applied sciences

MDPI

Article

Real-Time UAV Trash Monitoring System

Yu-Hsien Liao and Jih-Gau Juang *

Department of Communications, Navigation and Control, National Taiwan Ocean University, Keelung 20224, Taiwan; steveliao1688@gmail.com
* Correspondence: jgjuang@ntou.edu.tw

Abstract: This study proposes a marine trash detection system based on unmanned aerial vehicles (UAVs) and aims to replace manpower with UAVs to detect marine trash efficiently and provide information to government agencies regarding real-time trash pollution. Internet technology and computer–machine interaction were applied in this study, which involves the deployment of a marine trash detection system on a drone's onboard computer for real-time calculations. Images of marine trash were provided to train a modified YOLO model (You Look Only Once networks). The UAV was shown to be able to fly along a predefined path and detect trash in coastal areas. The detection results were sent to a data streaming platform for data processing and analysis. The Kafka message queuing system and the Mongo database were used for data transmission and analysis. It was shown that a real-time drone map monitoring station can be built up at any place where mobile communication is accessible. While a UAV is automatically controlled by an onboard computer, it can also be controlled through a remote station. It was shown that the proposed system can perform data analysis and transmit heatmaps of coastal trash information to a remote site. From the heatmaps, government agencies can use trash categories and locations to take further action.

Keywords: object detection; marine trash; streaming data; message queuing system; UAV

Citation: Liao, Y.-H.; Juang, J.-G. Real-Time UAV Trash Monitoring System. *Appl. Sci.* **2022**, *12*, 1838. https://doi.org/10.3390/app12041838

Academic Editor: Yosoon Choi

Received: 28 December 2021
Accepted: 8 February 2022
Published: 10 February 2022

1. Introduction

Marine pollution has been a critical environmental issue for decades. Most of the waste in the ocean comes from rivers. Oceanic debris frequently washes aground and is known as beach litter. Coastal trash is detected and monitored by certain agencies, usually using manpower. Fixed monitoring stations can also obtain trash information, but they only cover certain areas. In addition, these monitoring systems cannot automatically provide detailed information of the trash. Detailed information includes the category of the trash and the size of the area. In previous studies, Xu et al. [1] surveyed 40 projects of related monitoring systems which used the internet of things (IoT) and sensors for ocean sensing and monitoring, water quality monitoring, fish farm monitoring, coral reef monitoring, and wave and current monitoring. Ullo and Sinha [2] reviewed research on various environment monitoring systems used for air quality, water pollution, and radiation pollution. Although the performances of these monitoring systems were good, they all used fixed stations. Monitoring areas were limited to predefined areas. There is a lack of mobility. In addition, the IoT, web server, and wireless communications used in previous studies were only used for sensor data transmission. The integration of video streaming and image data analysis were not included. In our study, the proposed monitoring system had mobility by using unmanned aerial vehicles (UAVs) to obtain coastal images and capture trash information for further analysis.

With the rapid progress in artificial intelligence, communications engineering, and IC technology, ideas such as the IoT, the artificial intelligence of things (AIoT), industry 4.0, and smart factories have been practically implemented in recent years. In our previous studies, we successfully applied the IoT to cage culture by an unmanned aerial vehicle

(UAV) [3,4]. Nowadays, UAVs are popular research tools due to their high-quality aerial image camera and flexibility for quick inspection ability. UAVs have been used in many applications for years, such as construction inspection, aerial photography and videography, real estate photography, mapping and surveying, asset inspection, payload carrying, agriculture, bird control, and crop spraying. Further, many projects combine UAVs and the AIoT to build systems for monitoring pollution, livestock, and pipeline security, which bring convenience to human life. Plastic trash can be found anywhere around coastal areas, constituting a huge crisis in the marine ecosystem. To monitor plastic pollution, we combined UAVs, trash detectors, and the IoT to develop a trash monitoring system that can help government agencies to monitor coastlines efficiently.

A UAV was applied to patrol the beach [5], using aerial images with SegNet [6], a deep neural network, for semantic segmentation to classify beach litter in pixel units. In an earlier study [7], the authors used UAVs for a long-term monitoring program and studied the spatial and temporal accumulation of the dynamics of beached marine litter. Then, three supervised classification algorithms, including Maximum Likelihood, Random Forest, and Support Vector Machine, were applied to classify marine litter [8]. To improve the performance of trash detection, the authors [9] proposed the attention layer in the neuron network. A TACO trash dataset was built [10], which adopted the Mask R-CNN [11] to test the performance of litter detection. The authors [12] then proposed an improved YOLOv2 [13] model and built an automatic garbage detection system. Next, the authors [14] modified the loss function in YOLOv3 [15] and created an automated floating trash monitoring system based on UAVs. Although these studies provided several useful trash detection systems, real-time data analysis and monitoring were still not included.

In this study, we applied an intelligent controller to the UAV. A high-resolution aerial image trash dataset called HAIDA was created. We also built a trash detector based on the YOLO object detection algorithm with algorithm selection, hyper-parameter tuning, and model evaluation to obtain the best model for the lowest generalization error. The main purpose of this study was to construct a UAV and IoT architecture that includes a data streaming platform: Kafka, MongoDB database, web service, video streaming server, and control station for the data transmission. Real-time monitoring in a remote site was performed, and internet technology and computer–machine interaction were used in the system communications. In addition, the proposed system used a UAV to execute the monitoring mission; the UAV can automatically fly along a preset path and transmit trash information to the control center, which can save manpower in coastal environment surveillance.

2. System Description

The real-time UAV trash monitoring system consisted of nine parts, including UAVs, a message queuing system, database, video streaming server, connector, UAV control station, web service, UAV map, and data analysis, as shown in Figure 1. The video streaming server, Kafka, Kafka connector, MongoDB, and web service were built in the ground station.

