electronics

MDPI

Article

An Improved Deep Convolutional Neural Network-Based Autonomous Road Inspection Scheme Using Unmanned Aerial Vehicles

Syed-Ali Hassan, Tariq Rahim and Soo-Young Shin *

Department of IT Convergence Engineering, Kumoh National Institute of Technology, Gumi 39177, Gyeongbuk, Korea; syedali@kumoh.ac.kr (S.A.H.); tariqrahim@ieee.org (T.R.)
* Correspondence: wdragon@kumoh.ac.kr

Abstract: Recent advancements in the field of machine learning (ML) provide opportunity to conduct research on autonomous devices for a variety of applications. Intelligent decision-making is a critical task for self-driving systems. An attempt is made in this study to use a deep learning (DL) approach for the early detection of road cracks, potholes, and the yellow lane. The accuracy is not sufficient after training with the default model. To enhance accuracy, a convolutional neural network (CNN) model with 13 convolutional layers, a softmax layer as an output layer, and two fully connected layers (FCN) are constructed. In order to achieve the deeper propagation and to prevent saturation in the training phase, mish activation is employed in the first 12 layers with a rectified linear unit (ReLU) activation function. The upgraded CNN model performs better than the default CNN model in terms of accuracy. For the varied situation, a revised and enriched dataset for road cracks, potholes, and the yellow lane is created. The yellow lane is detected and tracked in order to move the unmanned aerial vehicle (UAV) autonomously by following yellow lane. After identifying a yellow lane, the UAV performs autonomous navigation while concurrently detecting road cracks and potholes using the robot operating system within the UAV. The performance model is benchmarked using performance measures, such as accuracy, sensitivity, *F*1-*score*, *F*2-*score*, and dice-coefficient, which demonstrate that the suggested technique produces better outcomes.

Keywords: autonomous navigation; autonomous road inspection; computer vision; drone; robots; neural network; UAV

Citation: Hassan, S.-A.; Rahim, T.; Shin, S.-Y. An Improved Deep Convolutional Neural Network-Based Autonomous Road Inspection Scheme Using Unmanned Aerial Vehicles. *Electronics* **2021**, *10*, 2764. https://doi.org/10.3390/electronics10222764

Academic Editor: Sung Ju Hwang

Received: 15 September 2021
Accepted: 3 November 2021
Published: 12 November 2021

1. Introduction

Machine learning (ML) is a subgroup of artificial intelligence (AI), and deep learning (DL) is a subgroup ML that has received a lot of attention recently. It is widely utilized in self-driving automobiles, among other things [1]. Unmanned aerial vehicles (UAVs) are a popular and fascinating platform for deploying deep learning-based systems. In a related study, consideration was given to the use of neural network-based algorithms in a drone. In addition, Ref. [2] discusses the use of neural networks to recognize drones. The convolutional neural network (CNN) is the most well-known deep learning-based neural network, and it performs well in object detection, particularly in real-time detection. As described in [3], a capable neural network CNN can be used for segmentation, recognition, and detection.

Additionally, CNN also pulls features from the image automatically, and its performance is continually improving. Previously, object identification using CNN has progressed significantly, and several CNN-based object detectors have been created, including faster R-CNN [4], you only look once (YOLO) [5], and single shot detector SSD [6]. Region-based and single-shot detectors are two types of CNN-based object detectors, respectively. In general, detectors that detect in relation to a region are computationally massive and require a high-configured system, particularly a high-powered GPU to execute, whereas

single-shot detectors only require one CNN to detect. YOLO is a real-time object detection system that produces excellent results [7].

Previously, most techniques relied on background subtraction, as discussed in [8], or classification techniques, such as the Haar cascade algorithm for object detection, as discussed in [9]. Manual inspections in forest fields take a long time and require a lot of manpower. The disease in radish fields is detected [10] by utilizing computer vision technology with a camera attached to a drone in order to change the manual inspection process into automatic. The CNN is utilized for detecting cattle by using a drone as discussed in [11] in order to analyze the real-time performance of cattle detection. An autonomous computer vision-based landing and detection system is introduced in [12] along with the PID controller for the detection of the target and for safe landing fuzzy logic controller is used. The drone wireless changing technique was presented [13] by employing a Hill-climbing algorithm to regulate the coupling between the transmitter and a receiver without the need of any sensor. CNN-based object identification has been used in medical science for rapid diagnosis during the last decade, as detailed in [14]. UAV inspection technology has been successfully applied in the realm of power grid; in [15], a multi-rotor UAV-based power grid intelligent patrol and maintenance system is presented.

The following are the most common road pavement flaws: Potholes and cracks are difficult to spot during a road inspection. Furthermore, manual examination of each road is difficult and costly because it is time-consuming and takes a substantial amount of staff to find potholes and cracks [16]. The default version of YOLO is utilized for pavement distress detection and classification in [17] and their achieved accuracy is 73.64%. In this paper, we improved the default YOLO CNN model in order to achieve better accuracy because 73.64% is not a good accuracy for real-time objects detection. The key contribution of this paper is the improvement in convolutional neural network by adding 6 convolutional layers in the model and implemented the mish activation function in first 12 layers to increase the detection accuracy. This study discusses an improved CNN-based detection system for autonomous road inspection with model optimization for real-time identification of potholes, cracks, and yellow lane. The yellow lane, in particular, is used as a reference for a drone to move autonomously while detecting cracks and potholes. The object detector YOLO is utilized and improved with respect to convolutional layers and activation function.

The sections of this paper are organized as follows: Section 2, define relevant work. Section 3 explains the object detection model for detecting potholes, cracks, and yellow lanes, as well as the tracking component of the yellow lane. Section 4 summarizes the dataset collecting process, as well as the final results. Finally, in Section 5, the study is wrapped up and future research is discussed.

2. Related Work

Automatic detection of potholes and cracks is proposed for speedy and reliable road defects analysis rather than depending on the laborious and time-consuming process of manual road inspection [18]. Autonomous navigation of drone for the inside environment by utilizing deep neural networks is discussed in [19]. Additionally, the outdoor environment navigation is utilized and discussed in [20] for shipment purpose.

UAVs are becoming increasingly popular. On certain occasions, they are physically controlled by a joystick in a mobile application, but, on others, they navigate autonomously by tracking an object, as seen in [21]. A drone is also guided by utilizing inertial navigation systems and GPS, which gives the altitude, velocity, and position which is critical for the navigation of the drone, as presented in [22]. Furthermore, a drone is used to rescue and search humans near the sea area by person detection system developed using CNN, as discussed in [23].

Furthermore, the CNN is utilized in autonomous drone racing, capable of beating a drone controlled by human as proposed in [24]. Moreover, the CNN is used to detect sidewalk area and implemented in a drone to fly autonomously, to deliver products,

without disturbing the traffic flow as presented in [25]. Drones are not only designed to control using a joystick or fly autonomously but they can also be controlled using hand gestures, which provides an opportunity to design a low-cost system as presented in [26].

