Small Flying Object Detection and Tracking in Digital Airport Tower through Spatial-Temporal ConvNets

Phat Thai (thai0009@e.ntu.edu.sg)
Nanyang Technological University
Sameer Alam (sameeralam@ntu.edu.sg)
Nanyang Technological University
Nimrod Lilith (nimrod.lilith@ntu.edu.sg)
Nanyang Technological University
Binh Nguyen (ngtbinh@hcmus.edu.vn)
Ho Chi Minh City University of Science

Research Article

Keywords:

DOI: https://doi.org/

Additional Declarations: No competing interests reported.

Small Flying Object Detection and Tracking in Digital Airport Tower through Spatial-Temporal ConvNets

Phat Thai[1,*], Sameer Alam[1,*], Nimrod Lilith[1], and Binh Nguyen[2]

[1]Saab-NTU Joint Lab, School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore
[2]Faculty of Mathematics and Computer Science, VNUHCM-University of Science, Ho Chi Minh City, Vietnam
[*]thai0009@e.ntu.edu.sg, sameeralam@ntu.edu.sg

ABSTRACT

Digital towers using high-resolution cameras that cover a 360-degree view of airports have recently been applied as a solution for some airports by replacing conventional towers. Although many computer vision systems have been developed as tools to assist tower controllers, small flying object detection remains challenging due to their small dimensions and unpredictable trajectories. This paper proposes a novel computer vision framework to detect, track and recognize small flying objects, namely aircraft and drones, in an airport environment. The framework creates a new Convolutional Neural Network which adapts to the unique characteristics of small flying objects. It also exploits the spatial-temporal information, as well as post-processing, to improve the performance. The proposed framework is validated on an airport dataset and Drone-vs-Bird public dataset. The results show that the framework can not only perform object detection in real-time, but also surpass the performance of state-of-the-art models in both datasets by a large margin.

1 Introduction

The digital tower concept has been developed as an alternative to conventional towers[1] for airport management. By using a network of high-resolution cameras that cover a 360-degree view of an airport, the digital tower can provide many advantages over a conventional tower[2]. To further enhance its capabilities, many computer vision systems[3–9] were developed to automatically detect and track airport objects. Since airport airside is a wide field of view environment with different types of vehicles (speed and dimension), these systems only focus on a particular application such as detecting aircraft on runway[4], tracking moving aircraft[3] or detecting objects in parking zone[7]. To our best knowledge, detecting and tracking small flying objects, especially drones, remains challenging due to small dimension. In this work, we propose a framework to detect, track and recognize small flying objects including drones and aircraft in videos captured by digital tower by using the state-of-the-art computer vision technique of Convolutional Neural Networks (ConvNets).

Unmanned aerial vehicles (UAVs), also known as drones, have recently been used in various fields such as wireless communications, security and surveillance, precision agriculture and civil infrastructure inspection due to their low maintenance cost and high mobility[10]. Great benefits could be realised if UAV operations are successfully integrated into civilian airspace[11,12]. Unfortunately, fully integrating Unmanned Aerial System (UAS) operations into a National Airspace System (NAS) is a long and uncertain road, because of the lack of a regulatory framework that allows UAVs to fly in civil airspace. Extensive UAS regulations can be found in the report[13]. The basic functional requirement is that UAVs must have the capability to detect, avoid and maintain a minimum distance between each other and to aircraft[14], either by built-in sensors or third party systems, such as cameras[15] and ground radars[16]. On the other hand, the number of unauthorized drones flying in airspace has increased significantly, resulting in flight disruptions to avoid potential collisions[17]. There were 404 drone sightings in the vicinity of major airports in the US alone between October and December 2018[17]. In particular, drone sightings causing huge costs have occurred at Dubai International Airport in 2016, Gatwick International Airport in 2018, Changi International Airport in 2019 and Newark Liberty International Airport in 2019, to name a few[17,18].

There have been many previous works[19–25] using different techniques applied to the detection of drones and birds. Video object detection in earlier works involved object detection in individual frames. By improving object detection performance on separate frames, video object detection performance is also improved[19,20,23]. However, detection on individual frames does not consider video characteristics. First, the relationship between spatial and temporal information is not exploited. Second, due to technical issues, some frames might suffer motion blur, noise or occlusion, resulting in low performance. Therefore, video object detection approaches exploit spatio-temporal information to address the above challenges[21,24,25]. Our framework improves detection performance in three ways. First, we use multiple frames as an input to exploit spatial-temporal

Figure 1. Leesburg Executive airport is captured by 1080p cameras. There are samples from seven cameras (over 14 cameras).

information. Since drone object sizes can be small, color features might not provide useful information. Instead, by stacking three consecutive gray images, a detection model can exploit temporal information with the same computational requirements as one RGB image. The model output is a bounding box, derived from the union of the three bounding boxes of the set of three consecutive frames. By doing this we increase the object sizes, which enables the model to detect objects more easily. Second, we create a new ConvNet which adapts to the unique characteristics of small flying objects. This customized network outperforms the state-of-the-art ConvNets. Third, a tracking algorithm is investigated, not only to track detected objects, but also to reduce false positive (false alarm) and false negative (miss detection) rates. In this algorithm, detected objects in current frames which do not link to previous frames and detected objects in previous frames which cannot be detected in current frames are re-detected. In addition, a majority vote algorithm is applied to adjust object classes if necessary.

The framework is validated on an airport dataset and the Drone-vs-Bird public dataset. The results show that the framework not only can perform small flying object detection in real-time, but also surpass the state-of-the-art models by a large margin.

The rest of this paper is organized as follows. In Section 2, a brief review of previous related work is presented, and video data is described in Section 3. Section 4 presents the proposed framework, with details of the experiment design outlined in Section 5. Results and comparison to other techniques is presented in Section 6. Finally, the paper concludes with summary and discussions in Section 7.

2 Related Work

2.1 Image Object Detection

ConvNets for image object detection can be clustered into one-stage approaches and two-stage approaches.

