Article

Moving Object Detection Based on Background Compensation and Deep Learning

Juncai Zhu*, Zhizhong Wang, Songwei Wang and Shuli Chen

School of Electrical Engineering, Zhengzhou University, Zhengzhou 450000, China; zhujuncai@gs.zzu.edu.cn (J.Z.); wzz1982@zzu.edu.cn (Z.W.); wangsongwei@zzu.edu.cn (S.W.)

* Correspondence: chensl@zzu.edu.cn

Received: 3 November 2020; Accepted: 23 November 2020; Published: 27 November 2020

Abstract: Detecting moving objects in a video sequence is an important problem in many vision-based applications. In particular, detecting moving objects when the camera is moving is a difficult problem. In this study, we propose a symmetric method for detecting moving objects in the presence of a dynamic background. First, a background compensation method is used to detect the proposed region of motion. Next, in order to accurately locate the moving objects, we propose a convolutional neural network-based method called YOLOv3-SOD for detecting all objects in the image, which is lightweight and specifically designed for small objects. Finally, the moving objects are determined by fusing the results obtained by motion detection and object detection. Missed detections are recalled according to the temporal and spatial information in adjacent frames. A dataset is not currently available specifically for moving object detection and recognition, and thus, we have released the MDR105 dataset comprising three classes with 105 videos. Our experiments demonstrated that the proposed algorithm can accurately detect moving objects in various scenarios with good overall performance.

Keywords: convolutional neural network; dynamic background; motion compensation; moving object detection; small target detection

1. Introduction

Moving object detection and recognition is an important research area in the field of computer vision, where it plays important roles in intelligent video surveillance [1,2], robot vision navigation [3,4], virtual reality [5], and medical diagnosis (cell state tracking) [6]. In particular, in recent years, the development of unmanned aerial vehicles (UAV) has increased the interest in detecting moving objects in video sequences [7,8]. UAVs have advanced imaging capacities, where the camera can operate with various degrees of movement and autonomy, but problems occur due to the moving background and motion blur. In addition, in outdoor conditions, due to light, occlusion, and shadows, the appearance of a moving object can change to affect the precision of moving object detection, therefore, developing a robust method for motion detection and recognition is particularly important.

In contrast to the moving object detection algorithms used for fixed cameras, such as optical flow [9], inter-frame difference [10], and background modeling [11], the image background often appears to undergo rotation, translation, and scaling when employing moving cameras. Therefore, it is difficult to model the background or detect moving targets based on the optical flow. To solve this problem, we propose a moving object detection method by combining background compensation [12–14] and deep learning [15–17]. The background compensation method is used to determine the motion information present in the image. First, the motion parameter is calculated according to the coordinate relationship of feature points in adjacent frames, and then the binary mask of the moving regions is obtained by using the inter-frame difference method. In order to make the matching of feature points more accurate, we restrict the feature points by spatial distribution and separate the outer points based
on the result of inter-frame object matching. The outer points are feature points on moving objects, which will have a bad influence on image registration. The deep learning method aims to locate the moving targets more precisely, in this paper, we proposed an object detection model—YOLOv3-SOD (YOLOv3 for small object detection)—which is faster and has a higher detection performance for small targets. Moving objects can be detected correctly by fusing the results obtained with these two methods. However, due to the influence of illumination variation, change in the appearance of moving objects, the presence of abrupt motion, occlusion and shadow, etc., the background compensation method often fails to detect the moving regions, resulting in moving objects escaping detection, which is called motion loss. To address this problem, we use the temporal and spatial information in adjacent frames to recall the lost targets.

In this study, we also developed a new dataset called MDR105 for moving object detection and recognition, which comprises three categories and 105 videos. All of the videos in our dataset were obtained in real-world situations and they contain different scenarios, such as multiple targets, complex backgrounds, and shadows. We only labeled the moving objects in the videos, i.e., some videos contained multiple cars but only the moving cars were considered to be correct. This dataset is available through the following link: https://figshare.com/articles/dataset/Images_zip/13247306. Experiments based on the MDR105 dataset demonstrated that our method performs well compared with other traditional algorithms.

The remainder of this paper is organized as follows. We describe related research in Section 2. In Section 3, we provide a detailed description of our proposed method. In Section 4, we present the dataset, experimental settings, and our comparative results. Finally, we give our conclusions in Section 5.

2. Related Work

2.1. Moving Object Detection Methods

In recent years, many promising applications have emerged for moving object detection and recognition using a motion camera, and thus, numerous studies have investigated this problem. These studies can be roughly assigned to three categories comprising background compensation [12–14], trajectory classification [18–20], and object tracking [21–23].