Airport road surface inspection is proposed in [27]. In the airport, the good condition of the road is very important for flight landing, the SSD mobile net and Mask-Rcnn models are utilized in this paper for real-time airport road surface inspection. In [28], UAV is utilized for catenary support device inspection with improved faster-Rcnn to ensure safety and reliability in railway systems. A low power drone is used in road assets classification. The dataset is trained on Mobilenet v2 which is a CNN based network by utilizing transfer learning in [29]. The accuracy which is achieved is 81.33% which is reasonable, but not the best and it can be further improved.The CNN model that is proposed in this paper achieved better accuracy.

Cracks on road can cause severe damages, to detect road cracks, the you only look once (YOLO) algorithm is used in [30] to detect the cracks on the road in real-time and to make the inspection process easy and quick. In order to connect IOT devices and to utilize communication technology for certain purposes, The Industrial Internet of Things (IIoT) is being implemented in [31].

3. Proposed Scheme

The faults on the road, such as potholes and cracks, are general issues and demand continuous surveillance using UAV with intelligent algorithms for timely localization. An autonomous road inspection method is proposed where Jetson hardware is utilized with the integration of Bebop drone by utilizing WIFI medium as communication. The algorithm running on Jetson hardware received the images from UAV to detect the trained objects. The robot operating system (ROS) is a resilient framework run on Jetson hardware with the deep learning object detector (YOLO) model. The real-time images acquired from the UAV are obtained and processed by the Jetson. The proposed deep learning model is trained on three classes named yellow lane, cracks, and potholes. The tracking command enables when the detected class match to yellow lane class. The distance and position of the detected object are determined to estimate altitude, roll, pitch, and yaw values by utilizing a tracking algorithm. UAV receives these calculated values to move forward by tracking and following the yellow lane on the road. The detected image will be forwarded to the server if the detected class is matched as a pothole or crack by utilizing Wi-Fi or 5G. Figures 1 and 2 depict the flowchart and the architecture of the proposed system in a detailed fashion. The Jetson is placed on the bebop drone in order to run the trained model as seen in Figure 3.

Technically, improved CNN model with ROS both are running on Jetson hardware. when the UAV take-off, it continuously sends video as a frame to Jetson hardware which is mounted on UAV. Furthermore, the Jetson hardware forward those frames to the YOLO algorithm which is responsible for detection, if the yellow lane class is detected then, the tracking algorithm activates, by utilizing ROS, the movement command forward, back to UAV. If YOLO detects cracks or pothole class, then it directly sends those detected frames to the server. This process is instructed to continue work until the UAV is turned off. The model is tested and works perfectly in daylight normal weather conditions. Further, for the future work night time inspection approach can be considered. The algorithm is able to detect different lighting conditions because before training the dataset, the color and position augmentation techniques are applied, such as scaling, rotation, flipping, brightness, saturation, etc.

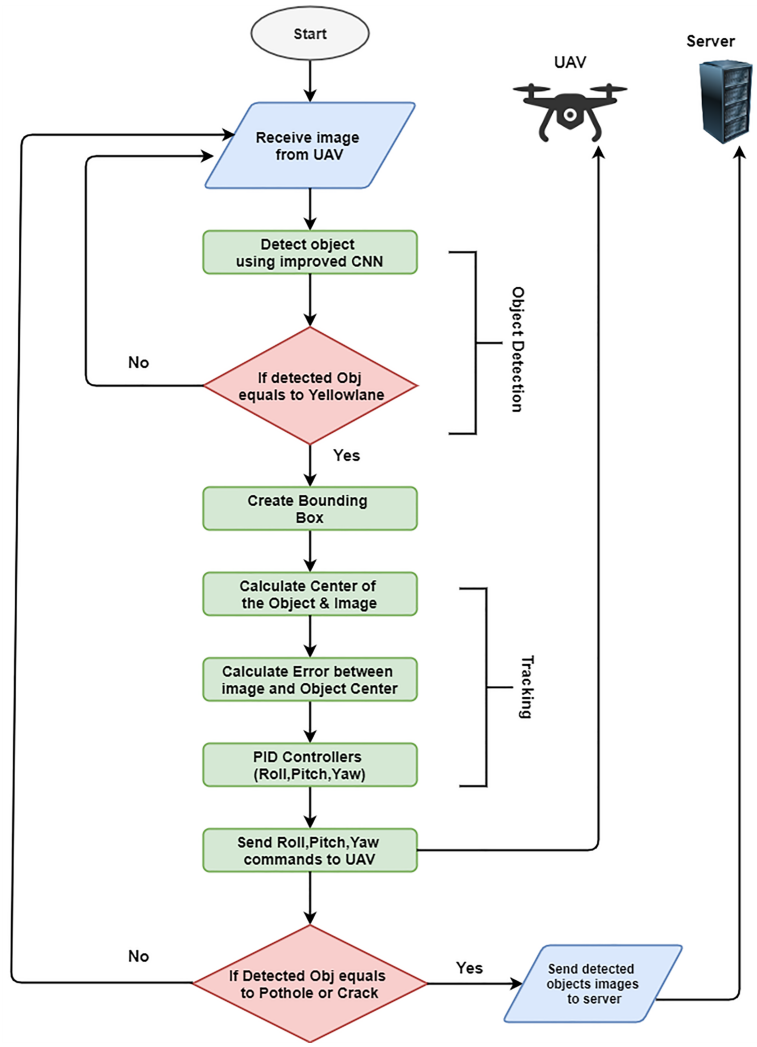

Figure 1. Flowchart of the proposed system.

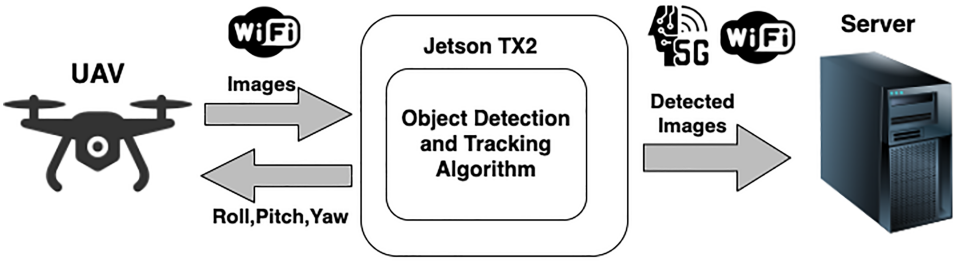

Figure 2. The proposed system architecture.

Figure 3. The mounted tx2 on the modified bebop drone.

