Fifteenth USA/Europe Air Traffic Management Research and Development Seminar (ATM2023)

Airside Surveillance by Computer Vision in Low-Visibility and Low-Fidelity Environment

Phat Thai
Saab-NTU Joint Lab
Nanyang Technological University
Singapore

Sameer Alam
Saab-NTU Joint Lab
Nanyang Technological University
Singapore

Nimrod Lilith
Saab-NTU Joint Lab
Nanyang Technological University
Singapore

***Abstract*—Low visibility can severely reduce the airside capacity of an airport and can cause ground delays and runway/taxiway incursions. With the advent of digital towers, enabled through live camera feeds, computer vision can contribute to airside surveillance to enhance safety and improve operational efficiency. However, digital camera technology presents its challenges where technical issues may affect the video quality, resulting in low-fidelity transmission effects such as blurring, pixelation, or JPEG compression. Furthermore, poor weather conditions in an aerodrome, including rain, fog, and mist, can greatly reduce visibility, whether based on digital video or out-of-tower view, which can reduce visual situational awareness for tower controllers. This paper proposes a computer vision framework and deep learning algorithms to detect and track aircraft in low-visibility (due to bad weather) and low-fidelity (due to technical issues) environments to enhance visibility using digital video input. The framework adopts a Convolutional Neural Network to detect aircraft and applies a Kalman Filter technique to track aircraft, especially under low visibility conditions. The performance of the proposed framework is further improved by pre/post-processing algorithms, including object filtering, corrupted image detection, and image enhancement. The proposed framework achieves a tracking accuracy of 0.91 for clean videos and 0.79 and 0.74 for low-fidelity and low-visibility environments, respectively. The framework is found to be effective on the airport video dataset from Houston airport in improving visibility in poor weather conditions.**

***Keywords*—Airside Surveillance; Computer Vision; Digital Tower; Low Visibility.**

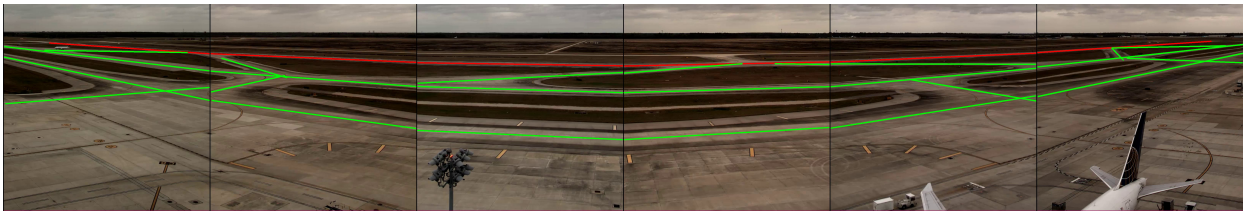
I. Introduction

Adverse weather conditions significantly impact air traffic, both on the ground and in the air. A study conducted by NTSB indicates that meteorological conditions contributed to 20 percent of total aviation incidents worldwide during the period of 2003 to 2007 [1]. Low visibility can lead to reduced airport capacity, delays, and runway/taxiway incursions [2]. Daily average delay data from US domestic flights between 2000 to 2005 reveals that delays on days with adverse weather conditions are 14 minutes longer than on clear days [3]. Similarly, O. R. Tambo International Airport reported 1425 hours of delay related to weather during the period of 2010 to 2013 [4]. Although the Advanced Surface Movement Guidance and Control System (ASMGC-S) has improved airside operations in poor visibility, the separation between aircraft still needs to be enlarged due to ground traffic control [5]. Therefore, effective airside surveillance systems are essential for better traffic management in adverse weather conditions.

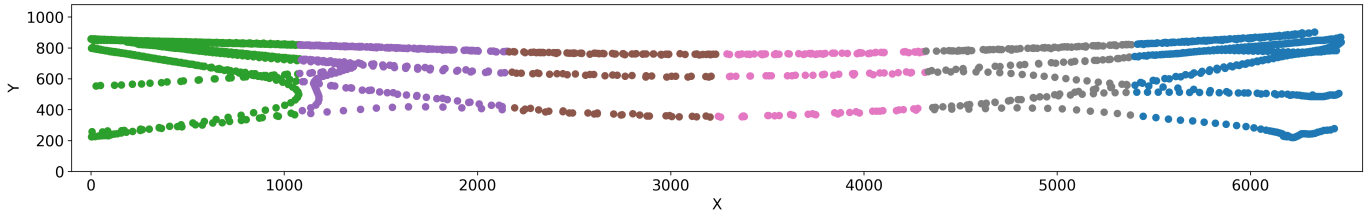
The concept of a digital remote tower has been developed to enhance safety and improve the operational efficiency of airport airside operations [6]. A digital remote tower utilizes a network of high-resolution cameras that covers a 360-degree view of airports to offer many advantages over an out-of-window view of a conventional tower [7]. However, several technical challenges need to be addressed to fully utilize the digitization of the tower environment. For example, the upper limit of transmission delay between the remote airport and the tower center should be below 500 ms [8]. The experiment in [8] shows that during good conditions such as high bandwidth and no image processing algorithms, the average delay of long-transmission time is 300ms. In addition, smart cameras that are optimized for video recording even under adverse environmental conditions can easily exceed a delay of 500 ms [8]. Furthermore, the continuous transmission of video data can also lead to image corruption, including dropped frames, blurring, or pixelation, due to factors such as data congestion, delay accumulation, and hardware problems.