Two-stage approaches, which were first introduced by R-CNN[26], include two separate modules which are the region proposal module and feature extraction module. The region proposal module uses an external algorithm, such as selective search[27], to propose typically 300 potential objects from an input image. Features, which are extracted from each proposal object by the feature extraction module, are classified into specific classes. R-CNN requires huge computational resources, because the network extracts features 300 times per image. To reduce computation, Faster R-CNN[28] introduces a Region Proposal Network to propose potential objects on feature maps which are extracted by feature extraction from an input image.

One-stage approaches[29,30] directly locate and classify objects on feature maps by taking a dense grid of bounding boxes, which are typically from $10,000$ to $100,000$ depending on target object sizes and input resolution. One-stage approaches detect objects faster than two-stage approaches, but achieve lower detection performance because of a high imbalance between objects and background. Fortunately, by introducing focal loss[31] to deal with this imbalance, one-stage approaches can outperform two-stage approaches. With further improvement one-stage approaches, such as Yolov4[32] and EfficientDet[33], have become the state-of-the-art models for object detection.

2.2 Video Object Detection

The major difference between image and video object detection is temporal information. By exploiting spatio-temporal information through multiple frames, features in current input frames will be enhanced, resulting in an increase in detection performance.

Previous works[34,35] combine features extracted at current frames with optical flow from previous frames. Optical flow is a method of estimating object motion between frames, which is calculated more rapidly than feature extraction. Hence, this method can either be used to speed up detection by replacing feature maps with flow field in some frames[34], or to improve detection accuracy by aggregating feature maps and optical field in every frame[35]. However, optical flow data is extremely difficult to obtain, and motion prediction is required to perform object detection.

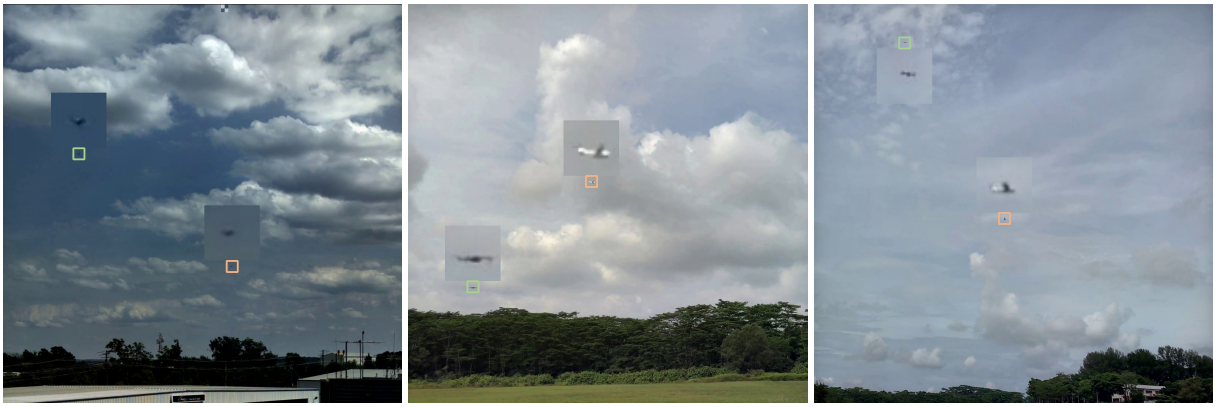

Figure 2. Blended frames of drones (green) and aircraft (orange). **Left:** A blended drone on digital tower frame. **Middle:** A blended aircraft on recorded frame. **Right:** A blended drone and a blended aircraft on a background frame.

Alternatively, temporal information is integrated with spatial information, which is extracted by ConvNets by a deformable convolutional network[36], recurrent neural network[37], or correlation tracker[38]. Recently, transformer model[39] is a prominent candidate for extracting temporal-spatial information due to an ability to model flexible frame sizes.

Several approaches focus on post-processing. Seq-NMS[40] uses bounding boxes of detected objects from previous frames to re-score detected objects in current frames. The authors in[41] propose a similarity function to link detected objects cross frames.

2.3 Drone Detection

Compared to general video object detection, drone detection has unique challenges. First, the object size is small, which makes spatial information not reliable. Second, drone trajectories are unpredictable because they can fly freely in every direction in 3D spaces with wide ranges of velocity and acceleration. Third, drones are captured by high definition cameras, typically 1920×1080 resolution, which can compromise real-time detection by the use of current the state-of-the-art ConvNets[32,33]. Therefore, ConvNets are modified to these specific challenges.

An initial approach for drone detection is to adopt state-of-the-art ConvNets, including SSD[19], Faster R-CNN[42], Yolov2[43], Yolov4[20] and more recently Yolov5[44,45]. Because these models are large and complex, synthetic data are required and the real-time constraint is removed. Moreover, since image object detection relies only on spatial information, the detection performance on small object size may not be sufficient. To improve the result, a super-resolution approach is explored[23]. Although performance increases, the model takes three seconds to detect one frame using an NVIDIA GeForce TITAN XP, which might be unacceptable for most applications.

To exploit temporal information, the detection is divided into two stages, which are motion extraction and object classification. First, motion is extracted by using Background Subtraction[21,22] or ConvNet-based networks[24,25]. Background Subtraction extracts the motion by calculating the background-foreground difference between consecutive frames. Since Background Subtraction ignore spatial information, it cannot distinguish drones when they stop moving or fly near other move objects. Moreover, moving background increases the number of detected motion which causes an increase in the classification time and the number of false positives. On the other hand, ConvNet-based networks apply convolutional operations on an input created by staking τ consecutive frames. Because more frames cost more computation, the model size is reduced[25] or gray scale frames are used instead of color (RGB) frames[24]. After the training process, these networks aim to extract motion from target objects, which reduces workload for the next stage. The extracted motion is fed to classification models to classify as specific objects including drones, birds, aircraft and noise. These extra models are either a customized ConvNet[21] or the state-of-the-art models[22,24,25]. Customized ConvNets normally detect faster than the state-of-the-art ConvNets since they create for particular requirements, but model optimization requires ton of knowledge and time.

