IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 9 June 2024, accepted 4 July 2024, date of publication 9 July 2024, date of current version 18 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3425572

RESEARCH ARTICLE

A Computer Vision-Based Standalone System for Automated Operational Data Collection at Non-Towered Airports

MOHAMMAD FARHADMANESH, ABBAS RASHIDI, (Member, IEEE), ABHISHEK KUMAR SUBEDI, AND NIKOLA MARKOVIĆ
Department of Civil and Environmental Engineering, The University of Utah, Salt Lake City, UT 84112, USA
Corresponding author: Mohammad Farhadmanesh (mohammad.farhadmanesh@utah.edu)

This work was supported in part by Utah Department of Transportation under Grant UT20.605, in part by the Airport Cooperative Research Program, and in part by the Mountain-Plains Consortium under Grant MPC-639.

ABSTRACT The accurate collection of operational data at airports is essential for ensuring the fair distribution of national funds. However, many U.S. airports lack control towers, which forces planners to rely on sound, radio, and transponder-based systems for detecting aircraft operations. While these methods are useful, they have limitations, such as low accuracy and an inability to identify specific aircraft. In our previous work, we developed a computer vision-based system capable of accurately counting and identifying aircraft. However, implementing this system requires powerful computing devices that are not typically found at non-towered airports. Additionally, cloud computing is not a viable option due to data transfer limitations and associated costs. To address these challenges, we propose an affordable solution utilizing edge computing. This paper describes the necessary software and hardware modules, including optimized machine learning methods and edge devices, for deploying the system at airports. We tested the system's performance using two independent standalone setups, created with NVIDIA edge kits, at three non-towered airports. The first setup aimed to obtain an accurate count of operations without detailed aircraft information, while the second setup was designed to extract an operations count with comprehensive aircraft information, including aircraft type recognition and identification. The accurate retrieval of aircraft information in the edge computing system is achieved through the introduction of a tailored CNN-based recognition model and the execution of a 3-step tail number identification algorithm. The results demonstrate the practical value of the proposed system, indicating that an accurate count of operations with detailed aircraft information can be obtained at a reasonable cost.

INDEX TERMS Edge computing, airport operations count, aircraft identification, computer vision, intelligent system.

I. INTRODUCTION

The majority of airports worldwide do not have control towers [1]. In the United States, for instance, there are nearly 20,000 non-towered airports compared to approximately 500 airports with control towers [2]. Control towers play a crucial role in identifying and documenting aircraft operations while efficiently coordinating aircraft and vehicle movements on the airport field [3]. However, at non-towered airports, alternative methods must be employed to ensure air traffic safety and security due to the absence of a reliable air traffic monitoring system [4]. The collected operations data from non-towered airports serve as a valuable foundation for resource allocation in airport industries, airport planning, staffing, and environmental analysis [5], [6]. Moreover, these records aid in estimating air pollution and noise levels associated with aircraft activities in the vicinity of non-towered airports [7].

The associate editor coordinating the review of this manuscript and approving it for publication was Rosario Pecora.

Developing intelligent systems that can automatically document airport activities and provide operational data to managers is crucial for non-towered airports. Currently, airport practitioners commonly rely on automated acoustical counters, radio click counters, and transponder-based counters [6]. However, these systems lack accuracy and identification capacity. In response, the authors have proposed a computer-vision-based method capable of accurately counting and identifying aircraft operations at non-towered airports [8], [9], [10]. The implementation of these algorithms requires powerful graphics processing units (GPUs) to handle the computational intensity. To evaluate the system's performance, the authors conducted tests by transmitting video data to a computer center located outside the airport. However, this approach encounters two challenges. Firstly, Wi-Fi availability is limited in the necessary areas of non-towered airport airfields, making it difficult to transmit data to remote servers. Additionally, the alternative solution of cellular data transmission can be cost-prohibitive. Secondly, the roundtrip delay of data transmission does not favor certain system applications, particularly those related to safety and security that require near-instantaneous decision-making.

This paper proposes an edge computing standalone system to address the issues mentioned above, which eliminates the need for large data transmission. The contributions of this paper are as follows:

- The development of an intelligent standalone system for fully-automated airport operation monitoring using NVIDIA developer kits. The system uses affordable hardware and fits within the typical budget for collecting operational data at non-towered airports.
- The optimization of the system software for efficient real-time operability on edge computing devices. Practical guidelines are provided for the development of optimized algorithms to obtain *operations count* and *aircraft information.*
- The development of an optimized custom convolutional neural network (CNN) classifier model for aircraft recognition. The classifier model is specifically designed with a focus on efficiency and precision when integrated with the edge computing system.
- The implementation of a 3-step identification process (Figure 1) on edge computing devices that employs a probabilistic method. This process utilizes predicted label sequences obtained from multiple video frames, ensuring precise tail number identification.

II. EXISTING SYSTEMS

Considering the type of sensors used, the current counter systems encompass automated acoustical counters, radio-based counters, and transponder-based counters. Additionally, we will focus on reviewing the relevant vision-based systems.

A. AUTOMATED ACOUSTICAL COUNTER

This device records the operations count by detecting the loud sound generated by aircraft engines during takeoff [11].

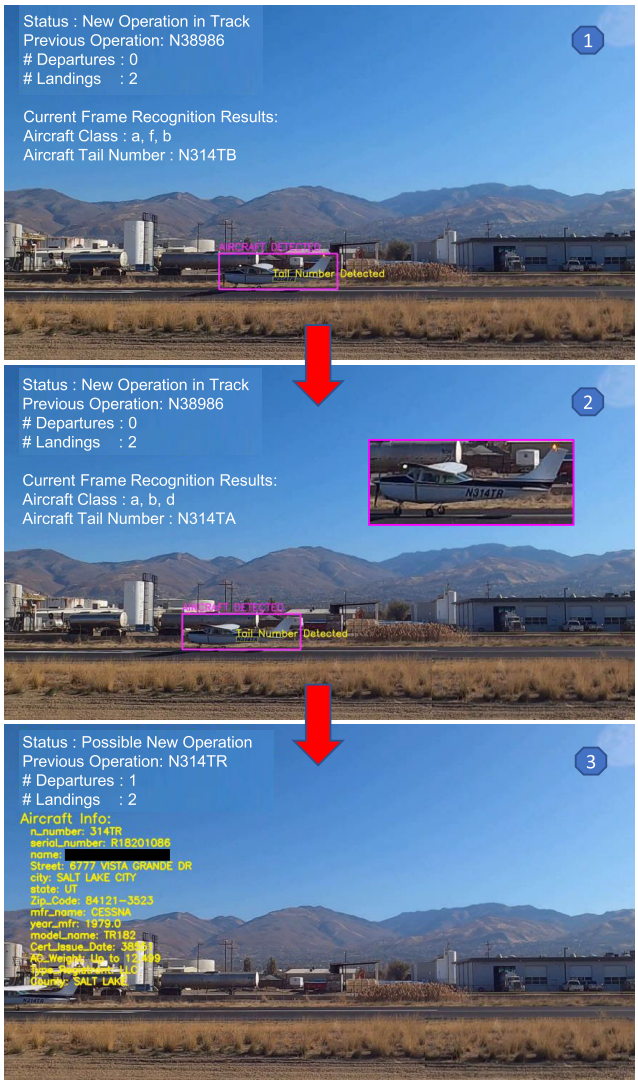


FIGURE 1. Extracted video frames from the proposed 3-step identification process.

The performance of acoustic counters is heavily dependent on their placement, requiring adherence to strict performance standards [12]. Consequently, it does not provide an accurate count for landing operations. Additionally, in the case of aircraft equipped with quiet engines, an acoustical counter may fail to detect and register the operation [6]. Weather conditions also present a significant challenge, impacting both the performance and accuracy of these devices [13]. Moreover, at airports where taxiways and taxilanes are in close proximity to each other, this system may result in an overcounting of aircraft operations [14]. Importantly, processing engine sounds alone cannot determine the specific identity of the operating aircraft. Research at Purdue University Airport highlighted the system's limitations, as it captured only 59.72% of 34,051 aircraft operations during a four-month period [11]. In addition, accurate operation counts necessitate the interpretation of sounds to distinguish between departures and non-departures [12].

B. RADIO-BASED COUNTER

A recent approach to counting airport operations involves recording the radio messages exchanged between aircraft pilots and the airport's universal communication (Unicom) station. In this method, the total operations count is estimated by dividing the overall number of radio messages by the average number of radio message transmissions for departure and landing operations. However, this average number can vary significantly depending on factors such as the pilot, airport location, and specific situation. Consequently, this method often produces inaccurate annual operations counts [15]. Another limitation of radio-based counters is the absence of an identification module comparable to acoustical counters. Additionally, the Unicom station frequencies at non-towered general aviation airports typically range between the 122 and 123 Mega Hz series. As a result, some adjacent airports might share the same frequency, leading to unreliable airport operational data. They are also susceptible to interference from other radio frequency signals, which can lead to data inaccuracies [16].

C. TRANSPONDER-BASED COUNTER

The spatiotemporal proximity of operating aircraft to the airport can be determined by decoding the signals transmitted through the aircraft transponder. Data-driven techniques have been developed to utilize this approach and detect aircraft operations at non-towered airports [17], [18]. Additionally, Mode S transponders utilize Automatic Dependent Surveillance-Broadcast (ADS-B) technology to identify the aircraft. Despite the promising nature of this technology, a significant number of general aviation aircraft are not yet equipped with the required transponders [19]. Furthermore, even among the equipped aircraft, many do not have Mode S transponders [20], which are necessary for accurate aircraft identification. Consequently, transponder-based counters have a limited capacity for aircraft identification. It is worth noting that most non-towered airports are not located within the airspace where the FAA mandates the use of ADS-B-enabled avionics [6]. Weather conditions significantly affect the performance of transponder-based counters by introducing variations in atmospheric pressure, leading to inaccurate altitude readings. Consequently, this can result in inaccurate operation counts, necessitating calibration to maintain accuracy [21]. Additionally, another challenge in employing transponders for navigating and identifying small aircraft is their vulnerability to signal interference [22].

D. VISION-BASED SYSTEMS

In an attempt to supplement the ADS-B technology for operations count, the system incorporated video/image detection (VID). This addition aimed to recognize aircraft registration numbers during taxiing into the centralized terminal, initially developed to automate landing fee billing [6]. The system aimed to utilize the capabilities of VID/ADS-B by utilizing supplemental FAA electronic-based near real-time traffic data from the National Airspace System (NAS), known as the Aircraft Situation Display to Industry (ASDI), as indicated in the Airport Cooperative Research Program (ACRP) Report 129 [6]. However, despite these efforts, the VID component proved unsuccessful in performing the operations count task. The ACRP Report 129 outlines several reasons for this failure, including the dependency on onboard ADS-B transponders for accurate counting, the high cost associated with system setup, and the limited applicability to airports with restricted access to the terminal area. A similar system, known as Vantage, combines radar flight data with registration numbers obtained through camera reading [23]. Another vision-based technique is apron surveillance. Additionally, apron surveillance has been explored as a vision-based technique. Thirde et al. [24] proposed a multi-camera system to monitor the activities of vehicles and personnel during airplane servicing operations at the apron. Similarly, Koutsia et al. [25] employed background extraction and data fusion to establish an activity-tracking network using these cameras and a central server. However, background extraction alone is insufficient for accurately detecting and differentiating between aircraft and service vehicles in general aviation airports due to their small sizes.

To address these issues, we propose a passive standalone system that performs aircraft operations count, including aircraft detection, tracking, and trajectory analysis, as well as aircraft identification through tail number detection and recognition, using efficient computer vision techniques and cost-effective edge computing devices. In addition, our system incorporates a deep neural network to classify the aircraft type in cases where the tail number is not visible or not imprinted on the aircraft fuselage. We also use a probabilistic aircraft identification procedure that builds upon techniques developed by Molina et al. [26] and Vidakis and Kosmopoulos [27], which both focus on feature-based optical character recognition (OCR). VaxOCR [28] is a similar commercially available technology for OCR services.

E. INTELLIGENT MODELS

To improve the performance of deep learning models for specific applications, various methods have been explored. The use of the sparrow search algorithm in combination with Support Vector Regression for parameter correction enhances deep learning model performance by refining their ability to accurately estimate parameters [29]. Similarly, the application of an improved YOLO v-5 algorithm demonstrates significant advancements in the field of object detection, specifically in detecting mineral zoning in spiral slope flows [30]. Additionally, the adoption of probabilistic prediction methods for estimating the remaining useful life of components using deep learning underlines the adaptability and effectiveness of probabilistic approaches in predictive analytics [31]. Furthermore, the development of low-rank tensor regularized graph fuzzy learning methods significantly advances multi-view data processing, enhancing clustering

accuracy by effectively managing high-dimensional multi-view data in consumer electronics [32]. Moreover, tensor voting methods have been applied to infer human mobility traces from incomplete and noisy data, demonstrating their effectiveness in recovering missing information and extracting meaningful patterns [33].