A computer vision-based framework is proposed in this paper for airside surveillance to track aircraft in low visibility and low fidelity environments. The framework is illustrated in Figure 4 and consists of modules to address specific environmental issues arising from low fidelity and low visibility. The first module uses AirNet [28], a customized Convolutional Neural Network, to detect aircraft. Then, by associating newly detected aircraft with aircraft from previous frames, these detected aircraft are tracked by a Kalman Filter. The framework also includes additional modules, such as object filtering to reduce false alarms, image enhancement to improve visibility during rain and fog, and image corruption detection to identify corrupted frames.

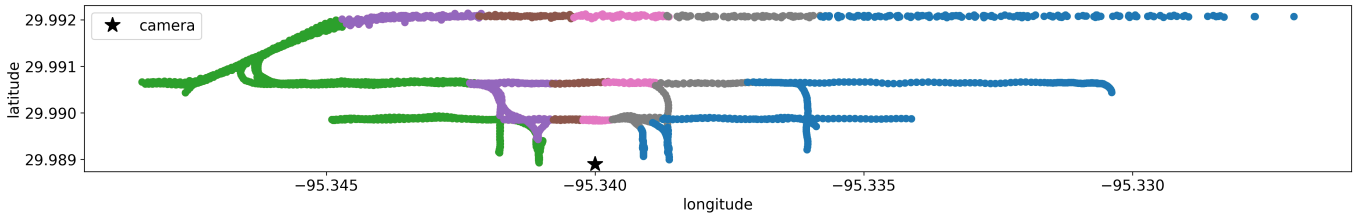
The remainder of this paper is organized as follows. In Section II, a review of previous related work is presented, and the video data and engineering approach are described in Section III. Section IV presents the overview of the proposed framework to address low visibility and low fidelity environmental conditions. The experimental design and the video dataset are outlined in Section V, followed by the results in Section VI. Finally, the paper concludes with a summary and discussions in Section VII.



(a) The Houston airport layout is captured by six surveillance cameras. The runway and taxiways are highlighted by red and green color, respectively.



(b) Aircraft trajectories are captured by the surveillance cameras. Each color represents a different camera.



(c) Aircraft trajectories are recorded by ASMGC-S. The different colors are corresponding with different cameras.

Figure 1: Houston airport airside layout captured from a camera system with the corresponding ASMGC-S information. The displayed trajectories from these two sources are identical.

II. Related Works

A. Single Object Tracking

Object tracking can be categorized as either single-object tracking or multiple-object tracking. Single-object tracking involves the tracking of each object by one tracker, whereas multiple-object tracking involves one tracker monitoring every object. As objects are required to be separated from each other in an airport airside environment, this research focuses on single-object tracking. However, since cameras lack depth information, multiple single trackers may be necessary to track objects that overlap with each other.

Object tracking algorithms can be grouped in various ways. As this paper focuses on low visibility, the object tracking algorithms are divided into temporal approaches and spatial approaches. Temporal approaches predict the future location given a set of previous locations with the Kalman filter [9], [10] being the best candidate for this approach. Although simple and fast, the Kalman filter requires frequent updates with actual object locations and a series of previous locations to stabilize the model.

Spatial approaches associate objects in current frames with target objects in previous frames. The state-of-the-art methods are based on correlation [12]. By performing correlation operations between the target (in previous frames) with the target candidate area (the current frame), the method can locate the target position in the current frame. The target and target candidate area are represented as features before applying the correlation operation. The features can be extracted by hand-crafted methods [13]–[15] or deep learning methods [16]–[18]. As these approaches rely on spatial information, a low visibility environment decreases the detection performance. Therefore, low-visibility handling algorithms are proposed to overcome this problem.

Therefore, a spatial-temporal approach is appropriate for low visibility, which combines spatial and temporal models [11], [19], [20]. When the target is captured in frames, it is tracked by a spatial model, and the captured target updates a temporal model. If the target cannot be captured because of low visibility, a temporal model predicts a potential location until the target is re-captured.

B. Low Visibility Handling

Object detection in low-visibility environments can be categorized into two groups: general and specific. The general category improves detection performance for every type of low visibility by integrating enhancements including sensor fusion [21], multiple scale network [21], [22], histogram equation [22] or larger networks [22]. In contrast, the specific category improves detection performance for only a particular condition, such as deblurring [23], rain removal [24], or defog [25]. Intuitively, specific approaches achieve better results than general approaches, as the particular environmental condition is known. However, they require low-visibility data for training, which are challenging to obtain.

Figure 2: Synthesized low-fidelity samples. Left to right: clean, corrupt level 1 and corrupt level 3 samples. Top to bottom: blurring, pixelating and JPEG compression.

Due to the data limitation, the proposed approach follows a general approach to improve overall detection performance. The detection model is trained only on clean (original) data while it is validated on different types of low-visibility data. The approach also divides low visibility into two scenarios: weather or technical issues. As technical issues cannot be predicted, a corruption detection model is developed to detect corruption events. Meanwhile, as weather conditions typically last longer, image enhancement is proposed to enhance image quality.

III. Dataset and Data Engineering

The airport video dataset is collected from George Bush Intercontinental Airport (IATA: IAH, ICAO: KIAH), also known as Houston Airport. This airport is referred to as Houston Airport for the remainder of the paper. The dataset comprises one hour of video data recorded by six cameras from a digital tower at Houston Airport, as shown in Figure 1. As the airport airside is a wide and complex environment, each camera captures a different size of the airport with different brightness conditions. Specifically, the areas captured by the edge cameras are larger than those covered by the central cameras, causing an aircraft to appear differently sized during its maneuvering phase. In addition, the images captured by the cameras exhibit distortions, making the runways and taxiways look curved instead of straight lines, thereby causing the aircraft trajectories to appear non-linear. Consequently, the Houston airport videos pose a significant challenge for object detection and tracking. To create the dataset, images were extracted from the six cameras every second, generating 21600 images, and the aircraft were manually labeled as bounding boxes. Subsequently, images that did not capture aircraft or had captured stationary aircraft were removed, resulting in 3725 images with a resolution of 1080×1080. Since obtaining low-visibility videos is difficult, this research initially synthesizes these corrupted videos.

