Aircraft Push-back Prediction and Turnaround Monitoring by Vision-based Object Detection and Activity Identification

Thai Van Phat*, Sameer Alam*, Nimrod Lilith*, Phu N. Tran† and Binh T. Nguyen‡
*Saab-NTU Joint Lab, Nanyang Technological University, Singapore
†Air Traffic Management Research Institute, Nanyang Technological University, Singapore
‡AISIA Research Lab, University of Science, Vietnam National University, Ho Chi Minh City, Vietnam

***Abstract*—An accurate prediction of aircraft readiness for departure can help Air Traffic Control (ATC) plan an optimal pre-departure sequence at which aircraft are dispatched from the parking stands. This dynamic mechanism between predicting when all ground handling activities end (Target Off Block Time) and the pre-departure sequencing (Target Start-up Approval time) is the core of Airport Collaborative Decision Making. This turnaround process consists of several activities (fueling, boarding/deboarding, loading/unloading, etc.) and involves several ground support types of equipment and vehicles. In this research, we propose a visual-analytic approach for detection, tracking such activities to predict the Target Off Block Time (push-back time). This research introduces a Convolutional Neural Networks based video-analytic framework that can monitor the aircraft turnaround processes, including object detection, object tracking, activity detection, and push-back prediction. It recognizes an aircraft type and retrieves turnaround process/activities from its Aircraft Performance Manual and then detects and tracks various activities to estimate their completion time with high accuracy. Live Gate Cam video data was collected from the Gate 3 at Tokachi-Obihiro airport, in Hokkaido, Japan. We used 16 videos with the corresponding lengths varying from 40 to 60 minutes for training and five videos for testing. The video-analytic framework achieves 100% accuracy in aircraft type recognition, while object detection achieves 0.9514 mean Average Precision. For activity detection, the median error is less than 6 seconds, which can be considered very low in the overall turnaround duration. Furthermore, the proposed framework provided accurate push-back prediction than the scheduled push-back time in four out of five or 80% of cases.**

***Keywords*—Aircraft Turnaround Monitoring, Computer Vision Surveillance System, Convolutional Neural Networks.**

I. Introduction

Aircraft turnaround (TA) is the process of preparing an aircraft for a flight; it begins once the plane has reached an airport apron and completes once the aircraft is ready to leave [1]. TA process involves many Ground Support Equipment (GSE), as shown in Figure 1. The process consists of many activities, including deboarding, catering, cleaning, fueling, cargo unloading and loading, and boarding, which can be performed simultaneously or sequentially depending on the regulations described in the Airplane Characteristics for Airport Planning Manual (APM) [2] (see Figure 3).

TA time plays a vital role in schedule planning, fleet planning, and operation planning. The TA's better prediction accuracy may enable ATC to forecast actual runway demand, which could help manage departure queues at the runway holding point. In turn, it may improve the predictability of runway demand and help determine an optimal push-back sequence to ensure smooth take-offs at the runways.

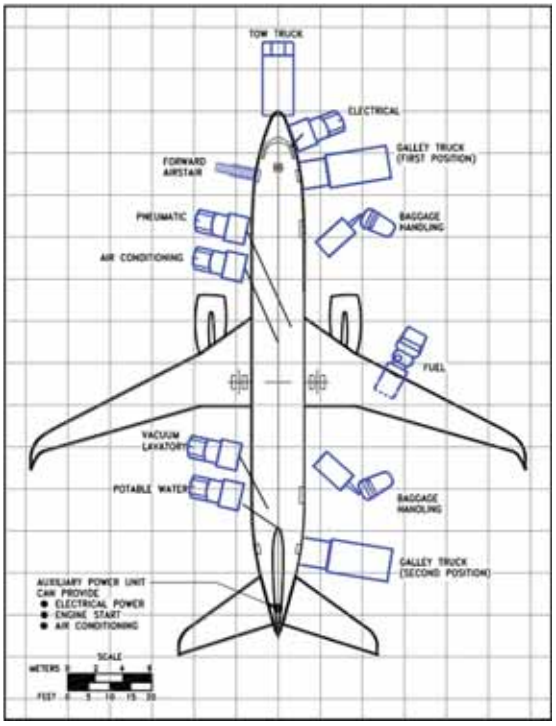

Figure 1. Aircraft turnaround arrangement involving many objects called ground support equipment of B737.

Previous works analyzed the TA time from various perspectives, including airport infrastructure and resources [3], [4], aircraft boarding [5], or arrival delay [6]. Another approach to TA analysis is through video analytic. In this approach, the authors introduced computer vision-based surveillance systems to detect and track aircraft in the airport airside recently [7], [8]. Additionally, a surveillance system, namely GAMTOS [9], was developed for monitoring aircraft turnaround. However, no validation has been reported for this system; thus, it might be

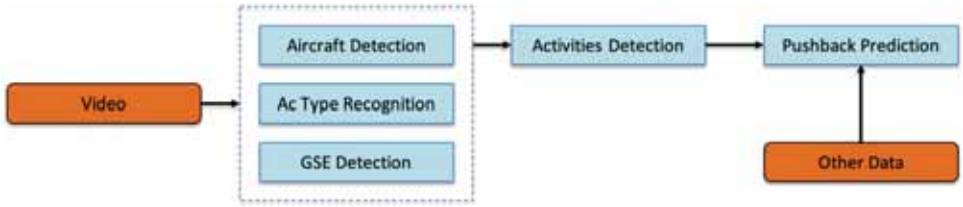

Figure 2. The proposed framework for monitoring aircraft turnaround process. By detecting aircraft, ground support equipment and recognizing aircraft type, the framework can detect activities and predict push-back time.