3.1. Integration of Detection Model in UAV

The identification and tracking techniques are implemented using ROS within the UAV in the proposed system. The ROS mostly deal with two nodes: node 01, which is in charge of object recognition and tracking, and node 02, which is in charge of UAV driver packages. Message communication between both nodes is accomplished through the use of ROS topics [32], as shown in Figure 4. Each topic is responsible for storing data as a message in order to communicate between two nodes. In the proposed system, we have employed */UAV/reset*, */UAV/land* as a reset and landing topics. */UAVtakeoff* as a UAV take-off topic and */cmd_vel (Command Velocity)*. /cmd_vel is responsible to send the altitude (Z), pitch, yaw, and roll commands to Bebop drone 2. Moreover, Node 01 subscribed two ROS topics like /UAV/front image_raw; and /UAV/nav data that are responsible for carrying video and navigation data, respectively. On the other hand, Node 01 published 04 ROS topics that are subscribed by Node 02 and Node 02 publishes 02 ROS topics, expressly, */UAV/front/image_raw* and */UAV/navdata*.

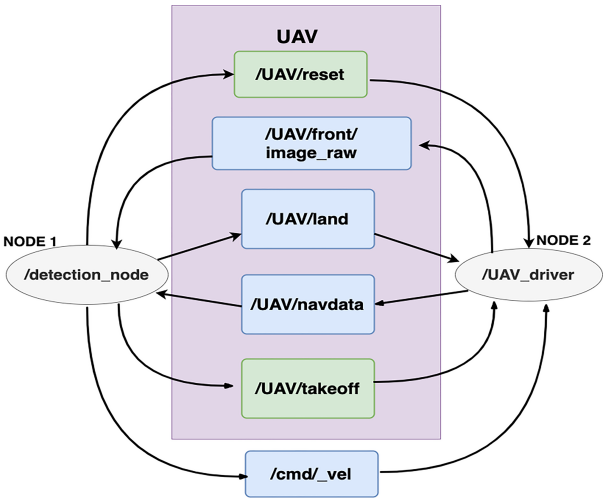

Figure 4. ROS node graph.

3.2. Improvements in the Detection Model

During training, the input image in the network is initially divided into a grid. Each cell is responsible for determining the Bounding box "B". The bounding box is made up of five primary characteristics: x, y in the centre, width, and height specified as h and w, respectively; and cs, which stands for confidence. The presence of an object in the bounding box is determined by confidence cs. The quickest CNN network detects potholes, yellow lanes, and cracks. The structure of the small YOLOv3 object detection model, which is based on CNN, has been modified to improve its accuracy. To run the object detection algorithm on less powerful devices such as Jetson and Raspberry Pi. The tiny version of YOLO is the best choice because it runs very fast and can be executed on low specification devices because it comprises of six polling layers and seven convolutional layers. However, fewer layers can decrease the accuracy but improved speed. The activation function leaky (Relu) is used in the default tiny version of YOLO and on its final layer consists of a linear activation function.

The same dataset is trained on both the default and upgraded versions of YOLO's CNN model, and after testing, it is confirmed that the improved version increased the accuracy. YOLO's default configuration includes 7 convolutional layers with Leaky ReLu as an activation function. Because the layers are insufficient and the network is not very deep, the default version of tiny YOLO cannot adequately extract the features from the image. To improve the model's accuracy, 13 convolutional layers are added, and the mish activation function is used in the first 12 layers, with a Softmax layer as the output layer and two fully connected layers (FCN). Figure 5 illustrates the architecture of the proposed deep CNN model for potholes, cracks and yellowlane detection and Figure 6 depicts

the inclusion of convolutional layers in the proposed enhanced YOLO model. Although the addition of extra layers can improve accuracy, it also increases the number of model parameters, which consumes memory resources and increases the number of calculations in the network. To minimize unnecessary processing, Resnet [33] proposed adding a 1×1 CNN layer to reduce the amount of computation. In this study, we applied this technique to propose a 1×1 convolution kernel. This strategy not only reduces calculation time by preserving memory resources, but it also enhances feature extraction and boosts the non-linear function of excitation. The activation function Mish, which replaced the ReLU activation function in the first 12 layers, is utilized for deeper propagation. Mish creates deeper propagation in the CNN layer as discussed in [34] as illustrated in Figure 7. Mish activation function was used to create self-regularization, deeper propagation of information, and better capping avoidance. After adding more convolutional layers and changing the activation function, the model detection accuracy was improved. Figure 8 shows the images utilized from the own created dataset and it is further discussed in Section 4.1.

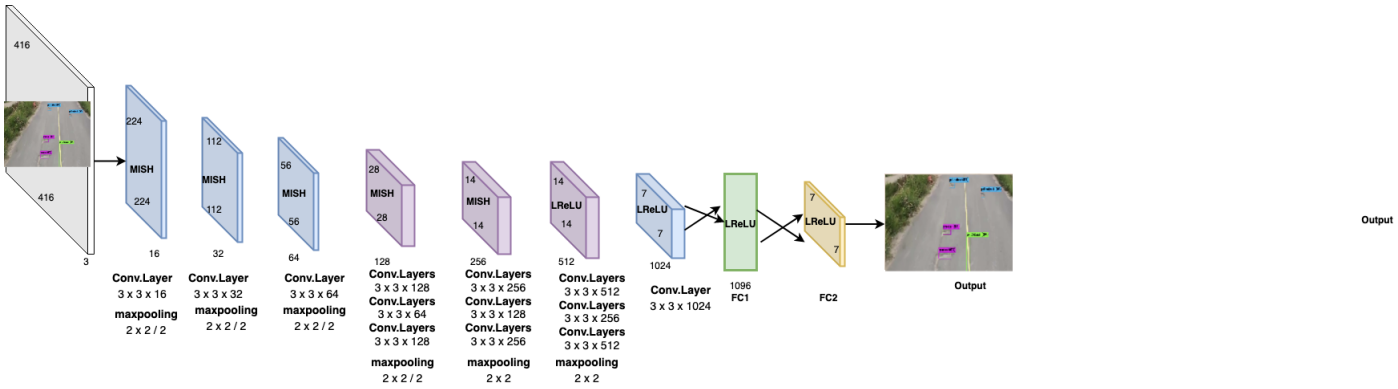

Figure 5. Architecture of the proposed deep CNN for yellow lane, cracks, and potholes detection.