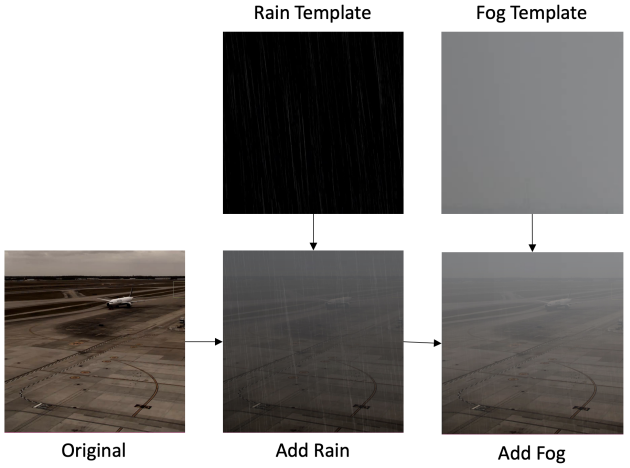


Figure 3: A low-visibility sample is synthesized by blending rain and fog templates to an original image.

A. Low Fidelity Image Generation

Although digital tower systems have numerous safeguards to prevent image corruption, most digital video systems are susceptible to corruption risks. To evaluate the performance of computer vision models in the presence of such corruption, intentional corruption was introduced to the images. The effect of corruption can vary depending on the encode/decode algorithms used. In this study, the three most common types of image corruption, which are blurring, pixelating, and JPEG compression, were chosen, as shown in Figure 2. Blurred images were created by convolving the image with a low-pass filter kernel. Pixelated images were produced by downsampling the image resolution and then upsampling it with a linear interpolation algorithm. JPEG-compressed images were generated by transforming an image into a vector using Discrete Cosine Transform [27], truncating the vector dimension, and then transforming it back. For each type of image corruption, three different levels of corruption are generated for each image.

B. Low Visibility Image Generation

This study examines the impact of weather on airside visibility, with a focus on two common weather phenomena: rain and fog. Figure 3 illustrates the image generation process to simulate these effects. First, a rain structure transfer algorithm [26] generates a synthetic rain image based on an exemplar rain image. As rain typically reduces the brightness, the "Lightness" channel in the HLS color space is reduced by half. The next step is to blend the rain image (**R**) to a fog image (**F**), using Eq. (1). The value of α channel increases in a near-distance view to generate less fog and decreases to generate more fog in a far-distance view. Similarly, image corruption is applied at three different levels.

$$\mathbf{I} = \alpha * \mathbf{R} + (1 - \alpha) * \mathbf{F}, \quad 0 < \alpha < 1 \tag{1}$$

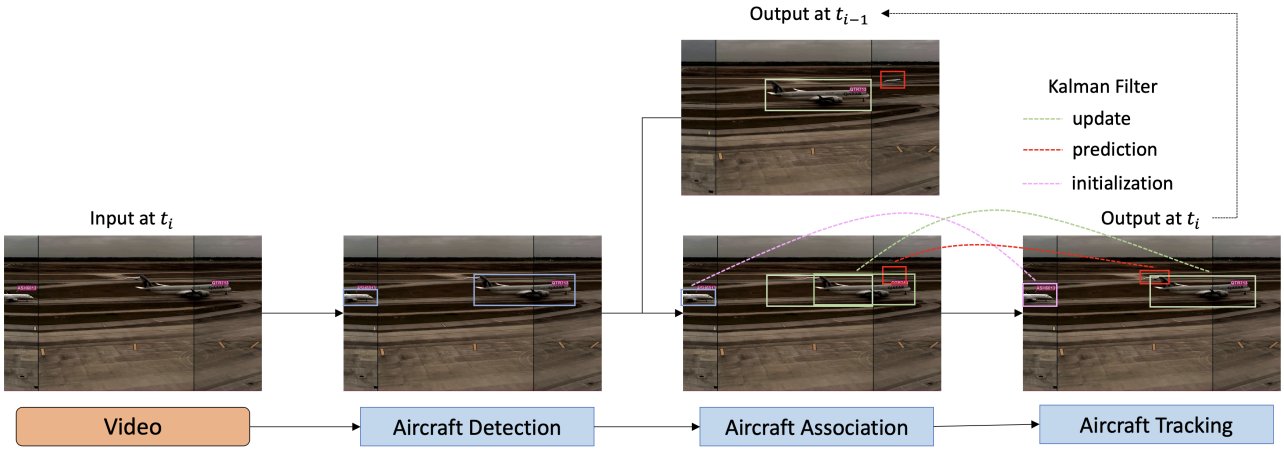


Figure 4: The proposed framework where aircraft are detected by customized Convolutional Neural Network, called AirNet. Detected aircraft are then associated with the previous frame. Finally, aircraft are tracked by a Kalman Filter based on the association results.

