Aquaculture Monitoring Systems Based on Lightweight Kubernetes and Rancher

Halim Fathoni[1,2], Chao-Tung Yang[3,4](✉), Chin-Yin Huang[1], Chien-Yi Chen[3], and Tí-Fēng Hsieh[3]

[1] Department Industrial Engineering and Enterprise Information, Tunghai University, Taichung, Taiwan (R.O.C.)
{D07330701,huangcy}@thu.edu.tw

[2] Department Ekonomi dan Bisnis, Politeknik Negeri Lampung, Bandar Lampung, Indonesia
fathoni@polinela.ac.id

[3] Department Computer Science, Tunghai University, Taichung, Taiwan (R.O.C.)

[4] Research Center for Smart Sustainable Circular Economy, Tunghai University, No. 1727, Sec. 4, Taiwan Boulevard, Taichung 407224, Taiwan (R.O.C.)
ctyang@thu.edu.tw

Abstract. Edge Computing is the new paradigm to process data at the edge of the network. The scenario of edge computing varies, depending on the case and problem. In this paper, we investigate an architecture that is suitable for Intelligence Aquaculture. This system will handle tasks such as collecting the water sensor data and running an Artificial Intelligence algorithm to train model prediction and run real-time object detection with a Deep learning algorithm and Deepstream. All applications were deployed with a docker container and managed with Lightweight Kubernetes (K3s). Rancher is also used to coordinate and visualize the resource system of the edge devices. This system architecture could be a reference for edge computing ecosystems and monitoring system of Aquaculture.

Keywords: Edge computing · Edge AI · Lightweight Kubernetes · K3s · Rancher · Aquaculture

1 Introduction

Nowadays, the use of the internet of things creates new challenges for cloud computing. In the cloud computing paradigm, data is stored centrally in the data center. However, sometimes real-time response and low latency become a burden for Cloud Computing when dealing with emerging technologies such as smart cities or Smart Aquaculture. Some researchers propose a new computing paradigm as extended of Cloud computing as known as Edge Computing. Edge computing is a paradigm where the resources of an edge server are placed at the edge of the network close to the things or data sources, integrating the capabilities of networks, storage, and applications [3]. Generally, the edge of the

Y.-B. Lin et al. (Eds.): SGIoT 2021, LNICST 447, pp. 38–45, 2022.
https://doi.org/10.1007/978-3-031-20398-5_4

networks is located only one hop away from the associated end devices. The purpose is to reduce network transmission, reduce response time and increase the security for data privacy. Some intelligence services also deploy to meet the key requirements of industry digitization for agile connectivity, real-time data optimization, and application intelligence [2]. Recently, Intelligence applications cannot be avoided by edge computing. Machine Learning and Deep Learning are commonly used in Traditional Cloud Computing. Data send to the cloud, and then the best model will support inference in the edge devices to deploy services.

The development of Edge Computing is still growing. Some researcher has proposed some edge computing architecture, for example, Cloudlet [5], Fog Computing [15], Microdata Center [8], Multi-Access Edge computing [7] and other systems [11] that work on the edge of the network which has same principles but different focuses. However, there has not yet been consensus on standardized definitions, architectures, and protocols of edge computing.

This work investigates an architecture that is suitable for Intelligence Aquaculture architecture. Aquaculture is one of the ways to ensure good seafood for the world [14]. Nowadays, traditional aquaculture faces severe environmental pollution, diseases, and a lack of product traceability [4]. Through Edge Computing principles, it is needed a fundamental architecture that can use for Intelligence Aquaculture. This Architecture will use a container to ensure an efficient deployment in every edge device, and all containers will be orchestrated with lightweight Kubernetes K3s.

This article is structured as follows: Sect. 2 Literature Review of the technologies in the architecture; Sect. 3 presents the design architecture; Sect. 4 presents the performance result of our architecture; Sect. 5 is discussion.