In this research, we explore the Hyperband search algorithm's application for optimizing convolutional layers in an aircraft recognition model, aiming to improve classification capabilities. For aircraft detection, the Haar cascade classifier is utilized, recognized for its efficiency in object detection within images. To further refine aircraft identification, we employ a convolutional recurrent neural network (CRNN) along with a probabilistic sequential prediction approach. This method leverages the strengths of CRNNs in processing sequential data and probabilistic methods in enhancing prediction accuracy, offering a robust solution for the challenges of aircraft identification. Through these explorations, our study further contributes to the expanding field of deep learning, illustrating the impact of tailored optimization and predictive modeling techniques in addressing specific application needs.

Deep learning technologies are applied across a wide array of domains, showcasing their adaptability and efficiency in solving complex problems. This approach has revolutionized the way we handle disease detection and classification, notably in agriculture for identifying plant diseases, thus making feature extraction more objective and reliable [34]. Similarly, advancements in CNN have led to significant improvements in detecting diseases in apple leaves, offering practical solutions in the agricultural sector [35]. Beyond agriculture, deep learning has also been instrumental in healthcare, specifically using segmentation and visualization techniques for Tuberculosis detection through chest X-rays [36]. Additionally, this technology has been leveraged in the energy sector to estimate photovoltaic output using sky images as input through innovative CNN methods [37]. Its application extends to aiding in the identification of mineral zoning in spiral slope flows [30]. The development of efficient and interpretable AI engineering products is crucial for achieving more substantial outcomes and enhancing customer acceptance of AI applications [38]. Our research utilizes deep learning for the novel purpose of counting and identifying aircraft at non-towered airports, further underscoring the broad applicability and potential of deep learning in addressing diverse challenges.

III. EDGE COMPUTING VERSUS CLOUD COMPUTING

There are two primary approaches to operating intelligent systems for decision-making: edge computing and cloud computing. Cloud computing-based intelligent systems provide computing services, including servers, databases, storage, networking, software, and analytics, through the internet. In this approach, data is collected by sensors and transmitted to the cloud for processing. While cloud computing offers benefits such as potentially unlimited processing power, it has limitations in terms of latency, bandwidth, and privacy concerns [39]. Additionally, data transmission costs can be prohibitively high in certain cases. In contrast, edge computing presents an alternative by enabling data storage, processing, and analysis at the network's edge [40]. Edge computing is a viable solution for achieving real-time stability and executing specific functions within a particular field [41]. The advantages of edge computing are as follows:

- Reduced data transmission costs by decreasing the size of the transmitted data. Only the final results of data processing (i.e., aircraft information, times, etc.) are transferred to the servers.
- Faster decision-making by reducing the response time [42] and removing the roundtrip data latencies that exist in the cloud computing approach.
- Secure data at the network edge (i.e., airport).

TABLE 1. Approximated cost of data transmission for a general aviation airport with normal traffic.

Factor	Remark	Total per Year
Operations Count	100 operations per day	36,500 operations
Video Data Size	80 MB per operation	2,920 GB
Cellular Data Transmission Cost	$15 per 1 GB data	$43,800.00

Note: The costs are obtained from the Verizon [43] carrier. Operational data from Skypark Airport [44] in Utah is used. Also, it is assumed that the cameras only transmit video data when a motion (i.e., possible aircraft operation) is detected.

In terms of expenses, cloud computing involves transmitting data to the associated cloud processing platform, which is typically achieved through a family of wireless network protocols known as Wi-Fi. However, non-towered general aviation airports often lack Wi-Fi coverage within their airfield, making cellular data service the only option. Table 1 provides estimates of the total cost of data transmission using cellular service and outdoor pan-tilt-zoom (PTZ) cameras with a motion detection module. To minimize the data transmission rate, we assumed that only video data linked to motion events are transmitted to the cloud processing platform. Despite this, the overall cost is exorbitant and does not warrant the use of a cloud computing setup. It is noteworthy that airports are dynamic environments, and many possible motions other than aircraft operations, such as nearby highway traffic at local airports, service vehicles and personnel, wildlife, and camera motions due to wind, can increase the cost of cloud computing several times more than what is calculated in Table 1.

IV. INTELLIGENT EDGE COMPUTING SYSTEM

The intelligent system is composed of both software and hardware components, which must be compatible with each other. Thus, the proposed intelligent edge computing system is specifically designed, taking into account the limitations of both software and hardware elements. The system is structured with efficient machine learning models

and utilizes NVIDIA edge computing processors to optimize performance. In the subsequent sections, we will provide a detailed explanation of how we developed a portable system capable of measuring aircraft operations in real-time at the network edge, where the sensors are located.

A. SOFTWARE

In this study, camera sensors are strategically positioned at entrance taxiways, designated locations for data collection. By providing coverage of these passages, the system can accurately measure arrival and departure operations on each runway. Figure 2 illustrates the camera layout implemented in a single-runway airport. In Farhadmanesh et al. [8], [9], we provide detailed explanations on how this camera layout can effectively cover aircraft operations in multi-runway airports, including intersecting and parallel runways. The two layouts are considered as Layout 1 and Layout 2. In Figure 2, the cameras from Layout 1 are labeled as 'S1', and those from Layout 2 are labeled as 'S2.'

Layout 1 features end of runway cameras, strategically placed at runway ends where pilots typically begin take-offs, enhancing safety by allowing for emergency aborts. These cameras capture take-off and landing activities. Ideally positioned beyond any bypass taxiway, they ensure comprehensive coverage of departures. In multi-runway airports, they are set up to avoid field-of-view overlap, with each camera focused on its respective runway.

In Layout 2, two types of cameras are employed for specific monitoring purposes. The first is positioned at the end entrance taxiways, capturing both the departure and arrival operations. The second camera type in Layout 2 is located at the mid entrance taxiways. Its primary function is to monitor arrival operations. This setup is potentially useful for observing intersection departures. In this study, layout 2 was the preferred choice for conducting our experiment. This selection is attributed to the layout's capability to allow edge devices additional time to detect aircraft, particularly as the aircraft's operational speed is slower when approaching these devices.

Our previous study tested the design of the edge nodes and experimented with them at five general aviation airports [8]. Each layout demonstrated a high level of precision in tracking operations independently, as detailed in Table 2. This table includes data from 21 collection sessions–18 during the day and 3 at night. While Layout 1 failed to record only 4 out of 378 operations, Layout 2 successfully documented all 149 operations. Notably, Layout 2 also possesses the capability to differentiate between arrivals and touch-and-go maneuvers. The data from these experiments are available in the previous study [45]. Figure 3 displays the placement of edge nodes in each camera layout for more complicated scenarios at an airport with a multi-runway configuration.

1) OPERATIONS COUNT

To ensure accurate tracking of the number of operations, the intelligent system integrates three modules in the specified order: (1) Aircraft Detection Module, (2) Aircraft Tracking Module, and (3) Trajectory Analysis Module. Figure 4 provides a visual representation of the sequential integration of these modules within the system.

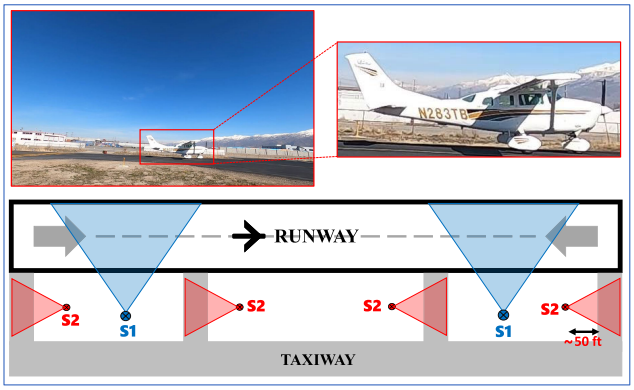


FIGURE 2. Camera layout on a single-runway airport. "S1" stands for Sensor for Layout 1 and "S2" stands for Sensor for Layout 2.

Aircraft Detection Module: The intelligent system is designed to activate only when motion is detected. To detect these motions within the field of view, we employ an adaptive background-foreground extraction algorithm called Gaussian Mixture model [46]. This algorithm identifies any captured movements. The parameters, as well as the number of mixture components, are continuously adjusted for each pixel [46]. Subsequently, an object detection model is utilized to localize any aircraft present in the field of view. Specifically, we utilize a pre-trained model of SSD [47] based on the MobileNetV2 architecture [48]. This makes the model more efficient and suitable for edge devices with limited computational resources. The SSD-Mobilenet-v2 model is trained on the 90-class Microsoft COCO dataset, which includes the aircraft object [49]. The dataset consists of images that capture complex daily scenes with common objects in their natural surroundings. The amount of identified instances per image supports the learning of contextual data. For faster inference on edge computing platforms, this object detection model is further optimized using NVIDIA TensorRT. TensorRT enables the optimization of neural network models trained on deep learning frameworks on separate host machines. By leveraging a combination of neural network layers and kernel optimization, TensorRT improves latency, memory consumption, and network throughput (refer to Figure 5).

Aircraft Tracking Module: The detected aircraft is tracked by the intelligent system for as long as it remains in the field of view. The proposed system uses a fast computing correlation-based object tracker, named MOSSE, to find the aircraft's position once it is detected. MOSSE, by calculating the correlation computation in the Fourier domain, accelerates the tracking process [50], which is necessary for real-time monitoring systems. Using geometric invariant features, MOSSE is robust with respect to variations in pose

TABLE 2. Distribution of the recorded video data of the aircraft operations based on the camera layout.

Layout	Total observed by human	Total captured by edge nodes	Fixed-wing single-engine (FWSE)	Fixed-wing multi-engine (FWME)	Missed operations camera view error (long landings)	Missed operations Layout 1 error (intersection departures)
Layout 1	378	374 (98.9%)	353	21	3 FWSE (out of 274 landings)	1 FWSE (out of104 departures)
Layout 2	149	149 (100%)	140	9	0	na

Note: Test locations selected are Bountiful Airport, Brigham City Municipal Airport, Spanish Fork Airport, Heber Valley Airport, and Logan-Cache Airport
na = not applicable.

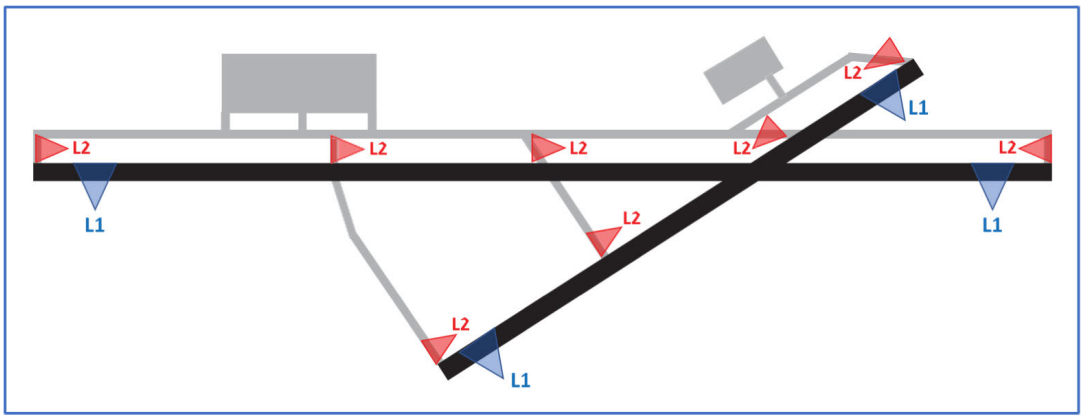


FIGURE 3. Camera Layout 1 and 2 on a multi-runway airport. "L1" refers to Camera Layout 1 and "L2" refers to Camera Layout 2.

TABLE 3. Constraint thresholds in the trajectory noise removal filter.

Constraint	Minimum threshold	Maximum threshold
Length (L_t)	$\frac{1}{10} f_w$	f_w
Duration (d_t)	$0.5s$	$30s$
Shape ($S_y(n)$)	$-\frac{1}{100} f_h$	$+\frac{1}{100} f_h$

Note: The numbers are calibrated with the collected video resolution and distances from the target aircraft. f_w = video frame width and f_h = video frame height.

and deformations. The pixel coordinates of the tracking box center are used as waypoints of the aircraft trajectory,

$$T(n) = (x_n, y_n), \tag{1}$$

where n is the waypoint counter, x_n is the horizontal pixel coordinate, and y_n is the vertical pixel coordinate.