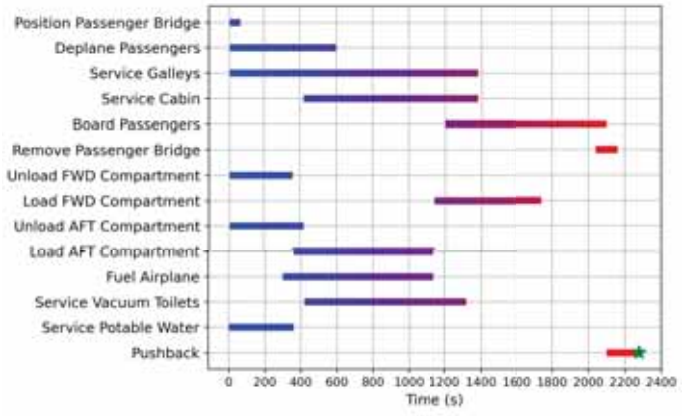

Figure 3. Aircraft turnaround process of B737 starting when an aircraft arrives (second 0) and ending when an aircraft pushes back. There are 14 activities which perform simultaneously or sequentially over time (from blue to red).

considered as a proof of concept system.

In this paper, we aim to investigate a computer vision-based framework to monitor the turnaround process to predict the aircraft's push-back time, as shown in Figure 2. Our framework adopts state-of-the-art models in computer vision, i.e., the Convolutional Neural Networks (ConvNets) [10]–[12], and is capable of accurately detecting different objects in the apron area, ranging from large aircraft to small fuel pipes. The framework also recognizes the aircraft type to retrieve the respective TA process information from the APM. The detection of the aircraft and GSE is then employed to identify different activities in the TA process and update the schedule progress in real-time. During the TA, the framework continuously predicts aircraft push-back time and revises it until the push-back event occurs.

The rest of this paper is organized as follows. We briefly review previous related works in Section II and describe the video data in Section III. We present our proposed framework in Section IV. The experimental results are discussed in Section V. Finally, the paper ends with conclusions and future work in Section VI.

II. Related Works

A. Convolutional Neural Networks

Although ConvNets [13] were introduced in the 1990s for handwriting recognition, they became widely-known after winning the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [14] in 2012, thanks to the development of computational power and the availability of dataset [15]. Since then, there have been several refinements to improve ConvNet performance, including inception modules [16] and residual modules [17]. Furthermore, with the introduction of the depthwise convolutional concept, MobileNet [18] achieved similar recognition accuracy with fewer parameters, while Xception [19] achieved higher accuracy with similar parameters compared with different famous models in 2017. In 2019, by combining depthwise convolutional operation with compound scaling method, EfficientNet [10] had been released with seven versions for the trade-off between accuracy and computation. By stacking a detection network on top of ConvNets, they were modified for object detection. To avoid confusion, we call ConvNets used for object recognition as recognizers and ConvNets used for object detection as detectors. Similar to recognizers, detector performance has increased significantly after several refinements, including region proposal networks [20], feature pyramid networks [21], and bidirectional feature pyramid networks [11].

To demonstrate the effectiveness of ConvNets, they are usually applied to a vast dataset, such as ImageNet dataset [14] with 1000 classes or COCO dataset [22] with 80 categories. It results in substantial network sizes, which require a high-end GPU to achieve real-time performance. Due to this reason, when applying ConvNets to specific applications, they are customarily modified based on the application requirements. Recently, we introduced AirNet [12], which has been designed to prioritize computational speed for high resolution (1920×1080) input videos for airport airside surveillance. In this project, we adapt AirNet for detecting relevant objects in the apron area and recognizing aircraft types.

B. Monitoring Aircraft Turnaround by Computer Vision

Although several previous works applied computer vision techniques to airport airside environments [7], [8], there has been only one that has monitored aircraft turnaround by using a camera system, namely GAMTOS [9]. The GAMTOS detected different activities by using a Deformable Part Model (DPM) [23] and predicted push-back time using Bayesian prediction in the specific gate area at different times of the day. However, the authors only described activity detection but did not mention anything about push-back prediction. GAMTOS used a sliding window with DPM for binary object classification for activity detection, which was heavily time consuming [24]. Since it used binary object classification, the

Figure 4. Videos collected from a live camera in Tokachi-Obihiro airport with different aircraft types, time and weather conditions.

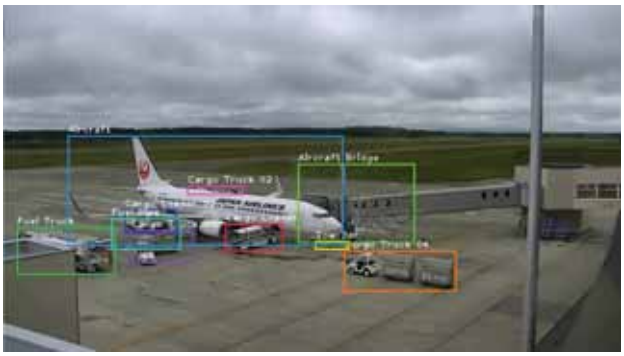

Figure 5. Objects are labelled as bounding boxes. The bounding boxes are heavily overlap which makes both labelling and detecting difficult.

system required five similar detectors to detect five activities, which seems ineffective. Notably, the paper only reported the feasibility study of using computer vision to monitor aircraft TA, as there was no validation in the report.

