This ACCV 2022 paper, provided here by the Computer Vision Foundation, is the author-created version.
The content of this paper is identical to the content of the officially published ACCV 2022
LNCS version of the paper as available on SpringerLink: https://link.springer.com/conference/accv

# AirBirds: A Large-scale Challenging Dataset for Bird Strike Prevention in Real-world Airports

Hongyu Sun, Yongcai Wang, Xudong Cai, Peng Wang, Zhe Huang, Deying Li, Yu Shao, and Shuo Wang

Renmin University of China, Beijing 100872, China
{sunhongyu,ycw,xudongcai,peng.wang,huangzhe21, deyingli,sy492019,shuowang18}@ruc.edu.cn

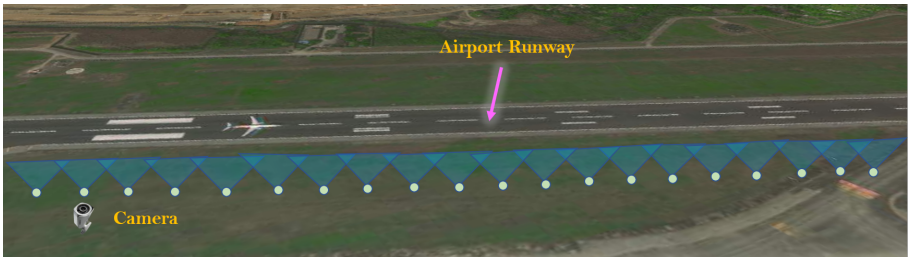
**Abstract.** One fundamental limitation to the research of bird strike prevention is the lack of a large-scale dataset taken directly from real-world airports. Existing relevant datasets are either small in size or not dedicated for this purpose. To advance the research and practical solutions for bird strike prevention, in this paper, we present a large-scale challenging dataset AirBirds that consists of 118,312 time-series images, where a total of 409,967 bounding boxes of flying birds are manually, carefully annotated. The average size of all annotated instances is smaller than 10 pixels in 1920×1080 images. Images in the dataset are captured over 4 seasons of a whole year by a network of cameras deployed at a real-world airport, covering diverse bird species, lighting conditions and 13 meteorological scenarios. To the best of our knowledge, it is the first large-scale image dataset that directly collects flying birds in real-world airports for bird strike prevention. This dataset is publicly available at https://airbirdsdata.github.io/.

**Keywords:** Large-scale Dataset · Bird Detection in Airport · Bird Strike Prevention.

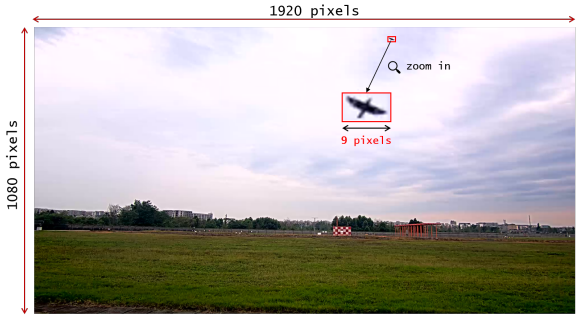
## 1 Introduction

Bird strike accidents cause not only financial debts but also human casualties. According to Federal Aviation Administration (FAA) [1], from 1990 to 2019, there have been more than 220 thousand wildlife strikes with civil aircraft in USA alone and 97% of all strikes involve birds. An estimated economic loss could be as high as $500 million per year. Furthermore, more than 200 human fatalities and 300 injuries attributed to bird strikes. Bird strikes happen most near or at airports during takeoff, landing and associated phrases. About 61% of bird strikes with civil aircraft occur during landing phases of flight (descent, approach and landing roll). 36% occur during takeoff run and climb[2]. It is the airspace that the airport should be responsible for so that the prevention of bird strikes is one of the most significant safety concerns. Although various systems are designed for preventing

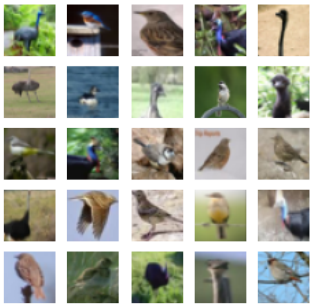
[1] https://www.faa.gov/airports/airport_safety/wildlife/faq/
[2] https://en.wikipedia.org/wiki/Bird_strike




(a) camera deployment alongside a runway in a real-world airport




(b) AirBirds



(c) Relevant Datasets

Fig. 1: (a) The deployment of a network of cameras in a real-world airport (b) A bird example in AirBirds (c) Examples of birds in CUB[27, 25], Birdsnap[1], NABirds[24] and CIFAR10[10].

bird strikes, accidents keep occurring with increasing commercial activities and flights. Improving the performances of bird strike prevention systems remains a research challenge. One fundamental limitation to the performances is the lack of large-scale data collected at real-world airports. On the one hand, real-world airports have strict rules on security and privacy regarding camera system deployment. On the other hand, it is inevitably expensive to develop a large-scale dataset that involves a series of time-consuming and laborious tasks.