Based on the above-mentioned literature, our approach tackles the challenges by many directions. We use stacking grayscaled frames to exploit temporal information without extra computational cost, which is similar to the work in[24]. However, we unify two stages into one customized ConvNet to adapt to the specific requirements. On top of that, post-processing is investigated to further improve performance.

3 Datasets and Data Engineering

As we focus on digital tower environment, our main dataset is obtained from a digital tower at Leesburg Airport. In addition, we use a public dataset, which is the Drone-vs-Bird Detection Challenge[46] for comparison.

Figure 3. Sample frames extracted from Drone-vs-Bird Dataset. The dataset is collected from multiple sources with different drone models, dimensions and background conditions.

3.1 Leesburg Executive Airport Dataset

The videos of flying aircraft are obtained from the digital tower installation at Leesburg Executive Airport in USA over one hour. The tower includes 14 cameras and each camera captured a fixed angle of the airport. In the videos, aircraft could be seen flying, taking off and landing as shown in Figure 1. These videos are used for small flying object detection where the pixel dimensions are from 10×7 to 28×20 which occupied on average 0.01% of the whole frame. Since drones are prohibited to fly in the airport, we record drone videos ourselves. The videos are recorded such that the drone sizes are similar to aircraft. We then blend drones into aircraft videos to create a dataset where aircraft and drones are on the same field of view. The videos are generated by Generative Adversarial Networks[47] to make the videos realistic. For data augmentation purpose, we also blend aircraft to the recorded drone videos, and blend aircraft and drones to different backgrounds as shown in Fig. 2

We select 20 1080×1080 videos, each with a frame rate of 30 frames per second. The dataset is divided into a training set of 18 videos and a testing set of 2 videos. Next, we extract frames from videos and manually label them with bounding boxes. As a result, we compose a dataset with 10000 training images and 1000 testing images.

3.2 Drone-vs-Bird Dataset

Drone-vs-Bird videos are obtained from different sources with high variability including complex backgrounds, different weather conditions or direct sun glare as shown in Figure 3. We use videos collected before the Drone-vs-Bird Detection Challenge 2019 for training, and after the Drone-vs-Bird Detection Challenge 2019 for testing. This data selection increases the detection difficulty, because of the difference between the training and testing video sets characteristics. Moreover, The datasets only have one class, drone, which means other flying objects, including birds and aircraft, are noise. Since we develop the framework for a digital tower environment, we only select static videos, which results in 39 training videos and 8 testing videos. Next, we extract frames from the videos, resulting in $26,201$ training frames and $6,944$ testing frames.

4 Methodology

Figure 4 illustrates the diagram of the proposed framework. We customize a ConvNet for the special requirements of the detection of small flying objects captured by static high resolution cameras. Next, by stacking multiple frames as an input, the ConvNet detector can exploit the temporal information, which increases the performance. The detected objects are not only tracked through videos, but also post-processed with the support of previous results by a tracking algorithm.

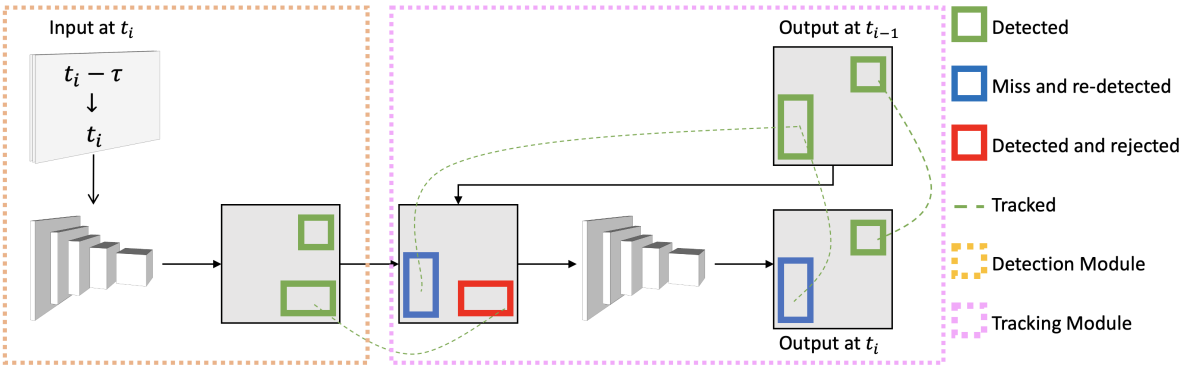

Figure 4. The diagram of the proposed framework. The ConvNet detects objects from multiple frames to exploit the temporal information. The tracking algorithm which stores previous results not only tracks detected objects but also processes current results.

4.1 Object Detection

The AirNet model was developed in previous work[48], and was specifically adapted to provide detection of aircraft and ground vehicles in airport environments. To detect small flying objects, we create a small version of AirNet, called SAirNet, as shown in Figure 5. The SAirNet architecture is described as follows.

- Convolutional operations. Similar to the AirNet, we mainly utilize depthwise convolutional operations to prioritise computational time. Standard convolutional operations are sometimes applied to learn cross-channel information.
- Backbone. The backbone consists of 4 identical blocks and a first convolutional layer which normalizes an input image $(W \times W \times 3)$ to a target shape $(W/2 \times W/2 \times C)$. Similar to the AirNet framework, the input size will be reduced by a factor of 2 and the number of channels will increase by k after passing each block.
- The number of feature scales. As SAirNet is designed to detect small objects, we build SAirNet with 3 feature map scales to cover the range of object sizes.
- Customize anchors. By using multiple frames as an input, the ratios between width and height of objects are unusual as shown in Figure 6. Therefore, we implement a grid search to find the most suitable anchors for the model.

The detector's input is τ consecutive frames and the output is bounding boxes which contain objects from these τ frames, as shown in Figure 7. By using multiple frames as an input, the detector can exploit the temporal information. By using crossing-frame bounding boxes, the detected targets become bigger, as shown in Figure 6 and Figure 7. However, increasing τ also increases the ratio between the width and height of the bounding boxes, which requires more anchors to cover. Therefore, we choose $\tau = 3$ to balance the object sizes and object ratios. Moreover, by choosing $\tau = 3$ and using gray images, there is no extra computational cost between 3-gray-frame input and 1-color-frame input. The object bounding box is reconstructed as an overlap of two detector bounding boxes, as shown in Figure 7.

