International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org Original Research Paper

Artificial Intelligence-Integrated Water Level Monitoring System for Flood Detection Enhancement

Kien-Fei Lee[1], Zi-Neng Ng[2], Kar-Ban Tan[3], Ruthramurthy Balachandran[4], Abraham Shiau-Iun Chong[5], Kah-Yoong Chan[1,*]

Submitted: 08/01/2024 **Revised**: 14/02/2024 **Accepted**: 22/02/2024

Abstract: Flash floods are increasingly becoming a common disaster in Malaysia, triggered by a combination of natural and human-induced factors. The natural factors include climate changes, landforms due to the environmental impacts, while the human-induced factors are associated with the negligence in river conservation, clogged drainage, and polluted water retention systems due to industrial and domestic wastes. These factors affect the water levels in rivers and drainage systems, leading to potential flash floods once the danger mark is exceeded. Flash floods could result in severe property damage and even loss of lives. Considering the devastating impact of flash floods, it is imperative to develop an early warning system that facilitates timely remedial measures. This system could monitor the water levels in rivers and other water retention areas. Herein, this study aims to design a water level monitoring system using a cost-effective camera module powered by the Internet of Things (IoT). The system, which includes an ESP32-Camera module powered by a solar panel, captures the water level data using OpenCV at one-minute intervals. Then, the data are made available on IoT platforms like ThingSpeak, enabling the authorized parties to keep track of the critical water levels in water retention areas.

Keywords: *Water level monitoring system, Internet of Things (IoT), ESP-32 Camera*

1. Introduction

The system used for monitoring or measuring water levels in rivers, ditches, and water retention areas has been explored and studied in the past. This is largely due to the recurring issue of flash floods, a problem that has plagued communities for generations. In recent times, flash floods have become an increasingly prevalent disaster in Malaysia and this could be attributed to a combination of factors, both natural and human-induced [1]. Natural factors such as climate change and landforms, driven by global warming, play a significant role in the occurrence of flash floods [2]. On the other hand, human-induced factors, are related to lack of proper care and preservation of our rivers, ditches, and water retention areas from the pollution caused by residential and industrial wastes. These factors bring impactful damages to the water levels in rivers and drainage systems, thus leading to potential flash floods in the event of excessive water overflow. Flash floods could result in severe property damages and, in severe cases, loss of human lives.

Considering the adverse effects caused by flash floods, it is essential to establish an effective warning system that allows for early intervention. This system should be capable of monitoring water levels in rivers, ditches, and water retention areas, and the data should be easily accessible for tracking. In this perspective, various technologies have been developed wherein different advantages and disadvantages are offered by these systems.

For example, the ultrasonic sensor is a commonly used device for measuring water levels. It is a popular choice due to its design simplicity and cost-effectiveness. However, the functionality is somewhat limited by its accuracy, particularly when water levels are low. In such cases, the sensor may lose track of the reflected ultrasound, refraining it from the detection of low water levels [3].

Considering these limitations, this study aims to develop a more effective water level monitoring system using a cost-effective camera module powered by the Internet of Things (IoT). IoT is tipped to transform the way we live and work in many aspects of our lives [4]. The proposed system here, which incorporates an ESP32-Camera module powered by a solar panel, captures water level data using OpenCV at one-minute intervals. This data is then made available on IoT platforms like ThingSpeak, enabling authorized parties to monitor critical water levels in any river, ditch, or water retention area.

---

[1] *Centre for Advanced Devices and Systems, Faculty of Engineering, Multimedia University, Persiaran Multimedia, 63100 Cyberjaya, Selangor, Malaysia*
ORCID ID: 0000-0003-1076-5034 (K-Y. C.),
** Corresponding Author Email: kychon@mmu.edu.my*
[2] *School of Electrical Engineering and Artificial Intelligence, Xiamen University Malaysia, Bandar Sunsuria, 43900, Sepang, Selangor, Malaysia*
ORCID ID: 0000-0003-2677-3055(Z-N. N.)
[3] *Department of Chemistry, Faculty of Science, Universiti Putra Malaysia, 43400 UPM Serdang, Selangor, Malaysia*
[4] *Department of Electronics and Communication Engineering, School of Electrical Engineering and Computing, Adama Science and Technology University, P O Box 1888, Adama, Ethiopia*
[5] *KESPRO Consultants Sdn Bhd, Bandar Sunway, Subang Jaya, Selangor, 47500, Malaysia*