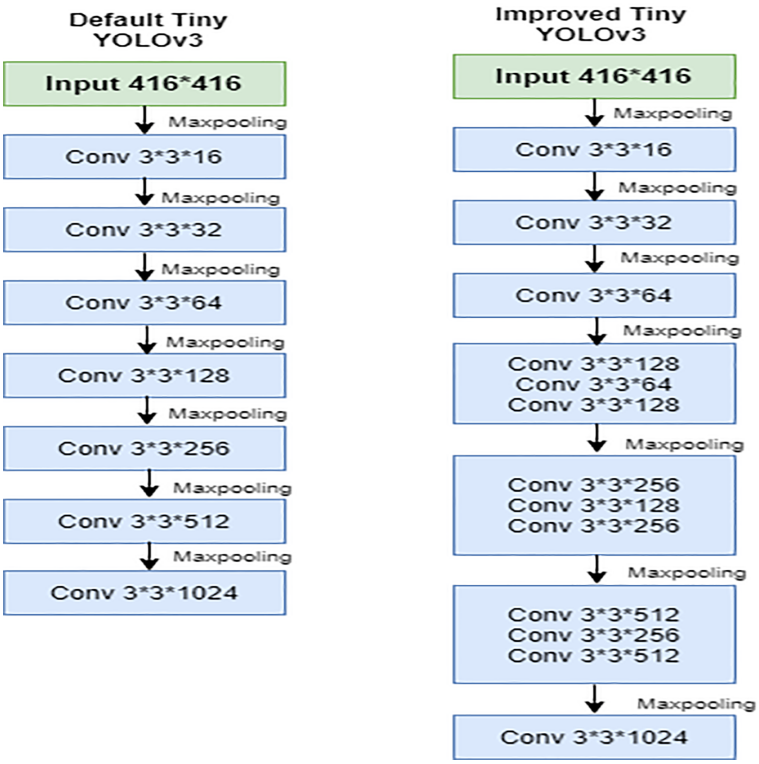

Figure 6. Improved and default model.

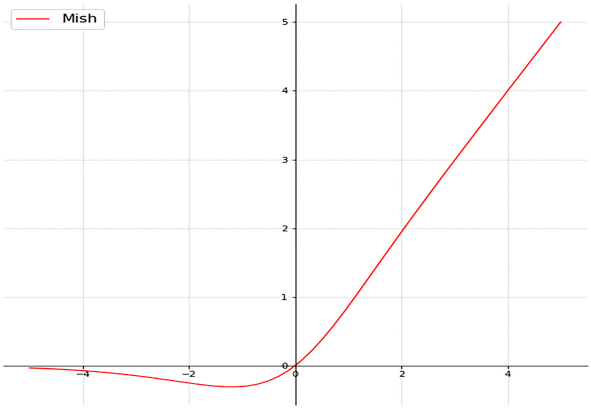

Figure 7. Mish activation function.

Figure 8. Some preliminary images from the own created dataset for the implementation.

3.3. Object Tacking and Navigation

When the model detects the yellow lane, the CNN show the boundary box which includes a tracked area of the detected class object, that is shown as pixel values; ($x_{\min}$, $y_{\min}$), ($x_{\max}$, $y_{\min}$), ($x_{\min}$, $y_{\max}$), ($x_{\min}$, $y_{\min}$) as illustrated in Figure 9. This boundary box holds the position of a detected object on an image. The center of an object is calculated as:

$$(x_o, y_o) = \left(\frac{x_{\min} + x_{\max}}{2}, \frac{y_{\min} + y_{\max}}{2} \right) \tag{1}$$

Object tracking required center values calculated as:

$$\begin{aligned} (x_i, y_i) &= \left(\frac{img_{width} - img_{width}}{2}, \frac{img_{height} - img_{height}}{2} \right) \\ (x_i, y_i) &= (0, 0) \end{aligned} \tag{2}$$

The image center value is (0, 0). The object and image center error is represented as:

$$\begin{aligned} e_x(t) &= x_o - x_i = x_o \\ e_y(t) &= y_o - y_i = y_o \end{aligned} \tag{3}$$

$$e_x(t) \text{ and } e_y(t) \tag{4}$$

By the equation above, for properly tracking the detected object, $e_x(t)$ and $e_y(t)$ must always near or be equal to zero. Moreover, for effective tracking, the center should be matched to the middle of the image to track properly. The middle of the yellow lane detects the value of the boundary box, which then utilize for tracking to follow the yellow lane by using these movement commands roll, pitch, altitude and yaw as illustrated in Figure 10. These four control commands are responsible to move the bebop drone. The responsibility of roll command is to navigate the UAV left or right, and for upward or downward movement pitch command is responsible. Yaw command is responsible for the rotation of UAV clockwise or counter clockwise and altitude is used for left or right movements. A control scheme that is based on PID controllers is illustrated in Figure 11 which shows the detected object tracking control scheme as a block diagram. The basic movements of the bebop drone are shown in Figure 12. Calculation of relative distance between the UAV and yellow lane is also completed. To measure the relative distance the width of the detected yellow lane is calculated. If the width of the bounding box is higher than the defined value, then the UAV will move backwards otherwise it will resume its forward movement.

Figure 9. Bounding box position with detected object.

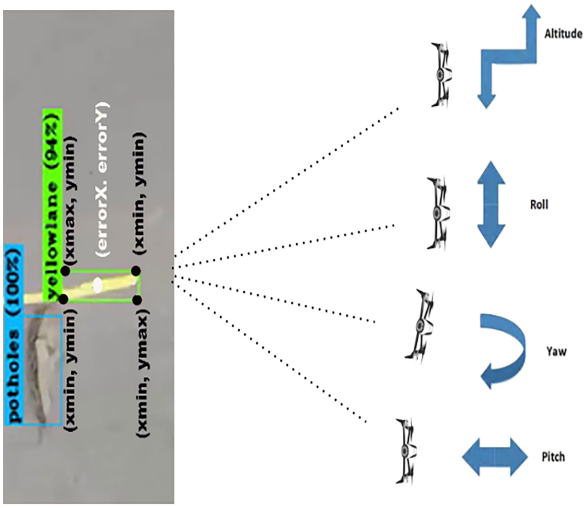

Figure 10. Autonomous UAV navigation.

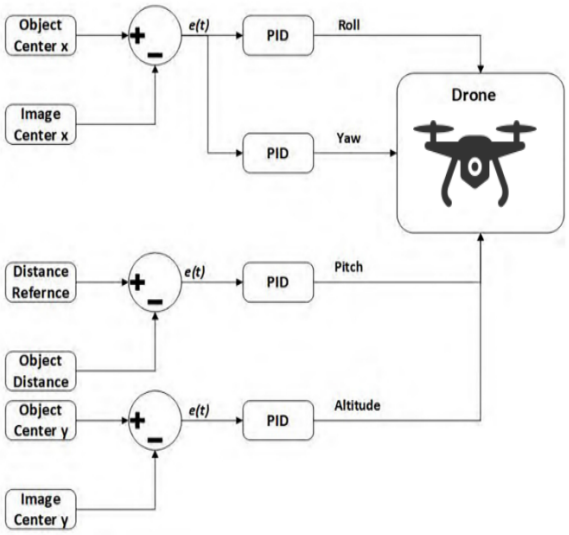

Figure 11. Diagram of the control scheme for detected object tracking.