4.2 Object Tracking

The detected objects are tracked by tracking-by-detection[49]. Moreover, the tracking algorithm also works as post-processing to improve detection performance. First, we initialize a list of objects (R) which aims to track detected objects through video. Each member has three attributes, which are location (x_1, y_1, x_2, y_2), probability of specific classes (aircraft or drone) and the last time of detection (t_0). At the time t, we calculate intersection over union (IOU) between objects in R and detected objects at time t (R^t) by Eq.(1). Two objects are considered to be the same object if their IOU is greater than 0 and the detected time difference $(t - t_0)$ is smaller than a threshold parameter. The value of this threshold parameter is dependent on the performance of the detection models. The value is small with high performance detection or large with low performance detection. There are three different outputs for this stage.

$$IOU(r_1, r_2) = \frac{Area(r_1) \cap Area(r_2)}{Area(r_1) \cup Area(r_2)} \tag{1}$$

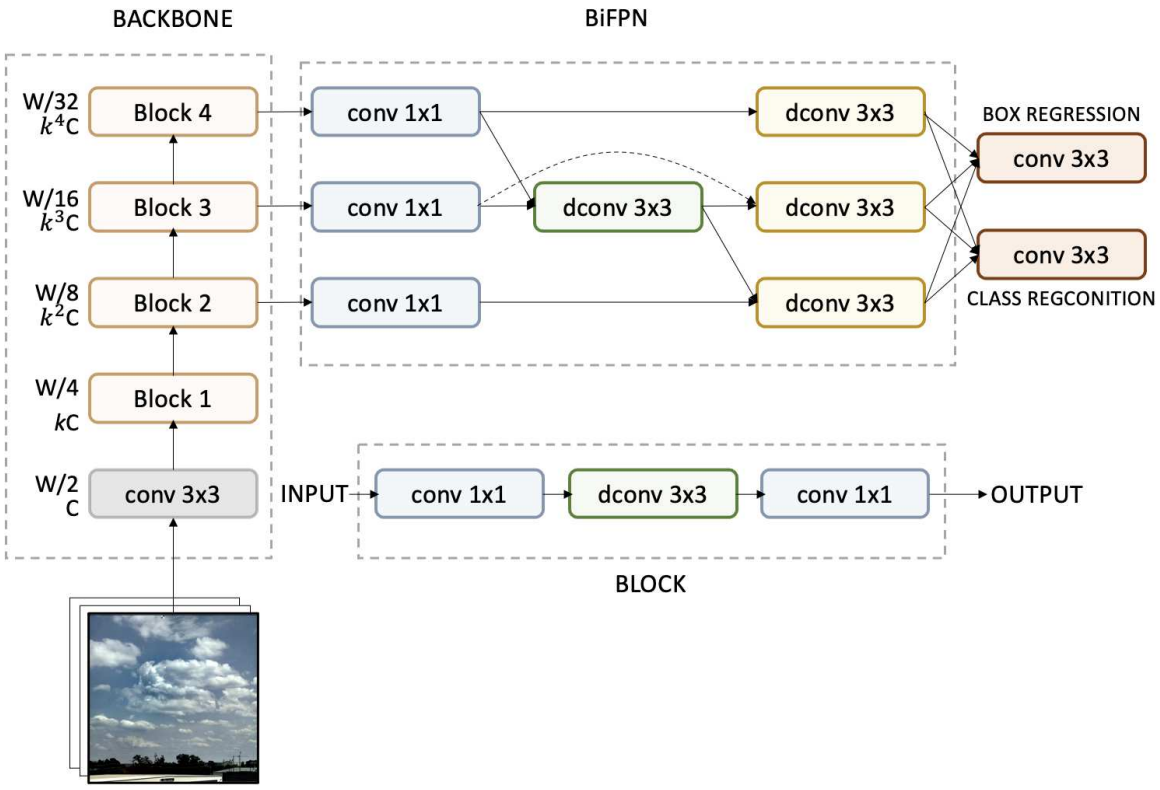

Figure 5. The SAirNet architecture. A backbone extracts features from an input image, creating feature maps with three scales. Then the feature maps are refined by BiFPN. Finally, the network detects objects including bounding box regression and class recognition.

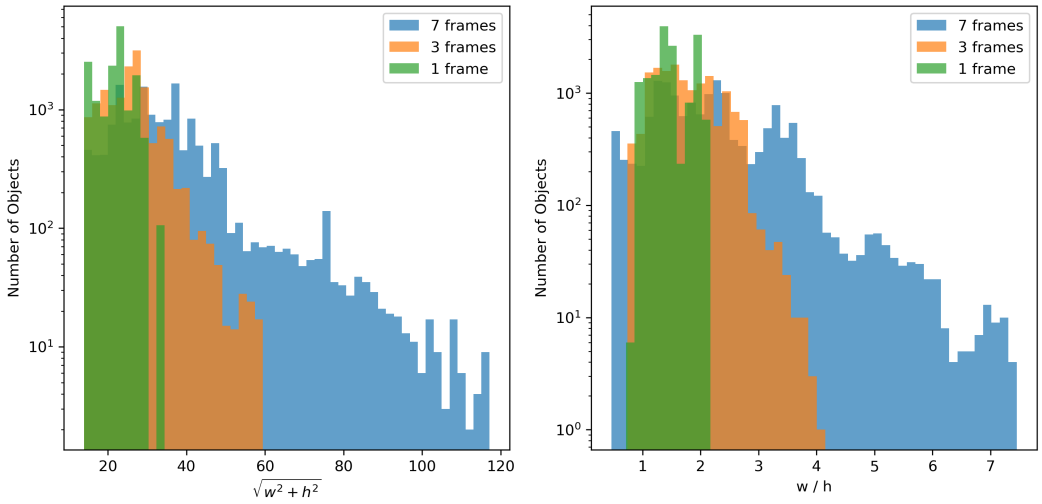

Figure 6. Distribution of bounding boxes sizes (left) and ratios (right) of the dataset with different number of frames. τ-frame bounding box is a bounding box which contain an object cross τ consecutive frames. Increasing number of frames increases not only the size but also the ratio.