Existing relevant datasets are either small in size or not dedicated for bird strike prevention. The wildlife strike database created by FAA provides valuable information, while each record in this database only contains a few fields in text form, such as date and time, aircraft and airport information, environment conditions, lacking informative pictures and videos. The relevant dataset developed by Yoshihashi *et al* aims at preventing birds from hitting the blades of turbines in a wind farm [29], rather than in real-world airports, and its size is less than one seventh of ours. Well-known datasets like ImageNet [5], COCO [14], VOC [6], CIFAR [10] collects millions of common objects and animals, including birds, but they are developed for the research of general image recognition, object detection and segmentation. Another branch of datasets, such as CUB series [27, 25], Birdsnap [1] and NABirds [24] containing hundreds of bird species, focus on

fine-grained categorization and part localization. And the size of these datasets is less than 50% of ours. One of the most significant differences between the above-mentioned datasets and ours is that birds in previous datasets are carefully selected and tailored, which means they are often centered in the image, occupy the main part of an image and have clear outlines, referring to Fig. 1c.

However, it is unlikely that birds in the images captured in real-world airports have these idealized characteristics. The deployment of a network of cameras around a runway in a real-world airport is shown in Fig. 1a. Each camera is responsible for monitoring an area of hundreds of meters so that flying birds that appear are tiny in size even in a high-resolution image. For example, in our dataset, the average size of all annotated birds is smaller than 10 pixels in the 1920×1080 images, taking up only ∼0.5% of the image width, shown in Fig. 1b.

To advance the research and practical solutions for bird strike prevention, we collaborate with a real-world airport for two years and finally present AirBirds, a large-scale challenging dataset consisting of 118,312 time-series images with 1920×1080 resolution and 409,967 bounding box annotations of flying birds. The images are extracted from videos recorded by a network of cameras over one year, from September 2020 to August 2021, thus cover various bird species in different seasons. Diverse scenarios are also included in AirBirds, *e.g.*, changing lighting and 13 meteorological conditions. Planning, deployment and joint commissioning of the monitoring system last for one year. The data collection process takes another whole year and subsequent cleaning, labeling, sorting and experimental analysis consume parallel 12 months. To the best of our knowledge, AirBirds is the first large-scale challenging image dataset that collects flying birds in real airports for bird strike prevention. The core contributions of this paper are summarized as follows.

- A large-scale dataset, namely AirBirds, that consists of 118,312 time-series images with 1920×1080 resolution containing flying birds in real-world airports is publicly presented, where there exist 409,967 instances with carefully manual bounding box annotations. The dataset covers various kinds of birds in 4 different seasons and diverse scenarios that include day and night, 13 meteorological and lighting conditions, *e.g.*, overcast, sunny, cloudy, rainy, windy, haze, etc.
- To reflect significant differences with other relevant datasets, we make comprehensive statistics on AirBirds and compare it with relevant datasets. There are three appealing features. (i) The images in AirBirds are dedicatedly taken from a real-world airport, which provide rare first-hand sources for the research of bird strike prevention. (ii) Abundant bird instances in different seasons and changing scenarios are also covered by AirBirds as the data collection spans a full year. (iii) The distribution of AirBirds is distinctive with existing datasets since 88% of instances are smaller than 10 pixels, and the remaining 12% are more than 10 and less than 50 pixels in 1920×1080 images.
- To understand the difficulty of AirBirds, a wide range of strong baselines are evaluated on this dataset for bird discovering. Specifically, 16 detectors

Table 1: Comparisons of AirBirds and relevant datasets. Density is the average instances in each image. Duration refers to the period of data collection.

| Dataset | Format | #Images | Resolution | #Instances | Density | Duration |
|---|---|---|---|---|---|---|
| FAA Database | text | - | - | 227,005 | - | 30 years |
| CUB-200-2010 [27] | image | 6,033 | ∼500×300 | 6,033 | 1.00 | - |
| CUB-200-2011 [25] | image | 11,788 | ∼500×300 | 11,788 | 1.00 | - |
| Birdsnap [1] | image | 49,829 | various | ∼49,829 | 1.00 | - |
| NABirds [24] | image | 48,562 | various | ∼48,562 | 1.00 | - |
| Wind Farm [29] | image | 16,200 | ∼5616×3744 | 32,000 | 1.97 | 3 days |
| VB100 [7] | video | - | ∼848×464 | 1,416 | - | - |
| **AirBirds** | image | 118,312 | 1920×1080 | 409,967 | 3.47 | 1 year |

are trained *from scratch* based on AirBirds with careful configurations and parameter optimization. The consistently unsatisfactory results reveal the non-trivial challenges of bird discovering and bird strike prevention in real-world airports, which deserve further investigation.

As far as we know, bird strike prevention remains a open research problem since it is not well solved by existing technologies. We believe AirBirds will benefit the researchers, facilitate the research field and push the boundary of practical solutions in real-world airports.

## 2 Related Work

In this section, we review the datasets that are either closely relevant to bird strike prevention or contain transferrable information to this topic.

**FAA Wildlife Strike Database**. One of the most relevant datasets is the Wildlife Strike Database[3] maintained by FAA. This database contains more than 220K records of reported wildlife strikes since 1990 and 97% of strikes attribute to birds. The detailed descriptions for each incident can be divided into the following parts: bird species, date and time, airport information, aircraft information, environment conditions, etc. An obvious limitation is the contents in this database are mainly in text form, lacking informative pictures and videos.

**Bird Dataset of a Wind Farm**. Yoshihashi *et al* develop this dataset for preventing birds striking the blades of the turbines in a wind farm [29]. 32,000 birds and 4,900 non-birds are annotated in total to conduct experiments of a two-class categorization. It is similar to us that the ratio of bird size and the image size is extremely small. However, compared to AirBirds' data collection process spanning a whole year, this dataset collects images only for 3 days so that the number of samples and scenarios are much less than those of AirBirds.