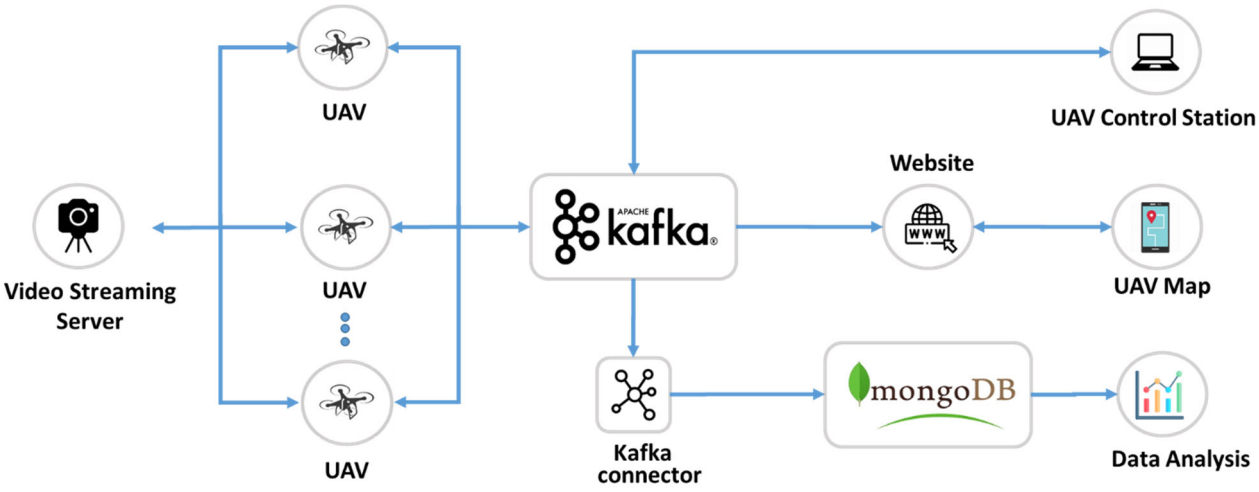


Figure 1. The structure of real-time unmanned aerial vehicle (UAV) trash monitoring system.

The UAV system included an intelligent UAV controller and a power system. The intelligent UAV controller used the master and slave structure to control the UAV. The power system was responsible for the lifting power of the UAV. A schematic of the UAV system is shown in Figure 2, where ESC is the electric speed controller.

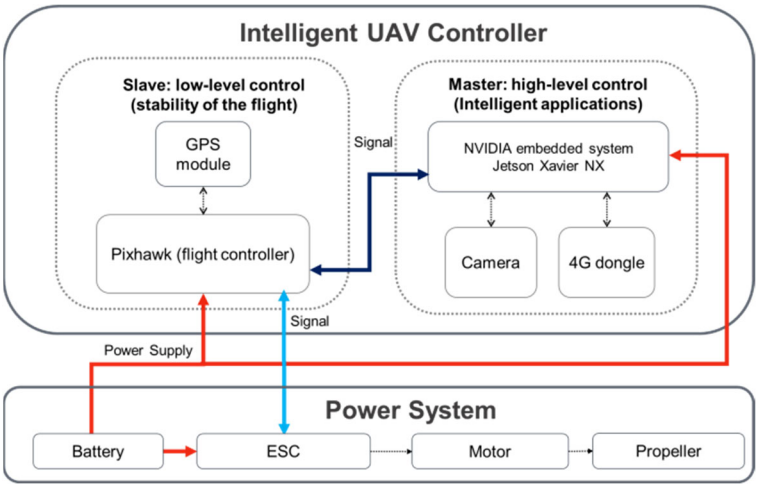


Figure 2. The layout of the unmanned aerial vehicle (UAV) system.

in a master and slave structure, the slave performs low-level control and is responsible for the stability of the flight. The master performs high-level control and sends the flight command to the slave, that is, intelligent applications. For the slave of the intelligent UAV controller, we used the Pixhawk flight controller [16] equipped with an Ublox M8N GPS module [17] to control the stability of the flight. The GPS error was 2 to 6 m, which depended on the GPS signal. We fine-tuned the UAV's proportional, integral, and derivative (PID) parameters in the UAV testing flight. The PID parameters were tuned in-flight. It used the auto-tuning technique [18], which is based on the Ziegler–Nichols closed-loop method. Figures 3 and 4 show the proposed quadrotor and the PID parameters.

Figure 3. The proposed quadrotor.

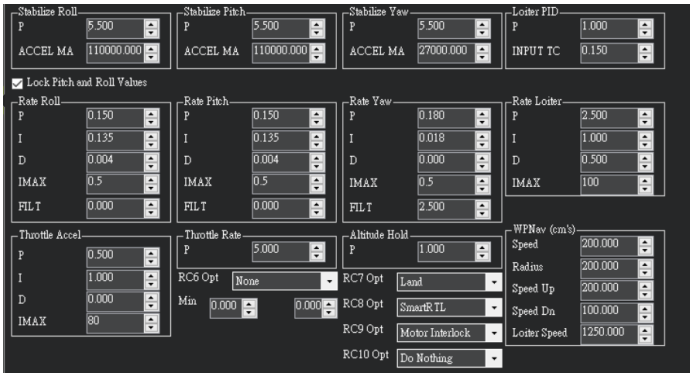


Figure 4. The proportional, integral, and derivative (PID) parameters of the quadrotor.

For the master of the intelligent UAV controller, we used the NVIDIA-embedded system Jetson Xavier NX [19] as the UAV onboard computer to carry out high-computational trash detection. The master was equipped with a Logitech BRIO webcam [20] and a Huawei E8372 4G dongle [21]. For the UAV trash detection task, we uploaded the predefined waypoints to the UAV first. Then, the UAV was controlled by the onboard computer to patrol the coastline along these waypoints automatically. While flying between two waypoints, the UAV took pictures, detected trash, and calculated the trash area. After data analysis, the image and the information regarding the trash were sent to the server of base station. Then, the UAV flew to next waypoint and repeated image processing and data transmission. After all waypoints were checked, the UVA flew back to the home position. A flowchart of the UAV trash detection task is shown in Figure 5.

