fire

MDPI

Article

Intelligent Methods for Forest Fire Detection Using Unmanned Aerial Vehicles

Nikolay Abramov [1,*], Yulia Emelyanova [1], Vitaly Fralenko [1], Vyacheslav Khachumov [1,2,3,4], Mikhail Khachumov [1,2,3,4], Maria Shustova [1] and Alexander Talalaev [1]

[1] Ailamazyan Program Systems Institute of Russian Academy of Sciences, 152021 Pereslavl-Zalessky, Russia; yuliya.emelyanowa2015@yandex.ru (Y.E.); alarmod@pereslavl.ru (V.F.); vmh48@mail.ru (V.K.); khmike@inbox.ru (M.K.); m.v.shustova@gmail.com (M.S.); arts@arts.botik.ru (A.T.)
[2] Federal Research Center "Computer Science and Control" of the Russian Academy of Sciences, 119333 Moscow, Russia
[3] The Ministry of Education and Science, Russia RUDN University, 117198 Moscow, Russia
[4] The Ministry of Education and Science, Russia MIREA-Russian Technological University, 119454 Moscow, Russia
* Correspondence: nikolay.s.abramov@gmail.com

Abstract: This research addresses the problem of early detection of smoke and open fire on the observed territory by unmanned aerial vehicles. We solve the tasks of improving the quality of incoming video data by removing motion blur and stabilizing the video stream; detecting the horizon line in the frame; and identifying fires using semantic segmentation with Euclidean–Mahalanobis distance and the modified convolutional neural network YOLO. The proposed horizon line detection algorithm allows for cutting off unnecessary information such as cloud-covered areas in the frame by calculating local contrast, which is equivalent to the pixel informativeness indicator of the image. Proposed preprocessing methods give a delay of no more than 0.03 s due to the use of a pipeline method for data processing. Experimental results show that the horizon clipping algorithm improves fire and smoke detection accuracy by approximately 11%. The best results with the neural network were achieved with YOLO 5m, which yielded an F1 score of 76.75% combined with a processing speed of 45 frames per second. The obtained results differ from existing analogs by utilizing a comprehensive approach to early fire detection, which includes image enhancement and alternative real-time video processing methods.

Keywords: deep learning; remote sensing; forest fires; fire monitoring; fire detection; semantic segmentation; unmanned aerial vehicle

Citation: Abramov, N.; Emelyanova, Y.; Fralenko, V.; Khachumov, V.; Khachumov, M.; Shustova, M.; Talalaev, A. Intelligent Methods for Forest Fire Detection Using Unmanned Aerial Vehicles. *Fire* **2024**, *7*, 89. https://doi.org/10.3390/fire7030089

Academic Editors: Demin Gao, Shuo Zhang and Cheng He

Received: 5 February 2024
Revised: 1 March 2024
Accepted: 10 March 2024
Published: 15 March 2024

1. Introduction

The problem of detecting forest fires using images and video streams from various technical vision systems is extremely relevant, considering the negative impact of fires on the socio-ecological-economic spheres observed in areas. Currently, fire detection systems in limited forest areas and national parks are built based on the installation of stationary video cameras. Due to limited time resources, unmanned aerial vehicles (UAVs) serve as auxiliary tools, but their usefulness can be substantial. For example, they can survey areas that are inaccessible to stationary cameras and facilitate the delivery of small rescue equipment to individuals in fire-affected areas. Moreover, the presence of professional UAVs equipped with firefighting capabilities [1,2] allows for early fire suppression using limited resources. Solving the problem of early fire detection allows for reducing the undesirable consequences of fires and requires the development of new autonomous methods to analyze photo and video streams from stationary and mobile cameras, installed, for example, on unmanned aerial vehicles.

The main requirement for early and accurate detection of anomalies is to obtain high-quality images, which is generally achieved through the use of special algorithms for processing the original video information.

An analytical review of the current scientific literature was carried out to identify significant work in this direction, which allowed singling out trends and promising architectures of artificial neural networks (ANNs), including convolutional neural networks (CNNs). The proposed approach involves intelligently processing the video information from the observed area with computer vision technologies. We divide the early fire detection problem into three sub-tasks. The first two of them relate to image preprocessing methods.

1. In the first stage, the overall quality of the video stream from the UAV's technical vision system is improved, including removing motion blur in frames and stabilizing the video stream. This stage helps minimize pictures' artifacts on video frames and potentially improves the quality of detection in regions of interest (fire and smoke).
2. In the second stage, the horizon line is detected, which allows areas with clouds to be cut off as they can be incorrectly recognized as smoke. his improves the quality of detection too.
3. In the third stage, for the detection of fires and smoke, we use two approaches: using the computer vision with the YOLO («You Only Look Once») [3] detection method and Euclidean–Mahalanobis distance [4].

This approach to the problem allows for optimal solutions to be selected at each stage, using both analytical capabilities and experimental research.

The rest of the paper is organized as follows:

- Section 2 contains a review of previous methods and approaches for the started tasks.
- All proposed methods are described in Section 3.
- The results of all the described methods are shown in Section 4.
- We discuss the limitations of the proposed algorithms and describe future work in Section 5.
- The conclusion and general results of this research are given in Section 6.

2. Related Works

Forest fires are a common occurrence, caused by lightning, human carelessness, and the impact of extreme heat and drought on fuel. Due to significant damage to natural and human resources, early warning schemes are necessary, as the initial stage is the best time to extinguish a fire. Let us consider relevant works that explore methods and approaches that can be used, including in the processing of images from UAVs. We will focus on the existing methods for the forest fire detection task as the primary focus, while considering the video stream improving methods as supporting approaches.

In [5], the authors propose a multi-stream hybrid deep learning model with a dual-stream attention mechanism for classifying wildfires from aerial and territorial images. The model is based on a pre-trained EfficientNetB7 and a customized Attention Connected Network with Bayesian optimization. The proposed model attains 97.45%, 98.20%, 97.10%, and 97.12% as accuracy, precision, recall, and F1 score, respectively, on the FLAME dataset.

In [6], a concept of fire detection is proposed for identifying incidents, tracking, and preventing forest fires on-site. The concept is based on data collection using sensors and network connectivity through cloud services on the Internet of Things (IoT). A model is proposed for predicting and monitoring forest fires during natural disasters, tracking the threshold value, and informing authorized personnel. Another work with a similar approach is described in [7]. A wireless sensor network was employed for forest data acquisition to identify abrupt anomalies when a fire ignition starts. We believe such sensor information could be used to optimize UAV flight routes in forest fire remote sensing.

A novel hierarchical domain-adaptive learning framework designed to enhance wildfire detection capabilities, addressing the limitations inherent in traditional convolutional neural networks across varied forest environments is described in [8]. The framework inno-

vatively employs a dual-dataset approach, integrating both non-forest and forest-specific datasets to train a model adept at handling diverse wildfire scenarios.

In [9], an automated early warning system for forest fires is described. The system utilizes an unmanned aerial vehicle equipped with multiple sensors and a deep learning algorithm. The sensors and algorithm process real-time images using the proposed deep learning model, the YOLO neural network. The system provides accurate real-time location detection of forest fires, allowing for tracking and reporting to relevant authorities.

Another algorithm based on the YOLO-V3 neural network for detecting forest fires using data collected from UAVs is described in [10]. The YOLO-V3 model supports multifunctional fire detection, including the real-time examination of small flame objects during feature extraction. Compared to alternatives, this model improves the accuracy of flame recognition in video images and reduces the frequency of errors.

An extended version of the YOLO-V7 model for forest fire smoke detection is proposed in the work [11]. The authors included the Convolutional Block Attention Module in YOLO-V7. To detect the initial stages of wildfires in the highway network, a component is added that extracts features of fine-grained details in aerial photographs using convolutional kernels of various sizes. A bidirectional feature pyramid network is used for feature fusion and obtaining more specific features, enhancing sensitivity to small fires in terms of area. The testing results of the system showed an average accuracy of 91.5%.

An efficient algorithm for quickly localizing fires based on low-level visual features is presented in [12]. It uses a multi-scale convolutional neural network architecture, the convolutional neural network, based on SPP (spatial pyramid pooling), which is trained on a dedicated dataset without the need for sample labeling. The algorithm consists of two stages: rapid determination of candidate fire regions and accurate identification of fire with different aspect ratios and scales. Experimental results show that the proposed approach provides accurate flame identification at different scales; the precision estimate (F-measure) given by the authors is 93.33%.

A new approach to detecting forest fires is explored in [13], using an artificial intelligence platform. Computer vision methods are employed for smoke and fire recognition and detection based on motionless images or video signals from cameras. Two classification algorithms were utilized in this study, with the first being the K-nearest neighbors algorithm, yielding 98.4% accuracy. Additionally, a random forest algorithm was implemented to improve accuracy, achieving 99.6% accuracy. Image processing methods are then applied to predict fire and smoke in the frames of the video during playback.

The approach proposed in the work [14] has several potential applications, including forest fire monitoring. The algorithm is built based on the YOLO-V8 network model. The proposed structure consists of four levels: (1) Smart fire detection system, (2) fog master node, (3) cloud layer, and (4) IoT layer. The method achieved an accuracy of 97.1%.

Article [15] provides a general analysis of mobile and immobile monitoring and surveillance systems for detecting forest fires and smoke. Two main approaches are identified: One based on machine learning and the other on extracting color characteristics from images or video frames. It is claimed that a combination of both methods is used to detect smoke and fire or smoke alone, guaranteeing an accuracy of more than 90%.

The method of ensemble convolutional neural networks (ECNN) is known. For instance, in the work [16], it is applied to eliminate the need for dimensionality reduction of the raw spectrum, along with other pre-processing operations. The authors delved into the internal learning mechanism of the deep CNN model to understand how it hierarchically extracts spectral information features. The experimental results demonstrate that the easy-to-use ECNN-based regression model achieves excellent prediction performance while maintaining interpretability.

The authors of the study [17] describe the basics of image processing methods, paying special attention to convolutional neural networks and their potential applications in the task of fire detection in images. They discuss existing datasets for training CNNs to detect fires and smoke. In the study [18], an evaluation was carried out on various combinations of

modern deep learning architectures, loss functions, and types of processed images, with the aim of determining the parameters that have the greatest impact on the results obtained in the task of semantic segmentation of images. The study [19] attempts to trace the evolution of approaches to solving the problem of smoke detection using machine learning methods based on characteristic features of motion, translucency, color, shape, texture, and fractal characteristics. They propose a CNN architecture with LSTM network elements that shows high accuracy rates.