2. Proposed System

The proposed IoT water level monitoring system deploys a programable camera to keep tracking the water level of rivers, ditches as well as water retention areas. The camera operates on power supply by an integrated battery system, which is chargeable by a solar module. The image processing is carried out using OpenCV, a real-time optimized Computer Vision library that offers a wide range of applications, including face recognition and object detection. OpenCV equips users with an extensive set of computer vision tools, enabling them to process the images according to their specific needs.

The system primarily relies on a solar panel to energize the rechargeable battery, which in turn powers the ESP32-CAM. This device continuously captures frames of the water level, which are then sent to OpenCV for image processing. The final results are subsequently displayed to the end-user. ThingSpeak is an open IoT platform with MATLAB analytics. It allows the calculated water level to be captured at one-minute intervals under normal circumstances. The recorded water level data is then computed to generate line graphs and relevant numerical values, which are displayed on the water level monitoring system’s channel under the respective fields of chart and numerical reading. When the water level reaches a predetermined threshold, the red LED on ThingSpeak will illuminate, transmitting an alert signal of the water level exceeding the critical limit. The block diagram of the overall proposed system is given in Figure 1.

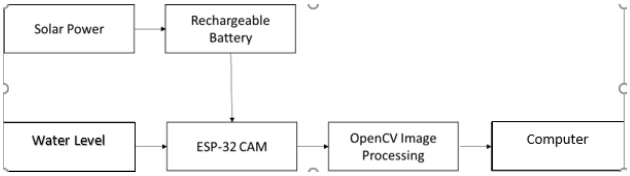


Fig 1. Block diagram of the overall proposed system.

3. Components of Hardware System

This section describes the technical details and roles of the hardware components deployed in the water level monitoring system, including both power supply unit, and frame capture unit, respectively.

3.1 Power Supply Unit

The power supply unit is composed of four key components: a solar panel [5], a 5 V regulator [6], a lithium battery charging module [7], and a 3.7 V lithium ion battery [8]. The solar panel harnesses sunlight, using photons to excite electrons from the valence band to the conduction band in a photovoltaic cell, thereby generating electricity. This electric power output is then directed to the 5 V regulator, which limits down the voltage from a high level to a safe 5 V to safeguard the lithium battery module. The lithium ion battery charging module, connected to the output of the 5 V regulator, will further step down the voltage to 4.2 V to charge the lithium battery safely.

Fig 2. Power Supply Unit Components: a) Solar Panel b) 5V Regulator c) Lithium Battery Charging Module d) Lithium Battery.

3.2 Frame Capture Unit

The video frame capture unit uses the ESP-32 Camera to capture the water level. The ESP32-CAM, equipped with 18 pins, is a versatile microcontroller that can be programmed for image or video capture applications. Apart from its function as a microcontroller, the ESP-32 also doubles as a WiFi module, featuring a built-in WiFi card.

4. Components of Software System

This section describes the technical details and roles of the software components deployed in the developed unit, including OpenCV, ThingSpeak, and Andriod Studio, respectively.

4.1 OpenCV

The image processing unit is tasked with the calculation of water level depicted in the image, achievable through the algorithmic processing. This algorithm is executed by OpenCV, an open-source image processing software [9][10]. OpenCV is a JavaScript-based software, which employs neural networks that can be trained to recognize various types of images, much like a human. It can distinguish different object images, such as airplanes, buses, and other objects. The accuracy of object recognition improves with increased training time, enabling the software to identify a wide array of more sophisticated objects.

4.2 ThingSpeak

ThingSpeak is an Internet of Things platform, which allows the user to write or read data [11]. In this IoT-enabled water level monitoring system, the OpenCV will write the data to ThingSpeak continuously by providing specific security writing API. When ThingSpeak receives the data from OpenCV, it will transfer those data to an App after a specific security reading API is provided.