We can calculate the detected trash pollution area by Equations (1)–(4), where α, β, H, L, W, W_i, L_i, W_o, L_o, A_o, and A_g are the camera's horizontal and vertical angle of view, the UAV's height, the camera's horizontal and vertical of view, the width and height of the camera's image, the width and the height of detected trash in the image, detected trash's area in the image, and the area of the detected trash in the real world, respectively, as shown in Figure 6, where FOV is field of view.

$$L = 2 \times H \times tan\left(\frac{\alpha}{2}\right) \quad (1)$$

$$W = 2 \times H \times tan\left(\frac{\beta}{2}\right) \quad (2)$$

$$A_o = W_o \times L_o \quad (3)$$

$$A_g = A_o \times \left(\frac{W_o}{W_i}\right) \times \left(\frac{L_o}{L_i}\right) \quad (4)$$

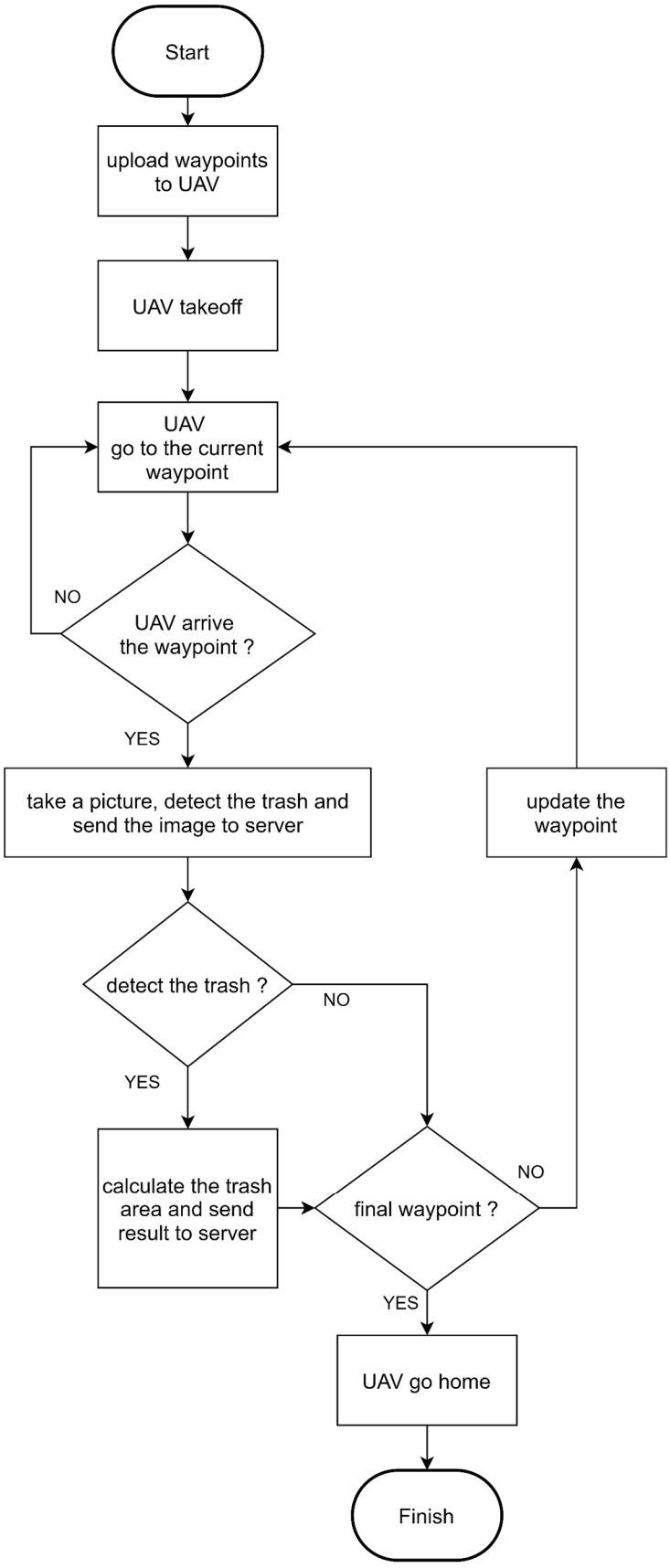


Figure 5. A flowchart of the UAV trash detection task.

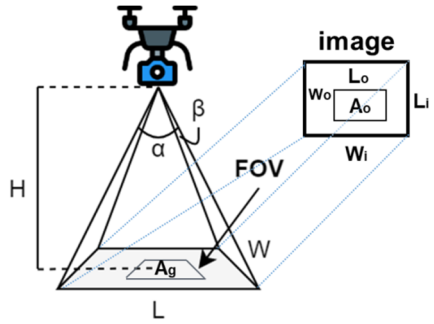


Figure 6. The calculation of the area of the detected trash.

The proposed quadrotor was set up based on a practical performance evaluation method [22]. This method provided necessary parameters' values for electric multi-copters. The performance of our proposed quadrotor is shown in Figure 7. The total weight of the quadrotor was 5.05 kg with a frame size of 700 mm. The quadrotor could fly for about 21.8 min at a minimum battery capacity of 25%.

Hovering Performance :

Hovering Time	: 26.84 min.
Throttle Percentage	: 57.6 %
ESC Current	: 5.71 A
Motor Speed	: 4326 rpm
Motor Power	: 109.2 W
Battery Voltage	: 24 V
Battery Current	: 26.8 A
Power Efficiency	: 67.2 %

Max. Throttle Performance :

Flight Time	: 8.1 min.
Total Lift	: 119.5 N
ESC Current	: 22.2 A
Motor Speed	: 6721.7 rpm
Motor Power	: 409.8 W
Battery Voltage	: 23.4 V
Battery Current	: 88.9 A
Power Efficiency	: 76.1 %

Integral Performance :

Normal Operation	: 21.8 min.
Total Weight	: 5.05 kg
Remaining Load	: 4.36 kg
Max. Takeoff Altitude	: 5.12 km
Max. Tilt Angle	: 57.5 °
Max. Forward Speed	: 20.1 m/s
Max. Flight Range	: 16.92 km
Wind Resistance	: 6 Degree

Figure 7. The performance of our proposed quadrotor.

3. Message Queuing System

The streaming data are produced by the data sources such as the sensors in the factory, personal e-commerce purchases, social website activities, or other log files. For the monitoring system, it is important to build a suitable application integration for streaming data processing, storing, and analyzing. The integration of one or more data sources to construct a large application is proposed. According to [23], the four main application integration styles are file transfer, shared database, remote procedure invocation, and messaging. The file transfer integration style means that the processor listens to the file in the folder. If the source creates the file, then the processor catches the file from the folder, as shown in Figure 8. Although the idea of file transfer integration is intuitive, it is not suitable for the data streaming system due to the high latency.