In [20], a CNN with the CSPdarknet53 architecture (based on YOLO) with an SPP layer is used to detect smoke. The proposed approach achieves a quality assessment of 97.9% (F-measure), with a data processing speed of 32 frames/s, which corresponds to real-time detection requirements. A real-time algorithm for fire detection based on the YOLO 5 s architecture CNN is described in [21]. The research showed that the model can automatically identify and detect flames and smoke at different stages and in different forms. The mAP@.5 (mean average precision) accuracy estimate given by the research authors indicates successful fire and smoke detection in 80.6% of cases, and the data processing speed is 31 frames/s.

The study [22] proposes a fire detection method based on a CNN with MobileNetV3 architecture. The described method shows 90.2% accuracy on the dataset created by the authors, while the model size is only 21.4 MB and the processing speed can reach 29.5 frames/s. An approach to fire recognition in a frame using deep learning with MobileNetV3 is proposed in [23]. Experimental results show that the number of parameters for the proposed CNN architecture is only 2.64 million: As compared to YOLO 4, this is a reduction of almost 95.87%. The approach shows the advantages of a model with fewer parameters, low memory requirements, and high output speed compared to existing algorithms. It was specifically designed for use as a monitoring system on a UAV platform. In [24], it is proposed to replace the YOLO 4 base network with the MobileNetV3 network. The experimental results show that the number of parameters in the proposed architecture decreases by 62.78% and the output speed increases by 3.04 times as compared to YOLO 4. At the same time, the accuracy of detection for the proposed algorithm reaches 66.6% on the mAP@.5 scale. The main value of the presented method is the possibility of implementing a mechanism for detecting wildfires and objects in real time on devices with low computing resources.

Next, Let us consider several methods for increasing the quality of video streams. Various preprocessing methods are widely used to reliably detect and recognize objects in video frames, including discarding non-informative frames in a series and enhancing the quality of the original images. Such methods are discussed, for instance, in [25]. The author of the study divides the informativeness evaluation methods into three categories: applying hash functions to reduced copies of images and comparing them, computing correlation, and comparing images using descriptors.

Attention has been given in UAV-based fire research to improving image processing accuracy by eliminating vibrations and stabilization. In [26], a method based on analyzing sequential image frames captured simultaneously by optical and infrared cameras from different perspectives is proposed. This approach reduces noise levels and eliminates camera displacement effects that occur during vehicle movement.

The thesis [27] presents evaluations of image quality and proposes a method for selecting the best frame from a video stream based on priority calculation using a Gaussian kernel. Well-known quality evaluations, such as PSNR, SSIM, and others, are widely used in image quality analysis [28]. Approaches to image quality assessments can be divided into two categories: Those that use complete information, meaning that the "reference" (high-quality) image is available, and those that rely only on the analyzed frame. Examples of methods belonging to the latter category include BRISQUE, which uses normalized brightness values in a window [29], GMLOGQA, which uses a gradient modulus map approach [30], and ILNIQE, which is built on the gradient and the Weibull distribution [31].

A method is known for "summing" multiple (N) frames of a video stream, which amplifies the useful signal by $\sqrt{N}$ times, while the noise is amplified only by N times [32]. Experiments have shown that the PSNR of the recovered image is higher than in cases where you use the standard approaches presented in [33]. Another approach is based on a modified MSR algorithm and involves nonlinear image enhancement while preserving local contrast in poorly and brightly lit areas, which improves the quality of the observed scene in the evening and under poor weather conditions [34].

Next, let us consider several methods for detecting the horizon line in the frame. In the work [35], the pixels located at the peak of the brightness gradient from light to dark in the vertical direction are recommended to be selected as a potential horizon line. In the study [36], DenseNet neural networks and dynamic programming methods are successfully used. Another example of using machine learning principles is [37], where automatic segmentation based on the DeepLabV3+ neural network and the conditional random fields method is applied. In the work [38], linear filters and local analysis of structures in the image are used. The method of separating the water surface and the sky in images is presented in [39], where the Gaussian filter is applied, and the areas potentially belonging to the horizon line are the areas containing the maximum value gradients of the integral image.

In summary, considering the aforementioned works, the task of forest fire detection and related tasks for improving the video stream are currently of utmost relevance and importance. The majority of the approaches heavily rely on the use of neural networks. These methods are continuously evolving, and given the socio-economic impact of early fire detection, there is a need for further research in this field. In the following sections, we will delve into the proposed methods to address the stated challenges.

3. Materials and Methods

All proposed methods are described in this section. Main task of forest fire detection is divided to several interrelated tasks.

- We propose improving the quality of the video stream method: Removing blur and video stream stabilization. This procedure is necessary in cases where, during sudden movements of the UAV caused by wind gusts, the image becomes unclear, which prevents good fire and smoke recognition. To solve this problem, various means of removing grease are used.
- Another approach that significantly affects the quality of recognition of fires is associated with removing part of the image from video frames, covering the upper part with the sky. This is an important step since most of the recognition errors are associated with weather conditions, such as when clouds appear in the sky adjacent to the forest area, as well as reflections during sunsets and sunrises, which leads to false positives of the recognition algorithm.
- Finally, we solve the problem of detecting forest fires with the modified CNN «YOLO». Fine-tuning of this network is provided through the selection of layers and neurons.

3.1. Deconvolution and Video Stream Stabilization

Real images from UAVs contain apparent motion blurs caused by the sharp camera displacement due to operator control inputs, wind loads, and the operation of orientation systems on the ground, including GLONASS and GPS. All these factors can occur simultaneously, which destabilizes the camera's position in space. In addition, objects such as cars, bicycles, pedestrians, and animals can also move during the capture of the same frame. Therefore, the camera may inadequately capture an object in the frame due to (1) rapid camera or object movement, or (2) suboptimal camera settings (for example, an excessively long exposure time, or a very slow or overheated matrix). Deconvolution is a method of compensating for these factors that cause video stream frames to become blurred. We propose to use ANN-based deconvolution for video stream improvement.

During video stream stabilization, the frame is cropped on all sides, and the remaining parts of the frame are used to compensate for motion and generate the final image. This compensation is usually performed using ANN and statistical methods based on previous frames, which requires maintaining a buffer of a certain size.

3.2. Image Semantic Segmentation

An algorithm has been developed for extracting regions of interest from full-color images that correspond to categories previously annotated by an expert in the training data of the knowledge base ("open fire", "smoke", "fire with smoke", "normal"). The algorithm is based on a scanning window method with pixel-wise shifting, and each position is processed using Euclidean–Mahalanobis distance. The generalized Euclidean–Mahalanobis metric was developed and investigated with the participation of some authors of this article [40]. It has shown its effectiveness in processing multispectral satellite imagery. In this case, it is reliable and works in all situations when the regions of interest are homogeneous in terms of spectral characteristics, where the Mahalanobis metric fails (the inverse correlation matrix underlying the metric becomes incomputable). Since the Euclidean–Mahalanobis metric is versatile, it is suitable for processing color intensity images, including fire detection.

For full-color images, statistical components of captured pixels are extracted from each position of the scanning window, as well as values of the Lavas features. The result is an image with regions of interest highlighted using color markers: Each pixel of the resulting image is colored based on the classification result of the individual scanning window position. "Normal" corresponds to green, "fire" to red, "fire with smoke" to orange, and "smoke" to yellow. The program can be used for the prompt detection of forest fire foci using Earth remote sensing data.

3.3. Horizon Line Detection

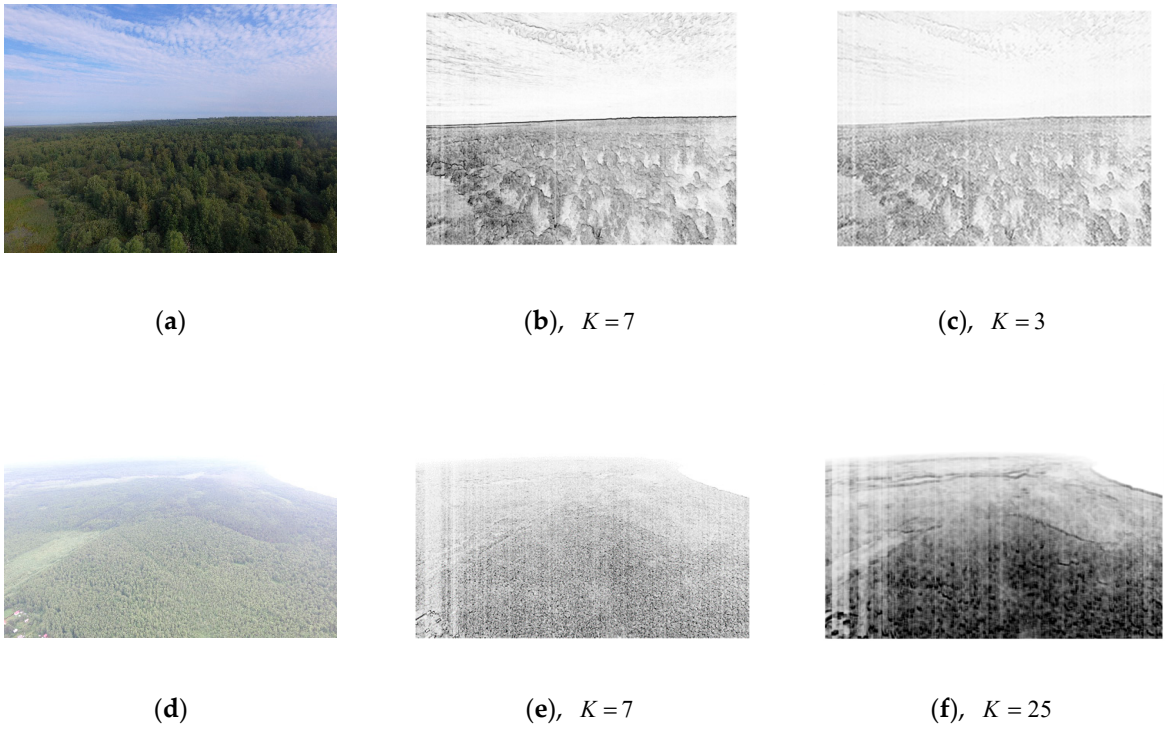
It follows from the analysis of sources (see «Related work», Section 2) that the existing methods for detecting the horizon line are adapted to specific situations. We propose an algorithm for extracting the horizon line from UAV images that allows for transforming the original information by removing the part of the image containing the sky, which interferes with the detection of smoke and fire. The algorithm is based on the principles of edge detection, defining informative image features, and building a multiple linear regression model. We will calculate the local contrast as a quantitative assessment of image informativeness in a sliding window [41]:

$$\partial[X] = \left| \frac{1}{K^2} \sum_{i=1}^{K^2} X_i^2 - \left(\frac{1}{K^2} \sum_{i=2}^{K^2} X_i \right)^2 \right|^{1/2}, \quad (1)$$

where K is the size of the square sliding window, X_i is the brightness value of the i-th pixel in the sliding window $K \times K$.