**Bird Datasets with Multiple Species**. Bird species probably provide valuable information for bird strike prevention. Another branch of the relevant

[3] https://wildlife.faa.gov

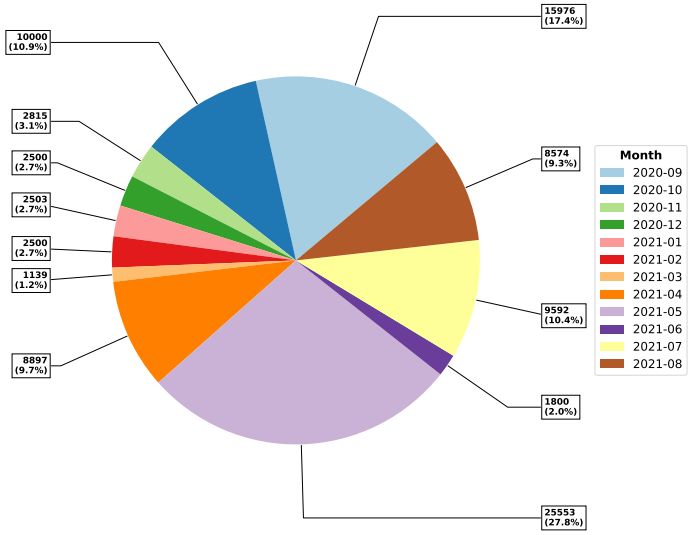



Fig. 2: The number of images per month in AirBirds.

datasets, such as CUB series [27, 25], Birdsnap [1], NABirds [24] and VB100 [7], focuses on fine-grained categorization of bird species. Images in these datasets are mainly collected from public sources, *e.g.*, Flickr [4], or by professionals. One of the most significant differences between the datasets in this branch and AirBirds is that birds in these datasets are carefully tailored, which means they are often centered in the image, occupy the main part and have clear outlines. However, it is unlikely for birds captured in real-world airports to have these wonderful characteristics. Moreover, bounding box annotations are absent in some of them, *e.g.*, VB100 [7], thus they are not suitable for the research of tiny bird detection.

**Well-Known Datasets Containing Birds**. Commonly used datasets in computer vision are also relevant as the `bird` belongs to one of the predefined categories in those datasets and there exist numbers of samples, such as ImageNet [5], COCO [14], VOC [6], CIFAR [10]. However, the above-mentioned datasets are dedicatedly designed for the research of general image classification, object detection and segmentation, not for bird strike prevention. And their data distributions differ from AirBirds, thus limited information can be transferred to this task.

The comparisons with related work are summarized in Tab. 1. AirBirds offers the most instances, the longest duration and the richest scenarios in image form.

[4] https://www.flickr.com/search/?text=bird

## 3 AirBirds Construction

This section describes the process of constructing the AirBirds dataset, including raw data collection, subsequent cleaning, annotation, splits and sorting to complete it.

### 3.1 Collection

To cover diverse scenarios and prepare adequate raw data, we decide to record in a real-world airport (Shuangliu International Airport, Sichuan Province, China) over 4 seasons of a whole year. The process of data collection starts from September 2020 and ends in August 2021.

Considering frequent takeoffs and landings, airport runways and their surroundings are major monitoring areas. We deployed a network of high-resolution cameras along the runways, as Fig. 1a shows. All deployed cameras use identical configurations. The camera brand is AXIS Q1798-LE[5], recording 1920×1080 images at a frame rate of 25. Due to the vast volume of raw data but a limited number of disks, it is infeasible to save all videos. We split into two parallel groups, one group for data collection and the other for data processing, so disk spaces can be recycled once the second group finishes data processing.

### 3.2 Preprocessing

This step aims to process raw videos month by month and save 1920×1080 images in chronological order. 25 frames per second in raw videos lead to numerous redundant images. To avoid dense distribution of similar scenarios, a suitable sampling strategy is required. One crucial observation is that the video clips where flying birds appear are very sparse compared to other clips. Hence, at first, we manually locate all clips where there exist birds, then sample one every 5 continuous frames in previously selected clips instead of all of them, resulting in an average of 300+ images per day, ∼10000 images per month, 118,312 in total. The number of images per month is shown in Fig. 2 and 13 meteorological conditions and the corresponding number of days are depicted in Fig. 3.

### 3.3 Annotation

To ensure quality and minimize costs, we divide the labeling process into three rounds. The first round that generates initial bounding box annotations for birds in the images is done by machines. The second round refines previous annotations manually by a team of employed workers. It should be noted that the team does not have to discover birds from scratch. In the third round, we are responsible for verifying those manual annotations and requiring further improvements of low-quality instances.

[5] https://www.axis.com/products/axis-q1798-le

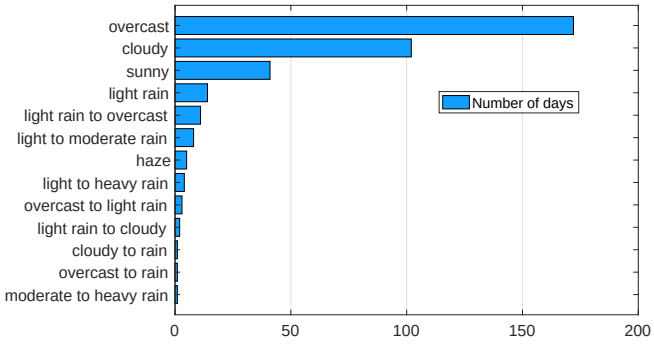



Fig. 3: The number of days of different weather in AirBirds.