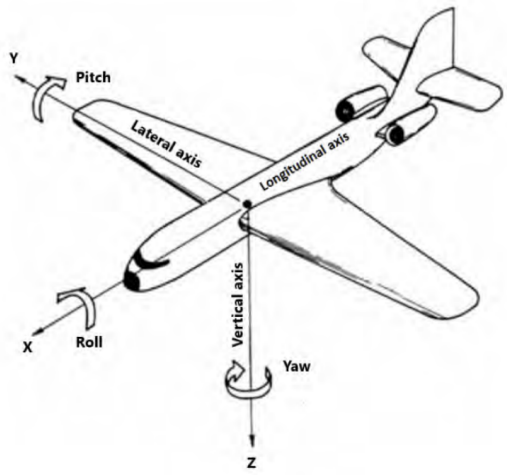

Figure 12. Drone movement axis.

4. Experimental Analysis

The gathered dataset is emphasized in detail and describes the results obtained by employing the enhanced model with specifications, as well as the results of the default model. Furthermore, the mean average precision(mAP) and accuracy reported above are compared for both the default and modified models.

4.1. Dataset Specifications

Due to the scarcity of the annotated dataset, this phase was considered very carefully by creating our dataset for real-time detection. Road crack, pothole, and yellow lane datasets were created by utilizing high definition camera. The dataset was split into 80% to train the model and 20% for validating the model. In total, ten thousand images were used. Three different classes are created such as pothole, cracks, and yellow lane, where 3333 images are used for pothole, 3334 for cracks and 3333 for yellow lane.

4.2. Results and Training

In training phases, it is instructed by programming to produced weights after every 10,000 iterations. The weight with highest mean average precision (mAP) is selected to test the model. After completing the training on the improved model, the highest achieve

mean average precision (mAP) is 94% as plotted in Figure 13. Furthermore, the detection of potholes, yellow lane and cracks are illustrated in Figure 14. To compare the results, the default YOLO model is also trained and the accuracy of default model is calculated as 89% and its highest mean average precision (mAP) is 89% as plotted in Figure 15; while, the improved model accuracy is noted as 95% which is a good improvement. Hence, it can be observed that modifying the activation function of the model and adding more CNN layers to create a model deeper ultimately improved its accuracy. A case study for detecting road pavement distress using CNN is reported in [35], and their attained accuracy is 83.8%, indicating that our improvement approach is superior to them because we achieved 95% accuracy with our improved model. The optimizer stochastic gradient descent SGD was utilized to train both models. The learning rate in both models is 0.001. The details about other parameters are shown in Table 1. In the training phase of the improved model, the training was stopped at 10,000 iterations. The algorithm is programmed to produce the best weight when the training is manually stopped. The accuracy of the model was calculated by utilizing the best output weight. The powerful GPU is used to train both models. Both default and improved models are trained with a subdivision of 4 and batch size of 64. Table 2 shows the performance results for both models where it can be observed that the performance of the proposed CNN model outperforms the default model. Table 3 illustrates the detection time for both models where it can see that the proposed CNN model detects the objects in 4.84 ms and on the other hand default model detects in 4.81 ms. It confirms that when modifying the model deeper not only increase its accuracy but also increase its detection time. The diagram shows the comparison of both models detection time, illustrated in Figure 16.

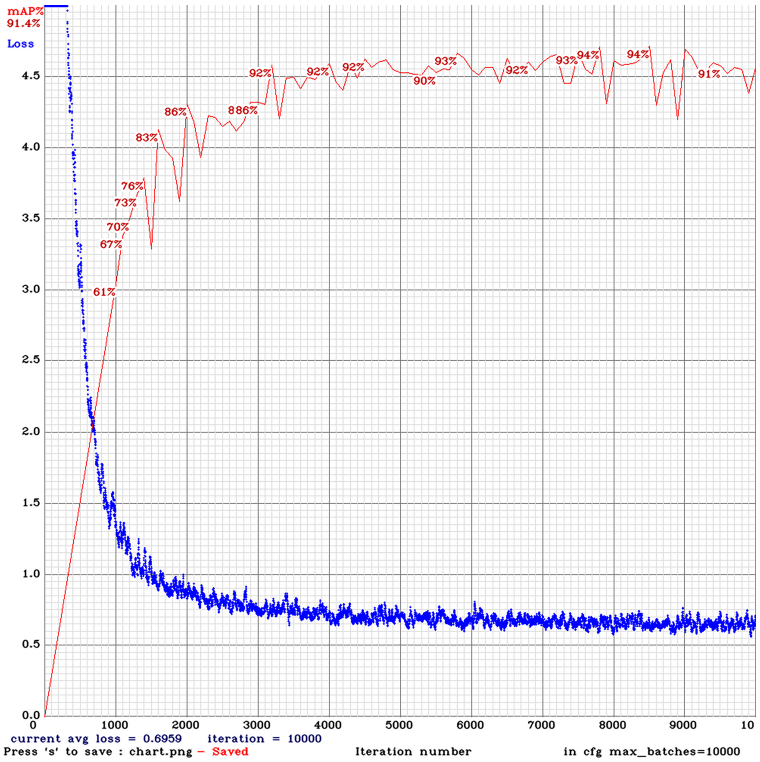

Figure 13. Training phase of the improved model.

For a fair comparison, an attempt for bench-marking the performance evaluation as a detection model is made to reflect the robustness of the proposed CNN model. The training parameters defined in Table 1 were kept constant for both models, and it can be observed in Table 4, that the proposed CNN model outperforms the default CNN model in terms of performance metrics opted for the detection of three classes. The model also suffers a high detection time-lapse which become unacceptable for real-time applications. In order

to recreate this experiment, the yolov3tiny model should be trained on the darknet by using the parameters mentioned in Table 1. For an improved version of YOLO extra layers should be added in the model with changes in the activation function.

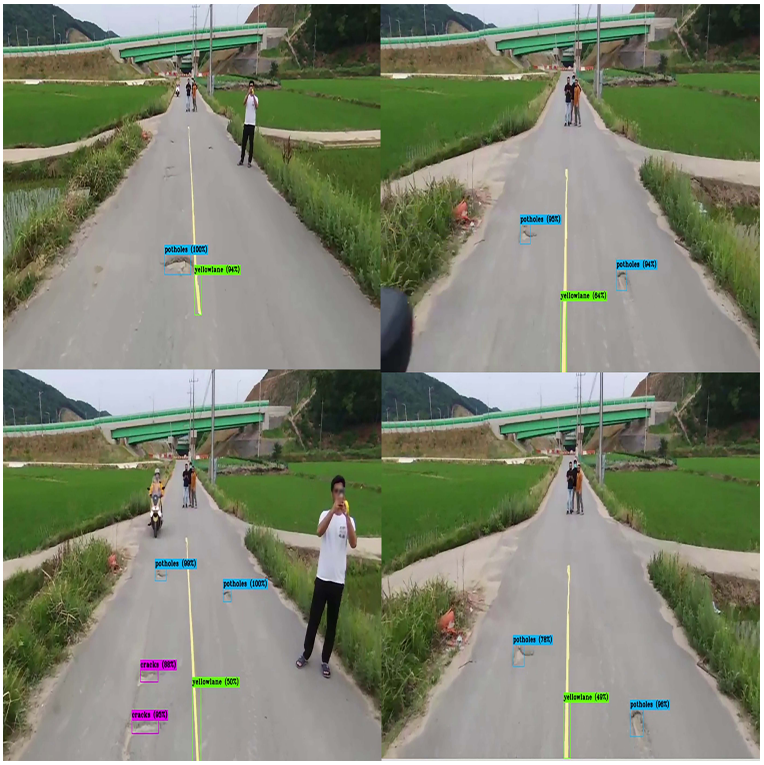

Figure 14. Real-time results of improved model.