The higher the value of $\partial[X]$, the more informative the image fragment. Thus, if we place the center of the sliding window on each pixel of the image, we evaluate its informativeness within the window [42], including the points of the horizon line. The root mean square deviation (RMSD) of the brightness of neighboring pixels determines local contrast [43].

The size of the square sliding window K has a key effect on the results of solving a number of computer vision problems [44]. Therefore, the choice of optimal K significantly affects the quality of the problem solution. The results of processing some landscape images with different K are presented in Figure 1.



(**a**) (**b**), $K = 7$ (**c**), $K = 3$

(**d**) (**e**), $K = 7$ (**f**), $K = 25$

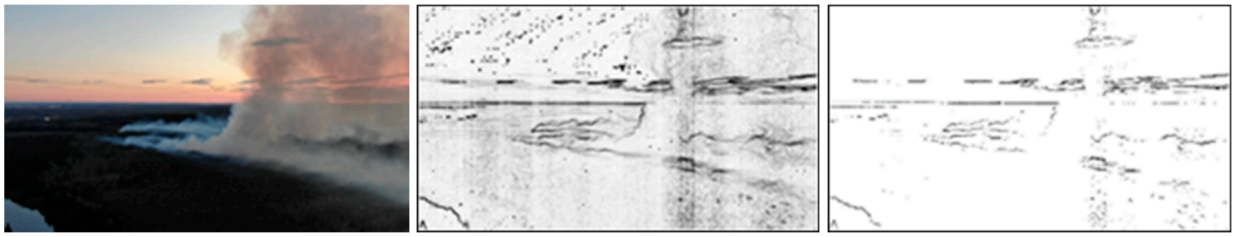
Figure 1. Initial images and results of assessing information content for different sizes of the scanning window (K): (**a**) An image with a clear horizon line; (**b**) the result of processing image (**a**) with a scanning window of size K = 7; (**c**) the result of processing image (**a**) with a scanning window of size K = 3; (**d**) an image with a blurry horizon line; (**e**) the result of processing image (**b**) with a scanning window of size K = 7; (**f**) the result of processing image (**b**) with a scanning window of size K = 25. Each pixel is assigned a brightness RMSD value $\partial[X]$ in the area of which it is the center. Next, a minimax normalization of the values $\partial[X]$ in each column of image pixels is performed. Normalized values are presented in (**b**,**c**,**e**,**f**). The maximum values of the standard deviation in the column correspond to the black color while the minimal ones correspond to the white color. A clearer horizon line will be associated with a higher RMSD and greater sharpness (**b**,**c**). (**a**–**c**) show that with a high contrast between the sky and the earth's surface, the horizon line can be calculated using a window of medium (K = 7) or small size (K = 3). On the contrary, a smooth transition from the sky to the Earth's surface is associated with less deep and more blurred grooves (**e**,**f**). From (**d**–**f**) it can be seen that the average window size is not enough to detect a weak horizon line (**e**). In this case, a large window such as 25 × 25 is required (**f**).

Let us set a certain threshold value of informativeness t such that $\partial[X] = 0$ when $\partial[X] < t$. Filtering out such low-informative points improves the quality and speed of solving image analysis tasks.

We carry out screening of points whose informativity is below the threshold value t. We will assume that the remaining points are candidates for belonging to the horizon line (see Figure 2). A finding from the horizon line algorithm will be presented below.

Figure 2 shows that a contour image is generated after filtering out the low-information pixels (Figure 2c). Several of the resulting contours are fragments of the horizon line. The problem is solved by combining fragments of the horizon line and restoring gaps between them.

Below is an algorithm for finding and merging horizon line points into a single set. A detailed description of the algorithm steps is presented in [45].



(**a**) (**b**), $K = 7$ (**c**), $t = 89.39$

Figure 2. Result of filtering out low informative values of RMSD: (**a**) initial image, (**b**) result of evaluation of information content of image fragments, (**c**) contour image.

The algorithm can be described as follows:

1. Calculation of factors $f_1, \ldots, f_{15}$ that affect the size of the sliding window K and the threshold of informativeness t for filtering out non-informative points. Factors characterizing the main features of displaying the horizon line on a photo are calculated, and models (equations) of multiple linear regression are constructed:

$$t = a_0 + a_1 N + a_2 f_2 + \cdots + a_{17} f_{17}, \quad (2)$$

$$N = d_0 + d_1 f_2 + d_2 f_3 + \cdots + d_{14} f_{15}, \quad (3)$$

where $a_0, a_1, \ldots, a_{17}$ and $d_0, d_1, \ldots, d_{14}$ are parameters of the regression model, $f_1, \ldots, f_{17}$ are factors (explanatory variables, features), and $f_1 = N$.

The N parameter obtained from Equation (3) is related to the size K by the ratio $K = 2N + 1$.

2. Normalization of the image brightness in grayscale.
3. Calculation and normalization of informative areas of the image.
4. Filtering informative pixels of the image based on the brightness of the areas above and below.
5. Calculation of factor values f_{16}, f_{17}, regression model coefficients a_{16}, a_{17} and threshold t using Equation (2).
6. Filtering out points with an informativeness level below the threshold t.
7. Grouping points into lines.

Denote the line as the set L_{yx}, for which (y, x) are the coordinates of the first element.

(7.1) Assigning a non-zero informative pixel $I_{yx} \neq 0$ to the set L_{yx}.

(7.2) Search for the pixel with the maximum information value I_{ij} in the column with the number $j = x + 1$ such that $i \in [y - (\alpha - N);\ y + (\alpha - N)]$. Further in experiments we take $\alpha = 10$. When working in a mountainous area we assume that $\alpha = 80$.

(7.3) If $I_{ij} \neq 0$, assign element I_{ij} to set L_{yx}; make a replacement: $x = j,\ y = i$; go to step 7.2. Otherwise, we search for the next pixel with non-zero information value, such that $x > j$, and go to step 7.1. If $j = J$, where J is the size of the image width in pixels, then go to step 8.

Examples of the resulting lines are indicated in green in Figure 3c–e.

8. Grouping lines.

Let us denote a group of lines as a set Z_{yx}, for which (y, x) are the coordinates of the first pixel in the first line, and (y_{last}, x_{last}) are the coordinates of the last pixel in the last line. Each line L_{yx}, represented by a set of its points, can belong to only one Z_{yx}, i.e., for any points (y_1, x_1) and (y_2, x_2) the condition $Z_{y_1 x_1} \cap Z_{y_2 x_2} = \varnothing$ is satisfied.

(8.1) Include in Z_{yx} the points of the first line L^k_{yx}, $k = 1$, i.e., $L^k_{yx} \subset Z_{yx}$.

(8.2) Search for a line L^{k+1}_{yx} with the maximum information value of the first point I_{ij}, $j = x_{last} + 1$, such that $i \in [y - \varepsilon;\ y + \varepsilon]$, where $\varepsilon = (\alpha - N)(j - x_{last})$.

(8.3) If $I_{ij} \neq 0$, then $L^{k+1}_{yx} \subset Z_{yx}$, perform the replacement $k = k + 1$ and go to step 8.2. Otherwise, go to step 8.1. If $j = J$, then go to step 9.

9. Selection of the line with the greatest length. The selected line is assigned as the "central part" of the horizon line. In Figure 3f–h, fragments of the "central part" are indicated in turquoise.
10. Restoration of gaps between the segments of the "central part" of the horizon line.

We introduce the following notations: $\left(y_f, x_f\right)$ are the coordinates of the beginning of the skip, (y_l, x_l) are the coordinates of the end of the skip, γ is the angle of the horizon line at the skip section.

(10.1) Search for a point with the maximum information value I_{ij} in a column with number $j = x_f + 1$, such that $i \in \left[y_f - \beta;\ y_f + \beta\right]$, where $\beta = \alpha - N$.

(10.2) If $I_{ij} \neq 0$, then perform the replacement $x_f = j$. Otherwise, calculate the coordinate i using the formula: $i = y_f + \left(j - x_f\right)\sin\gamma$, perform the replacement $x_f = j$. If $j = J$, then go to step 11. Otherwise, go to step 10.1.

11. Calculation of the "right part" of the horizon line.

We introduce the following notations: (y_{start}, x_{start}) are the first point's coordinates of the "central part" of the horizon line, (y_{end}, x_{end}) are the coordinates of the last one, (y_{last}, x_{last}) are the last point's coordinates of the "right part" of the horizon line, $\sin\varphi = (y_{end} - y_{start})/(x_{end} - x_{start})$.

(11.1) We take $(y_{last}, x_{last}) = (y_{end}, x_{end})$.

(11.2) Find a point with the maximum information value I_{ij} in the column numbered $j = x_{last} + 1$ such that $i \in [y_{last} - \beta;\ y_{last} + \beta]$.

(11.3) We assign the last point to a pixel with coordinates obtained by the formula:

$$(y_{last}, x_{last}) = \begin{cases} (i, j),\ if\ I_{ij} \neq 0 \\ (y_{start} + (j - x_{start})sin\varphi, j), otherwise \end{cases} \quad (4)$$

If $j = J$, then go to step 12, otherwise, go to step 11.2.

12. Calculation of the "Calculation of the "right part" of the horizon line.

Let us introduce the following notation: $\left(y_{first}, x_{first}\right)$ are the first point's coordinates of the "left part" of the horizon line.

(12.1) We take $\left(y_{first}, x_{first}\right) = (y_{start}, x_{start})$.

(12.2) Find a point with the maximum information value I_{ij} in the column numbered $j = x_{first} - 1$ such that $i \in \left[y_{first} - \beta;\ y_{first} + \beta\right]$. We assign the first point to a pixel with coordinates obtained by the formula:

$$(y_{first}, x_{first}) = \begin{cases} (i, j),\ if\ I_{ij} \neq 0 \\ (y_{start} + (j - x_{start})sin\varphi, j),\ otherwise \end{cases} \quad (5)$$

If $j = 0$, then go to step 13. Otherwise, go to step 12.2.