It is not a simple task for humans to discover tiny birds from collected images with broad scenes. In the first round, we develop an algorithm for generating initial annotations and run on a computer. The idea of this algorithm is related to background subtraction in image processing. In our context, cameras are fixed in real-world airports, thus the background is static in the monitoring views. Since the images in each sequence are in chronological order, considering two consecutive frames, by computing the pixel differences between the first and second frame, the static part, namely the background, is removed while other moving targets, such as flying birds in the monitoring areas, are probably discovered. Alg. 1 describes the detailed process. Initially, we treat the first frame as background, convert it to gray mode, apply Gaussian blur[6] to this gray image, and denote the output image as $b$, then remove the first image from the input sequence $\mathcal{S}$. The set of initial bounding box annotations $\mathcal{B}$ is empty. Then we traverse the image $I_i$ in $\mathcal{S}$. In the loop, $I_i$ is also converted to gray image $g_i$. After that, Gaussian blur is applied to $g_i$ to generate a denoised image $c_i$. Then we compute differences between $b$ and $c_i$, resulting in $d$. Fourth, regions in $d$ whose pixel values are in the range of $[min, max]$ are considered as areas of interest, *e.g.*, if the pixel differences of the same area in those 2 consecutive frames are more than 30, there probably are moving targets in this area. The dilation operation is applied to those areas to expand contours for finding possible moving objects $\mathbf{c}_i$, including flying birds. After that, heuristic rules are used to filter candidates according to the object size, *e.g.*, big targets like airplanes, working vehicles, workers, are removed, resulting in $\mathbf{b}_i$. Then $\mathbf{b}_i$ is inserted into $\mathcal{B}$. Finally, we set background $b$ as $c_i$ and move forward. The key steps of this algorithm are visualized in Fig. 4.

Refinement is required since previously discovered moving objects are not necessarily birds. In the second round, we cooperate with a team of workers to accomplish the task. According to the predefined instructions, every single image should be zoomed in to 250+% to check the initial annotations in detail

[6] https://en.wikipedia.org/wiki/Gaussian_blur

**Algorithm 1** The First Round of Annotations

**Input**: $\mathcal{S} = \{I_1, I_2, \ldots, I_n\}$, an image sequence, where $n$ is the sequence length. Constants $min$ and $max$

**Output**: $\mathcal{B} = \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n\}$, the bounding boxes set for birds in $\mathcal{S}$, where $\mathbf{b}_i = \{b_i^1, b_i^2, \ldots, b_i^m\}$, $m$ is the number of detected birds in image $I_i$

$g \leftarrow$ `imRead`$(I_1, 0)$ ▷ read image in gray mode
$b \leftarrow$ `gaussBlur`$(g)$ ▷ denoise the image
$\mathcal{S} \leftarrow \mathcal{S} \setminus I_1$ ▷ remove $I_1$ from $\mathcal{S}$
$\mathcal{B} \leftarrow \emptyset$ ▷ initialize $B$
**for** $I_i$ in $\mathcal{S}$ **do**
    $g_i \leftarrow$ `imRead`$(I_i, 0)$
    $c_i \leftarrow$ `gaussBlur`$(g_i)$
    $d \leftarrow$ `Diff`$(b, c_i)$ ▷ compute differences
    $d \leftarrow$ `Thresh`$(d, min, max)$ ▷ apply threshold
    $d_i \leftarrow$ `Dilate`$(d)$ ▷ dilate areas further
    $\mathbf{c}_i \leftarrow$ `findContours`$(d_i)$ ▷ find candidates
    $\mathbf{b}_i \leftarrow$ `Filter`$(\mathbf{c}_i)$ ▷ filter candidates
    $\mathcal{B} \leftarrow \mathcal{B} \cup \mathbf{b}_i$ ▷ insert annotations
    $b \leftarrow c_i$ ▷ move background next
**end for**

and the team mainly handles 3 types of issues that arose in the first round (i) add missed annotations, (ii) delete false-positive annotations, (iii) update inaccurate annotations. In the third round, we go through the annotations refined by the team, requiring further improvements where inappropriate.

### 3.4 Splits

To facilitate further explorations of bird strike prevention based on this dataset, it is necessary to split AirBirds into training and test set.

We need to pay attention to three key aspects when splitting the dataset. First, we should keep a proper ratio between the size of the training and the test set. Second, it is essential to ensure training and test sets have a similar distribution. Third, considering the characteristic of chronological order, we should put a complete sequence into either the training or the test set rather than split it into different sets.

At last, we divide 98,312 images into the training set and keep the remaining 20,000 images in the test set, a nearly 5:1 ratio. All images and labels are publicly available, but excluding the labels in the test set. The validation set is not explicitly distinguished as the primary evaluation should take place on the test set, and users can customize the ratio between training and validation set individually. We are actively building an evaluation server and the labels in the test will be kept there.

In addition, the images in AirBirds can also be divided into 13 groups according to 13 kinds of scenarios shown in Fig. 3. This division is easy to achieve since each image is recorded on a specific day and each day corresponds to one

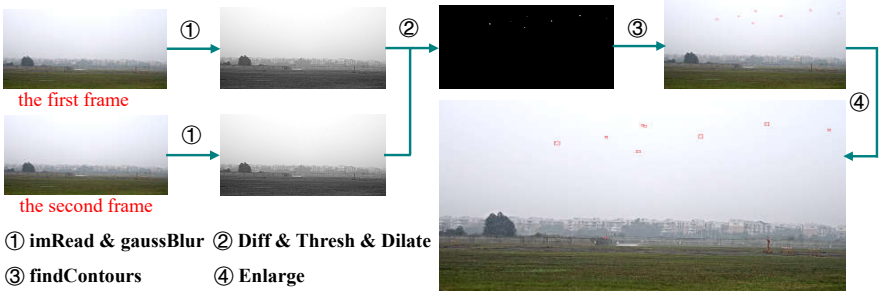



Fig. 4: Visualization of key steps in Alg. 1.

type of meteorological condition, according to the official weather report. Based on the division, we can evaluate the difficulty of bird discovering in different scenarios in real-world airports.