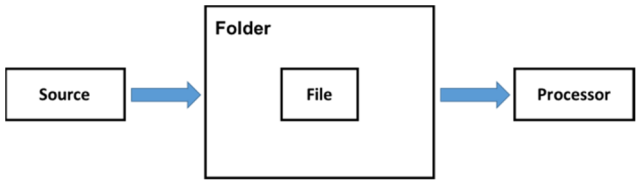


Figure 8. The file transfer integration style.

The shared database integration style facilitates the storage of data by multiple applications with a shared database, as shown in Figure 9. Frequent reading and

modification of the same data in the database causes a bottleneck to the performance and is not suitable for streaming data processing.

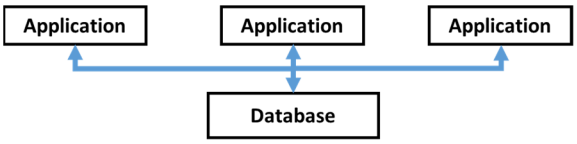


Figure 9. Shared database integration style.

In remote procedure invocation, the client sends a request and the server replies, as shown in Figure 10. The drawback of remote procedure invocation is that all the applications are tightly coupled, and it is hard to maintain integration. The messaging integration style is asynchronous messaging and can decouple all the applications so the sender does not need to wait for the receiver, and we can develop the application efficiently in the real-time monitoring system, as shown in Figure 11.

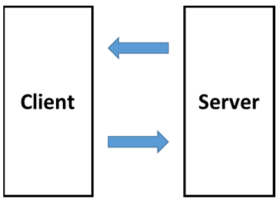


Figure 10. Remote procedure invocation integration style.

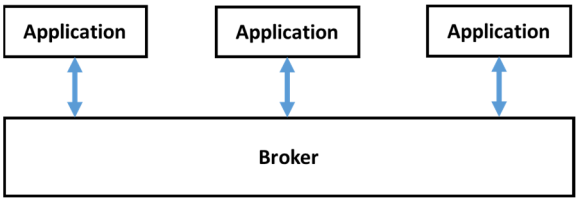


Figure 11. Messaging integration style.

In an earlier study [24], the authors compared five typical message queuing systems, including Kafka [25], RabbitMQ [26], RocketMQ [27], ActiveMQ [28], and Pulsar [29]. They also tested the latency and the throughput in the three scenarios, including the message size, number of producers/consumers, and number of partitions. The latency means how long the message transmitted between the applications is affected by packet metadata processing, packet replication, memory access latency, the message guarantee mechanism, and dequeuing latency. Here, RocketMQ showed the lowest latency across the three scenarios. The throughput measures the message bytes transmitted through the message queuing system per time unit. In the throughput test of five message queuing systems, Kafka received the highest throughput in message sizes of 4K bytes, one producer and one consumer, and 16 partitions. Considering the scalability of our real-time UAV trash monitoring system in the future, the high throughput of Kafka is needed. Although the latency of Kafka is about 70 ms and is higher than RocketMQ, Kafka is still acceptable in our system.

Kafka, a publish/subscribe messaging system designed by LinkedIn, is used as a distributed event streaming platform in most scenarios such as financial transactions, automotive industry tracking, and customer interactions in business. Kafka is good at handling multiple producers and multiple consumers and can scale up the brokers to handle any amount of data. Because of the disk-based retention of Kafka, if the data processing is in traffic, there is no danger of losing data. The data are retained in Kafka with custom retention rules. To validate the high performance of Kafka, we compared the throughput with one producer and one consumer in the topic (one partition) via three Python APIs.

We sent 1M messages at 100 (bytes) per message. Tables 1 and 2 show the high-performance test of the Kafka producer and consumer in our experiment with WD10EZEX-22MFCA0 hard disk drive and Intel i7-6700 CPU. In our experiment, the confluent-Kafka API had the best throughput both in the producer and the consumer.

Table 1. Comparison of the throughput to Kafka producer with three Python APIs.

API	Time(s)	MB/s	Msg/s
confluent-kafka	2.32	41.11	431,034.19
pykafka	89.48	1.07	11,176.24
kafka-python	250.72	0.37	3930.44

Table 2. Comparison of the throughput to Kafka consumer with three Python APIs.

API	Time(s)	MB/s	Msg/s
confluent-kafka	5.36	17.8	186,679.93
pykafka	130.84	0.73	7642.8
kafka-python	115.36	0.83	8668.59

The three topics *gcs*, *trash*, and *command* were used in the UAV trash monitoring system. We used one Kafka broker to handle the messages from the UAVs. The topic *gcs* was used to collect the stream of the UAVs' flight data, and each UAV produced one partition with one replica. The topic *trash* collected the trash data, which are detected by the UAV, and all the UAVs sent the data to the same partition in the topic. The UAV real-time maps consumed the topic *gcs* and *trash* to monitor the status of each UAV and the polluted area of the trash. The topic *command* collected the commands sent by the base station and was partitioned with the number of UAVs. Figure 12 shows the Kafka configuration in the UAV trash monitoring system. The fire-and-forget method was used to increase the throughput in the topic *gcs*. The asynchronous method was used to gain the good throughput/latency tradeoff in the topic *trash*. The synchronous method was used to guarantee the reliability of data in the topic *command*.