4.3 Android Studio

Android Studio is a platform for the development of Android based App, which supports programming languages, e.g. Java and Kotlin [12]. In this IoT-enabled water monitoring system, an App was developed to receive the data of water level from ThingSpeak and then display the information on several interfaces in the Apps.

5. Software Development

5.1 Frame Capture

The first process of frame capture is needed to include the board information, allowing the IDE to recognize the ESP32 board for further compilation process. When Arduino IDE successfully recognizes the ESP32 board, it can program the ESP32 to set up the definition of pins ESP32 relations to the camera.

Once the pin configuration is complete, the ESP32 camera is ready to capture frames. Each captured frame is then converted into JPEG format. These frames are subsequently transmitted to a web server, providing a path for other platforms to process the images. Upon successful connection of the ESP32-CAM to a wireless network, the Arduino IDE's serial monitor displays the IP address of the ESP32-CAM, confirming that it is ready online. The web server facilitating this process is constructed using HTM.

5.2 Image Processing

Image processing is carried out using a series of algorithms, with OpenCV that provides the majority of the functions. The initial algorithm applied is Gaussian Blur, which serves to reduce frame noise by blurring the image. Figure 3 shows the initial image and the image after the process by Gaussian Blur.

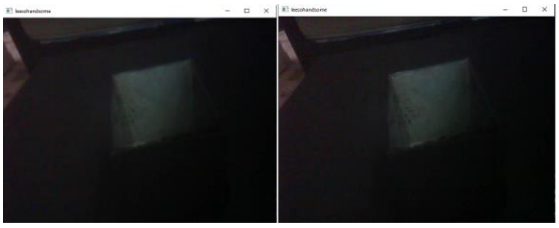


Fig 3. a) Blur Image b) Initial Image

Following the noise reduction or blurring of the image, the color is converted into the HSV format. Here, 'H' represents Hue, 'S' stands for Saturation, and 'V' denotes Value. Saturation quantifies the amount of grey in a particular color, ranging from 0 to 100 percent, with values nearing zero that has more grey with imparting a faded appearance [13]. Hue, expressed as a number from 0 to 360 degrees, defines the color aspect of the model. Value, also known as Brightness, works in tandem with saturation to determine the brightness or intensity of a color. It varies from 0 to 100 percent, where 0 signifies complete black and 100 indicates the brightest and most vivid colors. By applying the specific HSV values, unwanted parts of a frame can be filtered out, isolating only the white and black color portions. The result of this thresholding process is depicted in Figure 4.

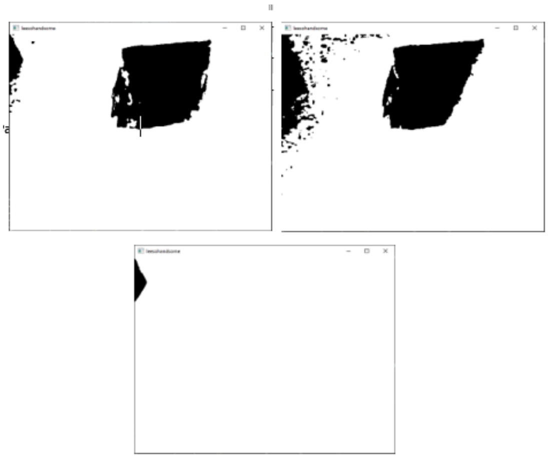


Fig 4. a) Optimized Threshold b) Under Threshold c) Over Threshold

Despite the use of HSV thresholding, some noises may remain unfiltered. To address this, the unwanted noise from the top left corner is removed by calculating the contour area of the image. This process returns an array, which is then sifted through to identify the largest contour area (Figure 5).

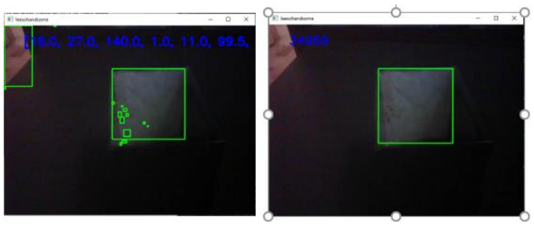


Fig 5. a) Unfiltered Image b) Filter Image

5.3 ThingSpeak

Once the water level measurement is successfully acquired, the readings are uploaded to ThingSpeak using the provided API keys. ThingSpeak offers both writing and reading API keys, enabling users to upload data to ThingSpeak and retrieve data from it.