2 Literature Review

2.1 LoraWAN

LoraWAN is an open standard for link and network layers that operate on top of LoRa as the physical layers [12]. LoraWAN is one of LPWAN technologies where it uses low power and long-range communication. This technology is suitable for rural areas with an insufficient infrastructure of Wi-Fi or a similar network for sending data to a server. LoraWAN can cover around 20km in the line-of-sight situation, and it was designed to work on unlicensed frequency bands (433MHz, 868MHz, or 915Mhz) depend on a regulator of the area. LoraWan focuses on uplinks for efficient power usage. The use of two-way communication will drain more energy from LoraWan devices which are generally powered by batteries. Therefore, LoraWAN defines classes based on the energy consumption constraints of different end devices. Class A implements the basic set of features; Class B enables scheduled listening; Class C has bi-directional communication.

2.2 Rancher Kubernetes Engine

Rancher Kubernetes Engine (RKE) is one of the container management that can help deploy Kubernetes engine anywhere. Rancher also unites the Kubernetes clusters with centralized authentication, access, and observability [13]. Rancher also offers K3s as Lightweight Kubernetes. K3s has a small memory footprint or binary, which contains the components to run a cluster. To keep the size small, K3s replace etcd by another datastore with sqllite3.

2.3 Docker

Docker is a lightweight container system to support rapid deployment [11]. Docker relies on the Kernel of the host OS, increasing the performance and reducing the footprint. Docker supports process-level isolation, with two concepts from the Kernel that are used: namespace and Cgroup. The namespace will isolate the communications in the container so the container cannot interact with other containers. Cgroup is used to specify which computer resources the container should be able to access, such as a certain amount of RAM, CPU cores, file system access.

2.4 InfluxDB

InfluxdB is an open-source tool for time series data [10]. Influxdb supports developers because it has many features that can be used for IoT development. Some features of InfluxdB are 1. SQL style query Language where it is has commonly used by a developer, 2. Retention Policies: Influxdb uses retention policies to handle data retention periods, 3. Automatically. Continuous Queries; these features make run as a continuous query which essentially means telling InfluxDB to run a query in the background and compare it automatically, 4. HTTP API-2 endpoints.

3 System Design

In this study, we addressed questions dealing with the configuration of an edge cluster for monitoring water with an edge computing approach. We configured Rancher Kubernetes Engine (RKE), Lightweight Kubernetes, and LoraWAN to run on edge devices. This configuration uses to collect data from the water sensor and analyze the data in the edge device. In addition, Fish Tracking with YOLO4 was inferences in Jetson Devices uses Deepstream pipeline and bundled in a Docker container to test the configuration performance under the RKE. The illustration of the systems is in Fig. 1.

3.1 Edge Devices

Edge Devices consists of three devices: Jetson-NX, Jetson Nano, and Raspberry-Pi4. All edge devices are installed with Lightweight Kubernetes (K3S), Docker, and some apps to support the tasks, such as Node-Red, InfluxDb, MQTT, and Grafana. Moreover, all devices are linked to the Rancher Kubernetes engine. One of the tasks on edge devices is receiving data from water sensors. Data in the

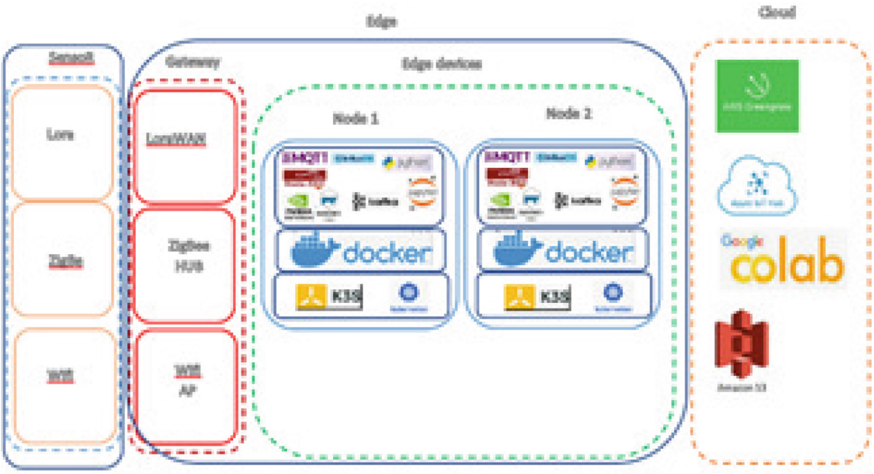

Fig. 1. Aquaculture monitoring system architecture.

