Machine Learning at the Edge: UAV Automatic Takeoff and Landing

Anastacia MacAllister, Rey Nicolas

General Atomics Aeronautical Systems, Inc.

Poway, CA

Anastacia.Macallister@ga-asi.com, Rey.Nicolas@ga-asi.com

Sairam Alavuru, Onkar Chougule, Divyansh Gupta, Alicja Kwasniewska, Mikael Sevenier, David Gamba

SiMa Technologies

San Jose, CA

{sairam.a, onkar.c, divyansh.gupta, alicja.kwasniewska, mikael.sevenier, david.gamba@sima.ai}

ABSTRACT

Artificial Intelligence (AI) and Machine Learning (ML) are an increasing area of emphasis for the Department of Defense (DoD). In their 2022 budget the Pentagon requested $4.3 billion dollars for AI/ML related efforts, an over 50% jump in a two-year span. Unfortunately, much of the driving technology is being developed for commercial applications due to the relatively small size and challenges associated with the defense market. Common commercial assumptions like ample compute power, internet connectivity, and stable power are not valid in DoD applications due to deployment in what is referred to as the edge on low size, weight, and power (SWaP) platforms. Edge AI/ML refers to processing data and running algorithms on the deployed device rather than centrally. This concept allows AI/ML to run without connectivity, an important feature for military platforms that can experience degraded communication. In addition, DoD edge computing often happens in low SWaP environments due to mission and cost requirements. This can be challenging for commercially developed technology that often is designed to live in large data centers with access to ample compute resources. As a result, to make AI/ML technology deployable and tactically relevant, edge computing and low SWaP deployment problems need to be addressed. This paper describes the process of deploying AI/ML models in low SWaP environments at the edge. Specifically, the paper looks at developing a vision based automatic takeoff and landing deep learning system for an unmanned aerial vehicle (UAV). The paper details the model development and optimization process for low SWaP deployment. In addition, the paper covers testing of the model using different deployment optimization strategies and the trade-offs associated with each. Ultimately, the paper provides the community with an example of how to address a pressing problem associated with deploying AI/ML for military applications.

ABOUT THE AUTHORS

Anastacia MacAllister, **Ph.D.**, is an AI/ML Solutions Architect for General Atomics Aeronautical Systems, Inc. (GA-ASI). Her work focuses on prototyping novel machine learning algorithms, developing machine learning algorithms using sparse, heterogenous or imbalanced data sets, and exploratory data analytics. Throughout her career she has provided key contributions in several interdisciplinary areas such as prognostic health management, human performance augmentation, advanced sensing, and artificial intelligence aids for future warfare strategies. This work has received several awards and commendations from internal technical review bodies. Dr. MacAllister has published over two dozen peer reviewed technical papers, conference proceedings, and journal articles across an array of emerging technology concepts. In addition, throughout her career, she has worked on several critical defense programs such as the F-35 Joint Strike Fighter, DARPA Air Combat Evolution (ACE) and the Predator family of unmanned aircraft systems from GA-ASI. Dr. MacAllister holds a bachelor's degree from Iowa State University (ISU) in Mechanical Engineering. She also holds a Master's and PhD from ISU in Mechanical Engineering and Human-Computer Interaction.

Rey Nicolas is Director of Software, Autonomy and AI Solutions and leads GA-ASI's Autonomy and AI teams developing next generation Autonomous Command and Control (C2) systems, Autonomous Processing, Exploitation, Dissemination (PED) systems, and AI edge processing for flight and sensor autonomy, air-to-air combat, and air-to-ground Intelligence, surveillance, and Intelligence (ISR) missions. Rey is also an Executive Committee Leader for SAE Standard Works that is developing AI in Aviation Safety Standards. Previously, Rey worked for Intel Corporation and Motorola/Google leading AI R&D and product development for multiple product lines in the smart and connected home, autonomous driving, and healthcare applications. Rey holds a bachelor's degree in Mechanical Engineering from University of California San Diego, Master's in Computer Science from the University of Illinois, and MBA from San Diego State University.