The background compensation method estimates the camera’s motion based on optical flow, block features [24], and point features, such as Harris [25], the scale-invariant feature transform (SIFT) [26], and oriented fast and rotated brief (ORB) [27]. For example, Adel Hafiane et al. [28] extracted a corner-based feature block to compensate for the moving background and their approach obtained good performance with videos captured by a camera on a UAV. Setyawan et al. [29] used the Harris corner detector to extract the feature points, and this approach was shown to be more robust and less sensitive to noise compared with other methods. The trajectory classification method involves selecting an appropriate tracker to calculate the long-term trajectory of the feature points, before using the clustering method to distinguish the trajectory belonging to the same target from others in the background. Yin et al. [18] used optical flow to compensate for the motion of the camera and then applied principal component analysis to reduce the number of atypical trajectories, before using the watershed transformation for clustering. Brox et al. [19] also employed the optical flow method to obtain point trajectories in video sequences, but using spectral clustering with spatial regularity to classify the foreground and background trajectories. The aim of this method is different but object tracking can also be regarded as moving object detection. Traditional object tracking uses statistical information such as colors, textures, histograms, and gradient maps to model objects, before locating the most similar object in a video sequence. A new approach for representing objects is the convolutional neural network (CNN), the ECO [23] algorithm proposed by Martin Danelljan introduced a factorized convolution operator on the basis of Discriminative Correlation Filter (DCF) to reduce the parameters of the model; it solved the computational complexity and overfitting problems,
therefore the tracking speed and performance were improved. Deep Sort [30] is a multi-target tracking algorithm. The algorithm firstly detects the targets of each frame by object detection model, then predicts the motion trajectory through Kalman filter, and uses the weighted Hungarian algorithm to match boxes, achieving a good effect in pedestrian multi-target tracking.

2.2. Object Detection Methods

Deep learning is better at extracting the characteristics of objects and representing objects, and thus, CNN has been applied in many areas, such as object detection, semantic segmentation, and target tracking. A well-trained CNN can extract object features in a rapid, robust, and stable manner. Many excellent CNN network structures are available, such as Faster R-CNN [17] based on a candidate region network for real-time target detection and the single shot multi-box detector network (SSD) [31] using multi-scale prediction. YOLOv3 [16] is recognized for its high speed and accuracy. YOLOv3 uses Darknet-53 as the feature extraction network, and is based on the capacity of ResNet [32] for solving the problem of degradation caused by increases in the network depth, and thus, the feature extraction capability of YOLOv3 is high. In addition, in order to increase the accuracy of small object detection, YOLOv3 employs multi-scale prediction to output feature maps with different fine-grained features, as well as using similar feature pyramid networks (FPN) [33] upsampling and fusion methods. In the present study, in addition to the moving object detection task, YOLOv3 is improved and used as an object detector to correct candidate moving objects.

3. Proposed Method

In this study, we propose a moving object detection method based on a combination of background compensation and object detection, which has three modules comprising motion detection, object detection, and object matching. The motion detection module obtains the rough motion position with the background compensation method. YOLOv3-Small Objects Detection (SOD) is introduced in the object detection module as an improved version of YOLOv3 for accurately detecting the locations of objects. In the object-matching module, a moving object is determined by matching the motion detection and object detection results, while temporal and spatial types of information from adjacent frames are used to recall the missing detections. The proposed approach is shown in Figure 1 and we explain the proposed method in the following.

Figure 1. Flowchart illustrating the proposed method.

3.1. Motion Detection

The background compensation method involves calculating the motion parameter model based on the coordinate relationships of the matched feature points. We compared the detection performance of several feature point detections algorithms and selected the SIFT algorithm. The SIFT algorithm is a feature point detection algorithm based on scale space, which exhibits good invariability with respect to image translation, rotation, scaling, shielding, noise, and brightness changes. Figure 2 shows the detection results obtained by the SIFT, Harris, features from the accelerated segment test (FAST) [34], and speeded up robust features (SURF) [35] feature point detection algorithms. Table 1 shows statistics
regarding the number of feature points detected and the time required. Experiments showed that the SIFT algorithm could extract the highest number of feature points with the greatest distribution, and the speed also satisfied the requirements.

![Figure 2](image)

**Table 1.** Comparison of four feature point detection algorithms.

| Methods    | Number of Feature Points | Time (s)  |
|------------|--------------------------|-----------|
| FAST [34]  | 108                      | 0.000180  |
| Harris [25]| 245                      | 0.037304  |
| SURF [35]  | 242                      | 0.162658  |
| SIFT [26]  | 532                      | 0.037742  |

The SIFT algorithm creates a descriptor for each feature point, which is represented by a 128-dimensional vector, and SIFT feature points with minimum Euclidean distance are matched in two adjacent frames. Figure 3a shows the matching result; we can see that the feature points are concentrated and form small clusters, and the area where the moving object is located contains a large number of feature points (called outer points), which is harmful to the accuracy of image registration. In the selection of feature points, we should consider the following factors:

1. When the intensity of the feature points is greater, they are less likely to be lost;
2. An excessive number of feature points can result in too many calculations when matching features;
3. An excessive concentration of feature points can lead to large errors for the motion parameters.

Thus, a reasonable distribution for the feature points should be broad and average, and not include outer points.