database analyze to create a model prediction of Dissolved Oxygen Water. The tools to analyze data are Miniconda and JupyterLab, which are installed in the edged devices. In addition, to investigate the uses of GPU in the jetson devices, we implement Fish Tracking. Deepstream pipeline will use for maximizing the GPU in Jetson devices, an architecture. All apps that support the Edge server were a bundle in Docker and managed by K3s, and all devices will monitor in the Rancher.

3.2 Network

This system uses two kinds of network systems. The first system is to collect data from the sensors. Second, to coordinate from the cloud to the edge devices. LPWAN has been used to collect data from the sensor to the edge devices. Data from sensors will send using LoraWAN to the Lora Gateway. This network use Lora class A. For deploying an application in edge devices, we use RKE through the internet.

4 Experimental Results

4.1 Edge Devices

All the edge devices have been successfully installed with lightweight Kubernetes (K3s) and connected with Rancher Kubernetes Engine. Three devices use ARM-based processors, and one node was built using VMware ESXi as a comparison, as shown in Fig. 2. Then each device will deploy an application that is related to Intelligence Aquaculture. Data from the sensors will be managed with MQTT. Then with Node-Red, data will be filtered based on the classification and input

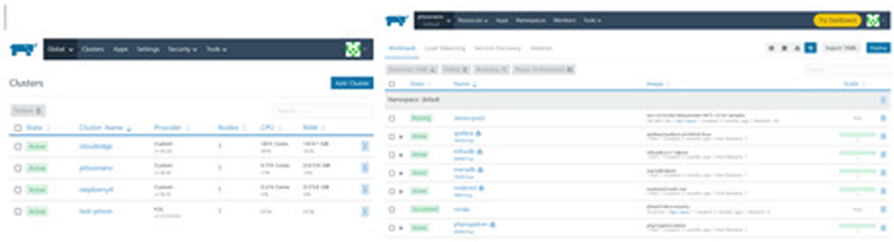

Fig. 2. Edge devices registered in rancher (A) and apps orchestration in every edge devices (B).

into the database. There are two databases Influxdb and MariaDB. Influxdb has a function to visualize data in real-time and has a function to set up the alert when the data has reached a threshold value. MariaDB is used to save data permanently and used to upload to the cloud. All applications deploy inside the Docker container. Another application that we try is real-time monitoring for fish detection on Jetson edge devices. We use the Nvidia Deepstream pipeline, which maximizes the GPU computation on Jetson devices [1]. Nvidia provides the Docker environment for Deepstream, so it can be applied on Lightweight Kubernetes (Fig. 3). The system used a camera to detect a fish in the Aquarium in real-time to test the Jetson device's performance [1]. The systems can identify a fish around 29 FPS. During real-time fish detection, the power consumption also rises around 8 W, wherein the idle condition is only about 3 W, as shown in Fig. 4. Fish detection builds using a YOLO4 and Deepsort [6]. A dataset for the trained model is available in mp4 format and converted into frames. The