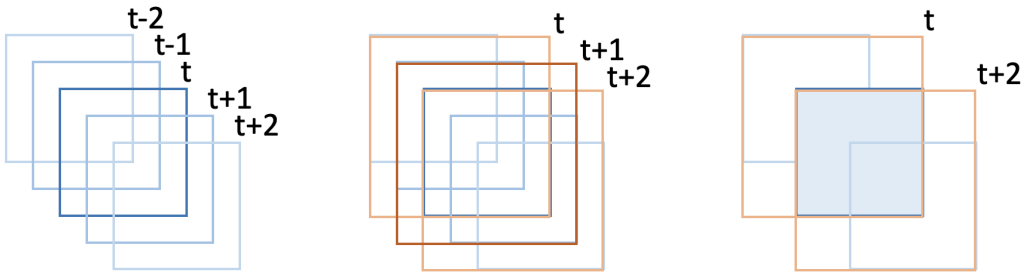

Figure 7. The detector bounding box (orange) is an union of object bounding boxes (blue) from three consecutive frames. The object bounding box is an overlap of two detector bounding boxes.

- **Case 1.** An object belongs to both R and R^t. This means the model successfully detects the object over time. Therefore, its attributes from R are updated according to R^t.
- **Case 2.** An object belongs to R but not R^t. This mean that the object detected in previous frames cannot be detected in the current frame (t). The reason for this occurrence might due to image blur, object occlusion or if the object is too small. Therefore, we focus to re-detect the object. We crop an image from the current frame which is extended from the object location (x_1, y_1, x_2, y_2) at time $t-1$ to ensure the image captures the object. To encourage the detection, we increase the cropped image size and lower the confidence threshold. If the object still cannot be detected, we repeat the process on the Δt next frames. Large Δt will increase chance of detection, but also increases computation cost. If the object is detected, we update it to R similar case 1. If the object is not detected over the Δt frames, we ignore the object.
- **Case 3.** An object belongs to R^t but not R. This means the object is newly detected, which also means there is a chance of it being a false detection. Therefore, similar to case 2, we re-detect it. The process is similar to case 2, except we raise the confidence threshold to discourage the detection. If the object is still detected, we update it to R as a new object.

5 Experiments

The SAirNet models are trained on Leesburg airport and Drone-vs-Bird data. The training process is performed on a computer with 4 Nvidia Quadro RTX 6000 GPUs, but the test process only uses 1 GPU to report the average running time per image. The hyper-parameters for SAirNet are input resolution (W), number of channels (C) and their scale (k), and the number of layers. The image resolutions are resized randomly, with a scale ranging from 1 to 2, to improve small object detection performance. However, doubling image resolution of 1080×1920 with factor 2 requires a huge computational cost. Therefore, we reduce image resolution to 512×512 by randomly cropping to speed up the training process. The training process is implemented as follows. First, we initialize an Adam optimizer[50] with a learning rate of 10^{-4} and batch size of 64. Next, the network is trained over 20 epochs with flexible image resolution from 512 to 1024. Then, we reduce the learning rate to 10^{-5} and train the network with the default input resolution until the network loss no longer decreases.

We implement Yolov4 as a baseline model on the Leesburg airport dataset with similar experimental setup mentioned in the paper[32]. However, due to hardware constraints, the batch size is 32. The SAirNet models are also compared with the best performing models from the Drone-vs-Bird detection challenge 2021[46]. The validation metric is average precision (AP) with IOU of 0.5, which is used for the Drone-vs-Bird detection challenge[46].

6 Results

The Table 1 and Figure 8 show the results of SAirNet with different settings on the Leesburg airport dataset. By reducing number of feature scales and applying depthwise convolutional operations, the models still achieve real-time detection requirement with high input resolution. Compare to Yolov4 at input resolution of 1024, SAirNet detects three times faster. By increasing the number of parameters and input resolutions, SAirNet models outperform Yolov4 by a high margin. Although the tracking algorithm improves detection performance, it requires more computational time than expanding the model size. However, it still applicable to improve the detection performance when increasing model size is not an option due to hardware constraints or overfitting.

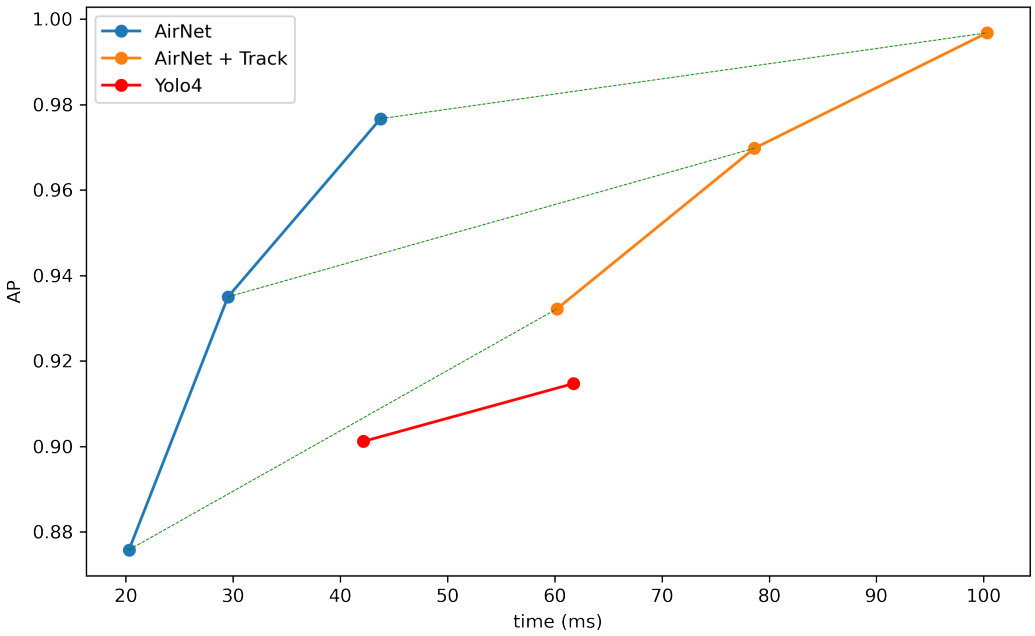

Figure 8. Average precision and corresponding running time on the Leesburg airport dataset. Tracking with post-process algorithm increases performance but also increases running time.