This paper's primary contributions are designing and validating a computer vision-based framework that implements a state-of-the-art ConvNet architecture to monitor aircraft turnaround activities and provide aircraft push-back prediction in real-time.

III. Data Collection

Videos were collected from a live camera at Tokachi-Obihiro airport, in Hokkaido, Japan[1]. We chose that airport as the videos are recorded continuously from close and with a high view, meaning we can clearly see every object. The camera statically captures boarding gate 3, which has up to four flights per day from Japan Airlines. Furthermore, there are only two types of aircraft, Boeing 737-800 and 767-300. Videos collected in July 2020 are used for training, while videos collected in August 2020 are used for testing. For the training set, we collected 16 videos with lengths of 40 minutes to 60 minutes. Videos were gathered at different times with various weather conditions, as shown in Figure 4. For the testing set, we collect five videos with similar conditions as the training set. To predict the actual push-back time, we collect schedule information from the APM [2] and departure flight schedule information from the airport website[2].

For labeling, we have object ground truths and activity ground truths. We label the following eight objects - aircraft, aircraft bridge, cargo truck, cargo loader, fuel truck, fuel pipe, tow bar, and tow truck. These objects are labeled as bounding boxes, as shown in Figure 5. For the aircraft class, we also label aircraft types, which are Boeing 737 and 767. Besides this, we also label the following nine activities: aircraft arrival, bridge attachment, bridge detachment, cargo loader attachment, cargo loader detachment, fuel pipe drawing in, fuel pipe drawing out, tow truck connection, and push-back as a duration (that is, starting and ending times).

[1]https://www.youtube.com/watch?v=9KPHePhqPRI
[2]https://obihiro-airport.com/flight/

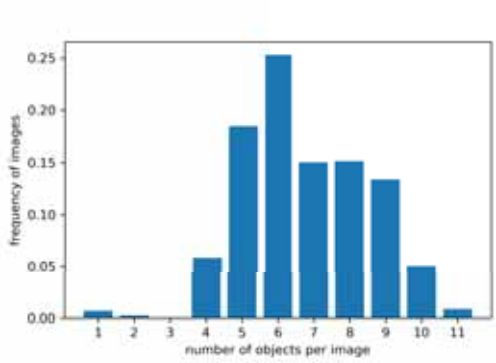

Figure 6. Frequency of number of objects per image. The number of objects per image commonly are from five to nine.

We extract images from videos every five seconds for the training/testing object detector, resulting in 15429 training images and 2593 testing images. In our opinion, this is a very challenging dataset due to several reasons. First, there are commonly five to nine objects per image, and there can be up to 11, as shown in Figure 6. Second, the objects often heavily overlap each other, leading to occlusion. For example, small objects (such as fuel pipe, rear cargo loader, and tow bar) can be occluded by larger objects, as shown in Figure 5. Third, the intra-variance of objects (that is, objects from the same class appearing differently) is high. Remarkably, the Boeing 737 process uses different cargo loaders and trucks compared to the Boeing 767 process. Finally, variations in time and weather conditions change the image brightness and can create noise such as shadows or reflections, as shown in Figure 4.

We similarly extract images to the object detection scheme for training and testing the aircraft type recognition model. Compared to the object detection dataset, the aircraft type dataset is less challenging as the aircraft images are quite large in the FHD images. Also, there are only two classes, which are B737 and B767.

IV. Methodology

Figure 2 describes the proposed framework, which monitors the aircraft TA process and predicts aircraft push-back time. From a video input, ConvNets are applied to detect aircraft and GSE and recognize aircraft types. Next, activities are detected.

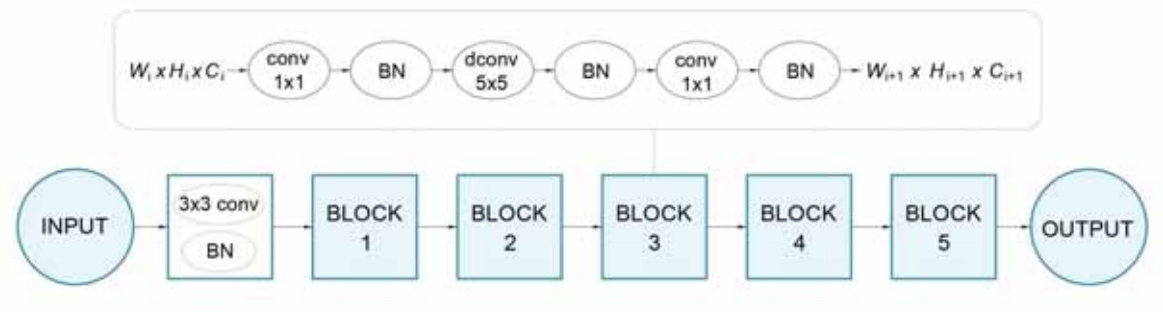

Figure 7. The AirNet recognizer adopted from [12] for aircraft type recognition.