Trajectory Analysis Module: This module performs an initial filtration of the generated trajectories by eliminating any noise that may arise from aircraft detection and tracking errors. These errors can lead to abnormal trajectories in terms of their shape, length, and duration. To address this, a noise removal filter is applied, effectively sifting out these abnormal trajectories. The filtered trajectories are subsequently classified into two categories: departure operations and arrival operations. This classification is based on the horizontal direction of the trajectory, $\vec{T}_x = \text{sign}(\Delta T_x)$. Table 3 represents the thresholds used in the trajectory noise removal filter.

2) AIRCRAFT INFORMATION

The aircraft information extraction process involves two approaches applied to the restored images of the aircraft from the operation time window. The first approach utilizes a custom CNN classifier model to directly classify and recognize the aircraft type from a set of predefined classes. On the other hand, the second approach retrieves aircraft information from the FAA registration database by recognizing the unique aircraft tail number. To optimize the search process within the FAA registration database and locate the target tail number efficiently, the second approach leverages the output of the first model, which provides the recognized aircraft type. By employing this information, the system filters out irrelevant tail numbers using a filtered database, as illustrated in Figure 6. This approach narrows down the search space and enables the identification of the desired tail number, allowing for the retrieval of the corresponding aircraft information.

Aircraft Type Recognition: In our study, we developed an aircraft classifier model using a CNN architecture. Figure 7 illustrates the structure of the customized CNN model for aircraft classification, achieved by layer arrangement optimization. In this optimization challenge, we established a framework consisting of 5 convolutional blocks (referred to as Conv B) and 3 fully connected layers as the core components of our custom CNN classifier. We then employed the HyperBand algorithm [51], integrated within Keras [52], to fine-tune and optimize specific parameters of these components:

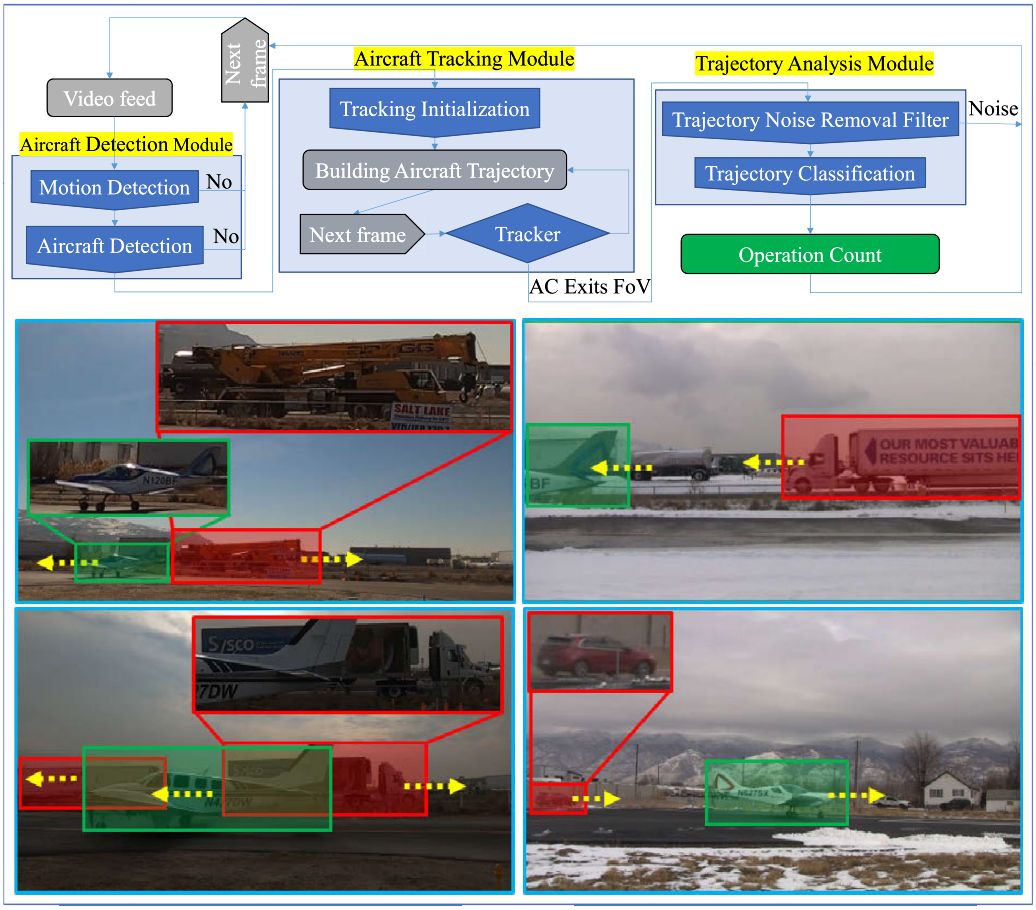


FIGURE 4. Operations count system flowchart. Green color-coded boxes refer to operational motions, and red color-coded boxes refer to non-operational motions. "AC" stands for aircraft.

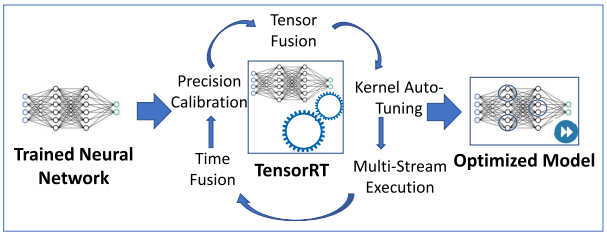


FIGURE 5. TensorRT framework.

- **Quantity of Convolutional Layers per Block:** This is restricted to a range of 1 to 5 layers in each block.
- **Convolutional Layer Output Dimensionality:** The output dimensions are limited to 16-64 for B1 layers, 32-128 for B2 layers, 64-256 for B3 layers, and 128-512 for both B4 and B5 layers, with step increments set to the respective lower bounds.
- **Fully Connected Layer Output Dimensionality:** The first two fully connected layers' output dimensions are capped between 512-1024 for FC1 and 256-512 for FC2, with step increments at half the lower bounds.
- **Optimizer Learning Rate:** Options for the learning rate include 5e-3, 1e-3, 5e-4, 1e-4, and 5e-5.

These defined boundaries were experimentally set to ensure optimized performance in recognition. The architecture consists of convolutional layers followed by a batch normalization layer, as suggested in [53], to accelerate the training process. The number of parameters in the CNN classifier was optimized for efficient deployment on edge devices and faster inference. The decreased size of the parameters facilitates integration and enhances processing speed on edge devices. To evaluate the performance of our custom CNN classifier model, we compared it with two state-of-the-art neural networks, ResNet50 [54] and Xception [55], which were initially trained on the ImageNet dataset [56] and later fine-tuned on our target dataset. The optimized CNN classifier demonstrated higher accuracy with fewer parameters on the FGVC-Aircraft dataset [57] compared to ResNet50 and Xception, as presented in Table 5. The deep classifier models were trained on a host machine equipped with an NVIDIA GeForce RTX 2080 Ti. Recognizing the aircraft type is instrumental in reducing the size of the registration database within the identification system. By filtering out tail numbers associated with other aircraft types, our system effectively streamlines the search process, improving overall efficiency. Table 6 provides a summary of

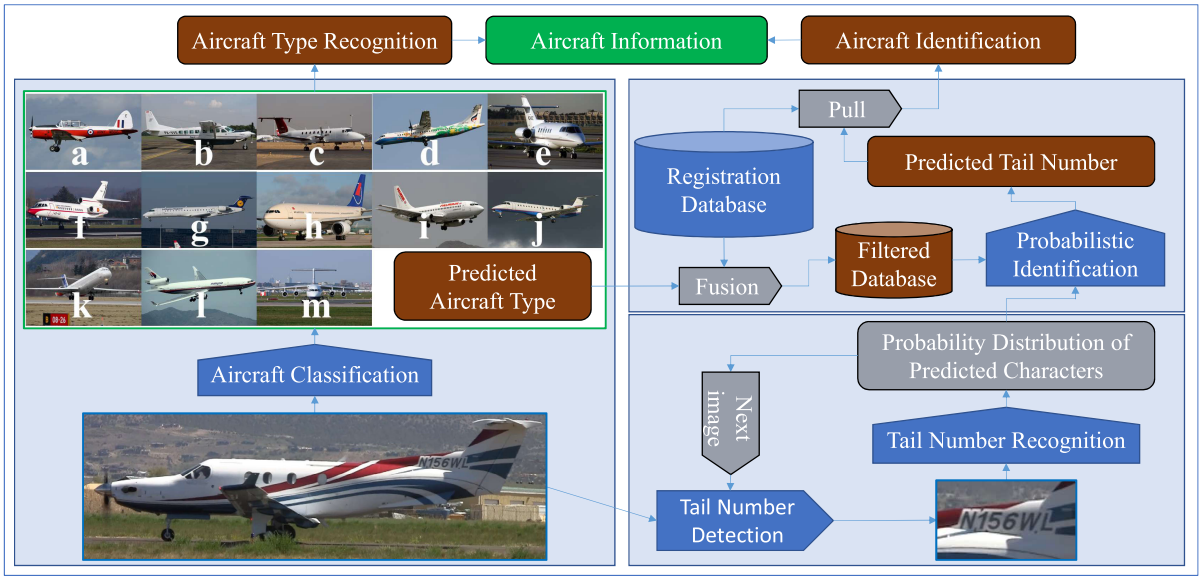


FIGURE 6. Aircraft information system flowchart.

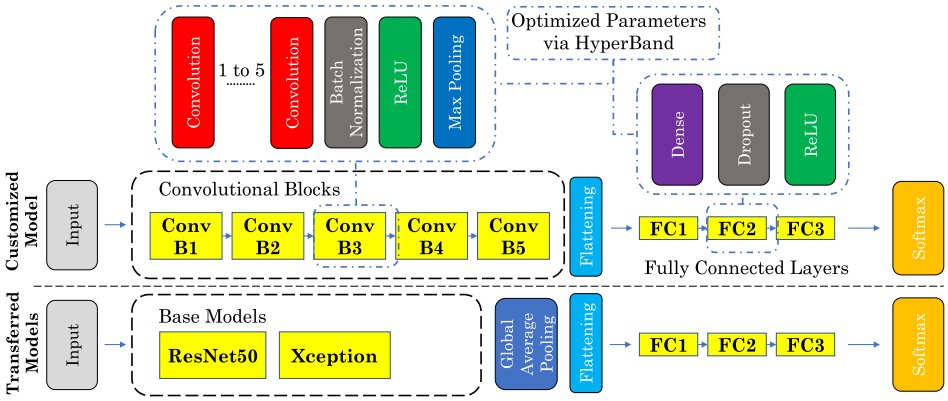


FIGURE 7. The structure of the CNN-based classifier models, where the HyperBand algorithm optimizes the number of convolutional layers in each Conv B and output dimensions.

the predefined 13 aircraft type classes that encompass the FAA database.

For our training, we utilized the FGVC-Aircraft dataset [57], which comprises 10,000 images of aircraft, each model variant represented by 100 images across 100 variants. The aircraft images possess variations in function, dimension, designation, structure, historical style, and brand identity. The image resolution ranges from approximately 1 to 2 Mpixels. This dataset is systematically categorized according to model, variant, family, and manufacturer. In our study, we have reclassified these into specific aircraft classes tailored to our identification system needs. These classes are shaped by the criteria set in the FAA's standard registration database. This database includes the *Aircraft Registration Master file* and the *Aircraft Reference file by Make/Model/Series Sequence* [58]. The database provides key visual attributes such as the type of aircraft, engine type, number of engines, size category (based on weight), and occasionally the model of manufacture. With these parameters, along with the models in the FGVC dataset, we established 13 distinct classes to cover a broad spectrum of aircraft commonly found at various airports. Table 4 displays the thirteen categories and the specific criteria utilized by the proposed approach to enhance the registration database, following the classification of the operational aircraft into one of these distinct categories.

In our analysis, different engine types like reciprocating, 4-cycle, 2-cycle, and rotary are categorized together due to their similar combustion processes. Turbo-prop aircraft, capable of carrying more payload than piston-powered ones, differ in appearance from them (class a and b). This logic also separates turbo-fan/jet from turbo-prop models, highlighted by the absence of propellers in the former. Figure 6 indicates that aircraft can be further grouped by engine number and weight within the same engine type. Additionally, the

TABLE 4. Aircraft classes; an example image from each class is presented in Figure 6.