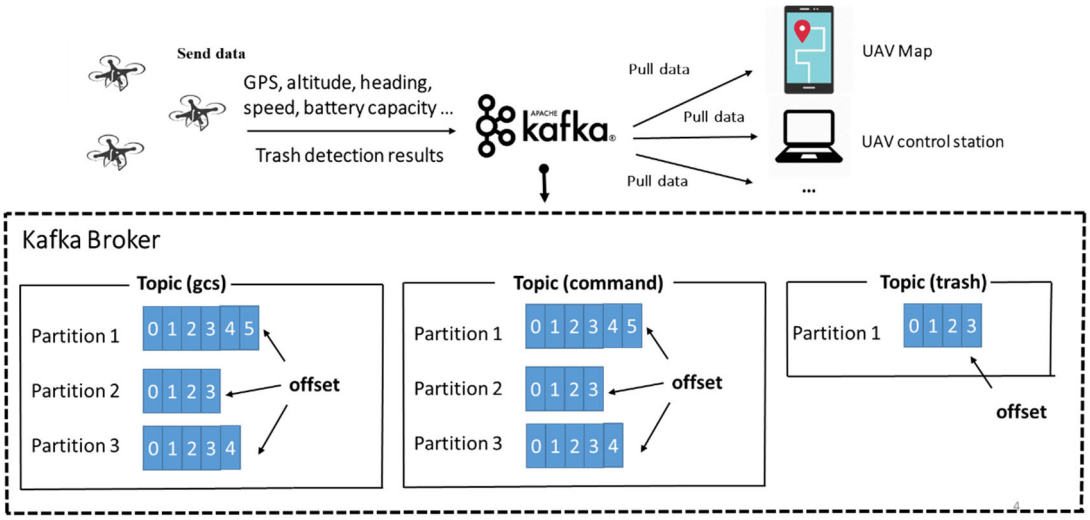


Figure 12. The Kafka configuration in the UAV trash monitoring system.

A web service is a good way to exchange data between applications that are built in different operating systems or programming languages. In the UAV marine trash monitoring system, the Python web framework Django was used [30] to develop a website that ran on a computer to listen to requests from clients via HTTP (HyperText Transfer Protocol). The real-time UAV trash map used Leaflet [31], a JavaScript library for interactive

maps, along with the Django website and the Kafka broker to develop a monitoring system for trash pollution. The consumers accessed the data from Kafka through the web service, and the trash information such as GPS and the polluted area, which is collected by the UAV, were displayed on the map.

For the design of a video streaming server, the UAVs used a 4G network to send the video stream to the server then publish it. Multiple users could subscribe to the video stream server with 30 FPS (Frames per Seconds) via the website. The UAV flew to the waypoints and used the current height and FOV (field of view) of an onboard camera to calculate the position where the UAV should detect the trash with no-repeat trash detection and sent the image back to the server. Additionally, the UAV sent the real-time video stream to the server continuously. ZeroMQ [32] is a concurrency framework based on the socket with in-process, inter-process, and TCP communications. The ImageZMQ [33] is a Python API built for video streaming based on the ZeroMQ framework. To suit the different network conditions, we used the ImageZMQ API with the auto image size tuning method, auto reconnection mechanism, and JPEG compression [34]. In the auto reconnection mechanism, we set the timeout to 1.8 s (this value depended on the network condition of the server and the UAV via trial and error). If the disconnection time was higher than the timeout, the UAV reconnected to the video streaming server automatically. The UAV recorded the number of timeouts that occurred and used it to tune the appropriate image size automatically (the more timeouts occurred, the smaller the image size via trial and error was) to send the video stream to the server.

In the UAV trash detection task, the UAV continuously consumed the messages from the Kafka topic to check if there were any commands from the control station. Each UAV consumed one partition in the topic. The database supported large volumes of data to store and query. The SQL (Structured Query Language) database, such as MySQL [35], stored the data in a predefined scheme with the table. The table was created by rows and columns, which meant that the data were stored in fixed columns that were not changeable. The addition of a new column for the table in the SQL database is time-consuming. The advantage of SQL is that it is more suitable for complex queries. The NoSQL (Non-Structured Query Language) database has a more flexible data storage format than the SQL database. In an NoSQL database such as Mongo [36], the JSON (JavaScript Object Notation)-like data are used to store in the database. The JSON data are key–value pairs and have no predefined structure to follow. We could add a new key to the data and store it in the database with a high level of freedom.

In [37], the authors compared the performance of seven databases with three SQL (Oracle, MySQL, and MsSQL) and four NoSQL (Mongo, Redis, GraphQL, and Cassandra) databases. The NoSQL databases were faster than the SQL databases, and Mongo had the best performance in the experiment, as shown in Tables 3 and 4.

Table 3. Query performance of databases with 10,000 records in milliseconds.

Type/Operation	Oracle	MySql	MsSql	Mongo	Redis	GraphQL	Cassandra
Insert	0.076	0.093	0.093	0.005	0.009	0.008	0.011
Update	0.077	0.058	0.073	0.008	0.013	0.007	0.013
Delete	0.059	0.025	0.093	0.01	0.021	0.017	0.018
Select	0.025	0.093	0.062	0.009	0.016	0.01	0.014

Table 4. Query performance of databases with 100,000 records in milliseconds.

Type/Operation	Oracle	MySql	MsSql	Mongo	Redis	GraphQL	Cassandra
Insert	0.091	0.038	0.093	0.005	0.01	0.008	0.011
Update	0.092	0.068	0.075	0.009	0.013	0.012	0.014
Delete	0.119	0.047	0.171	0.015	0.021	0.018	0.019
Select	0.062	0.067	0.06	0.009	0.015	0.011	0.014

We chose Mongo as our database due to the good performance and schema-free nature of the data storage. In Mongo, the collection is created to store the documents. The trash document that is sent by the UAV includes *id*, *time*, *lat*, *lon*, and *trash*, which are the ID of the UAV, the time that the trash is detected, the latitude and the longitude of the trash position, and the polluted area of the trash, respectively. In the UAV trash monitoring system, Kafka is used as the data streaming platform; the data need to be accessed and stored in the database. We built the Kafka connector that consumed the data with batch processing from Kafka and transferred the data into the database.