Finally, by combining the detected activities with other data, including TA manual schedules from APM, TA from historical data, and the departure flight schedule from the airport, push-back time is predicted and updated in real-time.

Figure 8 describes the algorithm in detail. First, a detector repeatedly detects aircraft. If an airplane is detected, the aircraft type is recognized, and GSE is also detected. The aircraft type is recognized once if the confidence is high, or several times when the confidence is low. Other data, including the TA manual schedule, TA history data, and departure flight schedule, can be retrieved based on the aircraft type. At the same time, GSE, including aircraft bridge, cargo loader, cargo truck, fuel pipe, fuel truck, tow bar, and tow trucks, are repeatedly detected. From the detected aircraft and GSE, activities (including bridge attachment/detachment, cargo loader attachment/detachment, fuel pipe drawing in/out, tow bar connection, and push-back) are detected. Next, the actual turnaround schedule is updated in real-time. After that, the push-back time is predicted and revised until it occurs. The algorithm finally ends when actual push-back is detected.

In our framework, the following three object categories are excluded from the detection algorithm. First, the framework does not consider objects that do not contribute to the push-back prediction algorithm. For example, cargo unload/load activities can be safely ignored; instead the algorithm needs only to detects cargo loader attachment/detachment. Second, we ignore activities that cannot be captured by the camera, such as passenger deboarding/boarding, service galleys, and service cabin. Third, activities that do not take place in the recorded videos are also excluded, e.g. GSE for service vacuum toilet and service potable water, although they exist in the TA manual.

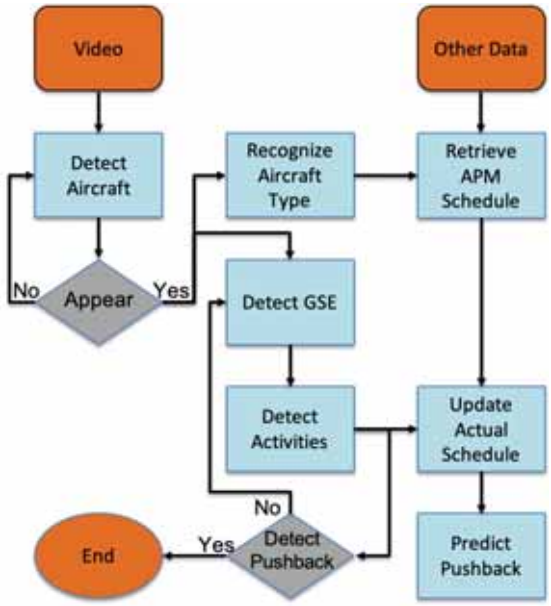

Figure 8. Flow chart of the algorithm. The main process starts when an aircraft is detected and ends when push-back is detected.

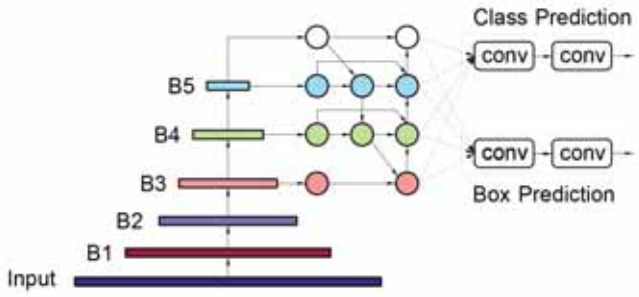

Figure 9. The AirNet detector adopted from [12] for object detection.

A. Aircraft Recognition and Object Detection

To achieve high performance, we adapt the state-of-the-art ConvNet architecture and build two different models: a recognizer for aircraft type recognition and a detector for object detection. As mentioned in Section III, object detection is very challenging, but aircraft type recognition is relatively straightforward. An important point is we want to build a generic framework that can be used in any airport, regardless of aircraft types or GSE. Therefore, to be able to control the framework fully, we design our ConvNets, instead of using pre-defined ConvNets [10], [11]. To this end, the AirNet [12] framework, released for aircraft detection in airport airside environment, is customized for object detection and aircraft type recognition. We use the lightest version of AirNet recognizer [12] with $141,740$ parameters for the recognizer, as shown in Figure 7. The recognizer consists of five identical blocks which exploit depthwise convolutional layers [18]. The most complex version of the AirNet detector [12], with

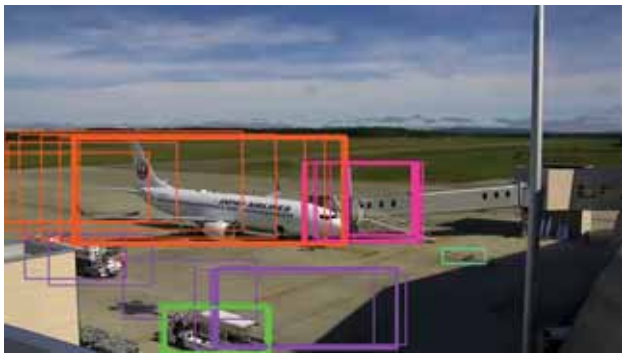

Figure 10. Aircraft tracking by detection. By comparing object bounding boxes and classes from previous frames, the algorithm can map them with the current frame.

$1,078,662$ parameters, is applied for the detector, as shown in Figure 9. The detector combines a recognizer with a bidirectional feature pyramid network [11].