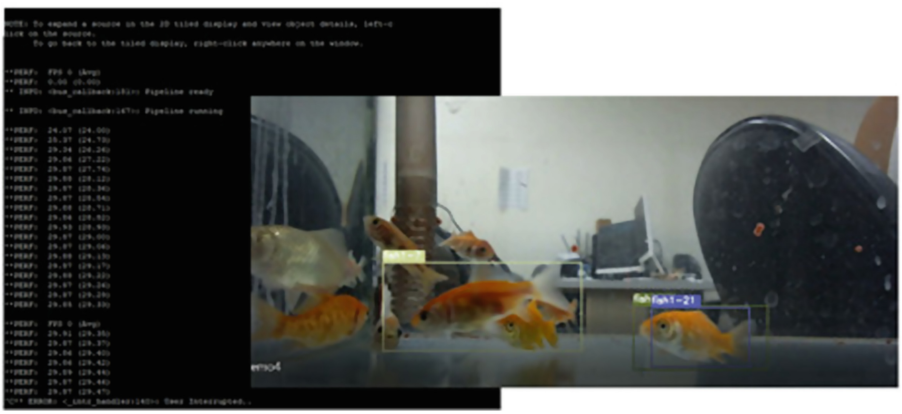

Fig. 3. Real-time monitoring with DeepStream for fish detection.

extracted frames were then annotated using the LabelImg tool [9] with YOLO label format. All datasets are then Trained with Google Colab using GPU. The convolution filter size was selected as 21×21 since only two classes. The model then inferences to Jetson nano use DeepStream. All the process was coordinated using RKE. One of the advantages of using RKE is the availability of a matrix for the resources used. The matrix is used to monitor the resources of our edge devices. The edge devices have small resources, so it is crucial to observe the best conditions continuously.

4.2 Networks and Sensors

The Lora Gateway is set up in the lab of Tunghai University. In this experiment, we use an indoor Lora gateway. We use three sensors: Temperature, Dissolve Oxygen water, Conductivity, and pH water. We build water monitor systems with Arduino board as the main controlling board and use an expansion board to carry more sensors on the same board. All the sensors' readings will send to edge devices through the Lora gateway. MQTT then managed a message from Lora based on topics. Node-red then uses to insert data to InfluxDB and visualize. All data from the sensors were then visualized with InfluxDB. In addition, power consumption monitoring is also used to understand the number of power consumption when running the object detection and data analysis (Fig. 4).

Fig. 4. Power consumption monitoring.

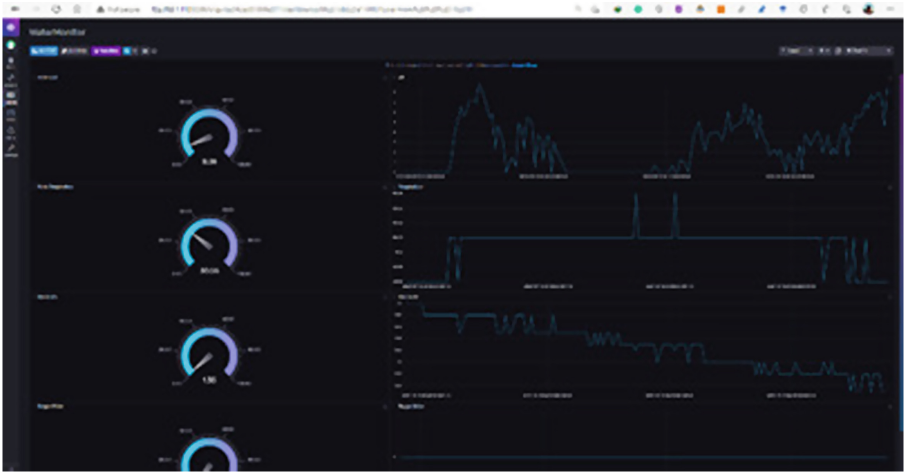

Fig. 5. A water sensor data collection

5 Discussion

The edge computing paradigm was run well in these systems. All edge nodes can collect data from sensors ideally. The graphic User Interface provided by RKE was easy to use and can help beginners deploy the system on edge devices. Ensure the installation process uses a correct RKE agent so that the edge devices can coordinate with the RKE servers (Fig. 5).

These systems collect not only data but also implement real-time fish detection. All tasks can run parallel and work perfectly. Deepstream pipeline optimizes the GPU on the edge devices for the video analytics process and only consumes less than 10 W. Unfortunately, RPI4 does not have GPU to process real-time video analytics, but we can use Movidius™ Neural Compute Stick for AI inferencing.