5.4 Apps

While ThingSpeak is a passive component awaits for the data writing from Python, the Apps is an active component that will continuously read data from ThingSpeak, displaying the output on the smartphone. Python needs to provide the writing API key to write data to Thingspeak while Apps requires a reading API key for the reading operation.

6.Result and Discussion

This section presents the results of the water level measurements and provides a discussion on each of these results to gain a better understanding of how these algorithms operate.

6.1 OpenCV Result

The height of the water container in the image is 14.8 cm, or 208 pixel. Equation (1) is used to calculate the ratio of length per pixel (cm per pixel):

$$Ration = \frac{14.8cm}{208} = \frac{0.0711538464cm}{pixel} \ldots (1)$$

As shown in Figure 6, any water level in pixels can be computed by taking the pixel difference between the water container and the current water level shown in Equation (2). The water level in centimeters is then computed using Equation (3).

$$Water\ Level\ in\ pixel = 208 - current\ pixel \ldots (2)$$

$$Water\ Level\ in\ centimeter = (Water\ Level\ in\ pixel)(ratio) \ldots (3)$$

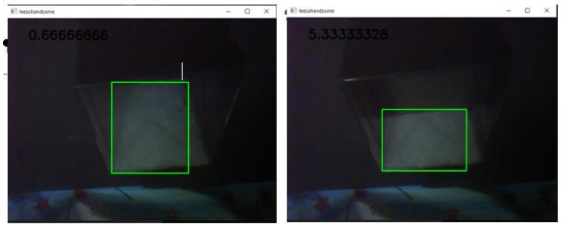


Fig 6. a) Water Level Reading in Empty Container b) Water Level Reading with Fill of Water

6.2 ThingSpeak

ThingSpeak is connected to OpenCV via a bridge written using Python code to transfer the information of the water level, and use the data to plot a line graph of water level with respect to time. It also displays water level as an integer value at any point of time, as shown in Figure 7.

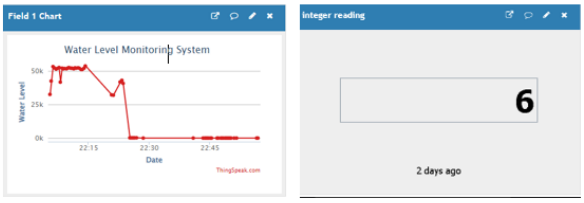


Fig 7. a) Line Graph b) Integer value

ThingSpeak offers a virtual LED feature, allowing users to recognize the signal specific conditions. For instance, a red LED can be activated when the water level surpasses a certain threshold, signifying that it has reached a critical level, as shown in Figure 8.

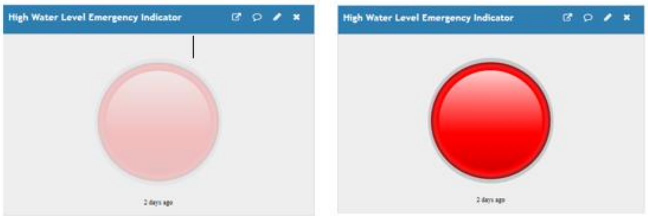
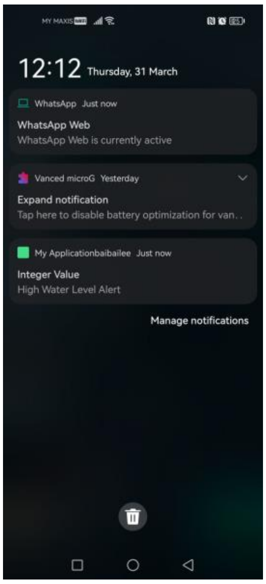


Fig 8. a) LED Turn OFF b) LED Turn ON

6.3 Apps

A notification will be sent out to alert the user when the water raises to its critical level. The figure below shows the prompt notification alerting the user that a critical level has reached.



12:12 Thursday, 31 March

WhatsApp Just now
WhatsApp Web
WhatsApp Web is currently active

Vanced microG Yesterday
Expand notification
Tap here to disable battery optimization for van. .