13. Checking the horizon line for truthfulness. The mean brightness values in grayscale above and below the horizon line are calculated and compared. Let l be the number of points on the horizon line, (y_i, x_i) be the coordinates of the i-th point, and $q(y_i, x_i)$ be the brightness in grayscale of the i-th point on the horizon line. If the condition

$$\frac{1}{l}\sum_{i=1}^{l} q_{(y_i - N, x_i)} > \frac{1}{l}\sum_{i=1}^{l} q_{(y_i + N, x_i)} \quad (6)$$

is met, where N is obtained from formula (3), the horizon line is considered calculated correctly and the algorithm proceeds to step 14. Otherwise (see Figure 3j), the size of the sliding window is increased: $N := N + 2$; the informativeness threshold is reduced: $t := \rho - 2(T - 1)$, where $\rho = 15$ is an empirical constant, T is the iteration number; the algorithm proceeds to step 3.

14. Smoothing of the "left part" of the horizon line.
15. Smoothing of the "right part" of the horizon line.
16. Displaying the horizon line on the image (see Figure 3i,k).
17. End.

The main stages of computing the horizon line under unfavorable conditions (partially obscured horizon line due to smoke and blurry horizon lines) are shown in Figure 3.

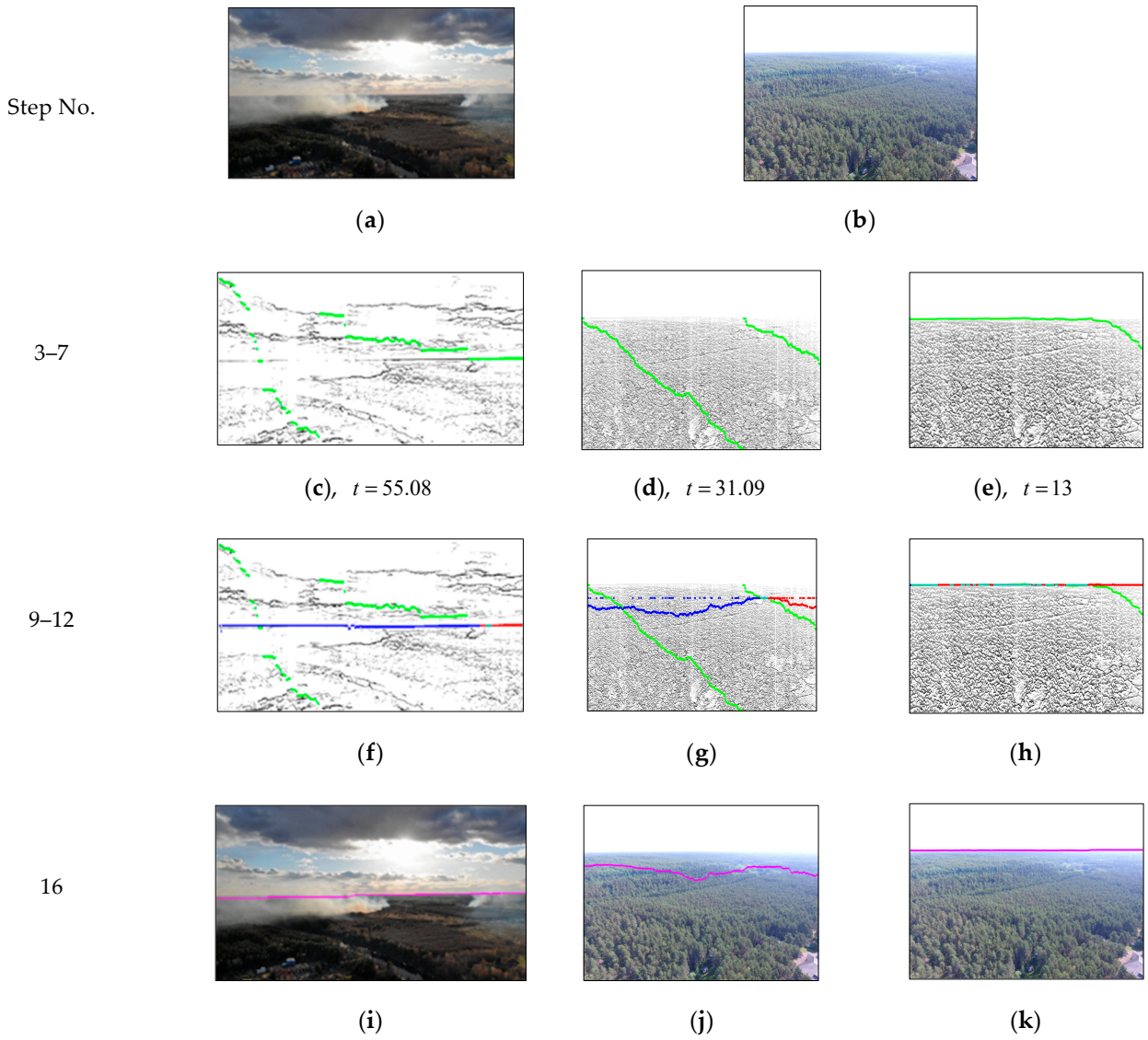


Figure 3. Stages of computing the horizon line on terrain photographs: (**a**) The original image with a partially obscured horizon line due to smoke; (**b**) the original image with a blurry horizon line; (**c**) the result of processing image (**a**) and the detected potential segments of the horizon line with an informativeness threshold of t = 55.08; (**d**) the result of processing image (**b**) and the detected potential segments of the horizon line with an informativeness threshold of t = 31.09; (**e**) the result of processing image (**b**) and the detected potential segments of the horizon line with an informativeness threshold of t = 13.00; (**f**) the computed horizon line (intermediate result) for image (**a**,**g**,**h**). The computed horizon lines (intermediate results) for image (**b**,**i**). The result of computing the horizon line for image (**a**,**j**,**k**). The results of computing the horizon lines for image (**b**).

The algorithm flowchart is shown in Figure 4.

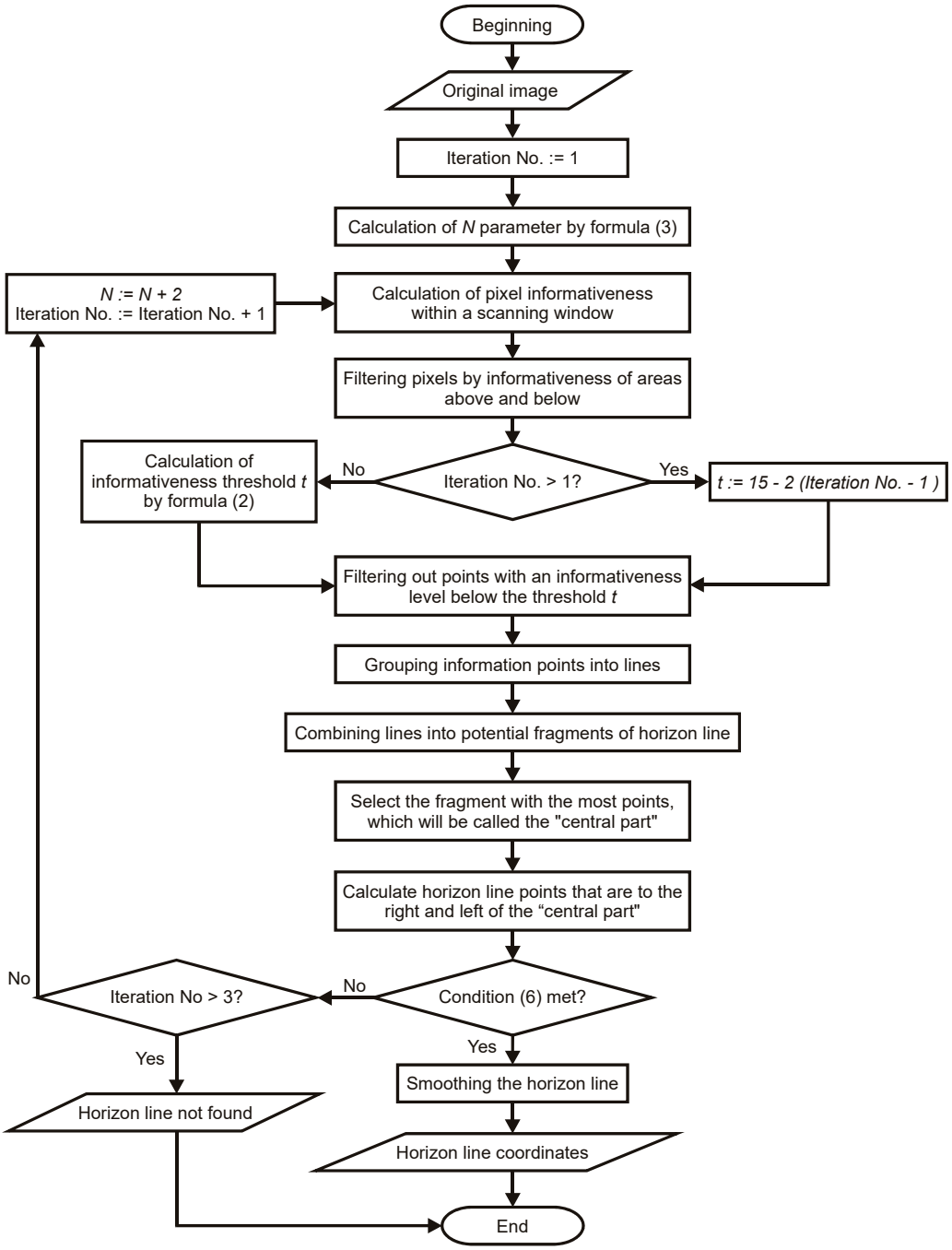


Figure 4. Horizon line detection algorithm flowchart.

3.4. Detection of Forest Fires with Convolutional Neural Networks

For the detection of forest fires, we propose approach based on various YOLO-architectures for detecting rectangular areas and accurately segmenting fire sources using CNNs, while ignoring the area above the horizon line. YOLO architecture has better machine learning robustness properties and object detection generalization capabilities than the larger transformer-based detection architectures, making it perfect for work in different weather conditions [46].