After objects are detected in each frame, they are tracked by mapping through the video, creating a sequence of bounding boxes for activity detection. Two factors are considered, classes and positions, when mapping objects. Specifically, objects from consecutive frames are considered the same object if they are from the same category and their positions overlap each other. Figure 10 shows the results of this tracking process, in which different colors represent different objects[3].

For training the models, the training images are split into training and validation sets, in a ratio of 9:1, respectively. Therefore, we have 13887, 1542, and 2593 images in the training, validation, and test sets. Two models are trained similarly with the binary (sigmoid) loss for the recognizer and categorical (softmax) loss for the detector. We use Adam optimization with learning rate 10^{-3}. We also implement a learning schedule and an early stop mechanism for the experiments. For the learning schedule, we reduce the learning rate by a factor of ten once validation loss no longer decreases in a period of three epochs. If validation loss does not drop in a period of ten epochs, then the training is stopped.

B. Activity Detection

Activity detection requires the relationship between the involved objects and the aircraft. For example, to detect bridge attachment/detachment, the position of bridge and planes are needed. We extract two pieces of information from the object position: speed and intersection over union (IOU) of the involved objects and the aircraft. To calculate object speed, we subtract the center points of the object bounding boxes over time. As we detect objects every second, the speed unit is the number of pixels per second. The IOU is calculated by the ratio between the overlapping area and the area of the union of the two objects. The left chart of Figure 11 shows an example of speed (green) and IOU (blue) over time. Because object bounding boxes are not stable over time, the speed and IOU exhibit fluctuation.

[3]https://www.dropbox.com/s/tze14zctbd7bblu/Object.avi?dl=0

Algorithm 1: Attachment/Detachment Detection

Result: Attachment at t_0 and Detachment at t_1
1 Calculate speed and IOU;
2 Initialize small number ϵ_1, ϵ_2;
3 t $\leftarrow$ 0;
4 **while** *True* **do**
5 **if** *speed[t]* $\geq \epsilon_1$ **then**
6 break
7 **end**
8 t $\leftarrow$ t + 1;
9 **end**
10 **while** *True* **do**
11 **if** *speed[t]* $\leq \epsilon_1$ *and IOU[t]* $\geq \epsilon_2$ **then**
12 break
13 **end**
14 t $\leftarrow$ t + 1;
15 **end**
16 $t_0 \leftarrow$ t;
17 **while** *True* **do**
18 **if** *speed[t]* $\geq \epsilon_1$ **then**
19 break
20 **end**
21 t $\leftarrow$ t + 1;
22 **end**
23 $t_1 \leftarrow$ t;
24 Return t_0, t_1;

The first and easiest activity is the arrival of an aircraft. An aircraft arrival happens when an aircraft's speed is close to zero, indicating the aircraft has stopped moving. Next, bridge attachment/detachment, cargo loader attachment/detachment, tow truck connection, and push-back are detected by Algorithm 1. First, speed and IOU are calculated. Then, for providing the resistance to fluctuation, we initialize two hysteresis thresholds for speed and IOU, namely ϵ_1 and ϵ_2. The loop from lines 4 to 9 detects the time that an object starts moving. The loop from lines 10 to 15 detects the time the object stops and overlaps with the aircraft, which signifies attachment, as shown on the upper right chart of Figure 11. The loop from lines 17 to 22 detects the time the object starts moving again, which signifies detachment, as shown on the lower right chart of Figure 11. Tow truck attachment is also detected from lines 10 to 15, while push-back is detected from lines 17 to 22.

As the fuel pipe is tiny and heavily overlapped, this object's detection accuracy is lower than for other objects. Fortunately, the fuel pipe cannot randomly appear in view. Therefore, the fuel pipe drawing in/out activities are detected as the first/last time the fuel pipe appears.

Activities are detected[4] and displayed in real-time, as shown in Figure 12. The timer begins to count at the moment that an aircraft arrives. Then, activities start when the attachment is detected and end when detachment is detected. The display ends when push-back is detected. The fuel pipe drawing

[4]https://www.dropbox.com/s/uoe9y5kbbkauvyh/Activity.mp4?dl=0

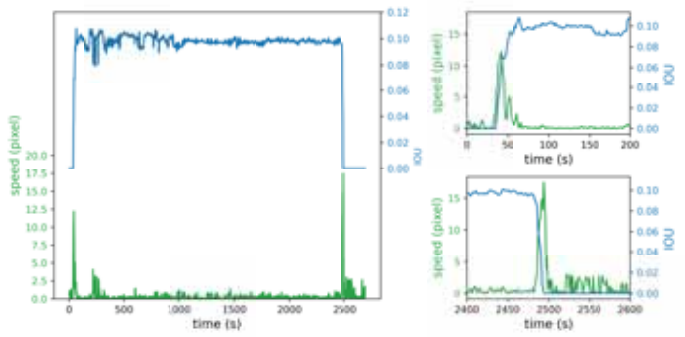

Figure 11. Activity detection is based on speed (green) of a involved object and intersection over union (blue) of the involved object and the aircraft. The left figure shows the whole process while the upper right and lower right focus on first 200 seconds and last 200 seconds respectively.