IV. Methodology

A. Overview

Figure 4 illustrates the computer vision framework for aircraft surveillance. First, aircraft are detected by a customized Convolutional Neural Network developed by authors, called AirNet [28]. The detected aircraft are then associated with the aircraft in the previous frame. Finally, the aircraft are tracked by a Kalman Filter [9] based on the association results. Specifically, if an aircraft is newly detected, a Kalman Filter tracker is initialized. If an aircraft is detected in both frames, the Kalman filter tracker is updated. If an aircraft from the previous frame cannot be detected in the current frame, the tracker predicts the aircraft's location.

The AirNet framework was developed to detect objects in an airport airside environment and outperformed the state-of-the-art ConvNets in an airport dataset [28], [29]. Generally, AirNet is customized to detect objects in a high-resolution image, which is 1080×6480 in this project. With multiple feature maps on different scales, the AirNet framework can cover a wide range of aircraft dimensions. Moreover, the AirNet architecture is divided into nearly identical blocks governed by a limited number of parameters. As a result, the AirNet framework is flexible to be adopted for the given problem.

A detected aircraft in the current frame ($\boldsymbol{B_t}$) is associated with the previous frame ($\boldsymbol{B_{t-1}}$) by the Hungarian method [30]. First, a distance matrix is constructed:

$$\Delta(B_t, B_{t-1}) = \{\delta_{i,j}\}, \delta_{i,j} = \|c_i - c_j\|$$
$$i \in [0, n_t], j \in [0, n_{t-1}]$$

where n_t and n_{t-1} are the number of detected aircraft in frames t and $t-1$ and c is the center of a detected aircraft. Then, based on the distance matrix, the Hungarian algorithm assigns detected aircraft between the current frame and the previous frame. However, when a distance is greater than a given threshold, an assigned pair is rejected. The algorithm outcome can be separated into three cases, as shown in Figure 4. The first case is new detection where a detected aircraft in the current frame is not associated with any aircraft in the previous frame. The second case is the successful detection of a previously detected aircraft where a detected aircraft in the current frame is associated with an aircraft in the previous frame. The third case is missed detection where an detected aircraft in the previous frame is not associated with any aircraft in the current frame.

The association result is used for aircraft tracking by a Kalman Filter [9]. Each aircraft is tracked by one Kalman Filter tracker. The Kalman Filter tracker uses aircraft history position to predict the next position. As the original Kalman Filter [9] is created for one-dimensional prediction, the tracker is modified to predict two-dimensional positions. Therefore, a state vector includes six variables which are $x, y, \delta x, \delta y, \delta^2 x, \delta^2 y$ where x and y are the position of the aircraft, δ and δ^2 are aircraft velocity and acceleration corresponding with the axes. The tracker tracks aircraft based on association results. In the first case, a new tracker is initialized while the tracker parameters are updated in the second case. In the third case, the tracker predicts the new location. Moreover, during the third case, the tracker can be terminated if the predicted position is out of the frame or the aircraft cannot be detected for a long period of frames.

B. Aircraft Tracking in Low Fidelity Environments

Aircraft tracking in low-fidelity environments is performed by the proposed framework as follows. It is to be noted that the AirNet framework in this research is trained only on clean images. To reduce false alarms on aircraft detection, this research incorporates negative samples during the training process, as suggested in [32]. False alarms are further reduced by an object filtering module, which rejects detected aircraft based on their sizes. In addition, an image corruption detector algorithm, based on a background subtraction algorithm [31], is implemented to detect corrupted images. The framework relies more heavily on temporal information for corrupted frames instead of spatial information for clean frames.

The object filtering module is created as a linear regression model. First, the wingspans and lengths of aircraft are collected from Aircraft Characteristics Database published by Federal Aviation Administration[1]. Then, the relationship between aircraft position ($\boldsymbol{X} = (x, y)$) and size ($\boldsymbol{S}$) is constructed, as shown in Eq. (2), where $S = \sqrt{wing^2 + len^2}$ and $z = \sqrt{width^2 + height^2}$ are the actual size and a pixel size of aircraft. Diagonal sizes are used to be invariant with aircraft orientation. W is the weight learned from the data during the training process, $f(.)$ can be viewed as the feature extraction function, and ϵ is the irreducible error that occurs when collecting the data. Naturally, $f(X)$ is chosen as a polynomial function with degree k, as shown in Eq. (3).

$$\frac{\tilde{S}}{z} = Wf(X) + \epsilon \tag{2}$$

$$f(x, y) = \sum_{i=0}^{k} \sum_{j=0}^{i} x^j y^{i-j} \tag{3}$$

During validation, the linear regression model predicts detected object sizes based on their position. Detected objects are rejected when their sizes are smaller than a certain threshold. In this research, the threshold is set to $100ft$.

The image corruption detector to detect corrupted frames is implemented as follows. First, the detector initializes a template from a history of clean frames. During the validation phase, the detector calculates the difference between the current frames and the template. When the difference exceeds a certain threshold, the frame is considered to be corrupted. If a current frame is not corrupted, the frame is used to update the template.

The spatial information in corrupted frames is not reliable. Therefore, the confidence scores of the newly detected aircraft are increased. In other words, the framework aims to maintain the tracking of existing aircraft and discourage the tracking of new aircraft. In addition, the tracking termination decision is disabled. That is, the framework does not terminate trackers in which an aircraft is not detected during corrupted frames.

C. Aircraft tracking in Low Visibility Environments

Aircraft tracking in low-visibility environments is performed by the proposed framework. Similarly, the AirNet framework is trained only on clean images and an object filtering module is applied to reduce false alarms. As the weather (rain and fog in this research) tends to last longer than image corruption caused by technical issues, the images are enhanced to improve detection performance. The enhancement, as shown in Figure 5, is inspired by the defogging process in [25]. First, the image brightness is increased from the low-visibility image. Due to the nature of fog, the near-view distance is brighter than the far-view distance. Therefore, a global increase in image brightness does not solve this problem. Hence, gamma correction changes the image brightness non-linearly, as shown in Eq. (4). With γ smaller than 1, an under-exposed image is created to focus on the far-view distance. Similarly, an over-exposed image is created to focus on the near-view distance with γ larger than 1. The final result is obtained by averaging these two differently exposed images.