The used activation function sigmoid-weighted linear unit (SiLU) resolves the issues of gradient decay and avoids the overfitting effect, which is found in the sigmoid function. This allows the model to train more efficiently and converge to the optimal solution faster. Initially, YOLO 5 used the Leaky ReLU activation function in the middle/hidden layers,

but now it uses SiLU as the default function. These changes were made by the YOLO 5 developers directly during the course of these research studies, supporting the conclusions made. Each detected fire source is described by the coordinates of a rectangle, which is generally rotated at an arbitrary angle at which the source can be placed. Each pixel is evaluated during the semantic segmentation of frames, and a decision is made as to whether it belongs to the area with fire. The training process is carried out once, and then the video stream is analyzed frame by frame using trained CNNs, which perform target area detection in the form of a rectangle and/or segmentation.

4. Results

In this section, we will describe the main results of the proposed methods. We used the following hardware and software for experiments: Intel Core i7-6850k processor (6 cores/12 threads, 4 GHz), 32 GB RAM, NVidia GeForce GTX 1080 Ti graphics card (11 GB video memory); CUDA 11.8 and PyTorch 2.2.1.

4.1. Results of Deconvolution and Video Stream Stabilization

Snapshots sized 1280 × 720 pixels obtained from a mobile camera (GoPro dataset [47]) that simulates the operation of UAV technical vision systems with a rate of 24 frames per second were used for experimental data in this study.

Table 1 below lists algorithmic and software solutions designed for the automated removal of blurs [48]. The best solutions were selected based on the speed and quality of the resulting images. Information is provided on the average processing time for individual frames. It should be noted that all current solutions are implemented as software code using graphics processors for calculations (full hardware configuration was described above). This study addressed the subtask of restoring areas of a video frame with a car number visible. UAVs are actively used worldwide for automated inspection and registration of car numbers, making this subtask relevant. In this case, images with car numbers. The metrics for comparing PSNR and SSIM image pairs for the damaged original blurred images of car numbers, respectively, are as follows: 18.42 and 0.57 (left frame); 18.79 and 0.56 (right frame).

Table 1. Results of the comparison of deconvolution methods.

Method	Processing Time, s	Fragment 1		Fragment 2	
		PSNR	SSIM	PSNR	SSIM
Test-time Local Converter (TLC) [49]	13.5	33.012	0.939	31.186	0.929
Multi-Axis MLP [50]	19.5	30.556	0.917	31.748	0.934
EFNet [51]	1.10	**35.184** [1]	**0.948** [1]	**35.695** [1]	**0.962** [1]
Learning degradation [52]	3.60	32.549	0.932	31.739	0.934
Deep Generalized Unfolding [53]	5.80	22.582	0.735	29.923	0.903
NAFNet width32 [54]	5.00	31.789	0.918	30.000	0.916
Stripformer [55]	**0.03** [2]	28.060	0.876	30.222	0.916
Uformer [56]	9.40	32.296	0.928	31.477	0.929
STDAN [57]	360	31.719	0.926	30.168	0.913
MMP-RNN [58]	108	26.668	0.844	30.413	0.910

[1] The best result in PSNR and SSIM. [2] The best result in processing time.

Experimental studies conducted using these and other images from the GoPro dataset have shown that the Stripformer neural network performs real-time processing while

producing high-quality output images. Stripformer has an architecture based on the idea of transformers. The "visual transformer" consists of a tokenizer, transformer, and projector, and a more detailed description of its operating principles can be found in the study [59]. Stripformer constructs internal and external strip tokens during the recalculation of informative image features in horizontal and vertical directions, allowing it to identify noisy fragments with different orientations (direction of blur propagation). The neural network combines external and internal strip attention layers to determine the degree and character of a blur. In addition to detecting specific blurred patterns of different orientations and sizes, Stripformer works successfully even without data on the dynamics of the scene and does not require a huge amount of training data. It is worth noting the high quality of the EFNet neural network, which, however, cannot be used in real-time mode. Neural networks that use multiple frames for restoration showed the longest processing time. Thus, the proposed solution allows for obtaining data from UAVs without motion blur in real time.

Now, we can proceed to the next step of improving the video stream. Table 2 lists the algorithmic and software solutions intended for automated UAV video stream stabilization. We use test sets of 100 frames (1280 × 720) restored using Stripformer, and fed at a rate of 24 frames per second. The average processing time for the entire video file is provided. Each solution is accompanied by a comment reflecting the acquired experience of its use. Some solutions are implemented with the support of graphics processing units (see full spec above at the beginning of this section).

Table 2. Results of the comparison of video stream stabilization methods.

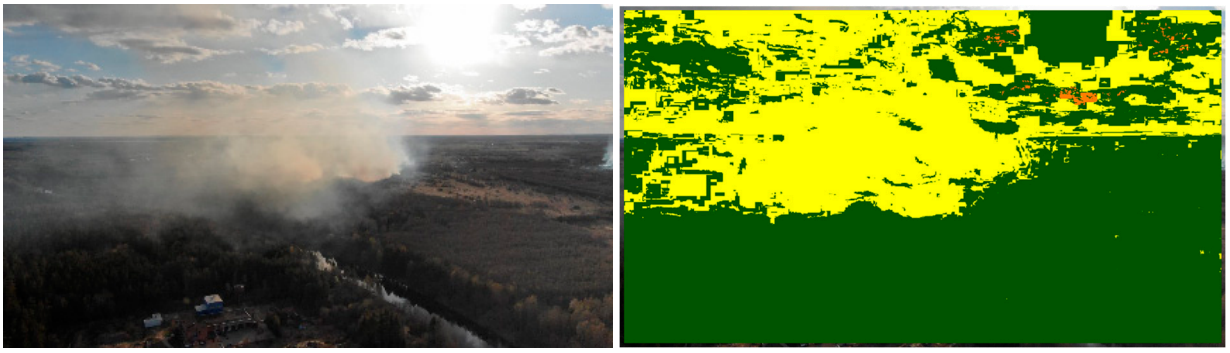
Method	Processing Time, s	Note
VidGear [60]	3.91	The frame borders are cropped using a border offset and scaling. Initialization of the stabilizer is required, with input data of about 24 frames used. The time required to prepare the stabilizer for operation is approximately 0.70 s. [1]
FuSta: Hybrid Neural Fusion [61]	1200	A graphics accelerator is used. The resulting video from these final frames is smooth, but there are artifacts at the boundaries of the "car" and "tree" objects.
MeshFlow: Minimum Latency Online Video Stabilization [62]	1260	Experiments were conducted with two settings profiles. The Original profile provides excellent quality without borders, and the frame is automatically cropped. The Constant High profile has artifacts in the form of unnaturally tilted objects, such as cars.
Video Stabilization with L1 optimal camera paths [63]	0.50	The resulting video is "wavy". The camera sometimes moves towards objects in the frame and then back.
Auto-Directed Video Stabilization with Robust L1 Optimal Camera Paths [64]	120	The resulting image is significantly cropped when high levels of stabilization are used. For example, at a threshold of 40 pixels, an image of 1280 × 720 is cropped at the edges to 1180 × 620. Using a lower threshold gives an insufficient level of video stabilization. Processing is divided into two stages: 2 min for preprocessing, followed by fast stabilization using the obtained data.
Video stabilization using homography transform [65]	4.18	The resulting image is significantly cropped, especially at high stabilization coefficients. The best results are obtained with real-time stabilization when the parameter is set to 20.
DUT: Learning Video Stabilization [66]	32.0	A graphics accelerator is used. The method relies on several pre-trained neural networks. The approach is based on self-learning on unstable video data. The car image in the resulting video is slightly offset left and right, but the overall resulting video can be considered of good quality.
Deep Iterative Frame Interpolation for Full-frame Video Stabilization [67]	100	A graphics accelerator is used. Processing is performed iteratively, with the number of iterations controlled by the n_iter parameter, which is set to 3 by default. The resulting video stream is unstable, and the overall quality of the resulting video stream is low.
PWStableNet [68]	8.00	A graphics accelerator is used. The resulting video has jerky movements.

[1] The best result.

The best stabilization results were achieved using the VidGear library, which accumulates an array of key points over a specified number of frames with subsequent real-time updating. The accumulated data are used to estimate the motion of the UAV camera with compensation for its jerks. The implementation corresponds to the approach used in the OpenCV library [69]. The overall processing occurs in a pipeline mode, as deconvolution and stabilization are performed in parallel.

4.2. Results of Image Semantic Segmentation

The proposed feature vector contains 147 texture and 35 statistical characteristics. The texture characteristics are computed by convolving the three RGB channels of the image with 49 Laws' energy masks [70]. To compute the statistical features, the image is considered a multidimensional signal $s(R, G, B, Gray, H, S, L)$, where R, G, and B are the color coordinates in the RGB color model; H, S, and L are the color coordinates in the HSL color model; and gray represents the grayscale intensity. Examples of the resulting semantic colorings are shown in Figure 5.



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

Figure 5. The result of fire detection using the Euclidean–Mahalanobis distance metric: (**a**) original image, (**b**) results of zone recognition.

It can be observed that the difficulty in solving the fire detection task is due to the high similarity between smoke and clouds or sky fragments, leading to errors [71].

Our hypothesis is that it may be possible to reduce the number of false fire signals during the patrolling of areas by UAVs by excluding the portion of the image that includes the sky. It is assumed that the information about the position of the horizon line can contribute to improving the accuracy of recognition and determining the scale of the fire.

4.3. Results of Horizon Line Detection

Figure 6 presents some results of the proposed algorithm under conditions of high smoke, semi-natural landscapes, and mountainous terrain.

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

Figure 6. Results of the proposed algorithm: (**a**) smoke zone, (**b**) semi-natural landscape, (**c**) mountains.

As shown in the figures, the developed algorithm allows for the detection of the horizon line in cases where it is partially covered by smoke (see Figure 6a). It is applicable in semi-natural landscapes (see Figure 6b) and mountainous terrain (see Figure 6c). The hypothesis test on the effectiveness of horizon segmentation as preprocessing for fire forest detection showed that the average accuracy without using horizon line information was 84.02%. The recognition quality increased to 95.03% after determining the horizon line. The application of the horizon line calculation algorithm improved the fire recognition accuracy by 11.01%.

4.4. Detection of Forest Fires with Convolutional Neural Networks

The original images from the UAV (FLAME dataset [72], files 9 and 10) with a size of 3840 × 2160 were converted to the new Yolo-compatible dataset [73], with training and test sets consisting of 4500 images each (with textual descriptions of fire zones). It is used for the tasks of fire source detection based on YOLO (versions of neural networks from 4 to 8). This dataset is most closely related to the problem being solved, as it contains real footage of a forest captured by UAVs.