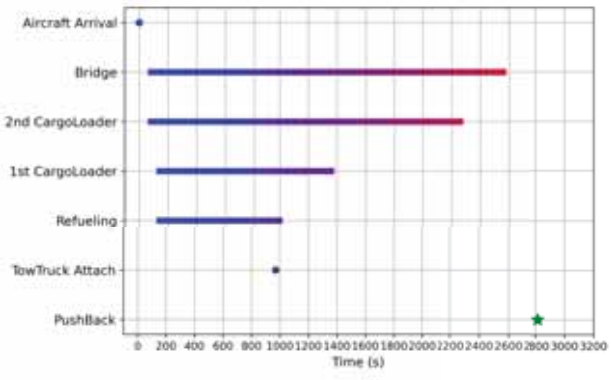

Figure 12. Detected activities is displayed by the framework starting when an aircraft arrives (second 0) and ending when an aircraft pushes back (green star) over time (from blue to red).

in/out signifies refueling. We do not distinguish between aft or forward cargo; therefore, we display those as the first cargo and the second cargo. Activities that have attachment and detachment are merged as a duration.

C. Push-back Prediction

From the TA history and manual, we establish the relationship between the time required to finish activities and the push-back time. Based on the TA history time, we calculate the average time to complete each activity and the ratios of these average times to the actual push-back time, as shown in Table I. Additionally, the predicted push-back time could not be shorter than the manual push-back time from APM.

The predicted time is updated in real-time, as shown in Figure 13. First, it is initialized to the departure flight schedule on the airport website. It remains at this value until the first activity (refueling) finishes. Since the refueling activity ends early, the predicted push-back time drops significantly. After

TABLE I. RATIO BETWEEN FINISH TIME OF ACTIVITIES AND ACTUAL PUSH-BACK TIME.

Activities	Bridge	1st Cargo Loader	2nd Cargo Loader	Refueling
Ratio	0.9263	0.5550	0.8319	0.3392

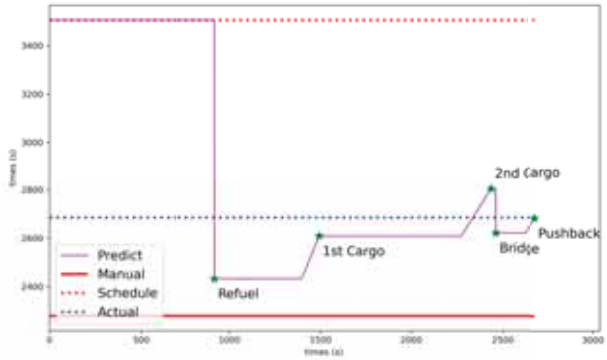

Figure 13. Push-back time prediction update every second. The predicted time is initialized as schedule time and updated based on the completion of detected activities.

that, the predicted time slowly increases as the first cargo loader does not finish early, as refueling did. After that, it slightly increases as the second cargo loader takes more time to complete than it should be. Then, the predicted time suddenly drops because the bridge detaches early. Finally, the predicted time increases since the actual push-back may occur later than the estimated one.

V. RESULTS

A. Aircraft Recognition and Object Detection

As mentioned in Section IV, we use the lightest version (i.e., a minimal number of trainable parameters) of AirNet for the aircraft type recognizer. At this minimal configuration, the recognizer yields an accuracy of 100% in detecting two aircraft types. If there are more than two aircraft types, high detection accuracy is still achievable by using a more heavyweight version of AirNet, i.e., introducing more trainable parameters.

Figure 14 shows the precision-recall curves of the detection for different objects. A high recall detector aims to detect as many objects as possible, which helps to reduce false negative. In contrast, a high precision detector aims to detect objects as precise as possible, which helps to reduce false positive. A precision-recall curve indicates the trade-off between precision and recall, which is important in determining the detection threshold. In Figure 14, the top-right corner is the most desirable region, where the detector is able to achieve high precision and high recall simultaneously. Since different thresholds are associated with different precision and recall values, Average Precision is an important metric that calculates the average precision over all possible thresholds. In other words, Average Precision, ranging from 0 to 1, indicates the detector's performance regardless of detection thresholds (i.e., the higher Average Precision the better performance). Table II shows the results of object detection, achieving a high-performance level with a mean Average Precision of 0.9514. Large objects (such as aircraft, bridge, tow truck, and fuel truck) are detected with very high precision. Cargo loader and cargo truck are detected with lower precision than other large objects as they suffer from intra-variance and overlapping. Due

TABLE II. AVERAGE PRECISION OF AIRCRAFT AND DIFFERENT GROUND SUPPORT EQUIPMENT.

Class	Aircraft	Bridge	Cargo Loader	Cargo Truck	Fuel Pipe	Fuel Truck	Tow Bar	Tow Truck	Mean
Average Precision	0.9996	0.9998	0.9304	0.9143	0.9028	0.9989	0.8663	0.9988	0.9514

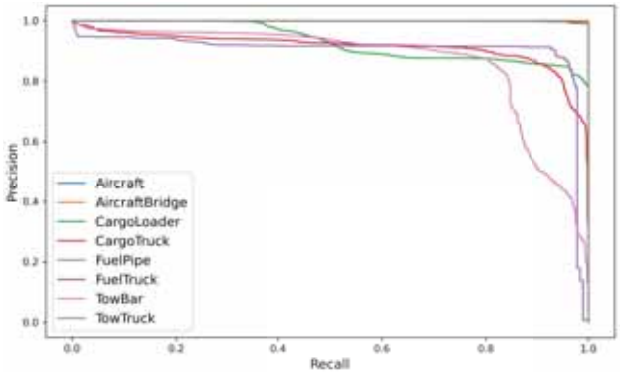

Figure 14. Precision-recall curves of the detection for different objects. The area under each curve indicates the performance of the corresponding detector (the larger area the better accuracy).