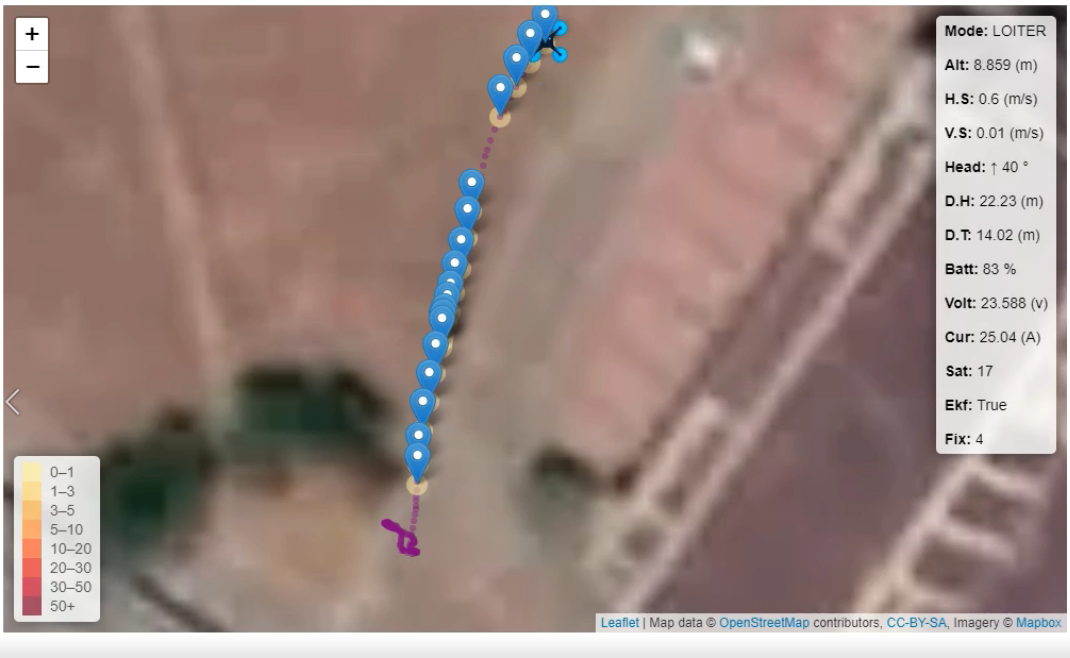
4. Experimental Results

We deployed the YOLOv4-Tiny-3l model [38] to the NVIDIA-embedded system Xavier NX on the UAV for the UAV trash monitoring system and tested it with three scenarios (NTOU campus, Badouzi fishing port, and Wanghai Xiang beach). The Mongo database was used in this study. The HAIDA dataset, which was collected in the NTOU campus, was applied to train the object detector (YOLO model). This dataset had 1319 pictures of trash, included two classes (3904 garbage objects and 2571 bottle objects) and 456 negative samples that were collected by the UAV. In this study, the dataset was split into three parts, a training set for model fitting, a validation set for hyper-parameter tuning and model selection, and a testing set for model evaluation. All models were trained on NVIDIA GeForce RTX-2080Ti GPU. After training, the YOLO model was implemented into the onboard computer Xavier NX. The YOLO model could process 22 FPS (frames per second) and identify trash objects with more than 70% AP50 (Test). Trash objects could be detected in real time. The flight time of the UAV was 21 min for each mission. The cruising speed of the UAV was 2 m/s. The UAV could fly 1 km in one mission, and the path was planned before the mission. The planned path was set in the flight controller of the UAV. Several waypoints were preset along the path. The UAV flew through all waypoints and obtained images of the coastal area. At the same time, the UAV sent a real-time video stream to the base station via mobile communication. In the video, one image frame covered a 10 m × 10 m area. Thus, the trash monitoring system processed one coastal picture and transmitted this still image to the base station every 5 s.

First, regular-shaped trash (bottle) detection was tested in the NTOU campus, as shown in Figure 13. As shown in Figure 13b, the confidence scores of small object detections were high (98.27%, 64.65%, and 91.86%).

(**a**)

(**b**)



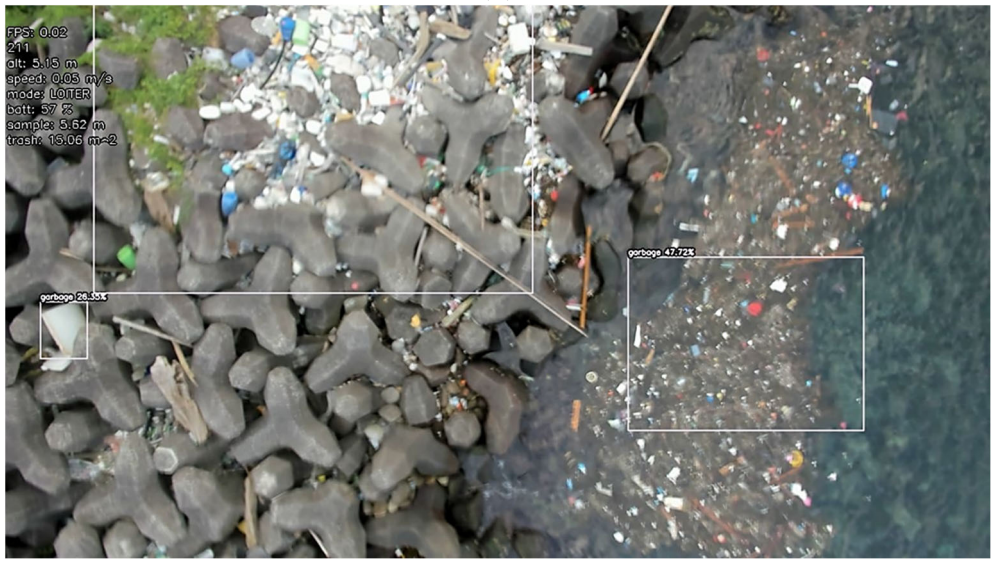
(**c**)

Figure 13. The result of the unmanned aerial vehicle (UAV) trash monitoring system in NTOU campus. (**a**) An area with the UAV and trash. (**b**) Video streaming server: detection result sent by the UAV. (**c**) UAV map: real-time monitoring by the UAV and the detection result.

In the Badouzi fishing port, the shape and the texture of the garbage vary due to natural erosion, thus decreasing the confidence scores of garbage detection (42.97%, 9.97%, 41.87%, 5.29%, and 8.49%). Thus, for the HAIDA dataset, there was a large bias to the trash of the Badouzi fishing port. The best solution for this problem was to collect more data for training. Figure 14a shows the UAV in the Badouzi fishing port, and Figure 14b,c show the detection of large, medium, and small objects. Figure 15 shows the UAV and trash information indicated on the proposed trash monitoring system.

(**a**)

(**b**)



(**c**)

Figure 14. The result of the unmanned aerial vehicle (UAV) trash detection at the Badouzi fishing port. (**a**) The UAV in Badouzi fishing port at 10 m height. (**b**) Video streaming server: the detection

result (a small area) in the Badouzi fishing port. (**c**) Video streaming server: the detection result (a large area) in the Badouzi fishing port.