The YOLO with different architectures [74] approaches was used for the experiments. However, the internal architectures of the networks were not changed, and hyperparameter tuning was not performed. Meanwhile, the following optimizers for neural networks were utilized [75]:

- Yolo 4 explicitly selected the ADAM optimizer.
- Yolo 5 explicitly selected the AdamW optimizer.
- Yolo 6 did not change the optimizer and used the default SGD in the code.
- Yolo 7 did not change the optimizer and used the default SGD in the code.
- Yolo 8 explicitly selected the AdamW optimizer.

The obtained results are presented in Table 3. Note that mAP is calculated without reference to the critical confidence threshold, which is common in the researched scientific literature: a certainty threshold, but column "Confidence threshold" contains values that are recommended for prediction (it provides the maximum F1 score).

Table 3. Results of YOLO application experiments.

ANN Version	Precision, %	Recall, %	F1 Score, %	Confidence Threshold, %	mAP@.5, %	mAP@.5:.95, %	Processing Speed, Frames per Second
4 csp mish	78.2	70.4	74.1	51.0	74.5	39.2	55
5s	79.4	71.1	75.1	51.5	76.7	40.7	49
5m with SiLU activation	79.3	**74.3** [2]	**76.8** [3]	44.4	**77. 9** [4]	41.3	45
5m with Mish activation	**80.2** [1]	72.1	75.9	49.9	77.1	41.1	43
5l	79.5	72.3	75.7	49.7	76.8	**41.8** [5]	35
6t	73.7	72.3	73.0	50.0	73.7	38.3	**122** [6]
7x	73.4	69.3	71.3	23.6	72.8	35.4	39
8l	77.3	70.3	73.0	30.1	76.1	36.4	39
8l Ghost	78.0	72.1	75.0	26.4	77.1	41.6	49

[1] The best result in Precision. [2] The best result in Recall. [3] The best result in F1 Score. [4] The best result in mAP@.5. [5] The best result in mAP@.5:.95. [6] The best result in Processing Speed.

YOLO 5m with SiLU activation provides high performance in solving the task at hand. Analysis of the results of the test dataset processing by the neural network revealed that explicit errors in fire detection are practically absent, but the detected fire is often

fragmented, and in many cases, such fragmentation does not correspond to the annotations in the dataset, which leads to a decrease in numerical indicators.

The same data as in the YOLO experiments (described above) were used for training and testing, but with pixel accuracy. Significantly better results were obtained compared to those achieved in the task of highlighting rectangular areas with target fire sources (see Table 4). Some neural networks are used in combination with a pre-trained backbone.

Table 4. Results of experiments on highlighting of fire source boundaries detection.

ANN Used	Normalized F1 Score, %	Frame Size, px × px	Processing Speed, Frames per Second
U-Net [76] with one convolution layer added to the coder and decoder	97.48	512 × 512	29.09
U-Net with VGG16 as a backbone	99.35	512 × 512	12.17
U-Net with ResNet18 as a backbone	**99.36** [1]	512 × 512	37.01
EfficientSeg [77]	97.47	512 × 512	54.99
TransUnet [78] with 6 vit blocks and 6 vit-headers, the mhsa block size is 512, the transformer size is 128	97.40	512 × 512	18.78
SwinUnet [79]	91.81	448 × 448	26.82
EffuNet [80]	97.54	512 × 512	**56.95** [2]
Linknet [81] with VGG16 as a backbone	99.30	512 × 512	11.62
Linknet with ResNet18 as a backbone	99.31	512 × 512	38.08

[1] The best result in Normalized F1 Score. [2] The best result in Processing Speed.

Replacing the VGG16 backbone with ResNet18 triples the speed of the neural network, but a small "fringe" appears at the boundaries of the designated fire areas, which is not critical and does not hinder the successful solution of the task. The main reason for the decrease in the performance of the neural network is that small fragments of the background next to the fire are incorrectly classified, including branches and needles.

5. Discussion

In this paper, we consider an approach to solving the problem of detecting forest fires using images and video streams from various technical vision systems. We solve the tasks of improving the quality of incoming video data, detecting the horizon line in the frame, and identifying fires using semantic segmentation with Euclidean–Mahalanobis distance and modified convolutional neural network YOLO.

There are certain limitations in the horizon line detection algorithm:

- Fog and snow in the processed image make the algorithm's operation challenging (increased processing time and decreased accuracy in detecting the horizon line).
- The applicability and effectiveness of the algorithm depend on the orientation of the UAV's camera.

During the processing of data obtained during daylight hours using video stream stabilization, deconvolution of data stream, and detection of forest fires based on CNN algorithms, no limitations were identified. The task of forest fire detection is significantly simplified during nighttime since there is a strong contrast in the image. Therefore, the article does not pay special attention to this issue.

The future work of the authors of the article includes a series of studies aimed at improving the obtained results and expanding the scope of the solved tasks. In particular, it involves addressing the following current issues:

- Processing satellite images of forest areas. Establishing aerospace monitoring will enhance the speed and accuracy of fire localization.
- Researching new architectures of neural networks, such as Baidu Rtdetr [82] and Yolo 9 [83].
- Addressing software and hardware acceleration issues to achieve real-time computations for processing multiple parallel incoming video streams.
- Addressing planning and optimization issues in monitoring forest areas using UAVs.
- Enhancing the reliability of UAV control through the development of an intelligent interface with speech and gesture recognition capabilities.
- Expanding the functions of UAVs through the ability to interact with multiple devices in a group.

All components of future work are aimed at building a comprehensive architecture of the software-hardware system for early automatic detection of forest fires and smoke.

6. Conclusions

The article presents the fundamentals of the algorithmic software for early fire detection in forest areas. The performed research contributes to the understanding of image processing processes from stationary surveillance cameras and unmanned aerial vehicles and the measurement of fire scales.

General results:

- The conducted research allowed to obtain a solution to two of the most important tasks of improving the video stream from UAVs: Removing blurs and stabilizing the video. The Stripformer neural network removes blurriness frame by frame in a video stream captured by a moving camera, and then the processed data are passed into the buffer of the stabilization module based on the VidGear library, which stabilizes the video stream. The final solution allows for the full cycle of video stream processing with a delay of no more than 0.03 s due to the use of a pipeline method for data processing. As a result, we obtain an enhanced image in real-time mode, which can be utilized for further research.
- A new approach for detecting smoke and fires is presented, applicable to challenging weather conditions and various types of landscapes. The approach is based on an algorithm that segments a portion of the image frame along the horizon line and applies a classifier in the form of a generalized Euclidean–Mahalanobis metric. This classifier relies on extracting a large number of fire and smoke features from the video stream frames. The calculation of horizon line points is based on determining the local contrast of the image within a sliding window, where the window size and the pixel informativeness threshold are determined for each image. Experimental results show that the main advantage of this approach is the ability to distinguish smoky areas from cloud-covered areas, which simplifies the task of early fire detection and, in some cases, leads to an improvement in accuracy of approximately 11%.
- An alternative approach to smoke and fire detection was explored, which does not require explicit feature extraction. The approach is based on analyzing different variants of CNN and fine-tuning them for smoke and fire classification. In the task of fire hotspot detection, versions of YOLO (ver. 4 to 8) were compared. The best results were achieved with YOLO 5m, which yielded an F1 score of 76.75% combined with a processing speed of 45 frames per second, i.e., in real time. In the segmentation task, the neural networks Linknet and U-Net with the ResNet18 backbone showed the best performance (by F1 score). During the experiments, some shortcomings and inaccuracies found in the current implementations of YOLO CNN were identified and addressed.

The obtained results differ from existing analogs by utilizing a comprehensive approach to early fire detection, which includes image enhancement and alternative real-time video processing methods. This expands the capabilities for potential users.

Author Contributions: Conceptualization, V.K. and M.K.; data curation, V.F.; formal analysis, V.K. and M.K.; funding acquisition, V.K. and M.K.; investigation, N.A., Y.E., V.F., M.S. and A.T.; methodology, V.F., V.K. and M.K.; project administration, V.K. and M.K.; resources, V.F., V.K. and M.K.; software, N.A., Y.E., V.F. and A.T.; supervision, V.K. and M.K.; validation, V.F. and M.S.; visualization, N.A., V.F. and A.T.; writing—original draft, N.A., Y.E., V.F., V.K., M.K. and A.T.; writing—review and editing, N.A., Y.E., V.F., V.K., M.K., M.S. and A.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by RUSSIAN SCIENCE FOUNDATION, grant number 22-11-20001, https://rscf.ru/en/project/22-11-20001/ (accessed on 12 March 2024) and a grant in the form of a subsidy from the regional budget to organizations in the Yaroslavl region (No 10/04-2022-7 June 2022).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The used datasets are released under CC BY 4.0 license and publicly available at https://paperswithcode.com/dataset/gopro (accessed on 12 March 2024) and https://ieee-dataport.org/open-access/flame-dataset-aerial-imagery-pile-burn-detection-using-drones-uavs (accessed on 12 March 2024).