## 4 Experiments

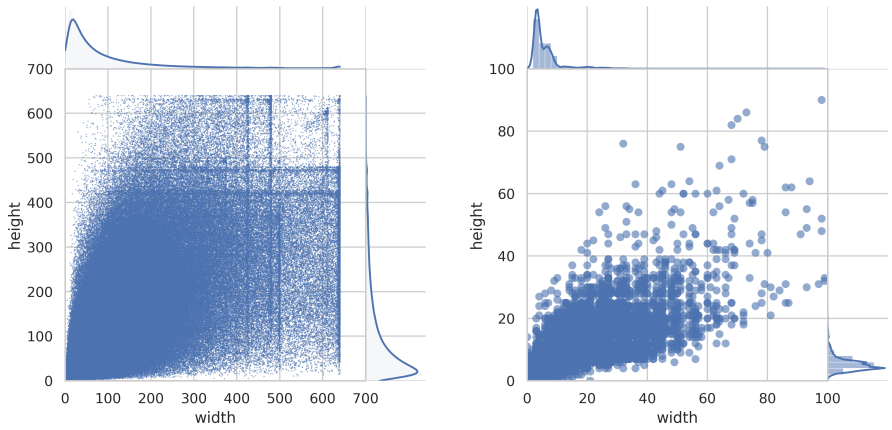
In this section, a series of comprehensive statistics and experiments based on AirBirds are presented. First, we investigate the data distribution in AirBirds and compare with relevant datasets to reflect their significant differences. Second, a wide range of SOTA detectors are evaluated on the developed dataset for bird discovering and the results are analyzed in detail to understand the non-trivial challenges of bird strike prevention. Third, the effectiveness of Alg. 1 is evaluated since it plays an important role in the first round of annotations when constructing AirBirds.

### 4.1 Distribution

In this subsection, we investigate the distribution of AirBirds and compare with relevant datasets. Figure 5 shows the distibution of width and height of bounding box in different datasets. Obviously, objects in AirBirds have much smaller sizes. Further, Fig. 6 depicts the proportion of objects with various sizes in relevant datasets. 88% of all instances in AirBirds are smaller than 10 pixels and the rest 12% are mainly in the interval [10, 50). Therefore, data distribution in real-world airports is significantly different from that in web-crawled and tailor-made datasets.

### 4.2 Configurations

A wide range of detectors are tested on AirBirds for bird discovering. Before reporting their performances, it is necessary to elaborate on the specific modifications we made to accommodate the AirBirds dataset and the detectors. Concretely, we customize the following settings.




(a) COCO (b) AirBirds

Fig. 5: Distributions of width and height of annotated instances in COCO and AirBirds.

**Models**. To avoid AirBirds preferring a certain type of detectors, various kinds of strong baselines are picked for evaluation, including one-stage, multi-stage, transformer-based, anchor-free, and other types of models, referring to Tab. 2.

**Devices**. 6 NVIDIA RTX 2080Ti GPUs are used during training and a single GPU device is used during test for all models.

**Data Format**. The format of annotations in AirBirds is consistent with YOLO [16] style. Then we convert them to COCO format when training models other than YOLOv5.

**Anchor Ratios and Scales**. We need to adapt the ratios and scales for anchor-based detectors to succeed in custom training because objects in AirBirds have notable differences in size with that in the commonly used COCO dataset. The k-means clustering is applied to the labels of AirBirds, finally the ratios are set to $[\frac{8}{13}, \frac{9}{12}, \frac{11}{9}]$ and the scales are set to $[2^0, 2^{\frac{1}{3}}, 2^{\frac{2}{3}}]$.

**Learning Schedules**. All models are trained *from scratch* with optimized settings, *e.g.*, training epochs, learning rate, optimizer, batch size, etc. We summarize these settings in Section 2 in the supplementary material.

**Alg.** 1. The thresholds of pixel differences in Alg. 1, *min* and *max*, are set to 25 and 255, respectively.

### 4.3 Results and Analysis

Both accuracy and efficiency are equivalently important for bird discovering in a real-world airport. The accuracy is measured by *average precision* (AP) and the efficiency is judged by *frames per second* (FPS). Results are recorded in Tab. 2.

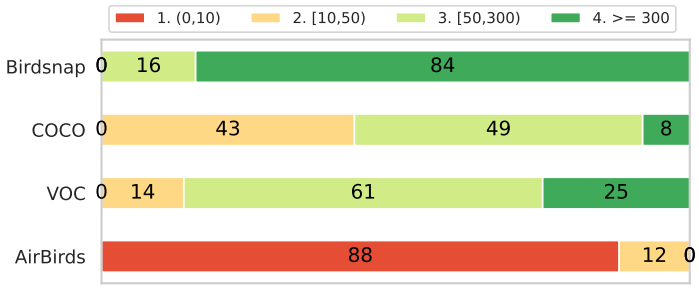



Fig. 6: Comparisons of the ratio of the number of objects with different sizes in the datasets Birdsnap, COCO, VOC and AirBirds. The numbers in each bar are in %. Here the object size in pixel level is divided into 4 intervals: (0,10), [10,50), [50, 300) and [300,+$\infty$).