[1]https://www.faa.gov/airports/engineering/aircraft_char_database

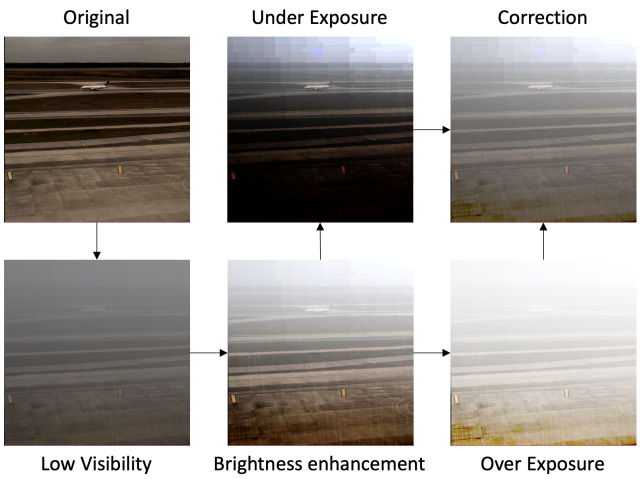


Figure 5: Image enhancement. First, The brightness of the low visibility image (bottom left) generated from the original image (top left) is increased (bottom middle). Then, under- (top middle) and over-exposed (bottom right) images are created by gamma correction. The final image (top right) is an average of two differently-exposed images.

$$O = \alpha I^{\gamma} \tag{4}$$

V. Experiments

The training process in this work is similar to AirNet [28]. However, negative images are included to reduce false alarms [32]. As the negative images should be diversified, they are collected from the COCO dataset [33]. Since the image resolution in COCO ranges from 480 to 640, a training sample is created by stitching four images together. The first half of the Houston airport video is used for training and the second half is used for testing. During the training process, the number of negative images is equal to the number of positive images. It is worth highlighting that the model is only trained and validated on clean (original) images.

The detection model and the tracking framework are tested separately. The detection model is tested on clean, low visibility, and low fidelity images. Low-fidelity images include three separate types of image degradation, which are blurring, pixelating, and JPEG compressing artifacts. In addition, each corrupted type has three degrees of corruption, as shown in Figure 2. Three evaluation metrics are used, which are precision, recall, and average precision (AP). A high recall detector aims to detect as many objects as possible, which helps to reduce false negatives. In contrast, a high-precision detector aims to detect objects as precisely as possible, which helps to reduce false positive detections (false alarms). Each detection threshold (confidence score) produces a different pair of precision and recall. Hence, AP is an important metric that calculates the average precision over all possible

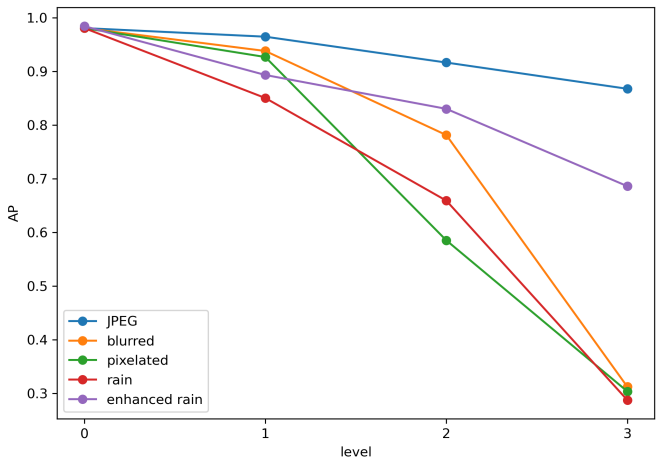


Figure 6: Detection results of different corruption types with multiple levels. Level 0 is a clean dataset.

thresholds. In other words, AP, ranging from 0 to 1, indicates the detector's performance regardless of detection thresholds (i.e., the higher AP the better performance).

The tracking framework is tested differently in different scenarios. The clean scenario can be used as a benchmark where the tracking framework is performed normally on clean, uncorrupted videos. As the weather normally occurs consistently, a low-visibility environment (rain and fog) is generated on the whole test videos with different levels of corruption (level 1 to 3). However, as technical issues occur randomly, a low-fidelity environment is generated randomly with different duration. For example, in the "low fidelity 2s" scenario, there are two seconds of corrupted frames with random types of corruption, for every four seconds. As the results are different every time an experiment is performed, the experiment is performed 10 times and the average results are reported. Four evaluation metrics are validated for the tracking framework: tracking length and accuracy are used to validate tracking performance [34], while precision and recall are used to compare with detection performance. Tracking length [34] reports the number of successfully tracked frames from the tracker's initialization to its first failure. This metric evaluates the robustness of the tracker. Accuracy [34] evaluates the accuracy of the tracker by the following formula: $\frac{TP}{TP+FN+FP}$ where TP, FN, and FP are true positive, false negative, and false positive, respectively.

VI. Results

A. Detection Results

Figure 6 shows the AP metric of AirNet for different image corruption types of different degrees. Naturally, the highest performance occurs on the clean dataset (level 0), which is 0.98 while the lowest performance occurs on the most corrupted datasets (level 3). The AirNet framework is more robust with JPEG compression, whose AP is 0.87 at level 3 while the AP of the remainder is around 0.3 at level 3. In addition, image enhancement improves the performance significantly where the AP at level 3 increases from 0.28 to 0.68.

