 December 2015; pp. 417–422.
7. Johnston, S.; Kroon, F.; Slavich, P.; Cibilic, A.; Bruce, A. Restoring the Balance: Guidelines for Managing Floodgates and Drainage Systems on Coastal Floodplains. Available online: http://www.dpi.nsw.gov.au/__data/assets/pdf_file/0007/167875/restoring-balance-guidelines.pdf (accessed on 1 October 2019).
8. LEGO Education. Design an Automatic Lego Floodgate to Control Water According to Various Precipitation Patterns. Available online: https://education.lego.com/en-us/lessons/wedo-2-science/prevent-flooding (accessed on 16 September 2019).
9. Liu, C.; Shao, W.; Zhang, L.; Chen, H.; Jiag, J.; Xiao, Z.; Meng, Y.; Liu, Z.; Chen, Y.; Li, G. Water Level Early-Warning Monitor for Power Grid Flood Control. Application Patent CN203672450U, 25 June 2014.
10. Zhu, X.; Liu, M.; Xia, X. Reservoir Flood Control Monitoring System Based on GPRS Communication and Internet. Application Patent CN105259887A, 20 January 2016.

11. Yamakawa, K. Water Gate Remote Monitoring System. Patent JP2002190079A, 5 July 2002.
12. Li, B. Farmland Water-Saving Irrigation Monitoring Management Device. Application Patent CN201328292Y, 21 October 2009.
13. Nguyen, T.-D. Energy efficient wireless sensor network and low power consumption station design for an urban water level monitoring system. In Proceedings of the 2016 3rd National Foundation for Science and Technology Development Conference on Information and Computer Science (NICS), Danang, Vietnam, 14–16 September 2016; pp. 252–256.
14. Dhandre, N.M.; Kamalasekaran, P.D.; Pandey, P. Dam parameters monitoring system. In Proceedings of the 2016 7th India International Conference on Power Electronics (IICPE), Patiala, India, 17–19 November 2016; pp. 1–5.
15. Nasser, N.; Ali, A.; Karim, L.; Belhaouari, S. An efficient Wireless Sensor Network-based water quality monitoring system. In Proceedings of the 2013 ACS International Conference on Computer Systems and Applications (AICCSA), Ifrane, Morocco, 27–30 May 2013; pp. 1–4.
16. Vacariu, L.; Cret, O.; Hangan, A.; Bacotiu, C. Water Parameters Monitoring on a Cyberwater Platform. In Proceedings of the 2015 20th International Conference on Control Systems and Computer Science, Bucharest, Romania, 27–29 May 2015; pp. 797–802.
17. Ilie, A.M.C.; Vaccaro, C.; Rogeiro, J.; Leitao, T.E.; Martins, T. Configuration, programming and implementation of 3 Smart Water network wireless sensor nodes for assessing the water quality. In Proceedings of the 2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), San Francisco, CA, USA, 4–8 August 2017; pp. 1–8.
18. Kudva, V.D.; Nayak, P.; Rawat, A.; Anjana, G.R.; Sheetal Kumar, K.R.; Amrutur, B.; Mohan Kumar, M.S. Towards a real-time campus-scale water balance monitoring system. In Proceedings of the 2015 28th International Conference on VLSI Design, Bangalore, India, 3–7 January 2015; pp. 87–92.
19. Yang, L.; Jiang, H.; Huang, X. IoT (Internet of Things) Based Centralized Riverbed Water Level Monitoring System with External Power Supply. Application Patent CN203012525U, 19 June 2013.
20. Zeng, L.; Tang, P.; Cui, F. Internet-of-Things Water Level Monitoring Device. Application Patent CN105786030A, 20 July 2016.
21. Li, Y.; Bi, Y.W.; Wang, Z.F.; He, J.G.; Wang, Y.T.; Li, X.H.; Li, Z.; Mu, X.D. Intelligent Lock Gate for Flood Protection Control Device and Its Control Method. Application Patent CN107988995B, 3 September 2019.
22. Rasin, Z.; Hamzah, H.; Aras, M.S.M. Application and evaluation of high power Zigbee based wireless sensor network in water irrigation control monitoring system. In Proceedings of the 2009 IEEE Symposium on Industrial Electronics & Applications, Kuala Lumpur, Malaysia, 4–6 October 2009; Volume 2, pp. 548–551.
23. Kamaruddin, N.A.; Tjahjanto, D. Automatic sliding gate used to control water level of a channel in flat urban area. In Proceedings of the National Seminar on Engineering Technopreneurship, Kuala Lumpur, Malaysia, 23–24 October 2008.
24. Madanhire, I.; Madaka, M.; Mbohwa, C. Design of an automated dam shutter control system: Case study. In Proceedings of the International Conference on Industrial Engineering and Operations Management (IEOM), Paris, France, 26–27 July 2018; pp. 2627–2639.
25. FRESNO. Series 7600 Fabricated Canal Gate—Technical Data Sheet. Available online: https://d2zm9amfddap0m.cloudfront.net/media/resources/Series%207600%20Tech%20Sheet%20update%20B&165.pdf (accessed on 31 August 2019).
26. Tharvin, R.P.; Thamari, S.; Muhammad, H.B.M.; Kabilan, N.S.; Nur, S.B.A. Developing an IoT Based Hydro-Gate System that Automate the Closure of Flood Gates Remotely, with and without the Intervention of Human Operators. Master's Thesis, University Tenaga Nasional (UNITEN), Bangi, Malaysia, 2017.
27. Al Blair, P.E. Low-Cost Automatic Gates for Irrigation Canals. Available online: http://www.twdb.texas.gov/publications/reports/contracted_reports/doc/0903580882_harlingen.pdf (accessed on 26 July 2019).
28. Dong, J. A kind of control method and application of the sluice gate on the lower Yellow River. In Proceedings of the 2011 Second International Conference on Mechanic Automation and Control Engineering, Hohhot, China, 15–17 July 2011; pp. 16–18.
29. Dolezilek, D.J.; Kalra, A.S. Automation of Water Flow in Networks. U.S. Patent US20140251478A1, 11 September 2014.