Class	Aircraft Models Selected from FGVC Dataset
a	Cessna 172, DH-82, DHC-1, DR-400, PA-28, SR-20
b	Cessna 208
c	Beechcraft 1900, DHC-6, EMB-120, B200
d	ATR-72, DHC-8-300, Fokker 50, Saab 2000
e	BAE-125, Cessna 525, Cessna 560, Falcon 2000
f	Falcon 900
g	CRJ-700, CRJ-900, E-170, E-190, E-195, Fokker 100, Tu-134
h	A300B4, A310, A318, A319, A320, A321, A330-200, A330-300
i	737-200/300/400/500/600/700/800/900, 757-200/300, 767-200/300/400, 777-200/300, Boeing 717
j	ERJ 135, ERJ 145, Global Express, Gulfstream IV, Gulfstream V
k	DC-9-30, MD-80, MD-87, MD-90
l	727-200, DC-10, L-1011, MD-11, Tu-154, Yak-42
m	707-320, 747-100, 747-200, 747-300, 747-400, A340-200, A340-300, A340-500, A340-600, A380, BAE146-200, BAE146-300,DC-8

Note: Class h consists of Airbus models.

FGVC-Aircraft dataset's heavy turbo-fan/jet images aid in identifying their manufacturers (class h-k), with similar models from Embraer, Bombardier, and Gulfstream classified together (class j).

a: TRAINING PROCEDURE

The FGVC-Aircraft image dataset was divided into three parts: 66% for training, 17% for validation, and 17% for testing. During training, we monitored the validation set's loss to identify the most effective model. The images were resized to fit the input requirements of different deep learning classifier models: 224 × 224 pixels for ResNet50, 299 × 299 for Xception, and 256 × 256 for our custom CNN classifier model. To enhance the training data, we employed data augmentation techniques, including random horizontal flips and rotations within a range of $[-\frac{2\pi}{10}, +\frac{2\pi}{10}]$ on batches of 32 images during each epoch. To prevent overfitting, we implemented two regularization methods: L2 regularizers with a 0.01 factor for the convolutional layers in the custom CNN classifier model and a dropout rate of 0.35 for the fully connected layers in all deep learning classifier models. The models were trained using categorical cross-entropy as the loss function and the Adam optimizer.

The three deep learning classifier models utilizing CNN were optimized for maximum efficiency, with each allowed a training period of up to 100 epochs. This training was conducted for the selection of the best architecture using the HyperBand algorithm. The optimal architectures identified from this process, using the HyperBand algorithm, are presented in Table 5. Additionally, a constraint was placed on the total number of parameters in the models by establishing fixed boundaries within the search space. After the initial 100 epochs of training, we continued to train each optimized model until the validation loss failed to improve for over 30 epochs. For the transferred models, we first allowed them to converge on the target dataset, then fine-tuned them by unfreezing the base model and continuing training. This utilized the full capacity of the models. During fine-tuning, we reduced the learning rate by a factor of 10 to ensure gradual improvement and prevent overfitting. The training validation curve is presented in Figure 8. The custom CNN classifier model attained the highest validation accuracy of 88% at epoch 132, and the training terminated after 162 epochs as there was no improvement seen in the validation loss. The highest training accuracy recorded was 92%.

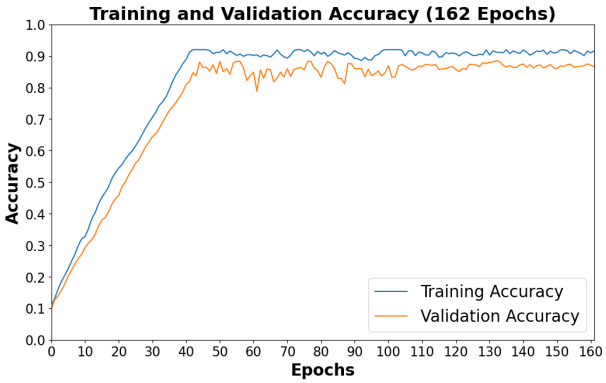


FIGURE 8. Training and validation Curve of the custom CNN classifier model for the Top 1 accuracy.

b: DEEP LEARNING CLASSIFIER MODEL

Figure 9 presents a comparison of three deep learning classifiers on the FGVC-Aircraft dataset's test set, using confusion matrices. These reveal the custom CNN classifier model's proficiency in categorizing heavyweight jet aircraft (classes g to m), with a consistent accuracy above 80%. On the other hand, ResNet50 and Xception demonstrate weaker performance for these classes; ResNet50's accuracy falls below 80% for classes i and j, and under 70% for class k, while Xception struggles with classes h, k, and l, maintaining accuracies below 80%. Additionally, the custom CNN classifier model occasionally misclassifies lightweight trijet aircraft (class f) as twinjet aircraft (class e), a trend observed in 18% of cases, likely due to a limited number of samples in class f. This issue of sample imbalance also affects the detection of single-engine turbo-prop aircraft (class b) in all models, resulting in slightly higher false positive rates. Moreover, the confusion matrices highlight the significant impact of an overrepresentation of BOEING images, particularly affecting the Xception model, which shows a 47% false positive rate in this respect. Overall, the custom CNN classifier model provides superior and consistent performance across all the classes compared to ResNet50 and Xception.

Table 5 indicates that when each of the three deep learning classifier models is applied to the test set, the top three predicted classes achieve an accuracy exceeding 96%. Therefore, in the process of filtering the registration database,

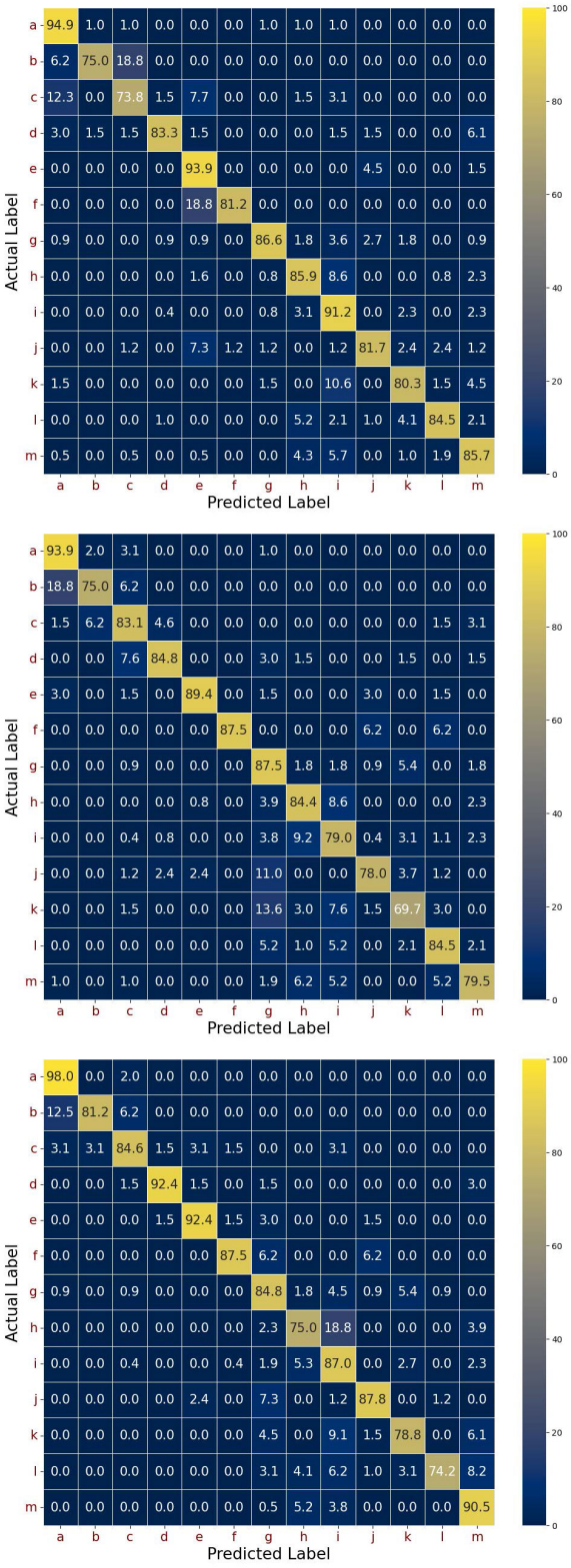


FIGURE 9. Visualization of confusion matrices depicting classification outcomes on the test set of the FGVC-Aircraft dataset. Top represents the Custom CNN classifier, Middle shows Transferred ResNet50, and Bottom illustrates Transferred Xception.

the focus is on these top three recognized classes, identified after the classifier is used on the detected aircraft in selected video frames during the operation period. This approach significantly reduces the likelihood of mistakenly excluding the correct tail number from the database. The concept of 'top-3 accuracy' refers to scenarios where the actual class is one of the three classes most likely to be predicted by the model. The *custom CNN classifier model* was chosen for the proposed aircraft identification system due to its advantages, such as fewer parameters (resulting in faster inference), lower loss, higher overall accuracy, and more consistent performance in recognizing individual aircraft classes. The custom CNN classifier model outperforms the state-of-the-art classifier models Xception and ResNet50 in these aspects.

To mitigate bias, the custom CNN classifier model was trained on the FGVC dataset and subsequently experimented using data collected from three distinct airports not included in the training set, ensuring robustness and generalizability across different environments.

Aircraft Identification: The tail number plays a crucial role in identifying aircraft. In our system, we adopted a detection model based on Haar-like features to locate the tail number within the recorded aircraft images. Although Textboxes, a fast text detector utilizing a deep neural network [59] was considered as an alternative, we decided against its usage due to the significant increase in runtime observed with deep text detectors. Instead, we chose the Haar cascade classifier, originally designed for face detection [60], as an efficient detection method suitable for embedded systems. The classifier was chosen for its balance of speed and accuracy, making it ideal for real-time applications on resource-constrained devices like edge devices. Since U.S. aircraft tail numbers begin with the letter "N," we trained a specialized "N" detector using Haar-like features. Our experiments revealed that extending the bounding box around the detected letter "N" four times captures the entire tail number image. The detector we developed uses 24 $\times$ 24 window sizes and consists of 15 rejection and acceptance stages. These stages are trained using Gentle Adaboost [61], following the instructions in [62]. Gentle AdaBoost utilizes weighted least-squares regression to establish a robust and stable ensemble of weak classifiers [63]. We selected a hit rate value of 0.997 because going any higher would noticeably slow down the training process. In our tests, a false alarm rate of up to 0.4 was found to be effective. Regarding the specific hyperparameters for the degenerate trees composed of weak classifiers, we established the weight trim rate at 0.95, limited the maximum depth of weak trees to 1, and capped the number of weak trees at each stage to 100. Additionally, we conducted a grid search across three potential maximum stages (namely, 10, 15, and 20) and two sizes of sample windows (20 $\times$ 20 and 24 $\times$ 24). For the training data, we used 248 positive image samples (images labeled as "N") and 415 negative samples, all sourced from randomly selected web images. A separate validation dataset comprising 100 aircraft images confirmed that the model, when using 24 $\times$ 24 window sizes, achieved a superior detection rate for tail numbers. The best-performing model

TABLE 5. Performance comparison of the optimized classifier with Xception and ResNet50. Top-"n" accuracy is the accuracy where the true class matches any of the "n" most probable classes predicted by the mode.

Variables	Optimized CNN Model		Transferred Models	
	# Conv Layers	Output Dimensionalities		
Conv B1	5	16, 48, 16, 16, 48	Xception	ResNet50
Conv B2	2	64, 64		
Conv B3	3	256, 64, 256		
Conv B4	2	512, 256		
Conv B5	4	512, 256, 128, 384		
FC1	NA	512	1024	1024
FC2	NA	256	512	512
FC3	NA	13 (aircraft classes)	13 (aircraft classes)	13 (aircraft classes)
Learning Rate	0.0001		0.0001	0.0001
# Parameters	**18,711,245**		23,491,125	26,217,357
Top-1 Accuracy	**87%**		86%	82%
Top-2 Accuracy	93%		**94%**	93%
Top-3 Accuracy	**97%**		**97%**	96%

TABLE 6. 13 aircraft classes (S: single-engine, M: multi-engine, R: reciprocating, TP: turbo-prop, P: piston, TF: turbo-fan, TJ: turbo-jet, weight class 1: upto 12,499 lb, weight class 2: 12,500-19,999, and weight class 3: 20,000 and over).