TABLE I. Detection results with different confidence scores for low fidelity environment.

Scenario	Metric	Confidence Scores		
		0.3	*0.5*	*0.7*
Clean	Recall	**0.9849**	0.9817	0.9722
	Precision	0.6951	0.8704	**0.9459**
Blur 1	Recall	0.9595	0.9539	0.9396
	Precision	0.4697	0.8277	0.9024
Blur 2	Recall	0.857	0.8094	0.7403
	Precision	0.1337	0.6876	0.8519
Blur 3	Recall	0.5052	0.4091	0.3018
	Precision	0.0893	0.1538	0.558
Pixel 1	Recall	0.9635	0.9539	0.9325
	Precision	0.1847	0.2596	0.8976
Pixel 2	Recall	0.753	0.6521	0.5036
	Precision	0.142	0.2388	0.8277
Pixel 3	Recall	0.4504	0.3384	0.2081
	Precision	0.1434	0.27	0.7988
JPEG 1	Recall	0.9778	0.9714	0.9492
	Precision	0.7095	0.8631	0.938
JPEG 2	Recall	0.9555	0.9349	0.8999
	Precision	0.2777	0.5447	0.8282
JPEG 3	Recall	0.9023	0.8292	0.6751
	Precision	0.3453	0.7856	0.9361

TABLE II. Detection results with different confidence scores for low visibility environment.

Scenario	Metric	Confidence Scores		
		0.3	*0.5*	*0.7*
Clean	Recall	**0.9849**	0.9817	0.9722
	Precision	0.6951	0.8704	0.9459
Rain 1	Recall	0.892	0.8761	0.8181
	Precision	0.6034	0.8277	0.9024
Enhance 1	Recall	0.8896	0.8777	0.8475
	Precision	0.9098	0.9501	**0.9656**
Rain 2	Recall	0.699	0.6068	0.4821
	Precision	0.712	0.8042	0.8634
Enhance 2	Recall	0.8491	0.8118	0.7228
	Precision	0.8444	0.9133	0.9539
Rain 3	Recall	0.3233	0.2351	0.1668
	Precision	0.5823	0.6948	0.7609
Enhance 3	Recall	0.6839	0.5941	0.467
	Precision	0.7455	0.8452	0.9116

AP compares the overall performance between models or datasets. Precision and recall with fixed confidence scores are used to evaluate models in real applications. Table I and Table II describe the precision and recall of the detection model in low-fidelity and low-visibility environments, respectively. Similarly, the clean dataset is used for benchmark purposes, and each corruption type has three levels of corruption. By increasing the confidence scores, the detection model detects fewer objects, resulting in high false negative (low recall) but low false positive (high precision). Therefore, confidence scores are chosen based on the application. Although the confidence scores range from 0 to 1, only three values are displayed, which are 0.3, 0.5, and 0.7. The precision and recall values can be interpreted as follows. The "Clean" scenario with a confidence score of 0.5 has a precision of 0.87 and a recall of 0.98. Simply put, for every 100 objects,

the detection model can correctly detect 98 objects and gives 14 false alarms. To provide another example, the "Rain 2" scenario with a confidence score of 0.5 has a precision of 0.8 and recall of 0.6. Meaning that for every 100 objects, the detector can correctly detect 60 objects and gives 15 false alarms.

B. Tracking Results

Table III shows the tracking results of the framework for different scenarios. The "Clean" scenario can be used as a benchmark. The tracking length is nearly 1, which means that objects are successfully tracked from the first initialization to the end. However, as the accuracy is only 0.91, there are some false negatives and false positives. By analyzing the results, the framework fails to detect aircraft at the early stage when they are relatively small in size. The precision and recall of the tracking framework are higher than the detector's, due to the effective prediction from the Kalman filter.

The results are similarly good in low-fidelity scenarios with different interval times. This demonstrates that the proposed framework can work well in low-fidelity environments. However, the framework performance is reduced in severe weather conditions. The lower visibility, the worse the performance. In addition, the precision values are relatively higher than the recall values. This can be interpreted as the framework tends to mis-detect aircraft rather than giving false alarms. This tendency is caused by two reasons. First, aircraft can be missed easily in low-visibility conditions. Secondly, the object filtering algorithm reduces detected objects by dimensions.

TABLE III. Tracking results of the framework for different scenarios.

Scenario	Length	Accuracy	Precision	Recall
Clean	0.9958	0.9174	0.9821	0.9329
Low Fidelity 1s	0.8806	0.7935	0.9488	0.8288
Low Fidelity 2s	0.9499	0.7974	0.9247	0.8527
Low Fidelity 4s	0.9687	0.8534	0.95	0.8936
Low Visibility 1	0.8920	0.8249	0.9609	0.8535
Low Visibility 2	0.8731	0.7407	0.9337	0.7818
Low Visibility 3	0.6695	0.5223	0.8660	0.5682

Figure 7 illustrates samples of the tracking framework in different scenarios. Each aircraft is displayed by a bounding box in the current frame and circles in previous frames. For visualization purposes, only 96 previous frames are displayed. In addition, the filled circles represent the center of detected bounding boxes while the unfilled circles represent the Kalman filter prediction, which means that the detector mis-detects aircraft. In the clean scenario, the trajectories are stable as the detector can detect aircraft most of the time. Even though the detector fails to detect the aircraft due to occlusion, the Kalman filter can predict the aircraft's position accurately. In a low-fidelity environment, the trajectories are not stable due to the detector's performance. However, with a maximum of four seconds of corruption, the framework still manages to track the aircraft given the clean frame after a series of corrupted frames. The most challenging performance occurs in the low-visibility environment. As low-visibility conditions