My Applicationbaibailee Just now
Integer Value
High Water Level Alert

Fig 9. Notification “High Water Level Alert”

7. Conclusion

In conclusion, an effective low-cost water level monitoring system integrated with IoT has been successfully developed and demonstrated. This system measures water levels using a pixel ratio in a given frame, calculated through specific algorithms. The transferred data to ThingSpeak are used for a clear and user-friendly presentation. This provides an alternative method for data reading, eliminating the need for constant video monitoring. An Android application was developed to continuously keep track of the data from ThingSpeak. Most importantly, the application shall alert the user when a pre-set critical water level has breached. The system is powered by a solar panel, aligning with the goal of using sustainable energy. This signifies the designed system is driven by renewable energy technology that also facilitates remote and, off-grid location installations.

Acknowledgements

This work was partly supported by the Telekom Malaysia Research and Development (TM R&D) under TM R&D Research Fund (Grant No: RDTC/231080, Project ID: MMUE/230009). The authors also want to acknowledge Multimedia University (MMU, Malaysia) for facilitating the development work.

Author contributions

Kien-Fei Lee: Conceptualization, methodology, formal analysis, investigation, writing—original draft preparation

Zi-Neng Ng: Writing— review and editing, Visualization, Validation

Kar-Ban Tan: Writing— review and editing, Visualization

Ruthramurthy Balachandran: Visualization, Validation

Abraham Shiau-Iun Chong: Visualization, Validation

Kah-Yoong Chan: Conceptualization, formal analysis, writing—review and editing, supervision, project administration, funding acquisition.

Conflicts of interest

The authors declare no conflicts of interest.

References

[1] Admad Fedzil Ismail, Mohammad Kamrul Hasan, Nor Bazilah Bopi, Suriza A. Zabidi, "Development of a River Basin Monitoring System for Malaysia," International Journal of u- and e- Service. Science and Technology, vol. 9, p. 1, 2016.

[2] Ching Yun Chen, Baharudin, Mohd Ekhwan T, Lee Yook Heng, Maimon A, Salmijah, "Impact of Climate Change on Flood Risk in the River Basin of Malaysia," Disaster Advances, vol. 6, no. 10, p. 1, 2013.

[3] Kodathala Sai Varun, Kandagadla Ashok, Vunnam Rakesh Chiowdary, C.S.K. Raju, "Water Level Management Using Ultrasonic Sensor (Automation)," International Journal of Computer Sciences and Engineering, vol. 6, no. 6, p. 801, 2018.

[4] Fathin Izzaty Fahmi, Kah-Yoong Chan, Chu-Liang Lee, Wai-Leong Pang, Gregory Soon How Thien, Zi-Neng Ng, An Intelligent Waste Management System Enabled by IoT, International Journal of Intelligent Systems and Applications in Engineering (IJISAE) (2023) 11(8s), 401–405.

[5] https://my.cytron.io/p-solar-panel-18v-555ma-10w.

[6] https://my.cytron.io/p-38v-step-down-5v-5a-regulator.

[7] https://www.instructables.com/DIY-Lithium-18650-Cells-Charger-by-Using-TP4056-Mo/.

[8] https://my.cytron.io/p-3.7v-2000mah-li-ion-battery.

[9] D. L. Baggio, "Mastering OpenCV with Practical Computer Vision Project," Packt Publishing Ltd, Birmingham, 2015.

[10] Gary Bradski, Adrain Kaehler, "Learning OpenCV," Media, Inc, CA, 2008.

[11] David Nettikadan, Subodh Raj M.S., "Smart Community Monitoring System using Thingspeak IoT Plaform," International Journal of Applied Engineering Research ISSN, vol. 13, no. 17, p. 13404, 2018.

[12] H. R. Esmaeel, "Apply Android Studio (SDK) Tools," International Journal of Advanced Research in, vol. 5, no. 5, p. 89, 2015 .

[13] Jun-Dong Chang, Shyr-Shen Yu, Hong-Hao Chen, and Chwei-Shyong Tsai, "HSV-based Color Texture Image Classification using Wavelet Transform," Journal of Computer, vol. 20, no. 4, p. 64, 2010.