30. Tech Support. The Ultrasonic Ranging Module HC-SR04. Available online: https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf (accessed on 31 August 2019).
31. Matthew, M. Using Ultrasonic Sensing to Monitor Level in Tanks. Available online: https://www.ti.com/lit/an/snaa270/snaa270.pdf (accessed on 24 July 2019).
32. Jeswin, C.; Marimuthu, B.; Chithra, K. Ultrasonic water level indicator and controller using AVR microcontroller. In Proceedings of the 2017 International Conference on Information Communication and Embedded Systems (ICICES), Chennai, India, 23–24 February 2017; pp. 1–6.
33. Karwati, K.; Kustija, J. Prototype of water level control system. In *IOP Conference Series: Materials Science and Engineering*; IOP Publishing: Bristol, UK, 2018; Volume 384, p. 012032.
34. Siddula, S.S.; Babu, P.; Jain, P.C. Water level monitoring and management of dams using IoT. In Proceedings of the 2018 3rd International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU), Bhimtal, India, 23–24 February 2018; pp. 1–5.
35. Hajjaj, S.S.H. Github Repository. Available online: https://github.com/sami-s-hajjaj/IoThydrogate (accessed on 20 October 2018).
36. Hut, T.P. HC-SR04 Ultrasonic Range Sensor on the Raspberry Pi. Available online: https://thepihut.com/blogs/raspberry-pi-tutorials/hc-sr04-ultrasonic-range-sensor-on-the-raspberry-pi (accessed on 10 November 2019).
37. Cohn, R.J.; Coppen, R.J.; Banks, A.; Gupta, R. MQTT (Message Queuing Telemetry Transport) Version 3.1.1 Plus Errata 1.0. Available online: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/errata01/os/mqtt-v3.1.1-errata01-os.html (accessed on 31 August 2019).
38. Bray, T. The I-JSON Message Format. Available online: https://tools.ietf.org/html/rfc7493 (accessed on 4 October 2019).

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).