Model	Res	Param	Time	AP	$AP_{aircraft}$	AP_{drone}
Yolov4	896	27.6M	42 ms	90.01%	84.78%	95.45%
Yolov4	1024	27.6M	62 ms	91.47%	84.01%	98.94%
SAirNet	1024	144K	20 ms	87.58%	86.58%	88.58%
SAirNet	1152	478K	29 ms	93.50%	94.63%	92.36%
SAirNet	1280	1.6M	43 ms	97.67%	96.21%	99.13%
SAirNet + Track	1024	144K	60 ms	93.22%	96.51%	89.94%
SAirNet + Track	1152	478K	78 ms	96.98%	99.27%	94.70%
SAirNet + Track	1280	1.6M	100 ms	99.68%	99.62%	99.74%

Table 1. The experimental results, including the input resolution, number of parameters, running time and average precision, from different settings for Leesburg airport dataset.

Video	SAirNet	SAirNet Track	OBSS	CARG	R-CNN	RetinaNet
GOPR5843_004	88.4%	99.1%	95.3%	62.8%	**100%**	98.9%
GOPR5847_001	88.0%	**96.9%**	66.3%	62.4%	69.3%	73.6%
GOPR5853_002	**94.9%**	89.7%	61.8%	28.6%	60.6%	47.0%
GOPR5868_001	95.5%	90.0%	95.1%	**100%**	98.9%	91.9%
C0001_27_46_solo	88.9%	**92.6%**	81.0%	46.3%	85.5%	87.3%
C0001_57_00_inspire	**96.7%**	94.3%	87.3%	73.6%	89.8%	73.7%
C0001_11_23_inspire	87.2%	**89.2%**	10.0%	08.8%	77.9%	78.8%
C0006_split	05.3%	**22.9%**	00.0%	0.00%	01.4%	08.0%
Overall	80.6%	**85.0%**	60.5%	47.8%	72.9%	67.6%

Table 2. The comparison for different models in the Drone-vs-Bird dataset. The AP is reported for each video in test set. OBSS and CARG are the winning entries of the Drone-vs-Bird Detection Challenge 2021. R-CNN and RetinaNet are the baselines from the organization.

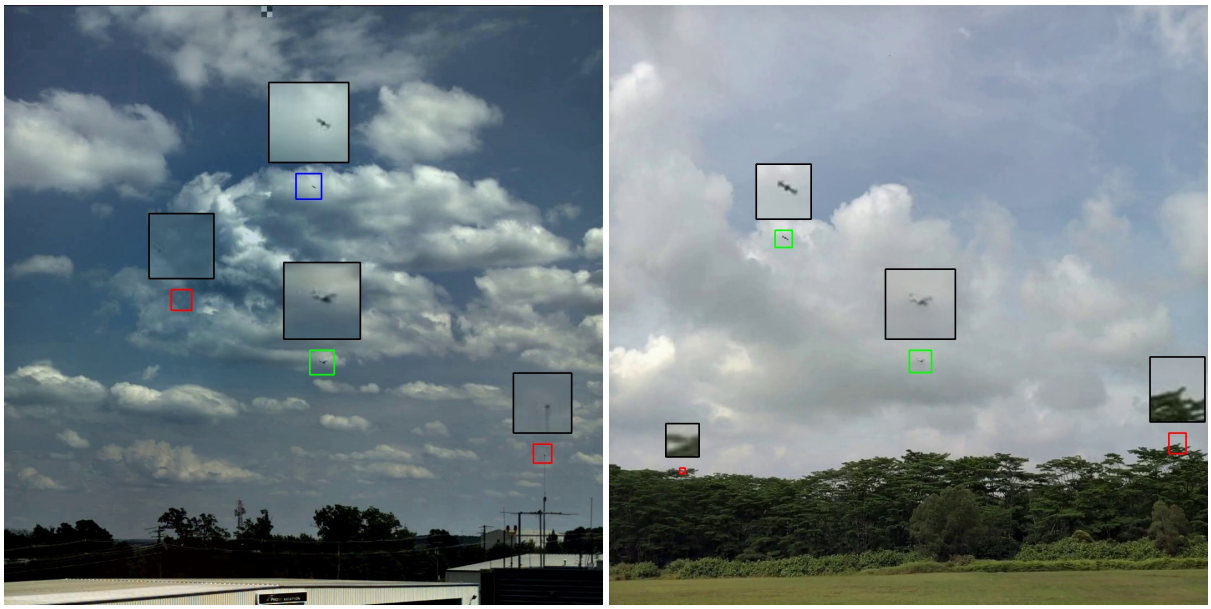

Figure 9. Samples of SAirNet detection results from different videos in the Leesburg dataset. The green, blue and red boxes indicate Case 1, Case 2 and Case 3 (described in Section 4.2), respectively.

Figure 10. Samples of SAirNet detection results from different videos in the Drone-vs-Bird dataset. The green, blue and red boxes indicate Case 1, Case 2 and Case 3 (described in Section 4.2), respectively.

Figure 11. Samples of SAirNet detection results from C0001_11_23_inspire_1m (left) and c0006_split_01_01 (right), which are the most challenging videos in the Drone-vs-Bird test set. The green, blue and red boxes indicate Case 1, Case 2 and Case 3 (described in Section 4.2), respectively.

Figure 12. Samples of color images (top) and motion images (bottom), which are an input of SAirNet. Static objects are monochrome, whereas moving objects are chromatic.