(**a**)

(**b**)

Figure 15. Trash and the unmanned aerial vehicle (UAV) information on the trash monitoring system. (**a**) A UAV map to monitor the UAV waypoints and the status in the Badouzi fishing port. (**b**) A UAV map to monitor the trash detection and the trash polluted area in the Badouzi fishing port.

In the third case, there was more plastic debris on the Wanghai Xiang beach. The system easily confused both humans and the trash, and thus, gave false positive (FP) detection (whether it was a stone, garbage, or some plastic debris), as shown in Figure 16b–d.


UAV

(**a**)


2021-05-04_17-51-40-707259
FPS: 0.38
250
alt: 6.28 m
speed: 2.73 m/s
mode: LOITER
batt: 78 %
sample: 6.86 m
trash: 7.95 m^2
bottle 5.63%
garbage 5.32%
garbage 24.96%
garbage 19.28%

(**b**)


2021-05-04_17-21-35-200610
FPS: 0.43
133
alt: 4.47 m
speed: 2.04 m/s
mode: LOITER
batt: 71 %
sample: 4.38 m
trash: 8.84 m^2
garbage 5.78%
garbage 10.33%

(**c**)

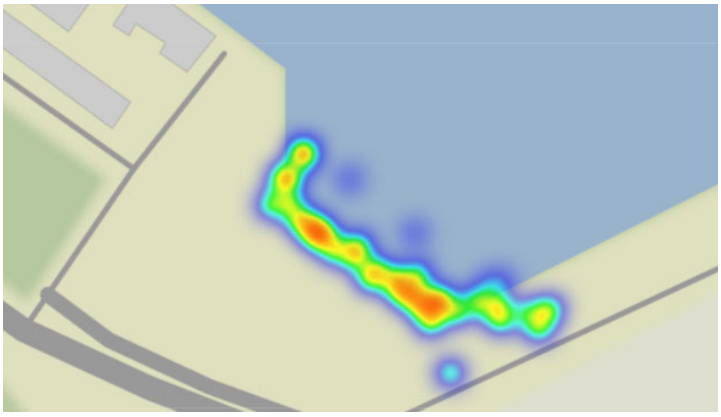
(**d**)

(**e**)

Figure 16. The result of the unmanned aerial vehicle (UAV) trash monitoring system for the Wanghai Xiang beach. (**a**) The UAV in the Wanghai Xiang beach. (**b**) Video streaming server: the detection result of a small area in the Wanghai Xiang beach. (**c**) Video streaming server: the detection result of a small area in the Wanghai Xiang beach. (**d**) Video streaming server: the detection result (a small area) in the Wanghai Xiang beach. (**e**) A UAV map to monitor the trash detection and the trash polluted area in the Wanghai Xiang beach.

After real-world testing for the Badouzi fishing port, we analyzed the trash data with the Mongo database. Figure 17 shows the degree of trash pollution map (based on trash polluted area) via our website. The red area means there was more trash.

(**a**)



(**b**)

Figure 17. Trash map: the heatmap of trash pollution areas. (**a**) A heatmap of trash pollution area (north of Taiwan). (**b**) A zoom in heatmap of trash pollution area (Badouzi fishing port).

In auto mode, the altitude of the UAV was set to 10 m on the cruising path so that the camera could obtain a picture with 10 m length. In auto mode, the speed of UAV was 2 m/s, which means every 5 s, the UAV flew 10 m (equal to the length of one picture). The proposed trash detection system sent one processed picture of the coastal image every 5 s so that the picture would not have overlap. The result of auto mode is shown in Figure 15. In manual mode, as shown in other figures, the altitude was not fixed. In order to show a clear view of different size of trash, the UAV was controlled manually and the altitudes were different.

5. Conclusions

In this study, we developed a marine trash detection and real-time monitoring system. The experiments show that the proposed system can successfully detect coastal trash, perform data analysis, and transmit trash information to a remote site. The onboard computer, video streaming server, internet communications, Mongo database, and Kafka connector were well integrated. We expect to contribute to global environmental protection with the proposed UAV trash monitoring system. The UAV trash monitoring system is designed for UAV swarm. In the future, multiple UAVs can be constructed across the coastline of target areas for trash monitoring. However, one issue was not included in this study, which is obstacle avoidance. In our next project, obstacle avoidance control will be

applied to the UAV monitoring system. For the commercialization, benchmarks of the proposed UAV trash monitoring system should be considered, such as the evaluation of real-world trash detection results, the reliability of the IoT architecture, and the cost of the hardware devices.