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

References

1. Flyox I Amphibious UAV—Aerospace Technology. Available online: https://www.aerospace-technology.com/projects/flyox-i-amphibious-uav/ (accessed on 24 January 2024).
2. Unmanned Aerial Systems (UAS) as Firefighters | Commercial UAV News. Available online: https://www.commercialuavnews.com/public-safety/unmanned-aerial-systems-uas-firefighters (accessed on 24 January 2024).
3. YOLO: Real-Time Object Detection. Available online: https://pjreddie.com/darknet/yolo/ (accessed on 24 January 2024).
4. Khachumov, M.V. Distances, Metrics and Cluster Analysis. *Sci. Tech. Inf. Process.* **2012**, *39*, 310–316. [CrossRef]
5. Islam, A.M.; Masud, F.B.; Ahmed, M.R.; Jafar, A.I.; Ullah, J.R.; Islam, S.; Shatabda, S.; Islam, A.K.M.M. An Attention-Guided Deep-Learning-Based Network with Bayesian Optimization for Forest Fire Classification and Localization. *Forests* **2023**, *14*, 2080. [CrossRef]
6. Parthipan, V.; Dhanasekaran, D. Preventing and Monitoring of Framework for Forest Fire Detection and Data Analysis Using Internet of Things (IoT). *Int. J. Eng. Adv. Technol.* **2019**, *8*, 691–695.
7. Brito, T.; Azevedo, B.F.; Mendes, J.; Zorawski, M.; Fernandes, F.P.; Pereira, A.I.; Rufino, J.; Lima, J.; Costa, P. Data Acquisition Filtering Focused on Optimizing Transmission in a LoRaWAN Network Applied to the WSN Forest Monitoring System. *Sensors* **2023**, *23*, 1282. [CrossRef]
8. El-Madafri, I.; Peña, M.; Olmedo-Torre, N. Dual-Dataset Deep Learning for Improved Forest Fire Detection: A Novel Hierarchical Domain-Adaptive Learning Approach. *Mathematics* **2024**, *12*, 534. [CrossRef]
9. Goyal, S.; Vohra, H.; Singh, A.; Kaur, A. A YOLO based Technique for Early Forest Fire Detection. *Int. J. Innov. Technol. Explor. Eng.* **2020**, *9*, 1357–1362. [CrossRef]
10. Wang, Z.; Zhang, H.; Hou, M.; Shu, X.; Wu, J.; Zhang, X. A Study on Forest Flame Recognition of UAV Based on YOLO-V3 Improved Algorithm. In *Recent Advances in Sustainable Energy and Intelligent Systems*; Li, K., Coombs, T., He, J., Tian, Y., Niu, Q., Yang, Z., Eds.; Communications in Computer and Information Science; Springer: Singapore, 2021; Volume 1468, pp. 497–503; ISBN 9789811672095.
11. Kim, S.-Y.; Muminov, A. Forest Fire Smoke Detection Based on Deep Learning Approaches and Unmanned Aerial Vehicle Images. *Sensors* **2023**, *23*, 5702. [CrossRef] [PubMed]
12. Cheng, Y.; Chen, K.; Bai, H.; Mou, C.; Zhang, Y.; Yang, K.; Gao, Y.; Liu, Y. An Efficient Fire Detection Algorithm Based on Multi-scale Convolutional Neural Network. *Fire Mater.* **2022**, *46*, 981–992. [CrossRef]
13. Pradeep Kumar, G.; Rahul, R.; Ravindharan, N. Early Forest Fire Detection Using Machine Learning Algorithms. *Int. J. New Technol. Res.* **2021**, *7*, 1–5.
14. Talaat, F.M.; ZainEldin, H. An Improved Fire Detection Approach Based on YOLO-v8 for Smart Cities. *Neural Comput. Appl.* **2023**, *35*, 20939–20954. [CrossRef]

15. Cruz, H.; Gualotuña, T.; Pinillos, M.; Marcillo, D.; Jácome, S.; Fonseca, C.E.R. Machine Learning and Color Treatment for the Forest Fire and Smoke Detection Systems and Algorithms, a Recent Literature Review. In *Artificial Intelligence, Computer and Software Engineering Advances*; Botto-Tobar, M., Cruz, H., Díaz Cadena, A., Eds.; Advances in Intelligent Systems and Computing; Springer International Publishing: Cham, Switzerland, 2021; Volume 1326, pp. 109–120; ISBN 978-3-030-68079-4.
16. Yu, Y.; Yao, M. When Convolutional Neural Networks Meet Laser-Induced Breakdown Spectroscopy: End-to-End Quantitative Analysis Modeling of ChemCam Spectral Data for Major Elements Based on Ensemble Convolutional Neural Networks. *Remote Sens.* **2023**, *15*, 3422. [CrossRef]
17. Geetha, S.; Abhishek, C.S.; Akshayanat, C.S. Machine Vision Based Fire Detection Techniques: A Survey. *Fire Technol.* **2021**, *57*, 591–623. [CrossRef]
18. Ciprián-Sánchez, J.F.; Ochoa-Ruiz, G.; Rossi, L.; Morandini, F. Assessing the Impact of the Loss Function, Architecture and Image Type for Deep Learning-Based Wildfire Segmentation. *Appl. Sci.* **2021**, *11*, 7046. [CrossRef]
19. Favorskaya, M.N. Early Smoke Detection in Outdoor Space: State-of-the-Art, Challenges and Methods. In *Advances in Selected Artificial Intelligence Areas*; Virvou, M., Tsihrintzis, G.A., Jain, L.C., Eds.; Learning and Analytics in Intelligent Systems; Springer International Publishing: Cham, Switzerland, 2022; Volume 24, pp. 171–208; ISBN 978-3-030-93051-6.
20. Huo, Y.; Zhang, Q.; Jia, Y.; Liu, D.; Guan, J.; Lin, G.; Zhang, Y. A Deep Separable Convolutional Neural Network for Multiscale Image-Based Smoke Detection. *Fire Technol.* **2022**, *58*, 1445–1468. [CrossRef]
21. Miao, J.; Zhao, G.; Gao, Y.; Wen, Y. Fire Detection Algorithm Based on Improved YOLOv5. In Proceedings of the 2021 International Conference on Control, Automation and Information Sciences (ICCAIS), Xi'an, China, 14–17 October 2021; pp. 776–781. [CrossRef]
22. Li, Y.; Zhang, W.; Liu, Y.; Jin, Y. A Visualized Fire Detection Method Based on Convolutional Neural Network beyond Anchor. *Appl. Intell.* **2022**, *52*, 13280–13295. [CrossRef]
23. Wang, S.; Zhao, J.; Ta, N.; Zhao, X.; Xiao, M.; Wei, H. A Real-Time Deep Learning Forest Fire Monitoring Algorithm Based on an Improved Pruned + KD Model. *J. Real-Time Image Process.* **2021**, *18*, 2319–2329. [CrossRef]
24. Wang, S.; Chen, T.; Lv, X.; Zhao, J.; Zou, X.; Zhao, X.; Xiao, M.; Wei, H. Forest Fire Detection Based on Lightweight Yolo. In Proceedings of the 2021 33rd Chinese Control and Decision Conference (CCDC), Kunming, China, 22–24 May 2021; pp. 1560–1565. [CrossRef]
25. Nebaba, S.G. In Methods of evaluation and preparation of images in the video stream for object recognition. In Proceedings of the 28th International Conference on Computer Graphics and Vision "GraphiCon 2018", Tomsk, Russia, 24–27 September 2018; pp. 450–453. (In Russian)
26. Sherstjuk, V.; Zharikova, M.; Dorovskaja, I.; Sheketa, V. Assessing Forest Fire Dynamicsin UAV-Based Tactical Monitoring System. In *Lecture Notes in Computational Intelligence and Decision Making*; Babichev, S., Lytvynenko, V., Wójcik, W., Vyshemyrskaya, S., Eds.; Advances in Intelligent Systems and Computing; Springer International Publishing: Cham, Switzerland, 2021; Volume 1246, pp. 285–301; ISBN 978-3-030-54214-6. [CrossRef]
27. Chernov, T.S. Mathematical Models and Algorithms for Image Quality Assessment in Optical Recognition Systems. Ph.D. Thesis, FRCCSC, Moscow, Russia, 2018. (In Russian)
28. Mohammadi, P.; Ebrahimi-Moghadam, A.; Shirani, S. Subjective and Objective Quality Assessment of Image: A Survey. *arXiv* **2014**, arXiv:1406.7799. [CrossRef]
29. Mittal, A.; Moorthy, A.K.; Bovik, A.C. No-Reference Image Quality Assessment in the Spatial Domain. *IEEE Trans. Image Process.* **2012**, *21*, 4695–4708. [CrossRef]
30. Xue, W.; Mou, X.; Zhang, L.; Bovik, A.C.; Feng, X. Blind Image Quality Assessment Using Joint Statistics of Gradient Magnitude and Laplacian Features. *IEEE Trans. Image Process.* **2014**, *23*, 4850–4862. [CrossRef]
31. Zhang, L.; Zhang, L.; Bovik, A.C. A Feature-Enriched Completely Blind Image Quality Evaluator. *IEEE Trans. Image Process.* **2015**, *24*, 2579–2591. [CrossRef]
32. Korovin, I.S.; Khisamutdinov, M.V. Method of Noise-Free Image Production Based on Video Sequence Handling. *AASRI Procedia* **2014**, *6*, 73–81. [CrossRef]
33. Khisamutdinov, M.; Kalyaev, A.; Korovin, I. New Method of Improving the Quality of Single Images in a Video Sequence. *DEStech Trans. Comput. Sci. Eng.* **2018**, *215*, 744. [CrossRef] [PubMed]
34. Zotin, A.G.; Pakhirka, A.I.; Buryachenko, V.V. Development of video surveillance system with visual quality enhancement. *Softw. Syst.* **2013**, *2*, 191–197. (In Russian)
35. Liu, Y.-J.; Chiu, C.-C.; Yang, J.-H. A Robust Vision-Based Skyline Detection Algorithm under Different Weather Conditions. *IEEE Access* **2017**, *5*, 22992–23009. [CrossRef]
36. Guo, F.; Mai, Y.; Tang, J.; Huang, Y.; Zhu, L. Robust and Automatic Skyline Detection Algorithm Based on MSSDN. *J. Adv. Comput. Intell. Intell. Inform.* **2020**, *24*, 750–762. [CrossRef]
37. Martinez-Sanchez, L.; Borio, D.; d'Andrimont, R.; Van Der Velde, M. Skyline Variations Allow Estimating Distance to Trees on Landscape Photos Using Semantic Segmentation. *Ecol. Inform.* **2022**, *70*, 101757. [CrossRef]
38. Ahmad, T.; Emami, E.; Čadík, M.; Bebis, G. Resource Efficient Mountainous Skyline Extraction Using Shallow Learning. In Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN), Shenzhen, China, 18–22 July 2021. [CrossRef]
39. Lin, C.; Chen, W.; Zhou, H. Multi-Visual Feature Saliency Detection for Sea-Surface Targets through Improved Sea-Sky-Line Detection. *J. Mar. Sci. Eng.* **2020**, *8*, 799. [CrossRef]