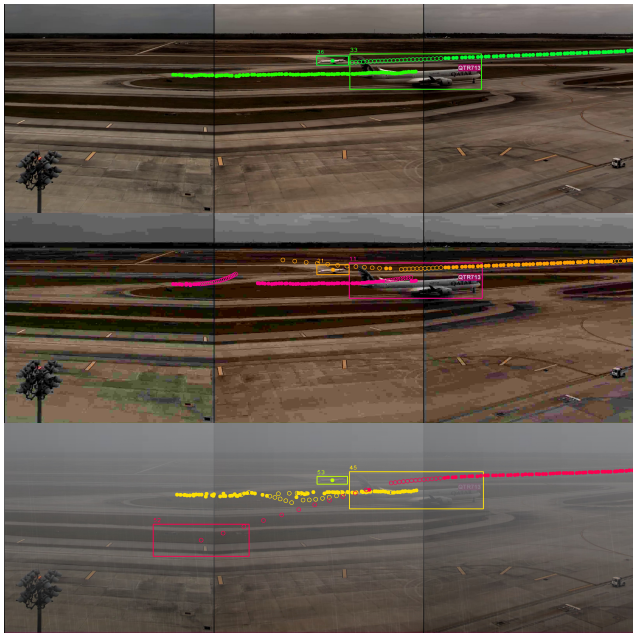


Figure 7: Samples of tracking framework in different scenarios. Top: clean scenario. Middle: low fidelity environment. Bottom: low visibility environment.

are present throughout the entirety of the video, the tracker error is accumulated due to the lack of detection performance. When an aircraft is not detected and the Kalman filter predicts the location of the aircraft incorrectly for a long time, the tracker fails to track the aircraft. As a result, a current tracker is terminated and a new tracker is initialized to track the same aircraft, as shown at the bottom of Figure 7. The detailed process leading to the failed tracking is described as follows. The 52-th aircraft moving on the runway was not detected because of the occlusion with the 45-th aircraft. Hence, the Kalman filter tracks the 52-th aircraft which indicates as unfilled circles. Since the runway is captured as a curve, the tracker predicts that the aircraft is moving down. When the 45-th aircraft moves partially to a different camera, the aircraft is captured with different brightness. Accumulating with low visibility, the detector detects 45-th aircraft as two separate aircraft. The right part of the detected aircraft is associated with 45-th aircraft while the left part is associated with the 52-th aircraft as it still is mis-detected due to occlusion. Consequently, the tracker predicts that the 52-th aircraft keeps moving further down. When the 52-th aircraft is out of occlusion, the distance between the real position and the prediction position is too large. Therefore, it is considered a new aircraft.

VII. Conclusion

The full capabilities of the digital tower environment can be harnessed by innovative computer vision techniques which can aid in the detection, tracking, and surveillance of airport airside objects. This paper proposes a computer vision framework to detect and track aircraft in low-visibility and low-fidelity environments for a digital tower. By integrating a Convolutional Neural Network (spatial information) and

a Kalman filter (temporal information), the framework successfully tracks aircraft in different environments. The results are further improved by additional modules including object filtering, corruption detection, and image enhancement.

The proposed framework demonstrates high performance in clean videos with an AP of 0.98 for detection performance and an accuracy of 0.91 for tracking performance. Although the detection results decrease in low-visibility and low-fidelity environments, the framework performance remains resilient to these challenges. Specifically, the framework manages to track aircraft through a series of low-fidelity frames. Additionally, the framework reduces the effect of adverse weather conditions by incorporating image enhancement.

In the future, new modules will be included in the framework to improve the performance. For example, aircraft position can be predicted in real-world coordinates which are not affected by distortion instead of pixel coordinates. However, the approach requires frequent conversion between two coordinate systems, which can increase processing time and accumulate conversion errors. Alternatively, since ADS-B data is now widely available, the camera system can be synchronized with ADS-B data. Then, aircraft can be detected from videos and predicted from ADS-B. Additionally, image enhancement could be applied across images instead of individual ones.

VIII. Acknowledgement

This research is supported by Saab Sweden. The authors are grateful to Saab Inc. Syracuse, USA, for providing airport surveillance videos from Houston Airport for this research.

References

[1] Ketabdari, Misagh, Emanuele Toraldo, and Maurizio Crispino. "Numerical risk analyses of the impact of meteorological conditions on probability of airport runway excursion accidents." Computational Science and Its Applications–ICCSA 2020: 20th International Conference, Cagliari, Italy, July 1–4, 2020, Proceedings, Part I 20. Springer International Publishing, 2020.

[2] Kulesa G. Weather and aviation: How does weather affect the safety and operations of airports and aviation, and how does FAA work to manage weather-related effects?. In The Potential Impacts of Climate Change on Transportation US Department of Transportation Center for Climate Change and Environmental Forecasting; US Environmental Protection Agency; US Department of Energy; and US Global Change Research Program 2003.

[3] Hsiao CY, Hansen M. Econometric analysis of US airline flight delays with time-of-day effects. transportation research Record. 2006;1951(1):104-12.

[4] Peck L. The Impacts of Weather On Aviation Delays At OR Tambo International Airport, South Africa (Doctoral dissertation, University of South Africa).

[5] Möller D, Wieprecht W, Hofmeister J, Kalass D, Elbing F, Ulbricht M. Fog dissipation by dry ice blasting: process mechanism. In 2nd Fog Conference 2001.