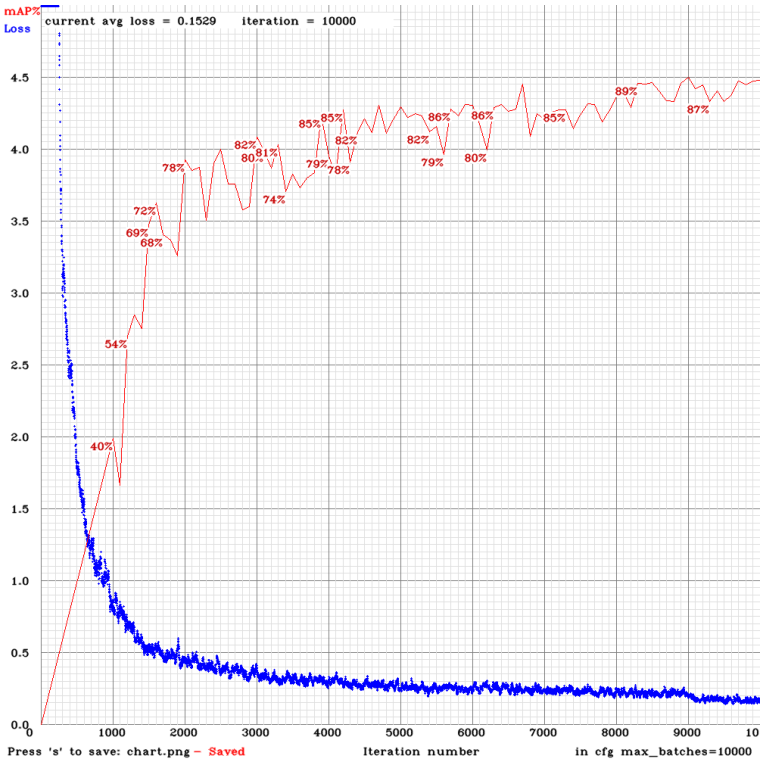

Figure 15. Training phase of the default model.

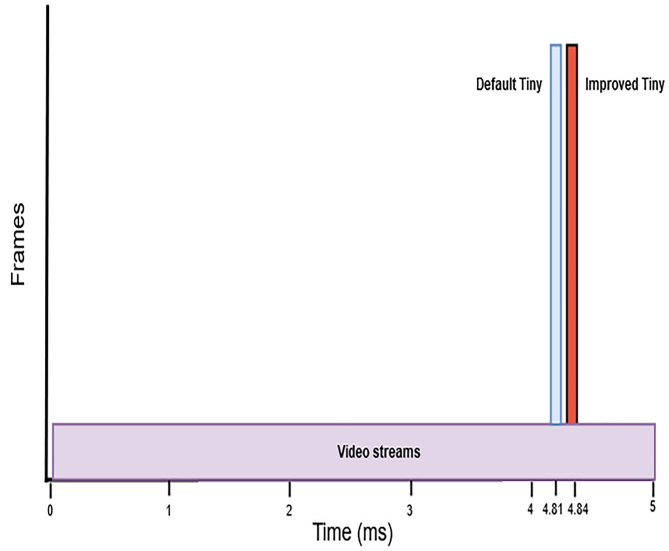

Figure 16. Detection time comparison of both models.

Table 1. Parameters opted during training phase for both models.

Parameters in Network	Configured Values
Input dimension	416 × 416
Learning rate (η)	0.001
Optimizer	SGD
Size of batch	64
Momentum	0.9
Stride	1
Iterations (t)	10,000
Subdivisions	4
Exposure	1.5
Saturation	1.5
Channels	3
Decay	0.0005
Momentum	0.9
Hue	0.1

Table 2. The performance comparison with respect to detection of the default and proposed model.

Data Set	Performance Metrics			
	Default Model (%)		CNN Model Proposed (%)	
Cracks Class	Pre	83.24	Pre	87.63
	Sen	82.81	Sen	84.02
	*F*1-*score*	83.02	*F*1-*score*	85.78
	*F*2-*score*	82.89	*F*2-*score*	84.71
	Dice-coefficient	83.02	Dice-coefficient	85.78
Pothole Class	Pre	97.58	Pre	98.26
	Sen	89.55	Sen	90.12
	*F*1-*score*	93.36	*F*1-*score*	94.04
	*F*2-*score*	91.04	*F*2-*score*	91.63
	Dice-coefficient	88.82	Dice-coefficient	91.04
Yellow Lane Class	Pre	94.92	Pre	93.26
	Sen	88.96	Sen	89.45
	*F*1-*score*	91.84	*F*1-*score*	91.31
	*F*2-*score*	90.09	*F*2-*score*	90.10
	Dice-coefficient	91.85	Dice-coefficient	92.11

Table 3. Both models results with comparison.

Model	Accuracy%	mAP%
YOLOv3TinyImproved	95.00	94.00
YOLOv3Tiny	89.00	89.00

Table 4. Performance comparison with default YOLOv3 model.

Data Set	Performance Metrics			
	CNN Model (%)		CNN model Proposed (%)	
Cracks Class	Pre	80.26	Pre	87.63
	Sen	80.07	Sen	84.02
	F1-score	79.85	*F1-score*	85.78
	F2-score	80.03	*F2-score*	84.71
	Dice-coefficient	81.11	Dice-coefficient	85.78
Pothole Class	Pre	90.15	Pre	98.26
	Sen	86.40	Sen	90.12
	F1-score	88.16	*F1-score*	94.04
	F2-score	85.37	*F2-score*	91.63
	Dice-coefficient	82.75	Dice-coefficient	91.04
Yellow Lane Class	Pre	90.15	Pre	93.26
	Sen	86.65	Sen	89.45
	F1-score	86.37	*F1-score*	91.31
	F2-score	85.58	*F2-score*	90.10
	Dice-coefficient	86.02	Dice-coefficient	92.11

4.3. Performance Metrics for Evaluation

The performance metrics which was utilized for the evaluation of the detection potholes, yellow lane, and cracks, are measured by utilizing some important parameters as discussed below:

True Positive (*TP*): It is classified as a true position when the centroid falls within the defined objects. When multiple true output detection happens in the frame then true positive is considered as one.