![Figure 3](image)

The feature points are restricted by their spatial distribution. First, the image is divided into a grid with a size of $S \times S$ in order to make the distribution of the feature points as uniform as possible, but with sufficient feature points, where we selected $S = 10$. The strongest feature points in the grid are then retained and other feature points are filtered out. Finally, according to the results obtained
after object-matching between frames, the feature points in the grids where the bounding boxes of objects in the previous frame are located are filtered out (designated as outer points). Figure 3 shows the matching results for the remaining feature points, where the distribution is more uniform and the points on the moving target have been removed.

The selected feature points still include a few outpoints, so the random sample consensus (RANSAC) [36] algorithm is used to calculate the perspective transformation parameters in order to estimate high-precision parameters from the data containing outer points. The perspective transformation can allow translation, scaling, rotation, and shearing, while it is more flexible and can accurately express the form of motion for the image background compared with the affine transformation. Equation (1) is the general perspective transformation formula

\[
[x', y', w'] = [\mu, \nu, w] \times T
\]

where \(\mu, \nu\) are the coordinates in the original image, \(x', y'\) are the coordinates in the transformed image, and \(T\) is the \(3 \times 3\) perspective transformation matrix. After background compensation, the three-frame difference method is used to obtain the binarization mask for the moving region. The formula for the three-frame difference method is as follows

\[
d_k(x, y) = \begin{cases} 1 & |l_{k-3}(x, y) - l_k(x, y)| > T, |l_{k+3}(x, y) - l_k(x, y)| > T \\ 0 & \text{others} \end{cases}
\]

where \(l_k(x, y)\) is the pixel value at position \((x, y)\) in frame \(k\), \(d_k(x, y)\) is the gray value of pixel \((x, y)\) in the binary image, 1 indicates that the pixel has moved, and \(T\) is the difference threshold. If the speed of movement for an object is excessively slow, the position of the moving target in the adjacent frames will not change greatly as if the object is not moving, so we set the interval between frames to three.

The inter-frame difference method outputs the binarization mask for the moving objects, but it contains a large amount of noise due to the uncertain threshold, background matching error, and changes in irregular objects such as light, shadow, cloud, and water grain in the image. We can apply morphological operations and analyze the properties of connected regions in binary images to reduce the influence of noise, but it is difficult to accurately and completely detect moving targets using only the inter-frame difference method. Figure 4 shows several problems associated with the inter-frame difference method. We apply CNN to solve these problems as described in the following section.

Figure 4. Problems associated with the inter-frame difference method: (a) shadow, (b) reflection, (c) occlusion, and (d) inaccurate registration.
3.2. Object Detection

We use the object detection network to detect all potential objects in the image regardless of whether they have moved or not. We propose two improvements to YOLOv3 for the moving object detection task. First, considering the possibility of deploying an UAV and surveillance camera, it is necessary to further reduce the size of the model and improve the speed of the model. Second, the moving object detection task involves a large number of small moving targets. We counted the numbers of pixels for the moving targets in the images and found that the targets with 20–50 pixels comprised the majority; therefore, the YOLOV3 network needs to be optimized for small-target detection. Thus, we introduce the YOLOv3-SOD network and Figure 5 shows the network structure for YOLOv3-SOD.

Compared with YOLOV3, we employ a shallower network where the backbone network comprises five residual modules with one, two, four, four, and two residual blocks. Due to the shallower network, YOLOv3-SOD is more likely to retain the small object features, and the shallower network improves the running speed and reduces the model size. In the CNNs, the deep layers contain semantic information, whereas the shallow layers contain position information. The small object features may be lost as the number of network layers deepens, so YOLOv3 uses upsampling to merge the different layers of the feature maps to solve this problem. We use the same method in YOLOv3-SOD to improve the ability to detect small targets. The $4 \times$ downsampling layer is selected as the final output layer of the network, the $8 \times$ downsampling layer is concatenated with the $4 \times$ downsampling layer, the $16 \times$ downsampling is concatenated with the $8 \times$ and $4 \times$ downsampling layers, and the $32 \times$ downsampling layer is concatenated with the $16 \times$ and $8 \times$ downsampling layers. A branch fusion structure is added so the shallow and deep information can be merged, thereby increasing the abundance of the location and semantic information for the small objects. In order to avoid the gradient vanishing during training and to enhance the reuse of features, inspired by the deconvolution single-shot multi-box detector (DSSD) [37] network, the convolutional layer of the YOLOV3 output layer is transformed into two residual units and a convolutional layer. The final output layer is shown in Figure 6.