Class	Aircraft Engine	Engine Type	Weight	#Engine	Manufacturer
a	Single-Engine	Reciprocating, Rotary	Class 1&2	1	ALL
b	Single-Engine	Turbo-prop	Class 1&2	1	ALL
c	Multi-Engine	Turbo-prop & piston	Class 1&2	2	ALL
d	Multi-Engine	Turbo-prop	Class 3	2	ALL
e	Multi-Engine	Turbo-fan, Turbo-jet	Class 1&2	2	ALL
f	Multi-Engine	Turbo-fan, Turbo-jet	Class 1&2	3	ALL
g	Multi-Engine	Turbo-fan, Turbo-jet	Class 3	2	ALL
h	Multi-Engine	Turbo-fan, Turbo-jet	Class 3	2	AIRBUS
i	Multi-Engine	Turbo-fan, Turbo-jet	Class 3	2	BOEING
j	Multi-Engine	Turbo-fan, Turbo-jet	Class 3	2	GULFSTREAM
k	Multi-Engine	Turbo-fan, Turbo-jet	Class 3	2	MCDONNEL-DOUGLAS
l	Multi-Engine	Turbo-fan, Turbo-jet	Class 3	3	ALL
m	Multi-Engine	Turbo-fan, Turbo-jet	Class 3	4	ALL

was further trained for 15 stages and concluded once it met the specified criteria for the overall false alarm rate. After locating the tail number, our intelligent system predicts the individual characters within the tail numbers using a text recognition model called the CRNN [64]. The CRNN model is pre-trained on two well-known synthetic text recognition datasets named MJSynth and SynthText [65]. Each predicted character in the CRNN model is represented as a probability distribution over the character classes, encompassing the English alphabet and numbers.

The softmax function is incorporated into the network after the bidirectional long short-term memory (BLSTM) layer in the recurrent layers. It serves to normalize the score list of predictions for each window (z_i) into a probability distribution across the predefined character classes. The softmax function is defined as follows:

$$\mathbb{P}(r_i = \omega) = \frac{e^{z_{i_k}}}{\sum_{k=1}^{L_r} e^{z_{i_k}}}. \tag{2}$$

Here $L_r = |\Omega|$ represents the number of character classes and ω is an element of set Ω. The softmax function ensures that the scores are normalized by exponentiating them and dividing by the sum of all exponentiated scores. To determine the final probability distribution of the recognized label sequence for a video frame, $\mathbf{r} = \{r_1, \ldots, r_{L_s}\}$, where L_s is the length of the predicted label sequence, any blanks and repeated labels (overlapped receptive fields) are removed.

In the initial step, for each video frame, we identify the most probable value of r_i, denoted as r_i^*:

$$r_i^* = \arg\max_{\omega \epsilon \Omega} \mathbb{P}\left(r_i = \omega\right). \tag{3}$$

The predicted label sequence for that specific video frame is represented by $\mathbf{r}^* = \{r_1^*, \ldots, r_{L_s}^*\}$, and $A = \{\mathbf{r}_1^*, \ldots, \mathbf{r}_{N_f}^*\}$ represents the set of predicted label sequences for all video frames, where N_f is the number of selected frames.

To determine the most frequently recognized label sequence during the operation time window, we select the label sequence with the highest frequency of occurrence among all the predicted label sequences:

$$T_{S1}^* = \arg\max_x |\{x \in A\}|. \tag{4}$$

In the second step, we alter the first step and associate the most frequent label sequence argument with a tail number lexicon:

$$T_{S2}^* = \arg\max_{x \in \Psi} |\{x \in A\}|. \tag{5}$$

Here, Ψ denotes the tail number lexicon, which is compiled by extracting the list of registered aircraft tail numbers relevant to the recognized aircraft classes from the database.

The third step further refines the second step, particularly in situations where there is no exact match between the tail number lexicon and the set of predicted label sequences during the operational time window. This step, specifically, helps with false tail number detection cases. In such scenarios, the tail number in the lexicon that yields the highest conditional probability in relation to the observation of r is chosen. This is expressed by:

$$\mathbf{s}^* = \arg\max_{T^D \epsilon \Psi} \mathbb{P}\left(T^D \middle| r\right). \tag{6}$$

In this scenario, $\mathbf{s}^*$ is deemed as the predicted label sequence for a specific video frame. $B = \{\mathbf{s}_1^*, \ldots, \mathbf{s}_{N_f}^*\}$ symbolizes the set of predicted label sequences for all video frames. T^D signifies a tail number recorded in the lexicon. The predicted label sequence is obtained by applying text recognition to the tail number image box, generating a set of predicted characters denoted by P_r. With this notation, the probability of each tail number in the database being the target tail number can be calculated as follows:

$$\mathbb{P}(T^D|P_r) = \frac{\sum_{i=1}^{N} \mathbb{P}(P_{r_i}|T_{(i)}^D)}{N}. \tag{7}$$

where N is the length of the tail number, and $T_{(i)}^D$ denotes the ith character of the target tail number. After processing all the images of the tail number captured during the aircraft operation time window, the intelligent system selects the tail number with the maximum probability.

Finally, we select the most frequently recognized tail number during the operation time window:

$$T_{S3}^* = \arg\max_x |\{x \in B\}|. \tag{8}$$

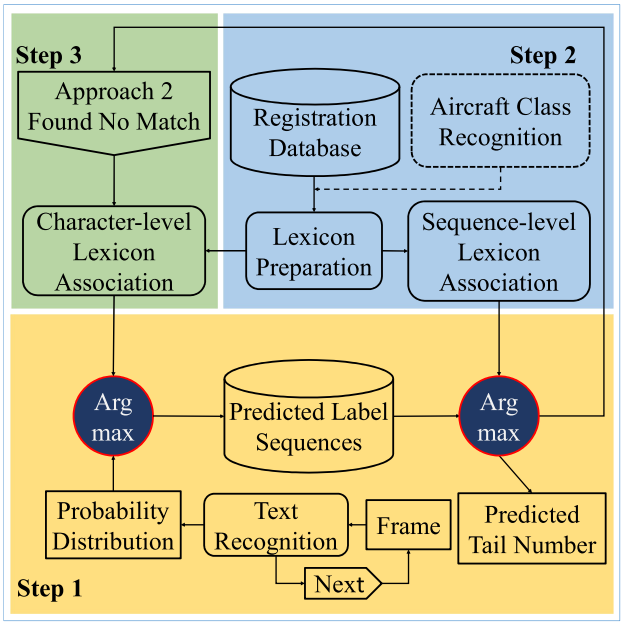


FIGURE 10. Identification steps for prediction of labels.

FIGURE 11. (a) NVIDIA Jetson Nano 4GB Developer Kit B01 (b) NVIDIA Jetson Xavier NX Developer Kit.

B. HARDWARE

1) PROCESSOR

Considering the computational intensity of the algorithms, we conducted experiments to test the system using two processing platforms. We utilized two commercially available Jetson developer kits to evaluate the performance of different system modules: (1) NVIDIA Jetson Nano 4GB Developer Kit B01 and (2) NVIDIA Jetson Xavier NX Developer Kit, as depicted in Figure 11. Both kits are equipped with NVIDIA GPUs that enhance deep learning computations, including convolution, pooling, and activation operations. However, the Jetson Xavier NX Developer Kit is approximately three times more powerful and expensive than the Jetson Nano 4GB Developer Kit B01. Moreover, the algorithms designed for extracting aircraft information require substantial computational resources. Consequently, we developed two separate standalone devices to accommodate the following two setups:

Standalone device #1:

- Hardware Module: NVIDIA Jetson Nano 4GB Developer Kit B01
- Software Module: Operations Count

Standalone device #2:

- Hardware Module: NVIDIA Jetson Xavier NX Developer Kit
- Software Module: Operations Count and Aircraft Information

NVIDIA Jetson Nano 4GB Developer Kit B01 has 128-core Maxwell GPU with 4 GB 64-bit LPDDR4 memory, 4K/HD/720p video decoder, microSD storage, and an M.2 key Wi-Fi connectivity (which requires an external Wi-Fi module to be installed). On the other hand, the NVIDIA Jetson Xavier NX Developer Kit is a more powerful edge device for its GPU with 384 NVIDIA CUDA cores and 48 Tensor cores. Additionally, it has an enhanced 8 GB 128-bit LPDDR4x memory and an included Wi-Fi/BT. The technical specifications of these two developer kits are summarized in Table 7.

The official operating system for the Jetson developer kits is the Linux4Tegra, which is based on Ubuntu 18.04. This operating system is available via the included SD card image. NVIDIA JetPack is used as the Software Developer Kit (SDK) for NVIDIA Jetson Nano Developer Kit and NVIDIA Jetson Xavier NX Developer Kit. NVIDIA JetPack SDK is a comprehensive solution for compiling end-to-end

TABLE 7. Technical specifications of NVIDIA Jetson Nano 4GB Developer Kit B01 and NVIDIA Jetson Xavier NX Developer Kit.

Developer Kit	Jetson Nano Developer Kit	Jetson Xavier NX Developer Kit
GPU	128-core Maxwell	384-core NVIDIA Volta GPU with 48 Tensor Cores
CPU	Quad-core ARM A57 @ 1.43 GHz	6-core NVIDIA Carmel ARM v8.2 64-bit CPU 6 MB L2 + 4 MB L3
Memory	4 GB 64-bit LPDDR4 25.6 GB/s	8 GB 128-bit LPDDR4x 51.2GB/s
Storage	microSD	microSD
Video Encode	4K @ 30 | 4x 1080p @ 30 | 9x 720p @ 30 (H.264/H.265)	2x 4K @ 30 | 6x 1080p @ 60 | 14x 1080p @ 30 (H.265/H.264)
Video Decode	4K @ 60 | 2x 4K @ 30 | 8x 1080p @ 30 | 18x 720p @ 30 (H.264/H.265)	2x 4K @ 60 | 4x 4K @ 30 | 12x 1080p @ 60 | 32x 1080p @ 30 (H.265) | 2x 4K @ 30 | 6x 1080p @ 60 | 16x 1080p @ 30 (H.264)
Camera	2x MIPI CSI-2 DPHY lanes	2x MIPI CSI-2 D-PHY lanes
Connectivity	Gigabit Ethernet, M.2 Key E	Gigabit Ethernet, M.2 Key E (Wi-Fi/BT included), M.2 Key M (NVMe)
Display	HDMI and display port	HDMI and display port
USB	4x USB 3.0, USB 2.0 Micro-B	4x USB 3.1, USB 2.0 Micro-B
Mechanical	69 mm x 45 mm, 260-pin edge connector	103 mm x 90.5 mm x 31 mm, 260-pin SO-DIMM connector

GPU-accelerated deep learning applications. In this study, we used JetPack 4.6 to flash both the NVIDIA Jetson Nano 4GB Developer Kit B01 and the NVIDIA Jetson Xavier NX Developer Kit.

An AC8265 Dual Mode Wireless NIC Wi-Fi Module (Figure 12a) is integrated into the NVIDIA Jetson Nano 4GB Developer Kit B01. Furthermore, a 4020 PWM cooling fan (Figure 12b) is placed on top of the Jetson Nano 4GB Developer Kit B01's heatsink. This cooling solution is necessary to prevent the kit's temperature from rising excessively during operation. For monitoring system performance at the airport site during the experiments, a 7-inch LCD display (Figure 12c) is connected to the Jetson Developer Kits. This display provides real-time information and allows for efficient observation of the system's performance.

2) SENSOR

We utilized two camera modules that are compatible with the NVIDIA Jetson Developer Kits: (1) Raspberry Pi Camera Module V2-8 Megapixel, 1080p, and (2) IMX477 Camera Bundle with 12.3MP HQ Camera Module, as shown in Figure 12d and e. The Raspberry Pi Camera Module V2 is installed on standalone device #1, while the IMX477 Camera Module is installed on standalone device #2.