For accuracy, the primary metric AP seems unsatisfactory, *e.g.*, the highest score achieved by YOLOv5 is only 11.9, and the scores of all other models are less than 10. We also compare the performances of those detectors on COCO and AirBirds, shown in Fig. 7. Under the same detector, however, the performance gap is surprisingly large. For instance, the AP score of EfficientDet-D2 on COCO exceeds the one on AirBirds by 41.5(=42.1-0.6).

Besides, precision-recall relationship are also investigated, and results are shown in Fig. 8. The trend in all curves is that precision decreases with increased recall because more and more false-positive birds produce as more and more birds are recalled. YOLOv5 outperforms others while precision drops to 0 when recall reaches 0.7.

At this point, we wonder whether these detectors are well trained on AirBirds. Hence, their training losses are visualized in Fig. 9. We observe the losses of all detectors drop rapidly in the initial rounds of iterations, then progressively become smooth, indicating the training process is normal and converges to the target.

In terms of efficiency, YOLOv5 also outperforms others, surpassing 100 FPS on a 2080Ti GPU. However, most of detectors *fail to operate* in **real-time efficiency** even with GPU acceleration, which deviates a key principle of bird strike prevention.

We also wonder why a wide range of detectors work poorly on AirBirds. Reasons are detailed in Section 3 in the supplementary material due to space limitation.

In short, existing strong detectors show decent performances on commonly used datasets *e.g.* COCO, VOC etc. However, even with carefully customized configurations, they have room for significant improvements when validating on AirBirds. The results also imply the non-trivial challenges of the research of bird strike prevention in real-world airports, where AirBirds can serve as a valuable benchmark.

Table 2: Comparisons of various kinds of object detectors on AirBirds test set. The column $AP_l$ has been removed as there are few large objects in AirBirds and the corresponding scores are all 0. EffiDet: EfficientDet-D2, Faster: Faster RCNN, Cascade: Cascade RCNN, Deform: Deformable DETR. These abbreviations have the same meaning in the following figure or table.

| Method | Type | Backone | AP | AP@50 | AP@75 | $AP_s$ | $AP_m$ | FPS |
|---|---|---|---|---|---|---|---|---|
| FCOS[22] | one-stage | ResNet50[9] | 0.3 | 1.3 | 0.0 | 0.3 | 0.2 | 18.5 |
| EffiDet[21] | | EffiNet-B2[20] | 0.6 | 1.0 | 1.0 | 0.6 | 4.3 | 4.88 |
| YOLOv3[17] | | DarkNet53[15] | 5.8 | 24.1 | 1.4 | 5.8 | 6.8 | 19.5 |
| YOLOv5[23] | | CSPNet[26] | 11.9 | 49.5 | - | - | - | 109.9 |
| Faster[18] | multi-stage | ResNet50[9] | 7.1 | 26.9 | 1.3 | 7.1 | 0.2 | 16.0 |
| Cascade[2] | | | 6.8 | 24.0 | 1.8 | 6.8 | 1.8 | 13.4 |
| DETR[3] | transformer | ResNet50[9] | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 19.7 |
| Deform[33] | | | 0.4 | 2.2 | 0.0 | 0.4 | 1.0 | 11.6 |
| FPN[12] | FPN | RetinaNet[13] | 2.9 | 12.5 | 0.3 | 2.9 | 8.0 | 21.3 |
| NASFPN[8] | | | 3.0 | 12.5 | 0.4 | 2.9 | 15.8 | 25.0 |
| RepPoints[28] | anchor-free | ResNet50[31] | 4.8 | 22.6 | 0.3 | 4.9 | 0.0 | 37.1 |
| CornerNet[11] | | HourglassNet | 4.5 | 19.5 | 0.6 | 5.0 | 2.5 | 5.5 |
| FreeAnchor[30] | | ResNet101[9] | 6.6 | 26.5 | 1.1 | 6.7 | 9.0 | 53.5 |
| HRNet[19] | high-resolution | HRNet[19] | 8.9 | 33.0 | 1.3 | 9.0 | 0.3 | 21.7 |
| DCN[4] | deformable | ResNet50[9] | 9.7 | 34.6 | 1.8 | 9.8 | 2.4 | 41.9 |
| DCNv2[32] | | ResNet50[9] | 4.2 | 17.5 | 0.5 | 4.6 | 0.0 | 14.2 |

### 4.4 Effectiveness of the First Round of Annotations

As mentioned in Section. 3, Alg. 1 provides the first round of bounding box annotations for possible flying birds and the annotations are saved. Here we validate its effectiveness and compare it with the best performing YOLOv5. Different from *average precision* that sets strict IoU thresholds between detections and groundtruth, actually precision, recall and f1 score are more meaningful metrics for evaluating initial annotations.

Table 3 shows Alg. 1 recalls more than 95% of birds in the initial round, which saves workers numerous efforts of discovering birds in subsequent rounds from scratch thus save costs. In addition, the results indicate that sequence information is helpful for tiny flying birds detection as the input images in Alg. 1 are in chronological order. The star symbol in the second row in Tab. 3 means the results of Alg. 1 are obtained on an ordinary computer(i5 CPU, 16GB memory), without GPU support.