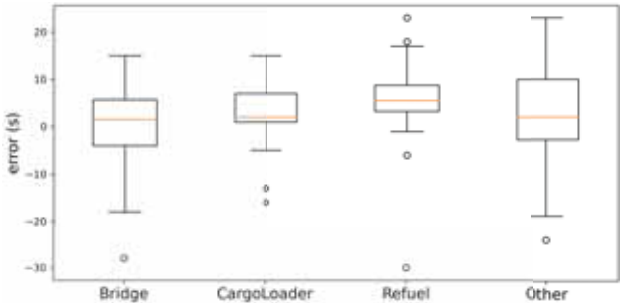

Figure 15. The errors between detected and ground truth activities. Negative errors indicate activities are detected earlier than ground truth, and vice versa.

to their small sizes and heavy overlapping by larger objects, the fuel pipe and tow bar have the lowest detection precision. However, this does not affect the activity detection as we use different methods to detect refueling, and the tow bar does not contribute to the activity detection.

B. Activity Detection

As ground truth activity time is labeled as a duration, while the detected activity time is estimated instantly, we normalize the ground truth activity time using the middle point of that duration. We divide nine activities into four groups, namely bridge (bridge attachment/detachment), cargo loader (cargo loader attachment/detachment), refueling (drawing in/out of fuel pipe), and other (aircraft arrival, tow truck connection, and push-back).

Figure 15 shows the errors between detected and ground truth activities represented as box plots. The errors are calculated by subtracting the predicted activity time from the ground-truth, as we want to keep both the magnitude and sign. Therefore, negative errors indicate activities that are detected earlier than the ground truth, and positive errors indicate activities that are detected later. As can be seen, the median errors for the Bridge, Cargo Loader, and Other groups range from 1.5 to 2 seconds. As the fuel pipe has low detection performance, refueling errors are not stable, resulting in more outliers. It results in a median error of the refueling time, only 5.5 seconds, which is still accurate compared to the overall turnaround duration.

C. Push-back Prediction

As there are only five testing videos, we display all these videos' prediction results in Figure 13 and Figure 16. Based on the five videos, one can see that the departure flight schedule times are very different, ranging from 2855 seconds to 3898 seconds compared to 2280 from the manual APM. Interestingly, the difference between actual and scheduled push-back times seems to be variable, regardless of duration. For example, in the two figures with actual times of approximately 2700 seconds, the scheduled times range from 2917 to 3506 seconds; while in the two figures with schedule times of about 3600 seconds, the actual times range from 2686 to 3446 seconds.

Moreover, there is no standard process for aircraft turnaround. For example, in the upper right chart of Figure 16, refueling time and first cargo loader time are similar to each other, while in the lower-left chart of Figure 16, the bridge detaches before the second cargo loader does. The framework can get more accuracy in predicting push-back, compared to the scheduled time, from the time of the first activity completion (refueling) in two of the five videos. Among these two videos, it can more accurately predict push-back from the time the second cargo loader detaches in one video and from the bridge detaches in the remaining video compared to the scheduled time. It results in a more accurate push-back prediction being available in four out of the five videos.

VI. CONCLUSION

We have proposed a novel computer vision-based framework that can monitor the aircraft turnaround process, including object detection, activity detection, and push-back prediction. By using ConvNets, aircraft type recognition achieves 100% accuracy, while object detection achieves 0.9514 mean Average Precision. For activity detection, the median error is smaller than 6 seconds, which can be considered very low in the overall turnaround duration. Furthermore, the framework provided more accurate push-back prediction that the airport schedule in four out of five, or 80% of cases.

In the future, we intend to further improve push-back prediction time by collecting Gate CAM videos from different airports and incorporating information from the A-CDM system such as Target Start-up Approval time and runway configuration and availability. It may lead to a reduction in