[6] Fürstenau N. Virtual and remote control tower. Switzerland: Springer. 2016.

[7] Frequentis AG. Whitepaper: Introduction to remote virtual tower.

[8] Schmidt M, Rudolph M, Fürstenau N. Remote tower prototype system and automation perspectives. Virtual and Remote Control Tower: Research, Design, Development and Validation. 2016:193-220.

[9] Welch G, Bishop G. An introduction to the Kalman filter.

[10] Najafzadeh N, Fotouhi M, Kasaei S. Object tracking using Kalman filter with adaptive sampled histogram. In2015 23rd Iranian Conference on Electrical Engineering 2015 May 10 (pp. 781-786). IEEE.

[11] Aslan MF, Durdu A, Sabanci K, Mutluer MA. CNN and HOG based comparison study for complete occlusion handling in human tracking. Measurement. 2020 Jul 1;158:107704.

[12] Zhou J, Yao Y, Yang R. Deep Learning for Single-object Tracking: A Survey. In2022 IEEE 2nd International Conference on Software Engineering and Artificial Intelligence (SEAI) 2022 Jun 10 (pp. 12-19). IEEE.

[13] Bolme DS, Beveridge JR, Draper BA, Lui YM. Visual object tracking using adaptive correlation filters. In2010 IEEE computer society conference on computer vision and pattern recognition 2010 Jun 13 (pp. 2544-2550). IEEE.

[14] Kalal Z, Mikolajczyk K, Matas J. Tracking-learning-detection. IEEE transactions on pattern analysis and machine intelligence. 2011 Dec 13;34(7):1409-22.

[15] Zhong W, Lu H, Yang MH. Robust object tracking via sparsity-based collaborative model. In2012 IEEE Conference on Computer vision and pattern recognition 2012 Jun 16 (pp. 1838-1845). IEEE.

[16] Danelljan M, Robinson A, Shahbaz Khan F, Felsberg M. Beyond correlation filters: Learning continuous convolution operators for visual tracking. InComputer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part V 14 2016 (pp. 472-488). Springer International Publishing.

[17] Danelljan M, Hager G, Shahbaz Khan F, Felsberg M. Learning spatially regularized correlation filters for visual tracking. InProceedings of the IEEE international conference on computer vision 2015 (pp. 4310-4318).

[18] Danelljan M, Bhat G, Shahbaz Khan F, Felsberg M. Eco: Efficient convolution operators for tracking. InProceedings of the IEEE conference on computer vision and pattern recognition 2017 (pp. 6638-6646).

[19] Stadler D, Beyerer J. Improving multiple pedestrian tracking by track management and occlusion handling. InProceedings of the IEEE/CVF conference on computer vision and pattern recognition 2021 (pp. 10958-10967).

[20] Al-Shakarji NM, Bunyak F, Seetharaman G, Palaniappan K. Multi-object tracking cascade with multi-step data association and occlusion handling. In2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS) 2018 Nov 27 (pp. 1-6). IEEE.

[21] Tomy A, Paigwar A, Mann KS, Renzaglia A, Laugier C. Fusing event-based and RGB camera for robust object detection in adverse conditions. In2022 International Conference on Robotics and Automation (ICRA) 2022 May 23 (pp. 933-939). IEEE.

[22] Hendrycks D, Dietterich T. Benchmarking neural network robustness to common corruptions and perturbations. arXiv preprint arXiv:1903.12261. 2019 Mar 28.

[23] Sayed M, Brostow G. Improved handling of motion blur in online object detection. InProceedings of the IEEE/CVF conference on computer vision and pattern recognition 2021 (pp. 1706-1716).

[24] Pal NS, Lal S, Shinghal K. A robust visibility restoration framework for rainy weather degraded images.

[25] Liu Q, Luo Y, Li K, Li W, Chai Y, Ding H, Jiang X. Single image defogging method based on image patch decomposition and multi-exposure image fusion. Frontiers in Neurorobotics. 2021 Jul 7;15:700483.

[26] Son CH, Zhang XP. Rain structure transfer using an exemplar rain image for synthetic rain image generation. arXiv preprint arXiv:1610.00427. 2016 Oct 3.

[27] Ahmed N, Natarajan T, Rao KR. Discrete cosine transform. IEEE transactions on Computers. 1974 Jan;100(1):90-3.

[28] Thai P, Alam S, Lilith N, Nguyen BT. A computer vision framework using Convolutional Neural Networks for airport-airside surveillance. Transportation Research Part C: Emerging Technologies. 2022 Apr 1;137:103590.

[29] Van Phat, Thai, et al. "Deep4air: A novel deep learning framework for airport airside surveillance." 2021 IEEE International Conference on Multimedia Expo Workshops (ICMEW). IEEE, 2021.

[30] Kuhn HW. The Hungarian method for the assignment problem. Naval research logistics quarterly. 1955 Mar;2(1-2):83-97.

[31] Piccardi M. Background subtraction techniques: a review. In2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583) 2004 Oct 10 (Vol. 4, pp. 3099-3104). IEEE.

[32] Gao, L., He, Y., Sun, X., Jia, X., Zhang, B. (2019). Incorporating negative sample training for ship detection based on deep learning. Sensors, 19(3), 684.

[33] Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL. Microsoft coco: Common objects in context. InComputer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13 2014 (pp. 740-755). Springer International Publishing.

[34] Čehovin, Luka, Aleš Leonardis, and Matej Kristan. "Visual object tracking performance measures revisited." IEEE Transactions on Image Processing 25.3 (2016): 1261-1274.