40. Khachumov, V.M.; Portnov, E.M.; Fedorov, P.A.; Kasimov, R.A.; Naing Linn, A. Development of an Accelerated Method for Calculating Streaming Video Data Obtained from UAVs. In Proceedings of the 2020 8th International Conference on Control, Mechatronics and Automation (ICCMA), Moscow, Russia, 6–8 November 2020; pp. 212–216. [CrossRef]
41. Zhu, D.; Wan, L.; Gao, W. Fusion Method Evaluation and Classification Suitability Study of Wetland Satellite Imagery. *Earth Sci. Res. J.* **2019**, *23*, 339–346. [CrossRef]
42. Vichevskaya, J.A.; Murynov, A.I. Structural analysis of images based on the use of the informative function. *Alm. Sovrem. Nauk. I Obraz.* **2010**, *4*, 53–55. (In Russian)
43. Tymchuk, A.I. On the choice of gray levels in the problem of texture segmentation of images based on the luminance dependence matrices. *Cybern. Program.* **2018**, *3*, 1–9. (In Russian) [CrossRef]
44. Zhurbin, I.; Bazhenova, A.; Shaura, A.; Zlobina, A. Determining the Sliding Window Size and the Optimal Number of Clusters in the Problem of Texture Segmentation of Multispectral Aerial Photography Data. *HFIM* **2020**, *4*, 434–447. (In Russian) [CrossRef]
45. Emelyanova, Y.G. Algorithm for finding the horizon line in images taken from an unmanned aerial vehicle camera. *Aerosp. Instrum. Mak.* **2023**, *1*, 40–53. (In Russian) [CrossRef]
46. Bakir, H.; Bakir, R. Evaluating the robustness of yolo object detection algorithm in terms of detecting objects in noisy environment. *J. Sci. Rep. A* **2023**, *54*, 1–25. [CrossRef]
47. GoPro DataSet. Available online: https://paperswithcode.com/dataset/gopro (accessed on 24 January 2024).
48. Awesome-Deblurring. Available online: https://github.com/subeeshvasu/Awesome-Deblurring (accessed on 24 January 2024).
49. Chu, X.; Chen, L.; Chen, C.; Lu, X. Improving Image Restoration by Revisiting Global Information Aggregation. In *Computer Vision—ECCV 2022*; Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T., Eds.; Lecture Notes in Computer Science; Springer Nature: Cham, Switzerland, 2022; Volume 13667, pp. 53–71; ISBN 978-3-031-20070-0.
50. Tu, Z.; Talebi, H.; Zhang, H.; Yang, F.; Milanfar, P.; Bovik, A.; Li, Y. MAXIM: Multi-Axis MLP for Image Processing. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 5759–5770. [CrossRef]
51. Sun, L.; Sakaridis, C.; Liang, J.; Jiang, Q.; Yang, K.; Sun, P.; Ye, Y.; Wang, K.; Gool, L.V. Event-Based Fusion for Motion Deblurring with Cross-Modal Attention. In *Computer Vision—ECCV 2022*; Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T., Eds.; Lecture Notes in Computer Science; Springer Nature: Cham, Switzerland, 2022; Volume 13678, pp. 412–428; ISBN 978-3-031-19796-3.
52. Li, D.; Zhang, Y.; Cheung, K.C.; Wang, X.; Qin, H.; Li, H. Learning Degradation Representations for Image Deblurring. In *Computer Vision—ECCV 2022*; Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T., Eds.; Lecture Notes in Computer Science; Springer Nature: Cham, Switzerland, 2022; Volume 13678, pp. 736–753; ISBN 978-3-031-19796-3.
53. Mou, C.; Wang, Q.; Zhang, J. Deep Generalized Unfolding Networks for Image Restoration. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022. [CrossRef]
54. He, W.; Yao, Q.; Yokoya, N.; Uezato, T.; Zhang, H.; Zhang, L. Spectrum-Aware and Transferable Architecture Search for Hyperspectral Image Restoration. In *Computer Vision—ECCV 2022*; Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T., Eds.; Lecture Notes in Computer Science; Springer Nature: Cham, Switzerland, 2022; Volume 13679, pp. 19–37; ISBN 978-3-031-19799-4.
55. Tsai, F.-J.; Peng, Y.-T.; Lin, Y.-Y.; Tsai, C.-C.; Lin, C.-W. Stripformer: Strip Transformer for Fast Image Deblurring. In *Computer Vision—ECCV 2022*; Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T., Eds.; Lecture Notes in Computer Science; Springer Nature: Cham, Switzerland, 2022; Volume 13679, pp. 146–162; ISBN 978-3-031-19799-4.
56. Wang, Z.; Cun, X.; Bao, J.; Zhou, W.; Liu, J.; Li, H. Uformer: A General U-Shaped Transformer for Image Restoration. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 17662–17672. [CrossRef]
57. STDAN. Available online: https://github.com/huicongzhang/STDAN (accessed on 24 January 2024).
58. MMP RNN. Available online: https://github.com/sollynoay/MMP-RNN (accessed on 24 January 2024).
59. Qian, Y.; Barthelemy, J.; Iqbal, U.; Perez, P. V2ReID: Vision-Outlooker-Based Vehicle Re-Identification. *Sensors* **2022**, *22*, 8651. [CrossRef]
60. abhiTronix/Vidgear: VidGear Stable v0.2.6 2022. Available online: https://zenodo.org/records/6926196 (accessed on 12 March 2024).
61. Liu, Y.-L.; Lai, W.-S.; Yang, M.-H.; Chuang, Y.-Y.; Huang, J.-B. Hybrid Neural Fusion for Full-Frame Video Stabilization. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 10–17 October 2021. [CrossRef]
62. Liu, S.; Tan, P.; Yuan, L.; Sun, J.; Zeng, B. MeshFlow: Minimum Latency Online Video Stabilization. In *Computer Vision—ECCV 2016*; Leibe, B., Matas, J., Sebe, N., Welling, M., Eds.; Lecture Notes in Computer Science; Springer International Publishing: Cham, Switzerland, 2016; Volume 9910, pp. 800–815; ISBN 978-3-319-46465-7.
63. Video Stabilization with L1 Optimal Camera Paths. Available online: https://github.com/ishank-juneja/L1-optimal-paths-Stabilization (accessed on 24 January 2024).
64. Grundmann, M.; Kwatra, V.; Essa, I. Auto-Directed Video Stabilization with Robust L1 Optimal Camera Paths. In Proceedings of the CVPR 2011, Colorado Springs, CO, USA, 20–25 June 2011; pp. 225–232. [CrossRef]
65. Video Stabilization Using Homography Transform. Available online: https://github.com/SergejVolkov/video_smoothing (accessed on 24 January 2024).

66. Xu, Y.; Zhang, J.; Maybank, S.J.; Tao, D. DUT: Learning Video Stabilization by Simply Watching Unstable Videos. *IEEE Trans. Image Process.* **2022**, *31*, 4306–4320. [CrossRef]
67. Choi, J.; Kweon, I.S. Deep Iterative Frame Interpolation for Full-Frame Video Stabilization. *ACM Trans. Graph.* **2020**, *39*, 1–9. [CrossRef]
68. Wang, M.; Yang, G.-Y.; Lin, J.-K.; Zhang, S.-H.; Shamir, A.; Lu, S.-P.; Hu, S.-M. Deep Online Video Stabilization with Multi-Grid Warping Transformation Learning. *IEEE Trans. Image Process.* **2019**, *28*, 2283–2292. [CrossRef]
69. Abhishek, S.T. Video Stabilization Using Point Feature Matching in OpenCV. Available online: https://learnopencv.com/video-stabilization-using-point-feature-matching-in-opencv/ (accessed on 24 January 2024).
70. Kvyetnyy, R.; Sofina, O.; Olesenko, A.; Komada, P.; Sikora, J.; Kalizhanova, A.; Smailova, S. Method of Image Texture Segmentation Using Laws' Energy Measures. In Proceedings of the Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics, Wilga, Poland, 7 August 2017; Romaniuk, R.S., Linczuk, M., Eds.; SPIE: Bellingham, WA, USA, 2017; p. 1044561.
71. Mukhiddinov, M.; Abdusalomov, A.B.; Cho, J. A Wildfire Smoke Detection System Using Unmanned Aerial Vehicle Images Based on the Optimized YOLOv5. *Sensors* **2022**, *22*, 9384. [CrossRef] [PubMed]
72. Shamsoshoara, A.; Afghah, F.; Razi, A.; Zheng, L.; Fulé, P. The Flame Dataset: Aerial Imagery Pile Burn Detection Using Drones (UAVS). 2021. Available online: https://ieee-dataport.org/open-access/flame-dataset-aerial-imagery-pile-burn-detection-using-drones-uavs (accessed on 24 January 2024).
73. Alarmod/Forest_Fire Datasets at Hugging Face. Available online: https://huggingface.co/datasets/alarmod/forest_fire (accessed on 24 February 2024).
74. Models Supported by Ultralytics. Available online: https://docs.ultralytics.com/models (accessed on 24 February 2024).
75. Loshchilov, I.; Hutter, F. Decoupled Weight Decay Regularization. *arXiv* **2017**, arXiv:1711.05101. [CrossRef]
76. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2015, Munich, Germany, 5–9 October 2015*; Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F., Eds.; Lecture Notes in Computer Science; Springer International Publishing: Cham, Switzerland, 2015; Volume 9351, pp. 234–241; ISBN 978-3-319-24573-7.
77. Yesilkaynak, V.B.; Sahin, Y.H.; Unal, G. EfficientSeg: An Efficient Semantic Segmentation Network. *arXiv* **2020**, arXiv:2009.06469. [CrossRef]
78. Ghali, R.; Akhloufi, M.A.; Mseddi, W.S. Deep Learning and Transformer Approaches for UAV-Based Wildfire Detection and Segmentation. *Sensors* **2022**, *22*, 1977. [CrossRef]
79. Cao, H.; Wang, Y.; Chen, J.; Jiang, D.; Zhang, X.; Tian, Q.; Wang, M. Swin-Unet: Unet-like Pure Transformer for Medical Image Segmentation. In *European Conference on Computer Vision*; Springer Nature: Cham, Switzerland, 2021.
80. Baheti, B.; Innani, S.; Gajre, S.; Talbar, S. Eff-UNet: A Novel Architecture for Semantic Segmentation in Unstructured Environment. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, WA, USA, 14–19 June 2020; pp. 1473–1481. [CrossRef]
81. Chaurasia, A.; Culurciello, E. LinkNet: Exploiting Encoder Representations for Efficient Semantic Segmentation. In Proceedings of the 2017 IEEE Visual Communications and Image Processing (VCIP), St. Petersburg, FL, USA, 10–13 December 2017; pp. 1–4. [CrossRef]
82. Lv, W.; Zhao, Y.; Xu, S.; Wei, J.; Wang, G.; Cui, C.; Du, Y.; Dang, Q.; Liu, Y. DETRs Beat YOLOs on Real-Time Object Detection. *arXiv* **2023**, arXiv:2304.08069. [CrossRef]
83. Wang, C.-Y.; Yeh, I.-H.; Liao, H.-Y.M. YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information. *arXiv* **2024**, arXiv:2402.13616. [CrossRef]