The IMX477 Camera Module includes a 6mm CS-mount lens, which allows for manual zoom adjustments, particularly

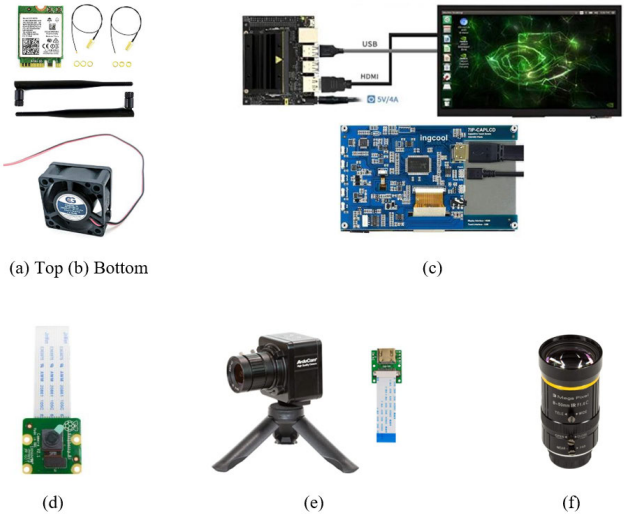


FIGURE 12. (a) Dual Mode (AC8265) Wireless NIC Wi-Fi Module (b) 4020 PWM cooling fan (c) Ingcool 7-inch HDMI LCD 1024 × 600 Resolution IPS Display Module (d) Raspberry Pi Camera Module V2-8 Megapixel,1080p (e) IMX477 Camera Bundle with 12.3MP HQ Camera Module (f) 8-50mm C-Mount Zoom Lens for IMX477 Raspberry Pi HQ Camera.

useful for capturing aircraft tail numbers and extracting aircraft information at airport sites. Since some airport safety areas may require placing the camera at longer distances from the taxiway and runway, we also employed a more robust zoom lens. For this purpose, an 8-50mm C-Mount Zoom Lens with a CS adaptor is used with the IMX477 Raspberry Pi HQ Camera, as depicted in Figure 12f.

To enhance the quality of video data acquisition, we installed the camera driver necessary for utilizing the Camera Serial Interface (CSI) on the Jetson Developer Kits (i.e., the host processors) instead of using USB cameras. This approach ensures improved performance and reliable data acquisition for the cameras connected to the Jetson Developer Kits.

V. CONCLUSION AND FUTURE SCOPE

A. DATA COLLECTION

We conducted our experiments at three non-towered general aviation airports located within the state of Utah: Heber Valley Airport, South Valley Regional Airport, and Logan-Cache Airport. The airports were selected due to their diverse range of aircraft types and varying traffic volumes. The Heber Valley Airport hosts over a hundred diverse aircraft, with transient traffic that includes everything from light piston planes to heavy turbine aircraft, such as the Gulfstream G650 and Global Express [66]. Runway 04/22 has a dual-wheel load capacity of 142,500 pounds and a pavement classification number of 32/F/B/X/T. Similarly, Logan Cache Airport is home to 176 based aircraft and supports roughly 137,900 takeoffs and landings annually [67]. Meanwhile, South Valley Airport accommodates a variety of aircraft, including 144 single-engine aircraft and 7 multi-engine aircraft [68]. We placed our system setup at the entrance

TABLE 8. Information on the data collection sessions.

Airport	# Session	Weather	# Operations
Heber Valley Airport	1	Cloudy with drizzle	12
	2	Cloudy	19
South Valley Regional Airport	3	Sunny	17
	4	Partly cloudy	10
Logan-Cache Airport	5	Sunny	22
	6	Sunny	29

taxiway, specifically the one leading to the runway's end, a layout previously tested in our earlier study [9]. The environmental setup for the three airports are illustrated in Figure 13. This specific arrangement was selected due to its advantage in providing the edge device with additional time to identify aircraft, as the speed of aircraft operations is slow while moving in front of the edge devices. The configuration was tested on a single node of the airport in order to evaluate the performance of the computer vision system and edge computing devices. This configuration was chosen because it allows the edge device more time to detect aircraft at their reduced speeds. To ensure adherence to safety regulations, the setup was situated outside both the runway safety area (RSA) and taxiway safety area (TSA). We performed a total of six data collection sessions, with each session taking place on a different day and lasting for four hours. The sessions were scheduled between 2 PM and 6 PM, during daylight hours. Table 8 provides comprehensive information regarding the weather conditions observed during each session, as well as the corresponding number of aircraft activities recorded in the targeted entrance taxiway.

Reduced visibility due to nighttime operations and adverse weather conditions is a common challenge for all vision-based systems. In the current study, data were collected from 2 PM to 6 PM, but in our previous study [69], we experimented with these challenges. Positioning edge devices closer to the taxiway and the lights present in airports helps improve visibility in these scenarios. Additionally, during adverse weather conditions, airports typically have less traffic. Figure 14 presents a screenshot from video footage capturing aircraft detection during nighttime operations in our previous study [69].

During the data collection sessions, we validated the collected data by comparing it to observations made by the data collection crew at the test locations (as shown in Figure 15). The crew was present during the four-hour data collection period and recorded the activities for comparison with the data collected by the two standalone devices. During each session, both standalone devices were placed next to the targetted end entrance taxiway, with their camera sensors aiming at the taxiway passage. To ensure sustained charging, a portable power station equipped with a 60 watts solar panel was used for each device. The two edge devices are efficient in terms of energy consumption, with Standalone device # 1 consuming 5 watts and Standalone device # 2 consuming 10 watts. This efficiency is notable given that typical security cameras enabled with solar panels consume between 2 to 15 watts on average. By conducting the experiments at three different airports, we were able to test the system in various scenarios, including different safety areas, distances from the target passage, and varying aircraft types.

B. SYSTEM INFERENCE SPEED

When assessing the system processing times on both standalone devices, we found that the operations count software module performed in real-time, as shown in Table 9. The highest processing time for this module occurred when SSD-Mobilenet-v2 was activated. As a result, standalone device #1 updates video frames at a minimum rate of 7 frames per second (FPS), and standalone device #2 updates video frames at a minimum rate of 19 FPS. This results from the processing times of 142 ms and 51 ms for SSD-Mobilenet-v2 on standalone devices #1 and #2, respectively, which directly determine the achievable FPS rates. During the active aircraft tracking module, both devices render more frames per second, with standalone device #1 rendering 13 FPS and standalone device #2 rendering 28 FPS. These frame rates are derived from MOSSE's processing times of 76 ms for standalone device #1 and 35 ms for standalone device #2. This indicates that both devices possess the capability to process an ample number of video frames, enabling the tracking and counting of aircraft operations within the time it takes for an aircraft to traverse an entrance taxiway. However, the aircraft information module is still computationally intensive to execute on standalone device #2, despite using a fast tail number detector. Therefore, this module is only executed after the operations count module has been processed. Nevertheless, the delay caused by processing the aircraft information module is still shorter than the waiting time between two consecutive aircraft operations, as pilots typically follow safety measures before entering the runway. As indicated in Table 9, the processing delay for aircraft information on standalone device #2 totals approximately 6.7 seconds.

C. SYSTEM ACCURACY

As illustrated in Figure 16, standalone device #2 achieved an accuracy rate of approximately 80% in operations counting. In contrast, standalone device #1, which has a less powerful processor, managed an accuracy of about 71% in the same task. It's noteworthy that software errors contributed to less than 4% of the counting errors in both devices. Due to the complexity of the aircraft information algorithms, only standalone device #2, with its more advanced processor, was able to execute the aircraft type recognition and identification tasks.

The majority of errors in both standalone devices were attributed to hardware-related issues, which are highlighted in dark blue. These errors refer to malfunctions of the standalone system during testing, such as losing some video frames due to camera-related errors or operating system malfunctions. This type of malfunction precedes software errors since they

FIGURE 13. Environmental setup for three airports.

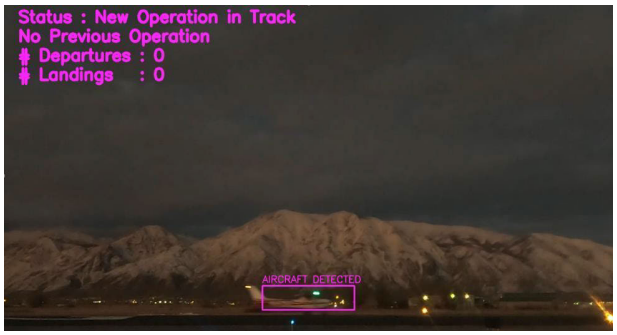


FIGURE 14. Screenshot of a nighttime operation.

FIGURE 15. On-site data collection.

TABLE 9. Processing times recorded by standalone devices.

Software Module	Algorithm Module	Model	Device #1 Processing Time	Device #2 Processing Time
Operations Count	Aircraft Detection	Gaussian Mixture	32 ms	17 ms
		SSD-Mobilenet-v2	142 ms	51 ms
	Aircraft Tracking	MOSSE	76 ms	35 ms
	Trajectory Analysis	Noise Removal Filter	5 ms	2 ms
Aircraft Information	Aircraft Type Recognition	Optimized CNN Classifier	N/A	393 ms
	Aircraft Identification	Tail Number Detector (Haar Cascade Model)	N/A	112 ms
		Tail Number Recognizer (CRNN)	N/A	814 ms
		Probabilistic Identifier	N/A	5,443 ms

Note: The processing times are the average of the times calculated by processing all captured aircraft operations. The processor of standalone device #1 is the NVIDIA Jetson Nano 4GB Developer Kit B01, and the processor of standalone device #2 is the NVIDIA Jetson Xavier NX Developer Kit. The sensors recorded video data with 1080x1920 resolution.

happen before the algorithms are triggered. The remaining errors were related to software issues.

Standalone device #2 was specifically utilized for extracting aircraft information. The optimized CNN classifier demonstrated an impressive aircraft type recognition rate of 76%, with an error rate of 7.4%. Additionally, the proposed probabilistic aircraft identification method successfully identified 68.8% of the exact tail numbers by referencing the FAA registration database. It is important to note that these results were obtained despite hardware malfunctions. If the hardware had been fully functional, it was estimated that the device could have achieved up to 85.3% accuracy in identifying the operating aircraft (as depicted in Figure 16). Notably, these identification accuracies were attained even though, on average, only 72% of the individual characters in the tail numbers were correctly recognized by the text recognition model (CRNN).

D. SYSTEM COST

The proposed system incorporates an NVIDIA developer kit and a solar-powered camera module. The operational cost of the standalone device can be segmented into hardware, installation, and maintenance expenses, as detailed in Table 10. However, the total cost of the system may fluctuate depending on the rental option selected and the duration of use. When accounting for the software subscription required for the operations count and aircraft information modules, an additional $4,000-$5,000 may be added to the annual

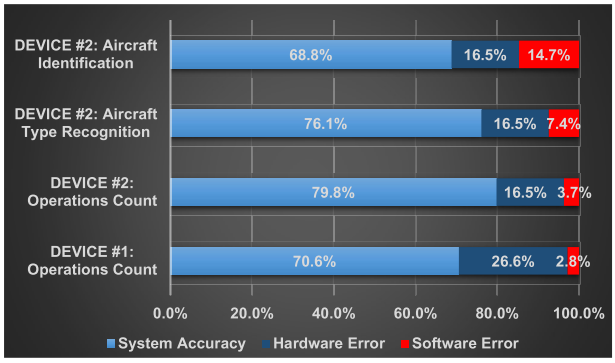


FIGURE 16. System accuracy and error.

system cost. This estimate is derived from currently available commercial computer vision solutions [70]. For a typical non-towered general aviation airport with four entrance taxiways, the estimated annual usage cost of standalone device #1 and standalone device #2 would be around $10,000 and $15,000, respectively.

E. COMPARISON WITH PREVIOUS MODELS

1) GENERAL AUDIO RECORDING DEVICE (GARD)

GARD calculates operation counts by considering the average number of radio transmissions during take-offs and landings. However, this count can be greatly affected by factors like the pilot, the specific situation, and the location, which can result in imprecise estimates. For instance, the Florida Department of Transportation (FDOT) has observed discrepancies in these estimates ranging from 40% to 60% [15]. Moreover, airports that share a Unicom frequency with neighboring airports are unable to utilize GARD. In contrast, our system provides an accuracy of 79.8% in tracking the operation counts of the aircraft.

2) TRANSPONDER

Only about 65% [71] of general aviation aircraft are equipped with transponders, and a substantial portion of these aircraft in the U.S., approximately 84% [72], do not possess transponders capable of transmitting Mode S signals that include the aircraft identity information. Given these challenges, our system provides a cost-effective and efficient approach for performing operation counts and aircraft identification.

3) AUTOMATED ACOUSTICAL COUNTERS

As previously discussed in Section II-A, AAC systems rely on sound to monitor aircraft operations, which presents challenges in accurately capturing all types of operations [6], [14]. Moreover, these systems cannot identify specific aircraft using acoustical data. Conversely, our system, utilizing a vision-based approach, demonstrates significant improvements, achieving 79.8% accuracy in operation count and 68.8% accuracy in aircraft identification.