The Table 2 shows the comparison between our models and the best models of Drone-vs-Bird detection challenge 2021. SAirNet achieves the highest overall performance. Moreover, the overall performance is further improved with the inclusion of the tracking algorithm.

Figure 9 and Figure 10 show the SAirNet detection samples. The green boxes indicate objects detected by SAirNet and accepted by the tracking algorithm (Case 1 described in Subsection 4.2). The model can detect small objects or occluded objects thanks to spatio-temporal information. The red boxes indicate objects detected by SAirNet but rejected by tracking algorithm (Case 2 described in Subsection 4.2) while the blue boxes indicate objects which cannot be detected by SAirNet but get re-detected by the tracking algorithm (Case 3 described in Subsection 4.2). The tracking algorithm can reject slow and occasionally moving objects, such as water waves, tree leaves or clouds, which improves the detection performance. However, the tracking algorithm cannot reject constantly moving objects including humans, aircraft, animals or tree leaves under high wind, which reduces detection performance.

Figure 11 shows the SAirNet detection samples from C0001_11_23_inspire_1m and C0006_split_01_01, which are the most difficult videos in the Drone-vs-Bird test set. For the C0001_11_23_inspire_1m video, drones are detected with difficulty due to the presence of occlusion. However, SAirNet easily overcomes this problem by motion information, as shown in the left of the Figure 12. Moreover, birds are another challenge which can cause many false positives. SAirNet only rejects slow-moving birds, but falsely detects fast-moving birds. C0006_split_01_01 is the most difficult video in the test set, in which every model failed to detect the drone. In this video a drone which is similar in color to a light traffic pillar descends vertically. As shown in the right of the Figure 12, there is rarely any motion sight since the drone moves very slowly. However, by tracking and post-processing, SAirNet can improve the detection performance, particularly when the drone is separated from the pillar or when it accelerates.

7 Conclusion

Small UAVs are required to be detected for the integration of UAV operations, or the prevention of drone incursion at airports. Advancement in digital tower and computer vision have offered new approaches to meet this requirement. This paper proposes a computer vision framework to detect, track and recognize small flying objects by exploiting spatial-temporal information with a state-of-the-art computer vision technique. The framework demonstrates high performance for the Leesburg airport dataset with 99.69% *AP*. Moreover, the framework outperforms the best algorithms of the public detection challenge by a large margin.

For future work, the framework will be extended to estimate the geographical location of detected objects. However, 3D position estimation is difficult to estimate by using only one camera. One possible solution is to mount multiple cameras at multiple locations, such as gates, fences or other static infrastructure.

Data Availability

There are two datasets in the study which are Leesburg airport dataset and drone-vs-bird dataset.

- The Leesburg airport videos in the current study are not publicly available because they are confidential videos from a private company but are available from the corresponding author on reasonable request.
- The Drone-vs-Bird videos that support the findings of this study are available from **International workshop on small-drone surveillance, detection and counteraction techniques** but restrictions apply to the availability of these data, which were used under license for the current study, and so are not publicly available. Data are however available from the authors upon reasonable request and with permission of **International workshop on small-drone surveillance, detection and counteraction techniques**. They can be reached at wosdetc@googlegroups.com.

References

1. Fürstenau, N. Virtual and remote control tower. *Switzerland: Springer* (2016).
2. Frequentis. White paper: Introduction to remote virtual tower (2018). https://www.frequentis.com/sites/default/files/support/2018-02/RVT_whitepaper.pdf.
3. Besada, J. A. *et al.* Airport surface surveillance based on video images. *IEEE transactions on aerospace electronic systems* **41**, 1075–1082 (2005).
4. Pavlidou, N. *et al.* Using Intelligent Digital Cameras to Monitor Aerodrome Surface Traffic. *IEEE Intell. Syst.* **20**, 76–81, DOI: 10.1109/MIS.2005.56 (2005).
5. Aguilera, J. *et al.* Visual surveillance for airport monitoring applications. In *11th Computer Vision Winter Workshop 2006*, 6–8 (2006).

6. Koutsia, A., Semertzidis, T., Dimitropoulos, K., Grammalidis, N. & Georgouleas, K. Automated visual traffic monitoring and surveillance through a network of distributed units. In *ISPRS* (Citeseer, 2008).

7. Bloisi, D., Iocchi, L., Nardi, D., Fiorini, M. & Graziano, G. Ground traffic surveillance system for Air Traffic control. In *2012 12th International Conference on ITS Telecommunications*, 135–139, DOI: 10.1109/ITST.2012.6425151 (IEEE, Taipei, Taiwan, 2012).

8. Lu, H.-L., Vaddi, S., Cheng, V. & Tsai, J. Airport gate operation monitoring using computer vision techniques. In *16th AIAA Aviation Technology, Integration, and Operations Conference*, 3912 (2016).

9. Vidakis, D. G. & Kosmopoulos, D. I. Facilitation of air traffic control via optical character recognition-based aircraft registration number extraction. *IET Intell. Transp. Syst.* **12**, 965–975 (2018).

10. Shakhatreh, H. *et al.* Unmanned aerial vehicles (uavs): A survey on civil applications and key research challenges. *Ieee Access* **7**, 48572–48634 (2019).

11. Sridhar, B. & Kopardekar, P. Towards autonomous aviation operations: What can we learn from other areas of automation? In *16th AIAA Aviation Technology, Integration, and Operations Conference*, 3148 (2016).

12. Dalamagkidis, K., Valavanis, K. P. & Piegl, L. A. *On integrating unmanned aircraft systems into the national airspace system: issues, challenges, operational restrictions, certification, and recommendations*, vol. 54 (springer science & Business Media, 2011).

13. Dalamagkidis, K., Valavanis, K. P. & Piegl, L. A. *On integrating unmanned aircraft systems into the national airspace system: issues, challenges, operational restrictions, certification, and recommendations*, vol. 54 (springer science & Business Media, 2011).

14. Wikle, J. K., McLain, T. W., Beard, R. W. & Sahawneh, L. R. Minimum required detection range for detect and avoid of unmanned aircraft systems. *J. Aerosp. Inf. Syst.* 351–372 (2017).