![Figure 5. Structure of YOLOV3-SOD network.](image-url)
3.3. Object Matching

A moving object is defined as an object with motion attributes, so the motion detection and object detection results must overlap for the image. We use intersection over union (IoU) to unify the two sets of detection results. The bounding box for motion detection is assumed to be $M$ and the bounding box for object detection is $D$, so the formula for calculating IoU is as follows.

$$IoU = \frac{\text{area}(M) \cap \text{area}(D)}{\text{area}(M) \cup \text{area}(D)}$$

(3)

The motion detection result is usually inaccurate and only the approximate area containing the moving object is provided. The object detection method can completely detect the target but it lacks motion information. Thus, we use IoU to add motion information for object detection, where the object detection box is used as the final output when the IoU value is greater than the threshold. Figure 7 shows an example of this process where there may be multiple objects in the image, including moving and non-moving objects. Suppose that there are $m$ objects in the image with $n$ moving objects, where $D_i$ is the object detection result for object $i$, $i \in (0, m]$, and $M_j$ is the motion detection result for moving object $j$, $j \in (0, n]$. For object $i$, Equation (4) can determine whether it is a moving object.

$$\text{Moving Object} = D_i \text{ if } \max \left(\text{IoU}(D_i, M_j)\right) > 0.2, j \in (0, n]$$

(4)

As the robustness of the motion detection module is not high, the motion information may be insufficient or not detected at all when the illumination changes, occlusion and appearance changes. Although the object detection module can accurately detect the objects, but the target cannot be
regarded as a moving object because of the lack of motion information. Therefore, we combine the motion detection module with the object detection module, however, the low recall rate of motion detection module limits the performance of the joint algorithm. Thus, we introduce inter-frame object-matching to recall the missing detections. The moving objects exhibit temporal and spatial continuity in the continuous image sequence, i.e., moving objects will not disappear suddenly in a short time and their positions will not change suddenly. Therefore, the bounding boxes of the same target will overlap greatly between adjacent frames, so it is possible to determine whether the two bounding boxes belong to the same object by calculating the IoU. In order to detect missing detections, we calculate the IoU for the bounding box detected by YOLOv3-SOD in the current frame and the bounding box detected in the previous frame. If the value of IoU is greater than 0.5, then the bounding box detected by YOLOv3 is used as the final detection result. In addition, to consider the situation where a moving object stops in the background, we count the number of frames with the lost detections in the motion detection module and the target is considered to have stopped moving when a certain threshold is satisfied.

Figure 7. Method for determining a moving object.

4. Experiments and Results

In the following, we first introduce our MDR105 dataset. We compared the results obtained using our method and other moving-object detection algorithms based on this dataset. All of the algorithms considered in this experiment were tested on an Intel Core i7-7700 CPU and NVIDIA GeForce RTX 2080Ti GPU. The proposed algorithm is implemented by Python language, using Tensorflow library, OpenCV library, Numpy library and CUDA.

4.1. Dataset

At present, a dataset is not available specifically for detecting moving objects in dynamic backgrounds because most of the datasets have a static background or they are used as target tracking datasets. Thus, we collected videos acquired by UAV onboard cameras and surveillance cameras to produce a new dataset for moving object detection and recognition called MDR105. This dataset comprises 105 videos belonging to three categories, where each has 35 videos; there may be multiple categories in a video. The size of each image is 640 × 360 pixels and the number of video frames ranges from 120 to 2700. All images are taken from the UAVs or ground in a distance and we only labeled moving objects in the videos. Figure 8 shows some representative images from the dataset.
In order to test the performance of moving object detection algorithms in various situations, MDR105 covers different scenarios, such as single and multiple objects, simple and complex backgrounds, static backgrounds, and moving backgrounds. In addition, several videos contain both moving and static objects for comparison. The videos in this dataset are all of actual scenes, thereby making it a highly valuable and representative dataset for moving object detection and recognition. Table 2 shows detailed statistical information for the MDR105 dataset. Figure 9a shows a histogram illustrating the sizes of the moving objects; many objects are smaller than 50 pixels in size, which demonstrates that the dataset contains many small targets. The IoU between the moving target bounding boxes in adjacent frames is used as the velocity of motion, and Figure 9b shows a histogram illustrating the velocities of motion.

**Table 2.** Detailed information for the MDR105 dataset.

| Class Name | Image Number | Object Number | Object Size (Pixel) |
|------------|--------------|---------------|---------------------|
|            |              |               | Max | Min | Average |
| person     | 11836        | 13393         | 225 | 18  | 70       |
| car        | 17714        | 21530         | 201 | 16  | 58       |
| aeroplane  | 11520        | 12827         | 406 | 16  | 97       |

**Figure 9.** (a) Histogram illustrating the sizes of the moving objects. (b) Histogram illustrating the velocities of motion.
4.2. Evaluation Metrics

In this study, we used the precision (P), recall (R), and F1 measure to evaluate the performance of the moving object detection algorithm. The mean average precision (mAP) was used to evaluate the performance of the object detection model. The detection results were divided into four cases according to the real category and the detection category: actual positive samples were recognized as true positive (TP), incorrect positive samples were recognized as false positive (FP), incorrect negative samples were recognized as false negative (FN), and real negative samples were recognized as true negative (TN). According to these definitions, the precision and recall can be expressed as follows.