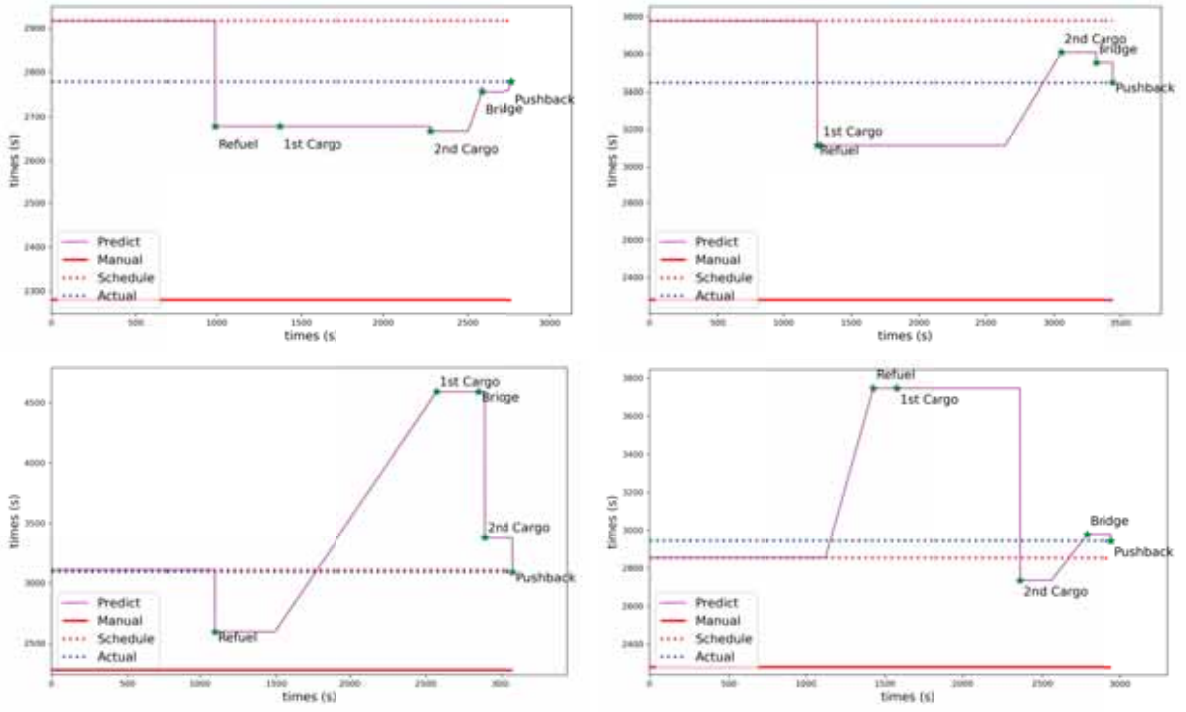

Figure 16. Push-back time prediction of different TA processes. The difference between actual and schedule push-back times seems to be variable, regardless of duration.

the aircraft waiting time at the runway holding points, reduce fuel consumption on the taxiways, and improve the passenger experience by having a smoother departure flow.

Acknowledgment

This work was conducted under the Saab-NTU Joint Lab with support from Saab AB, Saab Singapore Pte Ltd., and Air Traffic Management Research Institute, Nanyang Technological University, Singapore.

References

[1] M. Schmidt, "A review of aircraft turnaround operations and simulations," *Progress in Aerospace Sciences*, vol. 92, pp. 25–38, 2017.

[2] Boeing. (2011) Airplane characteristics for airport planning. https://www.boeing.com/commercial/airports/plan_manuals.page.

[3] M. M. Mota *et al.*, "Simulation-based turnaround evaluation for lelystad airport," *Journal of Air Transport Management*, vol. 64, pp. 21–32, 2017.

[4] S. Okwir *et al.*, "Managing turnaround performance through collaborative decision making," *Journal of Air Transport Management*, vol. 58, pp. 183–196, 2017.

[5] M. Schultz and S. Reitmann, "Machine learning approach to predict aircraft boarding," *Transportation Research Part C: Emerging Technologies*, vol. 98, pp. 391–408, 2019.

[6] B. Oreschko *et al.*, "Turnaround prediction with stochastic process times and airport specific delay pattern," in *International Conference on Research in Airport Transportation (ICRAT), Berkeley*, 2012.

[7] A. Koutsia *et al.*, "Automated visual traffic monitoring and surveillance through a network of distributed units," in *ISPRS*. Citeseer, 2008.

[8] N. Pavlidou *et al.*, "Using Intelligent Digital Cameras to Monitor Aerodrome Surface Traffic," *IEEE Intelligent Systems*, vol. 20, pp. 76–81, 2005. [Online]. Available: http://ieeexplore.ieee.org/document/1439483/

[9] H.-L. Lu *et al.*, "Airport gate operation monitoring using computer vision techniques," in *16th AIAA Aviation Technology, Integration, and Operations Conference*, 2016, p. 3912.

[10] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *arXiv preprint arXiv:1905.11946*, 2019.

[11] M. Tan *et al.*, "Efficientdet: Scalable and efficient object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 781–10 790.

[12] P. Thai *et al.*, "Deep4air: A novel deep learning framework for airport airside surveillance," 2020.

[13] Y. LeCun *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[14] J. Deng *et al.*, "Imagenet: A large-scale hierarchical image database," in *IEEE conference on computer vision and pattern recognition*, 2009.

[15] A. Krizhevsky *et al.*, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, May 2017. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3098997.3065386

[16] C. Szegedy *et al.*, "Going Deeper with Convolutions," *arXiv:1409.4842 [cs]*, Sep. 2014, arXiv: 1409.4842.

[17] K. He *et al.*, "Deep Residual Learning for Image Recognition," *arXiv:1512.03385 [cs]*, Dec. 2015, arXiv: 1512.03385. [Online]. Available: http://arxiv.org/abs/1512.03385

[18] A. G. Howard *et al.*, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint:1704.04861*, 2017.

[19] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[20] S. Ren *et al.*, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[21] T.-Y. Lin *et al.*, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[22] ——, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014.

[23] P. F. Felzenszwalb *et al.*, "Object detection with discriminatively trained part-based models," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 9, pp. 1627–1645, 2009.

[24] P. Sermanet *et al.*, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *arXiv preprint:1312.6229*, 2013.