Sairam Alavuru is a ML Applications Engineer at SiMa.ai, with 5 years of experience in the industry, in the AI field, He received his B.Tech from Engineering College, India in Electrical & Electronics Engineering. He worked as a Research Engineer in a product-based firm working on AI based Android applications. He currently works on developing various AI solutions in Computer Vision for Robotics & Surveillance markets.

Onkar Chougule is ML Applications Engineer at SiMa.ai. He has contributed to open-source tvm technology and has worked on developing end to end ML applications. He holds a B.Tech degree in Mechanical Engineering and has co-authored a paper that was accepted in SAE journal.

Divyansh Gupta received his B.S from Academy of Business and Engineering College, India in Computer Science. Gupta worked as a Senior Software Engineer in a product-based firm working on web automation applications. Also, worked on developing methods for Facial Recognition and Pose Estimation during internships. He completed his M.S. in Computer Engineering from Rochester Institute of Technology with 2 publications. He is currently work as a Software Application Engineer at Sima.ai and his research interests include Computer Vision, Deep Learning, and Pose Estimation.

Alicja Kwasniewska, Ph.D, is an ML Architect at SiMa.ai with 10 years of experience in the industry, author of 30+ high-impact journals and international conferences publications in the AI field. Lecturer, panelist, presenter and speaker for multiple international events. Topic Editor of MDPI Imaging Journal, and reviewer of various international journals and conferences: ICML, ICLR, NeurIPS, etc. Co-chair of Society of Automotive Engineers, focused on advancing mobility knowledge and solutions for the benefit of automotive engineering, specifically in the aerospace field. In her PhD dissertation she proposed novel AI models for thermal image processing, awarded with the best PhD dissertation in the AI field. Alicja is also a co-founder and organizer of the International Summer School on Deep Learning educating future generations on the fundamentals of deep learning methods since 2018 and a mentor in non-profit AI projects, e.g., HearAI. Currently she works on various AI solutions in the robotics, aerospace, autonomy, healthcare, smart vision, and surveillance markets with focus on their optimization and efficient execution on AI accelerators.

Mikael Bourges-Sevenier is an ML Fellow at SiMa.ai, previously worked as a multimedia software architect at Motorola Mobility focusing on user experience and multicore applications on mobile devices. He is co-editor of WebCL specification. Prior to Motorola, he was editor of various standards such as MPEG-4, X3D, U3D, and their implementation in products of Adobe, Sun, iVAST, France Telecom. Mikael has an MS in mechanical and electrical engineering from ECAM Lyon, France and an MS in signal and image processing from University Rennes I, France.

David Gamba is the Vice President of Marketing, Sales and Business Development at SiMa.ai and has over 25 years of sales & marketing experience at companies such as Xilinx and Altera and start-ups such as Aeluros and HotRail. David has led groups that have won the largest revenue generating design in FPGA history and have won awards for the introduction of hard floating point into FPGAs, the introduction of OpenCL compilers for FPGAs and sponsoring the very first Machine Learning in FPGAs seminar. David has a BSEE from UCLA, an MSEE from UC Berkeley and an MBA from Stanford University.

Machine Learning at the Edge: UAV Automatic Takeoff and Landing

Anastacia MacAllister, Rey Nicolas

General Atomics Aeronautical Systems, Inc.

Poway, CA

Anastacia.Macallister@ga-asi.com, Rey.Nicolas@ga-asi.com

Sairam Alavuru, Onkar Chougule, Divyansh Gupta, Alicja Kwasniewska, Mikael Sevenier, David Gamba

SiMa Technologies

San Jose, CA

{sairam.a, onkar.c, divyansh.gupta, alicja.kwasniewska, mikael.sevenier, david.gamba@sima.ai}

INTRODUCTION

Artificial Intelligence (AI) and Machine Learning (ML) are an increasing area of emphasis for the Department of Defense (DoD). In their 2022 budget the Pentagon requested $4.3 billion dollars for AI/ML related efforts, an over 50% jump in a two-year span (Deltek, 2022). This interest is due to AI/ML's potential to revolutionize the battlefield, especially for unmanned systems. Unmanned systems enabled by AI/ML driven autonomy have the ability to go into dangerous environments or perform tasks to help reduce cognitive loads on human operators at a fraction of the cost. As a result, several big-name programs have been devoted to exploring autonomy and AI/ML on unmanned platforms such as ACE and AlphaDogFight (Javorsek, 2019). In addition, numerous studies have also been conducted in this field, resulting in innovative application ideas and new research directions, e.g., UAVs Swarm Deployment Framework (Danilchenko & Segal, 2021), UAV Trajectory Pattern Recognition (Pan, Desbarats, & Chaumette, 2019) or UAV ground target tracking under obstacle environments (Li & Wu, 2020). However, while AI/ML has shown enormous potential, there exists a large gap between deploying this technology in a lab vs in the field due to limitations on hardware for DoD specific applications.