15. Agrawal, P., Ratnoo, A. & Ghose, D. Image segmentation-based unmanned aerial vehicle safe navigation. *J. Aerosp. Inf. Syst.* 391–410 (2017).

16. Sahawneh, L. R. *et al.* Ground-based sense-and-avoid system for small unmanned aircraft. *J. Aerosp. Inf. Syst.* **15**, 501–517 (2018).

17. Mott, J. H., Marshall, Z. A., Vandehey, M. A., May, M. & Bullock, D. M. Detection of conflicts between ads-b-equipped aircraft and unmanned aerial systems. *Transp. research record* **2674**, 197–204 (2020).

18. Nguyen, P. *et al.* Towards rf-based localization of a drone and its controller. In *Proceedings of the 5th workshop on micro aerial vehicle networks, systems, and applications*, 21–26 (2019).

19. Park, J., Kim, D. H., Shin, Y. S. & Lee, S.-h. A comparison of convolutional object detectors for real-time drone tracking using a ptz camera. In *2017 17th International Conference on Control, Automation and Systems (ICCAS)*, 696–699 (IEEE, 2017).

20. Singha, S. & Aydin, B. Automated drone detection using yolov4. *Drones* **5**, 95 (2021).

21. Schumann, A., Sommer, L., Klatte, J., Schuchert, T. & Beyerer, J. Deep cross-domain flying object classification for robust uav detection. In *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 1–6 (IEEE, 2017).

22. Seidaliyeva, U., Akhmetov, D., Ilipbayeva, L. & Matson, E. T. Real-time and accurate drone detection in a video with a static background. *Sensors* **20**, 3856 (2020).

23. Magoulianitis, V., Ataloglou, D., Dimou, A., Zarpalas, D. & Daras, P. Does deep super-resolution enhance uav detection? In *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 1–6 (IEEE, 2019).

24. Craye, C. & Ardjoune, S. Spatio-temporal semantic segmentation for drone detection. In *2019 16th IEEE International conference on advanced video and signal based surveillance (AVSS)*, 1–5 (IEEE, 2019).

25. Rozantsev, A., Lepetit, V. & Fua, P. Detecting flying objects using a single moving camera. *IEEE transactions on pattern analysis machine intelligence* **39**, 879–892 (2017).

26. Girshick, R., Donahue, J., Darrell, T. & Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 580–587 (2014).

27. Uijlings, J. R., Van De Sande, K. E., Gevers, T. & Smeulders, A. W. Selective search for object recognition. *Int. journal computer vision* **104**, 154–171 (2013).

28. Ren, S., He, K., Girshick, R. & Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, 91–99 (2015).

29. Redmon, J., Divvala, S., Girshick, R. & Farhadi, A. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 779–788 (2016).

30. Liu, W. *et al.* Ssd: Single shot multibox detector. In *European conference on computer vision*, 21–37 (Springer, 2016).

31. Lin, T.-Y., Goyal, P., Girshick, R., He, K. & Dollár, P. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, 2980–2988 (2017).

32. Bochkovskiy, A., Wang, C.-Y. & Liao, H.-Y. M. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934* (2020).

33. Tan, M., Pang, R. & Le, Q. V. Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10781–10790 (2020).

34. Li, M., Sun, L. & Huo, Q. Dff-den: Deep feature flow with detail enhancement network for hand segmentation in depth video. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, 1548–1552 (IEEE, 2018).

35. Zhu, X., Wang, Y., Dai, J., Yuan, L. & Wei, Y. Flow-guided feature aggregation for video object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, 408–417 (2017).

36. Bertasius, G., Torresani, L. & Shi, J. Object detection in video with spatiotemporal sampling networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 331–346 (2018).

37. Xiao, F. & Lee, Y. J. Video object detection with an aligned spatial-temporal memory. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 485–501 (2018).

38. Feichtenhofer, C., Pinz, A. & Zisserman, A. Detect to track and track to detect. In *Proceedings of the IEEE International Conference on Computer Vision*, 3038–3046 (2017).

39. Yang, H., Guo, L., Zhang, Y. & Wu, X. U-shaped spatial–temporal transformer network for 3d human pose estimation. *Mach. Vis. Appl.* **33**, 1–16 (2022).

40. Han, W. *et al.* Seq-nms for video object detection. *arXiv preprint arXiv:1602.08465* (2016).

41. Sabater, A., Montesano, L. & Murillo, A. C. Robust and efficient post-processing for video object detection. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 10536–10542 (IEEE, 2020).

42. Chen, Y., Aggarwal, P., Choi, J. & Kuo, C.-C. J. A deep learning approach to drone monitoring. *arXiv preprint arXiv:1712.00863* (2017).

43. Aker, C. & Kalkan, S. Using deep networks for drone detection. *arXiv preprint arXiv:1706.05726* (2017).

44. Akyon, F. C., Eryuksel, O., Ozfuttu, K. A. & Altinuc, S. O. Track boosting and synthetic data aided drone detection. In *2021 17th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 1–5 (IEEE, 2021).

45. Dadboud, F., Patel, V., Mehta, V., Bolic, M. & Mantegh, I. Single-stage uav detection and classification with yolov5: Mosaic data augmentation and panet. In *2021 17th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 1–8 (IEEE, 2021).

46. Coluccia, A. *et al.* Drone-vs-bird detection challenge at ieee avss2021. In *2021 17th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 1–8 (IEEE, 2021).

47. Wu, H., Zheng, S., Zhang, J. & Huang, K. Gp-gan: Towards realistic high-resolution image blending. In *Proceedings of the 27th ACM international conference on multimedia*, 2487–2495 (2019).

48. Thai, P., Alam, S., Lilith, N. & Nguyen, B. T. A computer vision framework using convolutional neural networks for airport-airside surveillance. *Transp. Res. Part C: Emerg. Technol.* **137**, 103590 (2022).

49. Oron, S., Bar-Hillel, A. & Avidan, S. Real-time tracking-with-detection for coping with viewpoint change. *Mach. Vis. Appl.* **26**, 507–518 (2015).

50. Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).