\[ P = \frac{TP}{TP + FP}, \quad R = \frac{TP}{TP + FN} \]  

(5)

The F1 measure was used to reflect the performance of the algorithms by comprehensively considering the precision and recall, as follows.

\[ F1 = \frac{2PR}{P + R} \]  

(6)

In the object detection task, the average precision (AP) was used to measure whether the model’s performance was good or bad for a given category. The mAP was calculated as the average value of all the APs to measure the model’s performance in all categories. We used an 11-point interpolation method to calculate the AP, as follows.

\[ AP = \frac{1}{11} \sum_{r \in \{0, 0.1, \ldots, 1\}} P_{\text{interp}}(r), \quad P_{\text{interp}}(r) = \max_{\tilde{r} \geq r} p(\tilde{r}) \]  

(7)

4.3. Experimental Results and Comparisons

For the purpose of detecting moving targets in the video, our algorithm contains three modules: motion detection, object detection and object matching. In this section, we verified the performance of each module. Section 4.3.1 discussed the effect of motion detection module. Section 4.3.2 evaluated the performance of object detection module. In Section 4.3.3, we combined the motion detection module, object detection module and object matching module as our proposed algorithm, then compared with each module and other algorithms on the MDR105 dataset. In addition, we also studied the effect of moving object size and velocity on algorithm performance.

4.3.1. Results on Motion Detection Module

Before conducting comparisons with other algorithms, we determined the threshold \( \delta \) used for background compensation, which may affect the detection performance. Figure 10 shows the binarization mask and detection box obtained by the motion detection module under different frame difference thresholds. In the figure, we can see that when the threshold is very low, the result contains a lot of noise, which may be considered as a moving object. However, when the threshold is too high, moving objects are not detected, resulting in a low recall rate of the algorithm. Therefore, it is necessary to choose an appropriate threshold.

Figure 11 shows the P, R, and F1 values for motion detection and our overall algorithm under different thresholds of \( \delta \). In the experiment, when the IoU between the bounding box for motion detection and the ground truth was greater than 0.3, the detection result was considered a TP. The experimental results showed that low values of \( \delta \) resulted in low R values for motion detection and low R values for the overall algorithm. When the value of \( \delta \) was excessively large, many edges were incorrectly detected and the precision rate decreased. Finally, we selected \( \delta = 19 \).
4.3.2. Results on Object Detection Module

We evaluated the performance of the YOLOv3-SOD model. In order to verify the performance of the object detection model for a motion scene, we have collected some images from other datasets and the Internet, most of which were taken from drones and distant places. There are 7684 train images in total, which also contain three categories—aeroplane, car, and person—and these three classes are evenly distributed. In addition, we selected 1900 images from the MDR105 dataset for testing. The test images were uniformly obtained from 105 videos, including different motion patterns, and there was no intersection between test images and train images. We manually labeled 9606 images; in contrast to the MDR105 dataset, all objects of interest in the images were annotated regardless of whether the objects were moving or not. This dataset is also available at the link above.

The algorithms compared with YOLOv3-SOD comprised YOLOv3, TinyYOLOv3, SSD, and FPN, where SSD and FPN are also specifically designed for detecting small targets. All algorithms were trained on the above dataset. During training, for data enhancement and prevention of overfitting, we set the parameter angle to 30, which means that each images will be randomly rotated by an angle between $[-30^\circ, +30^\circ]$. In addition, we set the initial learning rate to 0.001, the momentum to 0.9, the decay to 0.0005, and the batch to 32. Our model was trained for a total of 32000 iterations, and the final loss is 0.96763. We evaluated the performance of the models in terms of their accuracy, speed, and model size. The final results are shown in Table 3.
Table 3. Comparison results of object detection models.

| Method      | P   | R   | F1  | AP (%) | mAP (%) | FPS | Size (MB) |
|-------------|-----|-----|-----|--------|---------|-----|-----------|
| TinyYOLOv3  | 0.97| 0.44| 0.70| 85     | 58      | 65  | 42        |
| YOLOv3      | 0.96| 0.82| 0.89| 90     | 90      | 90  | 24        |
| SSD         | 0.96| 0.58| 0.77| 90     | 80      | 80  | 60        |
| FPN         | 0.65| 0.74| 0.69| 82     | 83      | 78  | 2         |
| YOLOv3-SOD  | 0.96| 0.86| 0.91| 91     | 91      | 91  | 27        |

For our algorithm, the recall was 0.86 and the F1 index was 0.91, which was the highest value. The mAP value was similar to that produced by YOLOv3, where both achieved state-of-the-art performance. However, the model size was more than twice as large for YOLOv3 compared with our model and the detection speed was slower. The size of our model was only 98 MB, but its accuracy was much higher compared with SSD and TinyYOLOv3. Our model obtained the best detection performance.