This challenge is driven by the fact that much of the technology behind AI/ML is being developed for commercial applications rather than defense. Commercial industry drives AI/ML investment because of its relative size to the defense market. The global AI market size was valued at $328.34 billion in 2021 and is projected to exhibit a Compound Annual Growth Rate (CACR) of 20.1% during next 10 years, reaching more than 4x increase in valuation by 2029 (Fortune Business Insights). Due to flexibility and shorter time to market, software is anticipated to hold the highest AI market share, followed by the services and hardware solutions crucial for ensuring support for increased AI system needs and continuously growing efficiency demand. This large commercial market scale gives companies an incentive to develop products for this market due to their ability to mass produce and take advantage of economies of scale, resulting in higher profits. Contrast this with the defense market. The AI/ML defense market size was $7.9 billion in 2021 and is forecasted to reach $15.1 billion by 2030 (Quadintel). For this market, similar to the global market, software is projected to witness the highest percentage of market share. However, the size difference between defense and commercial markets means that defense solutions are often more expensive to produce and cannot take advantage of economies of scale. As a result, the driving commercial providers of AI/ML hardware and software are not monetarily incentivized to focus on the needs of the DoD market.

As a result, unique DoD needs are not being considered when developing critical AI/ML components. Because DoD applications, especially for hardware, can greatly differ from commercial industry this lack of design for DoD applications creates deployability hurdles for the warfighter. For example, considerations in the DoD often not found in commercial applications are things like encountering interrupted or surging power, compute limitations, smaller footprint requirements, high-level data security, and lack of internet connectivity (Cameron, 2018). These types of considerations are found in the DoD because of what is often referred to as deployment on the edge on low size, weight, and power (SWaP) platforms. Edge AI/ML refers to processing data and running algorithms on the deployed device rather than centrally. This concept allows AI/ML to run without connectivity and with minimal response latency, an important feature for military platforms that can experience degraded communications (Qiu, Kung, & Gai,

2020; Ahamed, Tariq, & Nusir, 2019). In addition, cost minimization and modernization challenges also play a role in driving the DoD to edge computing in low SWaP environments. Due to the prohibitive cost of upgrading systems across the DoD, current legacy systems cannot all be expected to have state of the art computing hardware to run AI/ML algorithms. These compute, bandwidth, and power limitations can be challenging for commercially developed technology that often is designed to live in large data centers with access to ample compute resources. As a result, to make AI/ML technology deployable and tactically relevant, edge computing and low SWaP deployment problems need to be addressed by providing holistic solutions that enable ease of integration with existing infrastructure, seamless software migration and effortless deployment.

This paper describes the process of deploying AI/ML models in low SWaP environments at the edge. Specifically, the paper looks at developing a vision based automatic runway detection, tracking and re-identification deep learning system for unmanned aerial vehicle (UAV)'s automatic takeoff and landing procedures (ATLC). Focus of the work is to perform identification of the environment using only nose cam video, so that in case of other sensors' failure, it would be possible to safely return to the base. In addition, using only vision sensors on the aircraft for ATLC is important for operation in denied environments where other sensors might not be available or reliable, another DoD specific concern for AI/ML deployment. The proposed method is the first step towards building out a complete ATLC system, addressing a large technical hurdle with a DoD legacy low resolution vision system. The paper starts by detailing the runway detection model development and optimization process for low SWaP deployment. Then, the paper covers testing of the model using different deployment optimization strategies and the trade-offs associated with each. Ultimately, the paper provides the community with an example of how to address a pressing problem associated with deploying AI/ML for military applications.