## 5 Conclusion

In this paper, we present AirBirds, a large-scale challenging dataset for bird strike prevention constructed directly from a real-world airport, to close the no-

Table 3: Comparisons of Alg. 1 and YOLOv5 in terms of precision, recall and f1 score. Alg. 1 runs on a common computer and YOLOv5 is tested with a 2080Ti GPU.

| Method | Precision | Recall | F1 | FPS |
|---|---|---|---|---|
| YOLOv5 | **68.10%** | 55.50% | 61.16% | **109.89** |
| Alg. 1 | 58.29% | **95.91%** | **72.51%** | 67.44⋆ |

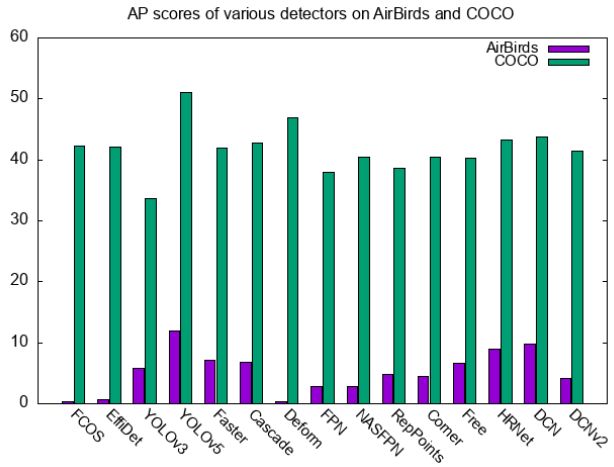
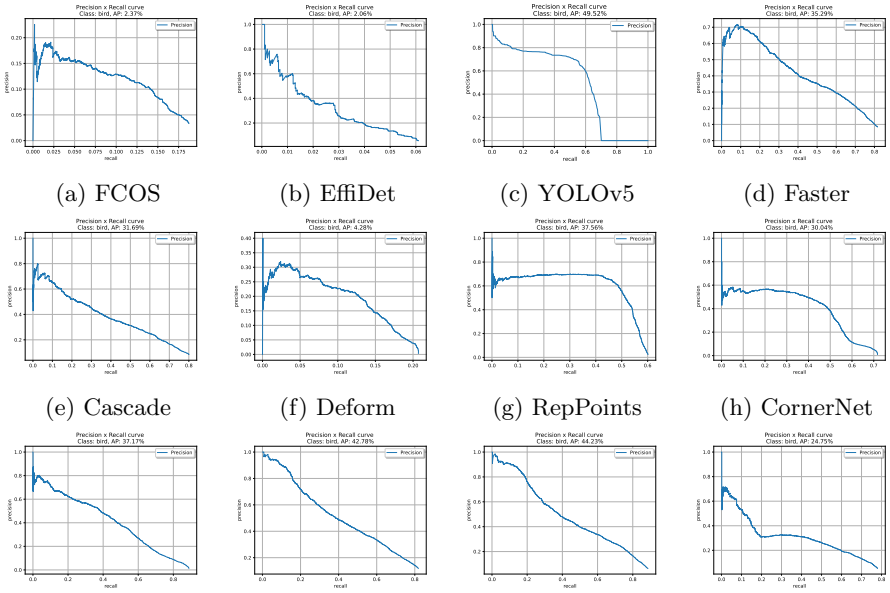



Fig. 7: Comparisons of the performances among representative detectors on AirBirds and COCO.

table gap of data distribution between real world and other tailor-made datasets. Thorough statistical analysis and extensive experiments are conducted based on the developed dataset, revealing the non-trivial challenges of bird discovering and bird strike prevention in real-world airports, which deserves increasing and further investigation, where AirBirds can serve as a first-hand and valuable benchmark.

We believe AirBirds will alleviate the fundamental limitation of the lack of a large-scale dataset dedicated for bird strike prevention in real-world airports, benefit researchers and the field. In the future, we will develop advanced detectors for flying bird discovering based on AirBirds.

**Acknowledgements** We thank all members who involved in the system deploying, data collecting, processing and labeling. This work was supported in part by the National Natural Science Foundation of China (Grant No. 61972404, 12071478).

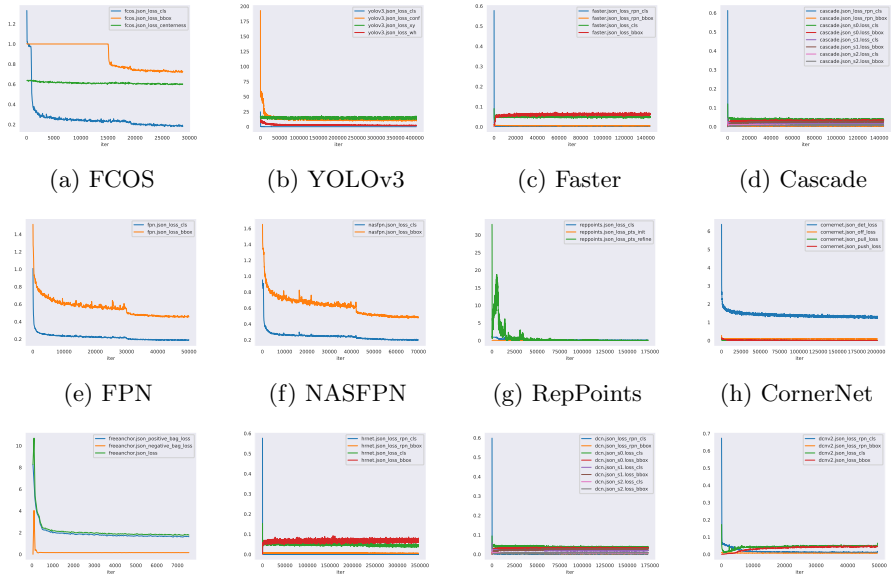


(a) FCOS (b) EffiDet (c) YOLOv5 (d) Faster

(e) Cascade (f) Deform (g) RepPoints (h) CornerNet

(i) FreeAnchor (j) HRNet (k) DCN (l) DCNv2

Fig. 8: Precision-Recall curves of different detectors in VOC[6] style.