Our model is optimized for small objects, so we determined the actual effects of optimization. We divided the objects into different levels according to the size of the bounding box and compared the recall values for several algorithms at different size levels. Table 4 shows the results obtained. Compared with the other algorithms, YOLOv3-SOD had the highest recall at less than 75 pixels but its detection performance for larger objects was not as good as TinyYOLOv3, YOLOv3, and SSD. However, the size of moving object under UAV perspective is small, and our optimization for small objects is successful.

Table 4. Recall with objects of different sizes.

| Method      | <25 | 25–50 | 50–75 | 75–100 | >100 |
|-------------|-----|-------|-------|--------|------|
| TinyYOLOv3  | 0.00| 0.31  | 0.82  | 0.77   | 0.89 |
| YOLOv3      | 0.86| 0.90  | 0.85  | 0.87   | 0.89 |
| SSD         | 0.10| 0.53  | 0.83  | 0.91   | 0.89 |
| FPN         | 0.76| 0.84  | 0.82  | 0.68   | 0.49 |
| YOLOv3-SOD  | 0.89| 0.90  | 0.89  | 0.80   | 0.66 |

4.3.3. Comparison Results on the MDR105 Dataset

The proposed algorithm contains three modules: motion detection module, object detection module and object matching module. We first evaluated the effects of these modules and then compare them with the ECO and Deep_Sort algorithms. Deep_Sort was trained with the same dataset for fairness of comparison. The results obtained based on the MDR105 dataset are shown in Table 5. In the “Method” column, “MD Module” denotes motion detection module based on the background compensation method. “OD Module” denotes object detection module using YOLOv3-SOD, the model uses the same weights as in Section 4.3.2. “OM Module” denotes object matching module. When multiple “4” appear in the table, it means that these modules are used in combination. In addition, we tested the performance with each video and then calculated the average value as the final performance in order to reflect the comprehensive performance in various situations.
Table 5. Comparison of different algorithms based on the MDR105 dataset.

| Method       | MD Module | OD Module | OM Module | P   | R   | F1   |
|--------------|-----------|-----------|-----------|-----|-----|------|
| Proposed     | 4         | 4         | 4         | 0.60| 0.75| 0.66 |
| method       | 4         | 4         | 4         | 0.81| 0.86| 0.83 |
| ECO          | 4         | 4         | 4         | 0.95| 0.80| 0.86 |
| Deep_Sort    | 4         | 4         | 4         | 0.96| 0.85| 0.90 |
|              | 0.86      | 0.73      | 0.79      |     |     |      |
| Deep_Sort    | 0.82      | 0.87      | 0.85      |     |     |      |

As shown in Table 5, the motion detection module obtained low P and R values, only 0.60 and 0.75. The object detection module has a high recall rate, but it cannot distinguish whether the target has been moved or not, so its P value is only 0.81. As can be seen from the Rows 3, the P values greatly improved after integrating object detection module and motion detection module, which indicates that our strategy of combining these two methods is correct. However, the problem of missing detections was not solved, so the R value was still very low; compared with the Rows 4, the R value increased by 5% after introducing the object matching module, demonstrating its effectiveness. ECO obtained good accuracy at single-target tracking, but MDR105 contains many multi-target movement and intermittent movement situations, so the overall performance of ECO was poor. Deep is a multi-target tracking algorithm; its recall rate is very high, but it cannot distinguish whether the object is moving or not, so the precision is not high.

Figure 12 shows the detection results of several algorithms on the MDR105 dataset. The motion detection module can detect the moving object in the image, but its detection is not accurate and contains a lot of noise. ECO is a single-target tracking algorithm and cannot determine whether the tracked target has stopped moving. The Deep_Sort algorithm is a multi-target tracking algorithm, which detects all objects in the image but cannot detect whether the object has moved. In conclusion, our method achieved state-of-the-art performance and it was more effective at moving object detection than the other methods.

![Detection results of several algorithms based on the MDR105 dataset.](image)

Figure 12. Detection results of several algorithms based on the MDR105 dataset. In the ground truth, green represents a moving object and red represents an object that is not moving.

Table 6 shows the recall rate obtained for the algorithms with moving targets of different sizes. YOLOv3-SOD generally achieved the highest scores at various size levels, thereby demonstrating its performance. However, from Rows 3, 4, we can see that the recall rate has dropped significantly after the object detection module is combined with the motion detection module. The obvious reason is that
the motion detection module reduces the recall rate of the entire algorithm. These findings show that we need to improve the capacity of the motion detection module.