True Negative (TN): It occurs when the detection is negative but true which means that the selected frames do not have defined objects.

False Positive (*FP*): In the class ground-truth the detected centroid does not fall in the defined objects.

False Negative (*FN*): Objects that are defined in the class are not available in the frame.

The parameters above are utilized to efficiently evaluate the performance of an improved model.

Precision: This metric is used to calculate how accurately the improved model recognizes the defined objects of the class.

$$\text{Precision (Pre)} = \frac{TP}{TP + FP} \times 100 \tag{5}$$

Sensitivity: This metric is also called recall or the true positive rate, which measures the proportion of the actual class of the defined object correctly.

$$\text{Sensitivity (Sen)} = \frac{TP}{TP + FN} \times 100 \tag{6}$$

*F*1-*score*and *F*2-*score*: *F*1- and *F*2-*score* can be defined by the harmonic mean between sensitivity and precision within a specified range of [0, 1]. In order to maintain the sensitivity and precision, both these scores were considered. The *F*1-*score* is given below:

$$F1 - score = \frac{2 \times \text{Sen} \times \text{Pre}}{\text{Sen} + \text{Pre}} \times 100 \tag{7}$$

while the *F*2-*score* defined and given as:

$$F2 - score = \frac{5 \times \text{Pre} \times \text{Sen}}{4 \times \text{Pre} + \text{Sen}} \tag{8}$$

Dice Coefficient: Below are the metrics used in order to compare the pixel-wise result between ground truth that ranges [0, 1] and prediction.

$$\begin{gathered} \text{Dice coefficient } (E, F) = \frac{2 \times |E \cap F|}{|E| + |F|} \\ = \frac{2 \times TP}{2 \times TP + FP + FN} \end{gathered} \tag{9}$$

4.4. Loss Function

The object detection algorithm YOLO overall process use the loss calculation commonly called sum square error [36]. The simple differences of addition, such as classification error, coordinate errors and IOU errors involved in the YOLO end to end network. In order to calculate the loss function, the following formula can be used. Loss function is also calculated in [37], CNN is used in medical science in order to detect the polyp in colonoscopy images and also improved to increase the accuracy of the model.

$$loss = \sum_{i=0}^{g^2} coordErr + iouErr + clsErr \tag{10}$$

Output weight of every loss function is calculated in order to estimate the total loss. At the time of training, the model indicates unstable behavior and divergence when the coordinate error is continual with a classification error. Hence, the $\lambda = 5$ is a coordinate error weight value. In order to evade the confusion between the grid consist of an object and the grid that does not consist of an object, YOLO assigns λ_{noobj} for the IOU error. During training the dataset the total loss function achieved can be defined as below:

$$\begin{gathered} loss = \lambda_{coord} \sum_{i=0}^{g^2} \sum_{J=0}^{B} l_{ij}^{obj} \left[(a_i - \hat{a}_i)^2 + \left(b_i - \hat{b}_i \right)^2 \right] + \lambda_{coord} \sum_{i=0}^{g^2} \sum_{j=0}^{B} l_{ij}^{obj} \\ \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] + \sum_{i=0}^{g^2} \sum_{j=0}^{B} l_{ij}^{obj} (c_i - \hat{c}_i)^2 + \lambda_{noobj} \sum_{i=0}^{g^2} \sum_{j=0}^{B} l_{ij}^{obj} (c_i - \hat{c}_i)^2 \\ + \sum_{i=0}^{g^2} l_i^{obj} \sum_{c \in class} \left(R_i(c) - \hat{R}_i(c) \right)^2 \end{gathered} \tag{11}$$

In Equation (11), g represents the number of grids while B is used to represent each number of cell related to the prediction boxes. The (a, b) is used in order to show the coordinate center of each cell. Additionally, its height and width are represented as h and w, respectively. Moreover, prediction confidence is defined as c; while R is used to label the object confidence in the class. The λ_{coord} is used to indicate the weight of the loss function position. λ_{noobj} represents the classification weight of loss function. Value is set to 1 when a trained object of the class is present otherwise the value is 0.

5. Conclusions and Future Work

The road cracks, potholes, and yellow lane were detected by implementing the improved deep CNN model. Navigation of the drone autonomously was accomplished by following and tracking the detected yellow lane. The purpose of following the yellow lane to autonomously move the drone and report road damages on the server in order to

perform autonomous road inspection. A high-quality dataset was collected to attain good results. Afterwards, the results achieved from both of the trained models were checked and compared with respect to the accuracy, detection time, and mean average precision (mAP).

The future work can be considered by creating a large dataset and compare the current improved model results with other object detectors, such as Mask-RCNN, etc.

Author Contributions: S.-A.H.: Model implementation, data set preparation, writing, and testing the model. T.R.: Conceptualization, writing, and data set labelling along with the performance metrics evaluation. S.-Y.S.: Supervision, visualization, review and editing, and resources. All authors have read and agreed to the published version of the manuscript.

Funding: This paper was supported by the National University Development Project in 2021.