4) VISION ALGORITHMS

The custom CNN classifier model, enhanced with the HyperBand algorithm, outperforms the advanced Xception and ResNet50 models by achieving a higher accuracy rate of 97% for the top 3 classes, while utilizing fewer parameters. This comparison is detailed in Table 5. The efficiency of the custom CNN model, processing at a rate of 4 frames per second, is attributed to its reduced parameter size, enabling quicker and more accurate results. It is particularly adept at classifying heavyweight jet aircraft (classes g to m), maintaining an accuracy above 80%. In contrast, ResNet50 and Xception demonstrate less consistent performance in these classes, as shown in Figure 9, with ResNet50 falling below 80% accuracy in classes i, j, and k, and Xception in classes h, k, and l. Thus, the custom CNN classifier is selected for the identification system, surpassing Xception and ResNet50 in terms of efficiency, accuracy, and consistent recognition of classes.

F. DISCUSSION AND CONCLUSION

This paper presents an affordable and intelligent data collection approach for counting and recognizing aircraft operations at non-towered airports, which represent the majority of U.S. airports. The availability of aircraft operational data is crucial for future planning and making timely decisions regarding safety and security. Our proposed intelligent system is structured with edge computing and optimized machine and deep learning-based modules to gather *operation counts* and *aircraft information*. We have developed two standalone setups: standalone device #1, dedicated to operation counting, and standalone device #2, which has a more powerful edge processor capable of recognizing aircraft information in addition to counting operations.

Both standalone devices were tested at three non-towered general aviation airports. We found that the majority of errors (26% in device #1 and 16% in device #2) originated from hardware components, while the software modules operated with accuracy. These tests were conducted for experimental purposes to evaluate the system's feasibility and performance and were not designed for long-term deployment. Future work will focus on addressing the challenges associated with long-term deployment and integrating more robust solutions for continuous operation. As such, future research should prioritize improving the hardware configurations. The advent of advanced edge computing technologies, such as the NVIDIA Jetson AGX Orin Developer Kit equipped with a 512-core NVIDIA Ampere Architecture GPU, presents promising opportunities for hardware enhancements. This advancement can facilitate real-time data storage, a feature not implemented in this study, while ensuring minimal impact on the devices' processing times. In future investigations, utilizing such advanced edge computing devices could enable longer deployment periods in coordination with existing systems. We estimate that the deployment of the proposed system for a one-year data collection period, inclusive of aircraft information, would cost approximately $15,000 at a typical general aviation airport. Additionally, the total cost

TABLE 10. System operation costs.

Item	Detail	Cost per item[1]	Quantity	Subtotal[2] Device #1	Subtotal Device #2
Hardware	NVIDIA Developer kit	∼ $300 - $1,300	4	$1,200	$5,200
	Camera Sensor	∼ $50 - $100	4	$200	$400
	Optical Zoom Lens	∼$50	4	$200	$200
	Power Station + Solar	∼$400	4	$1,600	$1,600
Installation	Electrician	∼$70 / hr	10 hrs	$700	$700
	Computer Engineer	∼$80 / hr	10 hrs	$800	$800
Maintenance	Technical Support	∼$100 each	1 per months	$1,200	$1,200
Subtotal			4 devices[3] + 1 year operation	$5,900	$10,100

Notes: [1] The costs are average costs found on the market.
[2] The mean costs are used for subtotals.
[3] For a typical non-towered general aviation airport with 4 entrance taxiways

TABLE 11. Comparison of the proposed system with the commercially available systems.

System	Accuracy	Operation Count	Aircraft Type(1)	Aircraft Tail	Approximate Cost(2)
Standalone Device #1	High accuracy if no visual obstruction	Computer vision	NA	NA	$10,000(3)
Standalone Device #2	High accuracy if no visual obstruction	Computer vision	Computer vision	Computer vision	$15,000(3)
AAC (Automated Acoustical Counter)	1- Misses landing ops and quiet aircraft 2- Over/undercount ops in airport with closely arranged taxiways	Acoustic	NA	NA	$15,000(4)
GARD (Radio & ADS-B-based counter)	1- Counts depend on average radio transmissions which drastically changes at different flight situations 2- Identification deficiency because many general aviation aircraft are not yet equipped with the required avionics. Additionally, many equipped aircraft do not have Mode S transponders (required for aircraft identification)	Radio records	NA	ADS-B	$7,000(5)
Vantage (Radar + Video)	1- Counting deficiency because many general aviation aircraft are not yet equipped with the required avionics 2- No strategy for aircraft type recognition in cases where no stencilled tail number	Transponder	NA	Camera detection	NA(6)

Notes: (1) Direct aircraft type recognition and not by cross referencing the tail number or aircraft ID identification result. This feature is usefull specifically for cases where the tail number is not imprinted on fuselage.
(2) For a typical non-towered general aviation single-runway airport. Costs are approximated and may vary for all systems depending on the runway lengths and entrances.
(3) Hardware, software, installation, and maintenance costs for one year are included.
(4) Hardware and software costs are included based on Muia and Johnson [6]. Multiple counters are needed for longer runways. Three counters are considered for a single 5,500 ft runway.
(5) Hardware and software costs are included based on Invisible Intelligence [73].
(6) No data is available

could potentially decrease for airports with fewer than four entrance passages to the runway.

Three commercially available systems commonly used by the aviation community for airport operational data collection are the Automated Acoustical Counter (AAC), the General Audio Recording Device (GARD), and Vantage. The GARD system relies on radio records for operation counts and uses ADS-B for aircraft identification. Conversely, Vantage is a transponder-based counter system that employs tail number camera detection for identification tasks. In contrast to these, our proposed system offers superior accuracy and more comprehensive information, as detailed in Table 11. It's worth noting that the tabulated costs for the AAC and GARD systems do not include installation and maintenance expenses, and cost data for Vantage is currently unavailable. Although cost comparisons can vary depending on the airport layout, our proposed system (i.e., standalone devices) has the potential to offer higher

accuracy than other systems, provided there are no visual obstructions.

The newly developed devices present a viable alternative to the current systems. Unlike acoustical and radio-based counters, which cannot identify the type or model of an aircraft, standalone device #2 can accurately identify aircraft by recognizing their tail numbers and integrating this data with the FAA database to gather detailed aircraft information. On the other hand, transponder-based counters face identification limitations as they require aircraft to be equipped with a specific type of transponder (i.e., Mode S transponders), which is not common in many general aviation fleets. This issue is resolved by the proposed system, which is solely based on computer vision. It operates as a self-sufficient passive system, independent of any onboard avionics equipment. Furthermore, this new system can recognize the aircraft type independently of tail number detection, a distinct advantage over current camera systems like Vantage.

Future research directions should emphasize the enhancement of hardware components in edge devices, with the objective of augmenting processing capacities and minimizing hardware errors. Alongside hardware improvements, it is also critical to advance software components to achieve better accuracy and enhance processing efficiency. In this context, exploring various techniques for optimization, such as model pruning, quantization, and knowledge distillation, could play a pivotal role in reducing the size and computational demands of deep learning models. This optimization is anticipated to significantly reduce processing times. Implementing a lightweight deployment strategy by removing non-essential connections between neurons and setting insignificant weight parameters to zero can help decrease processing time [74]. Collectively, these improvements can bring us closer to achieving optimal precision in both aircraft operation counting and aircraft identification. To facilitate further research, we have published the code, which can be found at https://github.com/Mohammad9292/Aircraft-Detection-Edge/tree/main.

ACKNOWLEDGMENT

The authors would like to thank the anonymous referees for their invaluable insights and contributions that have facilitated the improvement of this article.

REFERENCES

[1] M. Farhadmanesh, A. Rashidi, and N. Marković, "Implementing Haar cascade classifiers for automated rapid detection of light aircraft at local airports," in *Proc. Comput. Civil Eng.*, May 2021, pp. 17–25.
[2] *Operations at Nontowered Airports*, ASI, New Delhi, India, 2020.
[3] S. Kim, Y. Gan, and J. Irizarry, "Framework for UAS-integrated airport runway design code compliance using incremental mosaic imagery," *J. Comput. Civil Eng.*, vol. 35, no. 2, Mar. 2021, Art. no. 04020070.
[4] I. Song, I. Cho, T. Tessitore, T. Gurcsik, and H. Ceylan, "Data-driven prediction of runway incursions with uncertainty quantification," *J. Comput. Civil Eng.*, vol. 32, no. 2, Mar. 2018, Art. no. 04018004.
[5] M. J. Muia, *Counting Aircraft Operations at Non-Towered Airports*, vol. 4. Washington, DC, USA: Transportation Research Board, 2007.
[6] M. J. Muia and M. E. Johnson, *Evaluating Methods for Counting Aircraft Operations at Non-Towered Airports*. Washington, DC, USA: National Academies Press, 2015.
[7] B. Becerik-Gerber, M. K. Siddiqui, I. Brilakis, O. El-Anwar, N. El-Gohary, T. Mahfouz, G. M. Jog, S. Li, and A. A. Kandil, "Civil engineering grand challenges: Opportunities for data sensing, information analysis, and knowledge discovery," *J. Comput. Civil Eng.*, vol. 28, no. 4, Jul. 2014, Art. no. 04014013.
[8] M. Farhadmanesh, N. Marković, and A. Rashidi, "Automated video-based air traffic surveillance system for counting general aviation aircraft operations at non-towered airports," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2677, no. 3, pp. 250–273, Mar. 2023.
[9] M. Farhadmanesh, A. Rashidi, and N. Markovic, "General aviation aircraft identification at non-towered airports using a two-step computer vision-based approach," *IEEE Access*, vol. 10, pp. 48778–48791, 2022.
[10] B. Sherafat, A. Rashidi, and S. Asgari, "Sound-based multiple-equipment activity recognition using convolutional neural networks," *Autom. Construct.*, vol. 135, Mar. 2022, Art. no. 104104.
[11] J. H. Mott and N. A. Sambado, "Evaluation of acoustic devices for measuring airport operations counts," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2673, no. 1, pp. 17–25, Jan. 2019.
[12] F. Sanchez, "Acceptable method of counting aircraft operations at non-tower airports," Center Transp. Res. Educ., Iowa State Univ., Ames, IA, USA, 1996. [Online]. Available: https://rosap.ntl.bts.gov/view/dot/55942
[13] A. Sedunov, A. Sutin, N. Sedunov, H. Salloum, A. Yakubovskiy, and D. Masters, "Passive acoustic system for tracking low-flying aircraft," *IET Radar, Sonar Navigat.*, vol. 10, no. 9, pp. 1561–1568, Dec. 2016.
[14] M. L. Ford and R. Shirack, "Estimating aircraft activity at nontowered airports: Results of the aircraft activity counter demonstration project," *Transp. Res. Rec.*, vol. 958, pp. 24–29, Jan. 1984.
[15] Florida Department of Transportation. (2018). *Operations Counting at Non-Towered Airports Assessment*. Accessed: Feb. 4, 2021. [Online]. Available: http://www.invisibleintelligencellc.com/uploads/1/8/4/9/18495640/2018opscountprojectfinalreport09102018.pdf
[16] L. Rowley, "Radio interference from aircraft electrical equipment," *J. Inst. Electr. Eng. IIIA, Radiocommunication*, vol. 94, no. 13, pp. 484–492, Mar. 1947.
[17] J. H. Mott, "Measurement of airport operations using a low-cost transponder data system," *J. Air Transp.*, vol. 26, no. 4, pp. 147–156, Oct. 2018.
[18] J. H. Mott and D. M. Bullock, "Estimation of aircraft operations at airports using mode-C signal strength information," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 3, pp. 677–686, Mar. 2018.
[19] Federal Aviation Administration. (2022). *ADS-B in and out, Installation*. [Online]. Available: https://www.faa.gov/airtraffic/technology/equipadsb/installation/
[20] J. H. Mott, M. L. McNamara, and D. M. Bullock, "Accuracy assessment of aircraft transponder–based devices for measuring airport operations," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2626, no. 1, pp. 9–17, Jan. 2017.
[21] J. H. Mott, C. Yang, and D. M. Bullock, "Atmospheric pressure calibration to improve accuracy of transponder-based aircraft operations counting technology," *J. Aviation Technol. Eng.*, vol. 9, no. 2, p. 35, Jan. 2021.
[22] V. Yu. Meltsov, A. A. Lapitsky, N. A. Zhukova, and A. I. Vodyaho, "Unit of one-clock SRC error correction for transponder on FPGA," in *Proc. IEEE Conf. Russian Young Researchers Electr. Electron. Eng. (ElConRus)*, Jan. 2021, pp. 154–157.
[23] Vector. (2021). *Vantage*. Accessed: May 7, 2021. [Online]. Available: https://www.vector-us.com/vantage
[24] D. Thirde, M. Borg, J. Ferryman, F. Fusier, V. Valentin, F. Bremond, and M. Thonnat, "A real-time scene understanding system for airport apron monitoring," in *Proc. 4th IEEE Int. Conf. Comput. Vis. Syst. (ICVS)*, Aug. 2006, p. 26.
[25] A. Koutsia, T. Semertzidis, K. Dimitropoulos, N. Grammalidis, and K. Georgouleas, "Automated visual traffic monitoring and surveillance through a network of distributed units," in *Proc. ISPRS*. Princeton, NJ, USA: Citeseer, 2008, pp. 599–604.
[26] J. M. Molina, J. Garcia, A. Berlanga, J. Besada, and J. Portillo, "Automatic video system for aircraft identification," in *Proc. 5th Int. Conf. Inf. Fusion. (FUSION)*, Jul. 2002, pp. 1387–1394.
[27] D. G. Vidakis and D. I. Kosmopoulos, "Facilitation of air traffic control via optical character recognition-based aircraft registration number extraction," *IET Intell. Transp. Syst.*, vol. 12, no. 8, pp. 965–975, Oct. 2018.