Table 6. Recall values for objects of different sizes on the MDR105 dataset.

| Method       | Recall at Different Size of Bounding Box |
|--------------|-----------------------------------------|
|              | MD  | OD  | OM  | <25 | 25–50 | 50–75 | 75–100 | 100–125 | >125 |
| Proposed     | 4   | 4   | 4   | 0.44 | 0.80 | 0.85 | 0.72 | 0.84 | 0.86 |
| method       | 4   | 4   | 0.93 | 0.91 | 0.87 | 0.90 | 0.95 | 0.88 |
| Deep_Sort    | 4   | 4   | 0.77 | 0.85 | 0.89 | 0.84 | 0.75 | 0.72 |
| ECO          | 4   | 4   | 0.79 | 0.85 | 0.91 | 0.88 | 0.87 | 0.83 |

We also explored how the algorithms performed at different velocities of motion and the results are shown in Figure 13. The velocity was defined as the IoU for the same object’s boundary box in two adjacent frames. As the velocity of motion increased, the IoU decreased correspondingly. The results showed that the recall decreased for all methods as the velocity of motion increased. Clearly, when the velocity of motion increased, the variations in the background amplitude were excessively large and the background compensation results could be incorrect. The target tracking algorithms such as ECO and Deep_sort lost the objects because their spatial positions changed greatly. However, the same phenomenon also affected the object detection model in YOLOv3-SOD, possibly due to motion blur caused by the high velocity of motion, although the specific reason for this requires further study.

Figure 13. Recall values for objects with different velocities. “MD Module” denotes motion detection module. “OD Module” denotes object detection module. “MD+OD Module” refers to the method of combining motion detection and object detection. The proposed method combines three modules.

5. Conclusions

In this study, we introduced a new method for detecting and recognizing moving objects. First, our method uses the background compensation method to detect moving regions and this method is then combined with the lightweight YOLOv3-SOD network with a high capacity for detecting small targets, thereby allowing the positions of moving objects to be accurately detected. Finally, object
matching between frames is employed to recall any lost detections. We developed the MDR105 dataset to verify the performance of our proposed algorithm. The MDR105 dataset contains numerous videos and rich scenes, and it is highly valuable for studies of moving object detection. Our experiments showed that the proposed method obtained a better performance compared with other methods and it could accurately detect moving targets in various scenarios.

However, our algorithm still has some shortcomings. In particular, the performance of the motion detection module is not adequate, which restricts the accuracy and speed of our algorithm. We might consider using better-performing algorithms to obtain motion information and to establish an end-to-end network for detecting and recognizing moving objects.

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

**Funding:** This research was funded by the National Natural Science Foundation of China (NSFC) General Program, grant number 61673353.

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

**References**

1. Lee, S.H.; Kwon, S.C.; Shim, J.W.; Lim, J.E.; Yoo, J. WisenetMD: Motion Detection Using Dynamic Background Region Analysis. *Symmetry* 2019, 11, 621. [CrossRef]
2. Ahmadi, M.; Ouarda, W.; Alimi, A.M. Efficient and Fast Objects Detection Technique for Intelligent Video Surveillance Using Transfer Learning and Fine-Tuning. *Arab. J. Sci. Eng.* 2020, 45, 1421–1433. [CrossRef]
3. Genci, C. A vision-based approach for intelligent robot navigation. *Int. J. Intel!. Syst. Technol. Appl.* 2010, 9, 97–107.
4. Tsai, D.M.; Tseng, T.H. A template reconstruction scheme for moving object detection from a mobile robot. *Ind. Robot* 2013, 40, 559–573. [CrossRef]
5. Yin, G.; Li, Y.; Zhang, J. The Research of Video Tracking System Based on Virtual Reality. In Proceedings of the 2008 International Conference on Internet Computing in Science and Engineering, Harbin, China, 28–29 January 2008; pp. 122–127.
6. Singh, A.; Dutta, M.K. Imperceptible watermarking for security of fundus images in tele-ophthalmology applications and computer-aided diagnosis of retina diseases. *Int. J. Med. Inform.* 2017, 108, 110–124. [CrossRef] [PubMed]
7. Cho, J.; Jung, Y.; Kim, D.; Lee, S.; Jung, Y. Design of Moving Object Detector Based on Modified GMM Algorithm for UAV Collision Avoidance. *J. Semicond. Technol. Sci.* 2018, 18, 491–499. [CrossRef]
8. Lin, C.Y.; Muchtar, K.; Lin, W.Y.; Jian, Z.Y. Moving Object Detection Through Image Bit-Planes Representation Without Thresholding. *IEEE Trans. Intell. Transp. Syst.* 2020, 21, 1404–1414. [CrossRef]
9. Shin, J.; Kim, S.; Kang, S.; Lee, S.W.; Paik, J.; Abidi, B.; Abidi, M. Optical flow-based real-time object tracking using non-prior training active feature model. *Real-Time Imaging* 2005, 11, 204–218. [CrossRef]
10. Fan, X.; Cheng, Y.; Qiang, F. Moving Target Detection Algorithm Based on Susan Edge Detection and Frame Difference. In Proceedings of the International Conference on Information Science and Control Engineering, Shanghai, China, 24–26 April 2015; pp. 323–326.
11. Jarraya, S.K.; Hammami, M.; Hanène, B.-A. Accurate Background Modeling for Moving Object Detection in a Dynamic Scene. In Proceedings of the International Conference on Digital Image Computing: Techniques and Applications, Sydney, Australia, 1–3 December 2010; pp. 52–57.
12. Wang, P.; Yang, J.; Mao, Z.; Zhang, C.; Zhang, G. Object Detection Based on Motion Vector Compensation in Dynamic Background. *Ordnance Ind. Autom.* 2019, 38, 6–10.
13. Rodriguez, P.; Wohlberg, B. Translational and rotational jitter invariant incremental principal component pursuit for video background modeling. In Proceedings of the 2015 IEEE International Conference on Image Processing (ICIP), Quebec City, QC, Canada, 27–30 September 2015; pp. 537–541.
14. Suhr, J.K.; Jung, H.G.; Li, G.; Nob, S.I.; Kim, J. Background Compensation for Pan-Tilt-Zoom Cameras Using 1-D Feature Matching and Outlier Rejection. *IEEE Trans. Circuits Syst. Video Technol.* 2011, 21, 371–377. [CrossRef]

15. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. *arXiv* 2015, arXiv:1506.02640.

16. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* 2018, arXiv:1804.02767.

17. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intel.* 2017, 39, 1137–1149. [CrossRef] [PubMed]

18. Yin, X.; Wang, B.; Li, W.; Liu, Y.; Zhang, M. Background Subtraction for Moving Cameras based on trajectory-controlled segmentation and Label Inference. *KSII Trans. Internet Inf. Syst.* 2015, 9, 4092–4107.

19. Brox, T.; Malik, J. Object Segmentation by Long Term Analysis of Point Trajectories. In Proceedings of the 11th European Conference on Computer Vision, Heraklion, Greece, 5–11 September 2010; pp. 282–295.

20. Singh, S.; Arora, C.; Jawahar, C.V. Trajectory Aligned Features For First Person Action Recognition. *Pattern Recognit.* 2017, 62, 45–55. [CrossRef]

21. Comaniciu, D.; Ramesh, V.; Meer, P. Kernel-based object tracking. *Pattern Anal. Mach. Intell.* 2003, 25, 564–575. [CrossRef]

22. Hyeonseob, N.; Bohyung, H. Learning Multi-Domain Convolutional Neural Networks for Visual Tracking. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 4293–4302.

23. Martin, D.; Goutam, B.; Fahad, S.K.; Michael, F. ECO: Efficient Convolution Operators for Tracking. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6931–6939.

24. Raychaudhuri, A.; Maity, S.; Chakraborti, A.; Bhattacharjee, D. Detection of Moving Objects in Video Using Block-Based Approach. In Proceedings of the International Conference on Computational Intelligence, Communications, and Business Analytics, Kolkata, India, 24 March 2017; pp. 151–167.

25. Nozari, A.; Hoseini, S.M. Recognition of Moving Objects in Videos of Moving Camera with Harris Attributes. In Proceedings of the 2015 Fourteenth Mexican International Conference on Artificial Intelligence (MICAI), Cuernavaca, Mexico, 25–31 October 2015; pp. 42–45.

26. Guang, L.I.; Yan, F. Moving object detection based on SIFT features matching and K-means clustering. *J. Comput. Appl.* 2012, 32, 2824–2826.

27. Luo; Hao, W. A Moving Object Detection Algorithm Based on ORB under Dynamic Scene. *Appl. Mech. Mater.* 2014, 602–605, 1638–1641. [CrossRef]

28. Hafiane, A.; Palaniappan, K.; Seetharaman, G. UAV-Video Registration Using Block-Based Features. In Proceedings of the 2008 IEEE International Geoscience and Remote Sensing Symposium, Boston, MA, USA, 7–11 July 2008; pp. 1104–1107.

29. Setyawati, F.A.; Tan, J.K.; Kim, H.; Ishikawa, S. Detection of Moving Objects in a Video Captured by a Moving Camera Using Error Reduction. In Proceedings of the SICE Annual Conference, Hokkaido University, Sapporo, Japan, 9–12 September 2014; pp. 347–352.

30. Nicolai, W.; Alex, B.; Dietrich, P. Simple Online and Realtime Tracking with a Deep Association Metric. In Proceedings of the 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 17–20 September 2017; pp. 3645–3649.

31. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the 2016 European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; pp. 21–37.

32. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.

33. Lin, T.Y.; Dollar, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. *arXiv* 2017, arXiv:1612.03144.

34. Rosten, E.; Drummond, T. Machine learning for high-speed corner detection. In Proceedings of the 9th European Conference on Computer Vision, Graz, Austria, 7–13 May 2006; pp. 430–443.

35. Bay, H.; Tuytelaars, T.; Gool, L.V. SURF: Speeded up robust features. In Proceedings of the 9th European Conference on Computer Vision, Graz, Austria, 7–13 May 2006; pp. 404–417.
36. Fischler, M.A.; Bolles, R.C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* **1981**, *24*, 381–395. [CrossRef]

37. Fu, C.; Liu, W.; Ranga, A.; Tyagi, A.; Berg, A.C. DSSD: Deconvolutional Single Shot Detector. *arXiv* **2017**, arXiv:1701.06659.

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).