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

References

1. Mantoro, T.; Ayu, M.A. Multi-faces recognition process using Haar cascades and eigenface methods. In Proceedings of the 2018 6th International Conference on Multimedia Computing and Systems (ICMCS), Rabat, Morocco, 10–12 May 2018; pp. 1–5.
2. Hassan, S.A.; Rahim, T.; Shin, S.Y. Real-time UAV Detection based on Deep Learning Network. In Proceedings of the 2019 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Korea, 16–18 October 2019; pp. 630–632.
3. Audebert, N.; Le Saux, B.; Lefèvre, S. Segment-before-detect: Vehicle detection and classification through semantic segmentation of aerial images. *Remote Sens.* **2017**, *9*, 368. [CrossRef]
4. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 91–99. [CrossRef]
5. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once:Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
6. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In *Proceedings of the European Conference on Computer Vision*; Springer: Cham, Switzerland, 2016; pp. 21–37.
7. Redmon, J. Darknet: Open Source Neural Networks in C. 2016. Available online: http://pjreddie.com/darknet/ (accessed on 8 November 2021).
8. Philipps, J.J.; Bönninger, I.; Weigert, M.; Vásquez, J. Automatic tracking and counting of moving objects. In Proceedings of the 3rd IEEE International Work-Conference on Bioinspired Intelligence, Liberia, Costa Rica, 16–18 July 2014; pp. 93–97.
9. Budiman, R.A.M.; Achmad, B.; Arif, A.; Zharif, L. Localization of white blood cell images using Haar cascade classifiers. In Proceedings of the 2016 1st International Conference on Biomedical Engineering (IBIOMED), Yogyakarta, Indonesia, 5–6 October 2016; pp. 1–5.
10. Dang, L.M.; Hassan, S.I.; Suhyeon, I.; kumar Sangaiah, A.; Mehmood, I.; Rho, S.; Seo, S.; Moon, H. Seo, and Hyeonjoon Moon. UAV based wilt detection system via convolutional neural networks. *Sustain. Comput. Syst.* **2018**, *28*, 100250. [CrossRef]
11. Rivas, A.; Chamoso, P.; González-Briones, A.; Corchado, J.M. Detection of cattle using drones and convolutional neural networks. *Sensors* **2018**, *18*, 2048. [CrossRef]
12. Rabah, M.; Rohan, A.; Talha, M.; Nam, K.H.; Kim, S.H. Autonomous vision-based target detection and safe landing for UAV. *Int. J. Control. Autom. Syst.* **2018**, *16*, 3013–3025. [CrossRef]
13. Rohan, A.; Rabah, M.; Asghar, F.; Talha, M.; Kim, S.H. Advanced drone battery charging system. *J. Electr. Eng. Technol.* **2019**, *14*, 1395–1405. [CrossRef]
14. Rahim, T.; Usman, M.A.; Shin, S.Y. A Survey on Contemporary Computer-Aided Tumor, Polyp, and Ulcer Detection Methods in Wireless Capsule Endoscopy Imaging. *arXiv* **2019**, arXiv:1910.00265.
15. Liu, Q.; Zeng, H.; Ni, S.; Li, B.; Meng, J.; Zhang, Y. Design of Power Grid Intelligent Patrol Operation and Maintenance System Based on Multi-Rotor UAV Systems. *Electromagn.-Non Eval.* **2020**, *45*, 54.
16. Cafiso, S.; Di Graziano, A.; Battiato, S. Evaluation of pavement surface distress using digital image collection and analysis. In Proceedings of the Seventh International Congress on Advances in Civil Engineering, Istanbul, Turkey, 10–13 October 2006; pp. 1–10.
17. Du, Y.; Pan, N.; Xu, Z.; Deng, F.; Shen, Y.; Kang, H. Pavement distress detection and classification based on YOLO network. *Int. J. Pavement Eng.* **2020**, 1–14. [CrossRef]
18. Huang, Y.; Bugao, X. Automatic inspection of pavement cracking distress. *J. Electron. Imaging* **2006**, *15*, 013017. [CrossRef]
19. Padhy, R.P.; Verma, S.; Ahmad, S.; Choudhury, S.K.; Sa, P.K. Deep neural network for autonomous uav navigation in indoor corridor environments. *Procedia Comput. Sci.* **2018**, *133*, 643–650. [CrossRef]
20. Muñoz, G.; Barrado, C.; Çetin, E.; Salami, E. Deep reinforcement learning for drone delivery. *Drones* **2019**, *3*, 72. [CrossRef]

21. Boudjit, K.; Larbes, C. Detection and implementation autonomous target tracking with a Quadrotor AR. Drone. In Proceedings of the 2015 12th International Conference on Informatics in Control, Automation and Robotics (ICINCO), Colmar, France, 21–23 July 2015; Volume 2, pp. 223–230.
22. Ding, W.; Wang, J.; Almagbile, A. Adaptive filter design for UAV navigation with GPS/INS/optic flow integration. In Proceedings of the 2010 International Conference on Electrical and Control Engineering, Wuhan, China, 25–27 June 2010; pp. 4623–4626.
23. Wang, S.; Han, Y.; Chen, J.; Zhang, Z.; Wang, G.; Du, N. A Deep-Learning-Based Sea Search and Rescue Algorithm by UAV Remote Sensing. In Proceedings of the 2018 IEEE CSAA Guidance, Navigation and Control Conference (CGNCC), Xiamen, China, 10–12 August 2018; pp. 1–5.
24. Rojas-Perez, L.O.; Martinez-Carranza, J. DeepPilot: A CNN for Autonomous Drone Racing. *Sensors* **2020**, *20*, 4524. [CrossRef]
25. Bidare, M.; Srivastav, A.; Khuu, T. CNN-based robust sidewalk identification for autonomous drone applications. Avalable online: http://cs230.stanford.edu/projects_spring_2020/reports/38903149.pdf (accessed on 8 November 2021)
26. Begum, T.; Haque, I.; Keselj, V. Deep Learning Models for Gesture-controlled Drone Operation. In Proceedings of the 2020 16th International Conference on Network and Service Management (CNSM), Izmir, Turkey, 2–6 November 2020; pp. 1–7.
27. Guo, W.; Wang, N.; Fang, H. Design of airport road surface inspection system based on machine vision and deep learning. *J. Phys. Conf. Ser.* **2021**, *1885*, 052046. [CrossRef]
28. Liu, J.; Wang, Z.; Wu, Y.; Qin, Y.; Cao, X.; Huang, Y. An Improved Faster R-CNN for UAV-Based Catenary Support Device Inspection. *Int. J. Softw. Eng. Knowl. Eng.* **2020**, *30*, 941–959. [CrossRef]
29. Mohan, S.; Shoghli, O.; Burde, A.; Tabkhi, H. Low-Power Drone-Mountable Real-Time Artificial Intelligence Framework for Road Asset Classification. *Transp. Res. Rec.* **2021**, *2675*, 39–48. [CrossRef]
30. Hassan, S.A.; Han, S.H.; Shin, S.Y. Real-time Road Cracks Detection based on Improved Deep Convolutional Neural Network. In Proceedings of the 2020 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), London, ON, Canada, 30 August–2 September 2020; pp. 1–4.
31. Gao, H.; Qin, X.; Barroso, R.J.D.; Hussain, W.; Xu, Y.; Yin, Y. Collaborative learning-based industrial IoT API recommendation for software-defined devices: The implicit knowledge discovery perspective. *IEEE Trans. Emerg. Top. Comput. Intell.* **2020**. [CrossRef]
32. Quigley, M.; Conley, K.; Gerkey, B.; Faust, J.; Foote, T.; Leibs, J.; Wheeler, R.; Ng, A.Y. ROS: An open-source Robot Operating System. *ICRA Workshop Open Source Softw.* **2009**, *3*, 5.
33. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
34. Misra, D. Mish: A self regularized non-monotonic neural activation function. *arXiv* **2019** arXiv:1908.08681.
35. Zhang, C.; Nateghinia, E.; Miranda-Moreno, L.F.; Sun, L. Pavement distress detection using convolutional neural network (CNN): A case study in Montreal, Canada. *Int. J. Transp. Sci. Technol.* **2021**. [CrossRef]
36. Ranjbar, M.; Mori, G.; Wang, Y. Optimizing complex loss functions in structured prediction. In *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 580–593.
37. Rahim, T.; Hassan, S.A.; Shin, S.Y. A deep convolutional neural network for the detection of polyps in colonoscopy images. *Biomed. Signal Process. Control* **2021**, *68*, 102654. [CrossRef]