[28] Vaxtor. (2021). *Vaxocr*. Accessed: May 7, 2021. [Online]. Available: https://www.vaxtor.com/products/vaxocr/tailfin/

[29] Y. Keshun, Q. Guangqi, and G. Yingkui, "Remaining useful life prediction of lithium-ion batteries using EM-PF-SSA-SVR with gamma stochastic process," *Meas. Sci. Technol.*, vol. 35, no. 1, Jan. 2024, Art. no. 015015.

[30] Y. Keshun and L. Huizhong, "Feature detection of mineral zoning in spiral slope flow under complex conditions based on improved YOLOv5 algorithm," *Phys. Scripta*, vol. 99, no. 1, Jan. 2024, Art. no. 016001.

[31] Y. Keshun, Q. Guangqi, and G. Yingkui, "Optimizing prior distribution parameters for probabilistic prediction of remaining useful life using deep learning," *Rel. Eng. Syst. Saf.*, vol. 242, Feb. 2024, Art. no. 109793.

[32] B. Pan, C. Li, H. Che, M.-F. Leung, and K. Yu, "Low-rank tensor regularized graph fuzzy learning for multi-view data processing," *IEEE Trans. Consum. Electron.*, vol. 70, no. 1, pp. 2925–2938, Feb. 2024.

[33] E. Pan, M. Pan, and Z. Han, "Tensor voting techniques and applications in mobile trace inference," *IEEE Access*, vol. 3, pp. 3000–3009, 2015.

[34] L. Li, S. Zhang, and B. Wang, "Plant disease detection and classification by deep learning—A review," *IEEE Access*, vol. 9, pp. 56683–56698, 2021.

[35] P. Jiang, Y. Chen, B. Liu, D. He, and C. Liang, "Real-time detection of apple leaf diseases using deep learning approach based on improved convolutional neural networks," *IEEE Access*, vol. 7, pp. 59069–59080, 2019.

[36] T. Rahman, A. Khandakar, M. A. Kadir, K. R. Islam, K. F. Islam, R. Mazhar, T. Hamid, M. T. Islam, S. Kashem, Z. B. Mahbub, M. A. Ayari, and M. E. H. Chowdhury, "Reliable tuberculosis detection using chest X-ray with deep learning, segmentation and visualization," *IEEE Access*, vol. 8, pp. 191586–191601, 2020.

[37] Y. Sun, G. Szűcs, and A. R. Brandt, "Solar PV output prediction from video streams using convolutional neural networks," *Energy Environ. Sci.*, vol. 11, no. 7, pp. 1811–1818, 2018.

[38] Y. Keshun, W. Puzhou, and G. Yingkui, "Toward efficient and interpretative rolling bearing fault diagnosis via quadratic neural network with bi-LSTM," *IEEE Internet Things J.*, vol. 11, no. 13, pp. 23002–23019, Jul. 2024.

[39] S. Parikh, D. Dave, R. Patel, and N. Doshi, "Security and privacy issues in cloud, fog and edge computing," *Proc. Comput. Sci.*, vol. 160, pp. 734–739, Jan. 2019.

[40] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[41] J.-H. Huh and Y.-S. Seo, "Understanding edge computing: Engineering evolution with artificial intelligence," *IEEE Access*, vol. 7, pp. 164229–164245, 2019.

[42] W. Yu, F. Liang, X. He, W. G. Hatcher, C. Lu, J. Lin, and X. Yang, "A survey on the edge computing for the Internet of Things," *IEEE Access*, vol. 6, pp. 6900–6919, 2018.

[43] Verizon. (2023). *How Much Does Verizon Charge Per GB of Data*. Accessed: May 10, 2023. [Online]. Available: https://www.verizon.com/support/the-verizon-plan-faqs/#:~:text=Nomatterwhatsizedata,closetoyourdataallowance

[44] Airnav. (2023). *Skypark Airport (KBTF) Operations*. Accessed: May 10, 2023. [Online]. Available: http://www.airnav.com/airport/KBTF

[45] M. Farhadmanesh. (2022). *General Aviation Aircraft Labeled Image Dataset*. Mendeley Data. [Online]. Available: https://data.mendeley.com/datasets/wbnsf4ydc5/2

[46] Z. Zivkovic, "Improved adaptive Gaussian mixture model for background subtraction," in *Proc. 17th Int. Conf. Pattern Recognit. (ICPR)*, Sep. 2004, pp. 28–31.

[47] W. Liu, "SSD: Single shot multibox detector," in *Proc. 14th Eur. Conf.* Amsterdam, The Netherlands. Cham, Switzerland: Springer, Oct. 2016, pp. 21–37.

[48] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.

[49] X. Chen, H. Fang, T.-Y. Lin, R. Vedantam, S. Gupta, P. Dollar, and C. L. Zitnick, "Microsoft COCO captions: Data collection and evaluation server," 2015, *arXiv:1504.00325*.

[50] D. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 2544–2550.

[51] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "HyperBand: A novel bandit-based approach to hyperparameter optimization," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 6765–6816, 2017.

[52] T. O'Malley, E. Bursztein, J. Long, F. Chollet, H. Jin, and L. Invernizzi. (2019). *Kerastuner*. [Online]. Available: https://github.com/keras-team/keras-tuner

[53] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.

[54] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[55] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1800–1807.

[56] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[57] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, and A. Vedaldi, "Fine-grained visual classification of aircraft," 2013, *arXiv:1306.5151*.

[58] Federal Aviation Administration. (2021). *Releasable Aircraft Database*. Accessed: Jan. 12, 2021. [Online]. Available: https://www.faa.gov/licensescertificates/aircraftcertification/aircraftregistry/releasableaircraftdownload/

[59] M. Liao, B. Shi, X. Bai, X. Wang, and W. Liu, "TextBoxes: A fast text detector with a single deep neural network," in *Proc. AAAI Conf. Artif. Intell.*, vol. 31, no. 1, 2017, pp. 1–7.

[60] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Dec. 2001, p. 1.

[61] J. Friedman, R. Tibshirani, and T. Hastie, "Additive logistic regression: A statistical view of boosting (With discussion and a rejoinder by the authors)," *Ann. Statist.*, vol. 28, no. 2, pp. 337–407, Apr. 2000.

[62] R. Lienhart, A. Kuranov, and V. Pisarevsky, "Empirical analysis of detection cascades of boosted classifiers for rapid object detection," in *Proc. Joint Pattern Recognit. Symp.*, Magdeburg, Germany. Cham, Switzerland: Springer, Sep. 2003, pp. 297–304.

[63] J.-J. Lee, P.-H. Lee, S.-W. Lee, A. Yuille, and C. Koch, "AdaBoost for text detection in natural scene," in *Proc. Int. Conf. Document Anal. Recognit.*, Sep. 2011, pp. 429–434.

[64] B. Shi, X. Bai, and C. Yao, "An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 11, pp. 2298–2304, Nov. 2017.

[65] J. Baek, G. Kim, J. Lee, S. Park, D. Han, S. Yun, S. J. Oh, and H. Lee, "What is wrong with scene text recognition model comparisons? Dataset and model analysis," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 4714–4722.

[66] Heber Valley Airport. (2024). *Airport Information*. [Online]. Available: https://hebervalleyairport.com/airportinfo

[67] Aviation View Magazine. (2024). *Logan-Cache Airport*. [Online]. Available: https://aviationviewmagazine.com/logan-cache-airport/

[68] AirportGuide.com. (2024). *South Valley Regional Airport (U42) Information*. [Online]. Available: https://airportguide.com/airport/info/U42

[69] M. Farhadmanesh, A. Rashidi, and N. Markovic, "Image processing and machine learning techniques for automated detection of planes at Utah airports," Utah Dept. Transp. Res. Division, Salt Lake City, UT, USA, Tech. Rep. UT-21.28, 2021.

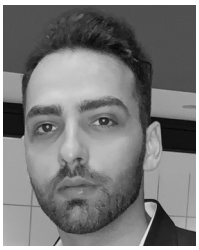
[70] Qualitas. (2022). *How Expensive are Machine Vision Solutions*. Accessed: May 10, 2023. [Online]. Available: https://qualitastech.com/image-acquisition/how-expensive-is-machine-vision-solution/#:~:text=Thetypicalrangeofa,computationcosts%2Candstoragecosts

[71] Federal Aviation Administration. (2021). *ADS-B in and Out, Installation*. Accessed: May 17, 2021. [Online]. Available: https://www.faa.gov/nextgen/equipadsb/installation/

[72] C. Yang, J. H. Mott, B. Hardin, S. Zehr, and D. M. Bullock, "Technology assessment to improve operations counts at non-towered airports," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2673, no. 3, pp. 44–50, Mar. 2019.

[73] I. Intelligence. (2023). *General Audio Recording Device Pricing*. [Online]. Available: http://www.invisibleintelligencellc.com/purchase.html

[74] Y. Keshun and L. Huizhong, "Intelligent deployment solution for tabling adapting deep learning," *IEEE Access*, vol. 11, pp. 22201–22208, 2023.



MOHAMMAD FARHADMANESH received the B.S. and M.S. degrees in civil engineering from Iran University of Science and Technology, Tehran, Iran, in 2014 and 2017, respectively, and the Ph.D. degree in civil engineering from the Department of Civil and Environmental Engineering, The University of Utah, in 2022. His main research interests include machine learning, computer vision, and image processing.

ABBAS RASHIDI (Member, IEEE) received the M.S. degree in civil and environmental engineering from Tehran Polytechnic, in 2004, and the M.S. degree in electrical and computer engineering and the Ph.D. degree in civil and environmental engineering from Georgia Tech, in 2013 and 2014, respectively.

His dual background in civil engineering and electrical and computer engineering, has enabled him to conduct multidisciplinary research and to implement electrical engineering tools and computational techniques to solve complex civil engineering problems. In particular, he is interested in applying audio/image/video processing techniques to analyze and model complex civil engineering systems. He is currently an Associate Professor of civil engineering with The University of Utah. He currently serves as a member for several professional committees, including the Signal Processing in Acoustics Committee of the Acoustical Society of America and the Data Sensing and Analysis (DSA) Committee of the American Society of Civil Engineers (ASCE). He is an Associate Editor and a member of the Editorial Board of two ASCE journals, including *Journal of Construction Engineering and Management* (ASCE) and *Journal of Performance of Constructed Facilities* (ASCE).

ABHISHEK KUMAR SUBEDI received the B.E. degree in civil engineering from Tribhuvan University, Nepal, in 2019. He is currently pursuing the master's degree in civil engineering with the Department of Civil and Environmental Engineering, The University of Utah. His main research interests include machine learning and transportation engineering.

NIKOLA MARKOVIĆ received the B.S. degree in transportation engineering from the University of Belgrade, Serbia, in 2009, and the Ph.D. degree in transportation engineering from The University of Maryland, College Park, MD, USA, in 2013. He is currently an Assistant Professor with the Department of Civil and Environmental Engineering, The University of Utah. He focuses on the development and application of operations research and data science methods to enhance the efficiency of transportation systems. His current research is supported by two grants from the National Science Foundation. He received the Glover-Klingman Best Paper Award, in 2015, and the Transportation Science and Logistics Best Paper Award, in 2022.