(a) FCOS (b) YOLOv3 (c) Faster (d) Cascade

(e) FPN (f) NASFPN (g) RepPoints (h) CornerNet

(i) FreeAnchor (j) HRNet (k) DCN (l) DCNv2

Fig. 9: Training losses of different types of detectors.

## References


1. Berg, T., Liu, J., Lee, S.W., Alexander, M.L., Jacobs, D.W., Belhumeur, P.N.: Birdsnap: Large-scale fine-grained visual categorization of birds. In: Proc. Conf. Computer Vision and Pattern Recognition (CVPR) (June 2014)
2. Cai, Z., Vasconcelos, N.: Cascade r-cnn: Delving into high quality object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
3. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M. (eds.) Computer Vision – ECCV 2020. pp. 213–229. Springer International Publishing, Cham (2020)
4. Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., Wei, Y.: Deformable convolutional networks. In: Proceedings of the IEEE international conference on computer vision (2017)
5. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 248–255 (2009). https://doi.org/10.1109/CVPR.2009.5206848
6. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. International Journal of Computer Vision **88**(2), 303–338 (Jun 2010)
7. Ge, Z., McCool, C., Sanderson, C., Wang, P., Liu, L., Reid, I., Corke, P.: Exploiting temporal information for DCNN-based fine-grained object classification. In: International Conference on Digital Image Computing: Techniques and Applications (2016)
8. Ghiasi, G., Lin, T.Y., Le, Q.V.: Nas-fpn: Learning scalable feature pyramid architecture for object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 770–778 (2016). https://doi.org/10.1109/CVPR.2016.90
10. Krizhevsky, A.: Learning multiple layers of features from tiny images. Tech. rep., University of Toronto (2009)
11. Law, H., Deng, J.: Cornernet: Detecting objects as paired keypoints. In: 15th European Conference on Computer Vision, ECCV 2018. pp. 765–781. Springer Verlag (2018)
12. Lin, T.Y., Dollar, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (July 2017)
13. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollar, P.: Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (Oct 2017)
14. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) Computer Vision – ECCV 2014. pp. 740–755. Springer International Publishing, Cham (2014)
15. Redmon, J.: Darknet: Open source neural networks in c. http://pjreddie.com/darknet/ (2013–2016)

16. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 779–788 (2016). https://doi.org/10.1109/CVPR.2016.91
17. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement (2018)
18. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence **39**(6), 1137–1149 (2017). https://doi.org/10.1109/TPAMI.2016.2577031
19. Sun, K., Xiao, B., Liu, D., Wang, J.: Deep high-resolution representation learning for human pose estimation. In: CVPR (2019)
20. Tan, M., Le, Q.: EfficientNet: Rethinking model scaling for convolutional neural networks. In: Chaudhuri, K., Salakhutdinov, R. (eds.) Proceedings of the 36th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 97, pp. 6105–6114. PMLR (09–15 Jun 2019), https://proceedings.mlr.press/v97/tan19a.html
21. 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 (CVPR) (June 2020)
22. Tian, Z., Shen, C., Chen, H., He, T.: Fcos: Fully convolutional one-stage object detection. arXiv preprint arXiv:1904.01355 (2019)
23. Ultralytics: Yolov5. https://github.com/ultralytics/yolov5 (2021)
24. Van Horn, G., Branson, S., Farrell, R., Haber, S., Barry, J., Ipeirotis, P., Perona, P., Belongie, S.: Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2015)
25. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The Caltech-UCSD Birds-200-2011 Dataset. Tech. Rep. CNS-TR-2011-001, California Institute of Technology (2011)
26. Wang, C.Y., Liao, H.Y.M., Wu, Y.H., Chen, P.Y., Hsieh, J.W., Yeh, I.H.: Cspnet: A new backbone that can enhance learning capability of cnn. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops (June 2020)
27. Welinder, P., Branson, S., Mita, T., Wah, C., Schroff, F., Belongie, S., Perona, P.: Caltech-UCSD Birds 200. Tech. Rep. CNS-TR-2010-001, California Institute of Technology (2010)
28. Yang, Z., Liu, S., Hu, H., Wang, L., Lin, S.: Reppoints: Point set representation for object detection. In: The IEEE International Conference on Computer Vision (ICCV) (Oct 2019)
29. Yoshihashi, R., Kawakami, R., Iida, M., Naemura, T.: Construction of a bird image dataset for ecological investigations. In: 2015 IEEE International Conference on Image Processing (ICIP). pp. 4248–4252 (2015). https://doi.org/10.1109/ICIP.2015.7351607
30. Zhang, X., Wan, F., Liu, C., Ji, R., Ye, Q.: FreeAnchor: Learning to match anchors for visual object detection. In: Neural Information Processing Systems (2019)
31. Zhou, X., Wang, D., Krähenbühl, P.: Objects as points (2019)
32. Zhu, X., Hu, H., Lin, S., Dai, J.: Deformable convnets v2: More deformable, better results. arXiv preprint arXiv:1811.11168 (2018)
33. Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable {detr}: Deformable transformers for end-to-end object detection.

In: International Conference on Learning Representations (2021), https://openreview.net/forum?id=gZ9hCDWe6ke