Author Contributions: Conceptualization, Y.-H.L. and J.-G.J.; methodology, Y.-H.L. and J.-G.J.; software, Y.-H.L.; validation, Y.-H.L. and J.-G.J.; formal analysis, Y.-H.L. and J.-G.J.; investigation, Y.-H.L. and J.-G.J.; resources, Y.-H.L. and J.-G.J.; data curation, Y.-H.L.; writing—original draft preparation, Y.-H.L.; writing—review and editing, J.-G.J.; visualization, Y.-H.L. and J.-G.J.; supervision, J.-G.J.; project administration, J.-G.J.; funding acquisition, J.-G.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Ministry of Science and Technology (Taiwan), grant number MOST 110-2221-E-019-075-MY2. The APC was funded by National Taiwan Ocean University and Ministry of Science and Technology.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Xu, G.; Shi, Y.; Sun, X.; Shen, W. Internet of things in marine environment monitoring: A review. *Sensors* **2019**, *19*, 1711.
2. Ullo, S.L.; Sinha, G.R. Advances in smart environment monitoring systems using IoT and sensors. *Sensors* **2020**, *20*, 3113.
3. Chen, C.X.; Juang, J.G. Vision based target recognition for cage aquaculture detection. *J. Mar. Sci. Technol.* **2020**, *28*, 480–490.
4. Cai, Y.E.; Juang, J.G. Path planning and obstacle avoidance of UAV for cage culture inspection. *J. Mar. Sci. Technol.* **2020**, *28*, 444–455.
5. Bak, S.H.; Hwang, D.H.; Kim, H.M.; Yoon, H.J. Detection and monitoring of beach litter using UAV image and deep neural network. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.-ISPRS Arch.* **2019**, *42*, 55–58.
6. Badrinarayanan, V.; Kendall, A.; Cipolla, R. SegNet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 2481–2495.
7. Merlino, S.; Paterni, M.; Berton, A.; Massetti, L. Unmanned aerial vehicles for debris survey in coastal areas: Long-term monitoring programme to study spatial and temporal accumulation of the dynamics of beached marine litter. *Remote Sens.* **2020**, *12*, 1260.
8. Escobar-Sánchez, G.; Haseler, M.; Oppelt, N.; Schernewski, G. Efficiency of aerial drones for macrolitter monitoring on baltic sea beaches. *Front. Environ. Sci.* **2021**, *8*, 283.
9. Tharani, M.; Amin, A.W.; Maaz, M.; Taj, M. Attention Neural Network for Trash Detection on Water Channels. Available online: https://arxiv.org/abs/2007.04639 (accessed on 10 December 2020).
10. Proença, P.F.; Simões, P. TACO: Trash Annotations in Context for Litter Detection. 2020. Available online: http://arxiv.org/abs/2003.06975 (accessed on 20 November 2020).
11. Pedropro/TACO, Trash Annotations in Context Dataset Toolkit. Available online: https://github.com/pedropro/TACO (accessed on 21 March 2021).
12. Liu, Y.; Ge, Z.; Lv, G.; Wang, S. Research on automatic garbage detection system based on deep learning and narrowband internet of things. *J. Phys.* **2018**, *1069*, 12032.
13. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings the 30th IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, Hawaii, 21–26 July 2016; pp. 6517–6525.
14. Niu, G.; Li, J.; Guo, S.; Pun, M.O.; Hou, L.; Yang, L. SuperDock: A deep learning-based automated floating trash monitoring system. In Proceedings of the 2019 IEEE International Conference on Robotics and Biomimetics, Dali, China, 6–8 December 2019, pp. 1035–1040.
15. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. 2018. Available online: http://arxiv.org/abs/1804.02767 (accessed on 3 August 2020).
16. ArduPilot, Pixhawk Overview Copter Documentation. Available online: https://ardupilot.org/copter/docs/common- pixhawk-overview.html (accessed on 7 July 2020).
17. U-blox, NEO-M8 Series. Available online: https://www.u-blox.com/en/ product/neo-m8-series (accessed on 21 July 2020).
18. Astrom, K.J.; Wittenmark, B. *Adaptive Control*; Addison Wesley: New York, NY, USA, 1995.
19. NVIDIA, Jetson Xavier NX Developer Kit. Available online: https://developer.nvidia.com/embedded/jetson-xavier-nx-devkit (accessed on 11 August 2020).

20. Logitech, Professional Webcam for High Definition Streaming and Video Calls. Available online: https://www.logitech.com/en-roeu/product/ brio-stream-4k-hd-webcam (accessed on 30 July 2020).
21. Huawei Global. Huawei 4G wingle E8372 specifications. Available online: https://consumer.huawei.com/en/routers/e8372/ specs/ (accessed on 14 August 2020).
22. Shi, D.; Dai, X.; Zhang, X.; Quan, Q. A practical performance evaluation method for electric multicopters. *IEEE/ASME Trans. Mechatron.* **2017**, *22*, 1337–1348.
23. Hohpe, G.; Woolf, B. *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*; Addison-Wesley Professional: Boston, MA, USA, 2003.
24. Fu, G.; Zhang, Y.; Yu, G. A fair comparison of message queuing systems. *IEEE Access* **2021**, *9*, 421–432, .
25. Kreps, J.; Narkhede, N.; Rao, J. Kafka: A distributed messaging system for log processing. In Proceedings of the 6th International Workshop on Networking Meets Databases, Athens, Greece, 12–16 June 2011; Volume 11, pp. 1–7.
26. Dixit, S.; Madhu, M. Distributing messages using rabbitmq with advanced message exchanges. *Int. J. Res. Stud. Comput. Sci. Eng.* **2019**, *6*, 24–28.
27. Yongguo, J.; Qiang, L.; Changshuai, Q.; Jian, S.; Qianqian, L. Message-oriented middleware: A review. In Proceedings of the 2019 5th International Conference on Big Data Computing and Communications (BIGCOM), Qingdao, China, 9–11 August 2019; pp. 88–97.
28. Christudas, B. ActiveMQ. In *Practical Microservices Architectural Patterns*; Amazon: Seattle, WA, USA, 2019; pp. 861–867.
29. Ramasamy, K. Unifying messaging queuing streaming and light weight compute for online event processing. In Proceedings of the 13th ACM International Conference on Distributed and Event-based Systems, Darmstadt, Germany, 24–28 June 2019; p. 5.
30. Django, The Web Framework for Perfectionists with Deadlines. Available online: https://www.djangoproject.com/ (accessed on 11 September 2020).
31. Leaflet, a JavaScript Library for Interactive Maps. Available online: https://leafletjs.com/ (accessed on 18 September 2020).
32. ZeroMQ. Available online: https://zeromq.org/ (accessed on 19 September 2020).
33. GitHub, a Set of Python Classes That Transport OpenCV Images from One Computer to Another Using PyZMQ Messaging. Available online: https://github.com/jeffbass/imagezmq# why-use-imagezmq (accessed on 20 September 2020).
34. Shaban, M.; Ani, A.; Awad, F.H. The JPEG image compression algorithm. *Int. J. Adv. Eng. Technol.* **2013**, *6*, 1055–1062.
35. MySQL. Available online: https://www.mysql.com/ (accessed on 21 September 2020).
36. MongoDB: The Most Popular Database for Modern Apps. Available online: https://www.mongodb.com/ (accessed on 22 September 2020)
37. Čerešňák, R.; Kvet, M. ScienceDirect comparison of query performance in relational a non-relation databases. *Transp. Res. Procedia* **2019**, *40*, 170–177.
38. Wang, C.Y.; Bochkovskiy, A.; Liao, H.Y.M. Scaled-yolov4: Scaling cross stage partial network. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Virtual, 19–25 June 2021; pp. 13029–13038.