6 Conclusion

Edge computing is a new paradigm in this era. There is much architecture to implement the Edge Computing. In this paper, we investigate the edge computing paradigm for implementing Intelligent Aquaculture. In this case, we emphasize real-time collecting data and fish detection. All systems must be based on containerization cause to help deployment proses of Intelligence Aquaculture system. Rancher Kubernetes Engine can fulfill our requirements, and Lightweight Kubernetes (K3s) can handle the artificial intelligence model bundles with Docker. Lightweight Kubernetes can maintain our model on the best performance. This architecture we will use to deploy our artificial intelligence model with the custom dataset in the future.

Acknowledgements. This work was supported by the Ministry of Science and Technology, Taiwan (R.O.C.), under grants number 110-2221-E-029-020-MY3 and 110-2811-E-029 -003.

References

1. Bakre, P.: I at the edge with K3s and Nvidia Jetson Nano: object detection and real-time video analytics (2021). https://www.suse.com/c/ai-at-the-edge-with-k3s-nvidia-jetson-nano-object-detection-real-time-video-analytics-src/. Accessed 16 Sept 2021
2. E.C. Consortium: Edge computing reference architecture 2.0 (2017). http://en.ecconsortium.net/Uploads/file/20180328/1522232376480704.pdf. Accessed 18 Sept 2021
3. Debauche, O., Mahmoudi, S., Mahmoudi, S.A., Manneback, P., Lebeau, F.: A new edge architecture for AI-IoT services deployment. Procedia Comput. Sci. **175**, 10–19 (2020)
4. Farmaki, E.G., et al.: Advanced multivariate techniques for the classification and pollution of marine sediments due to aquaculture. Sci. Total Environ. **763**, 144617 (2021)
5. Gupta, A., Mukherjee, N.: A cloudlet platform with virtual sensors for smart edge computing. IEEE Internet Things J. **6**(5), 8455–8462 (2019)
6. Liu, S., et al.: Embedded online fish detection and tracking system via YOLOv3 and parallel correlation filter. In: OCEANS 2018 MTS/IEEE Charleston, pp. 1–6. IEEE (2018)
7. Martínez-Casanueva, I.D., Bellido, L., Lentisco, C.M., Fernández, D.: An initial approach to a multi-access edge computing reference architecture implementation using Kubernetes. In: Gao, H., Durán Barroso, R.J., Shanchen, P., Li, R. (eds.) BROADNETS 2020. LNICST, vol. 355, pp. 185–193. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-68737-3_13
8. Mondal, S.K., Pan, R., Kabir, H., Tian, T., Dai, H.N.: Kubernetes in it administration and serverless computing: an empirical study and research challenges. J. Supercomput. **78**, 2937–2987 (2021). https://doi.org/10.1007/s11227-021-03982-3
9. NaelsonDouglas: Labelimg (2021). https://github.com/tzutalin/labelImg. Accessed 16 Sept 2017
10. Nasar, M., Kausar, M.A.: Suitability of InfluxDB database for IoT applications. Int. J. Innov. Technol. Explor. Eng. **8**(10), 1850–1857 (2019)
11. Pääkkönen, P., Pakkala, D., Kiljander, J., Sarala, R.: Architecture for enabling edge inference via model transfer from cloud domain in a Kubernetes environment. Future Internet **13**(1), 5 (2021)
12. Queralta, J.P., Gia, T.N., Zou, Z., Tenhunen, H., Westerlund, T.: Comparative study of LPWAN technologies on unlicensed bands for M2M communication in the IoT: Beyond LoRa and LoRaWAN. Procedia Comput. Sci. **155**, 343–350 (2019)
13. Rancher: Kubernetes management for dummies (2021). https://rancher.com/. Accessed 26 Sept 2021
14. Yue, K., Shen, Y.: An overview of disruptive technologies for aquaculture. Aquacult. Fish. **7**, 111–120 (2021)
15. Zahmatkesh, H., Al-Turjman, F.: Fog computing for sustainable smart cities in the IoT era: caching techniques and enabling technologies-an overview. Sustain. Urban Areas **59**, 102139 (2020)