BACKGROUND

Edge and Low SWaP Computing

Edge deployment is becoming a necessity for operational AI/ML deployment in latency-critical DoD systems. However, much work in the AI/ML space to date has focused on cloud computing. Such progress has been driven by the Big Data trend along with cheaper, more advanced, and accessible input sensors. Clouds have become a new computing paradigm and standard for data processing and storage. However, they have their own set of challenges and for many applications it's not sufficient to rely on the cloud due to problems with higher latency, limited bandwidth, connectivity issues and security concerns (Parikh, Dave, Patel, & Doshi, 2019).

Edge computing fills the gaps introduced by the cloud by mitigating latency challenges and significantly enhancing capabilities of existing applications deployed locally (Yang, et al., 2021). Edge computing shifts the paradigm for data collection, analysis, and storage by placing them closer to the point of use (Dahad, 2020). As a result, delays in communications can be reduced to address the round-trip latency time (RTT) of cloud-based systems (Carvalho, et al., 2019). These features help edge devices ensure continuous operation in case of poor connectivity or deployment in restricted areas, as resources usually reside on end devices that don't have to be connected to the backbone network (IEC, 2017), all critical features in UAV use cases (Pandey, et al., 2021).

Sensing at the edge is changing the very nature of computing and its growth is accelerating as conventional platforms struggle to achieve greater performance and efficiency on low SWaP devices. As a result, there is an increasing demand for holistic innovative AI accelerators and specialized software platforms to deploy models on the edge (Shaf, 2020). In addition, demand for purpose-built computer hardware is also increasing for military applications, where the use of low-SWaP devices has significantly increased over the past decade (Friedl, 2018). SWaP considerations are critical for ensuring that no significant weight is added to the system and that it can continuously operate without additional power sources. By addressing the SWaP and edge concerns with a holistic hardware plus software solution smaller footprints and increased efficiency can gained for AI model deployment. This can lead to improved mobility and flexibility of current solutions and makes AI for sensor fusion, decision support or automated environmental awareness more effective for unique DoD applications.

One of the most important use cases where SWaP constraints are crucial are UAVs. The U.S. Department of Defense has recently indicated the need for innovative computing platforms that would allow for lowering the SWaP without

sacrificing performance (Iriarte, 2019). This need is driven by the fact that UAVs are becoming more and more autonomous and thus require enablement of new features such as the ability to execute multiple missions with single payloads (Iriarte, 2019) and further automation of their critical functions, e.g., automated take-off and landing (Cho, et al., 2008). With this increase in mission scope one possible solution would be to enhance the sensing capabilities of the platform with an increased number of sensors, however this creates issues in a low SWaP environment where resources are at a premium. In order to meet these increasing mission requirements, novel AI techniques that use fewer and less expensive sensors to deliver the same functionality are being explored.

Taking this into account, this work focuses on designing automated take-off and landing machine learning pipelines using only vision inputs of relatively low-resolution existing sensors. Efficiency and high performance are achieved by using custom optimization techniques combined with a purpose-built embedded platform that could be easily integrated into UAVs. This directly translates into more compact autonomous systems that can operate longer and produce more complex responses due to more advanced sensing capabilities. As a result, the warfighters can access enhanced surveillance, and reconnaissance (ISR) information (e.g., buildings occupancy, target enemy threats, or locations of mines and weapons) even in case of communication shortfalls.

Computer Vision Pipelines

Before the meteoric growth of Computer Vision methods, understanding the vision inputs was a very laborious task that was rarely deployed in applications due to high complexity and expense. Computer Vision enables machines to interpret visual data and provides computers with an understanding of digital videos and images. It is often used to solve challenging tasks like image recognition, object detection (Akbar, Shahzad, Malik, UL-Hasan, & Shafait, 2019), pose estimation (Gupta, Artacho, & Savakis, 2022), action recognition, segmentation (Chen, Papandreou, Kokkinos, Murphy, & Yuille, 2018), and motion tracking. With the evolution of convolutional neural networks (Krizhevsky, Sutskever, & Hinton, 2012), deep learning, plus increasing amount of data, computer vision has made great strides in both accuracy and speed.

As a result, advancements in Computer Vision techniques have enticed DoD to integrate these cutting-edge technologies into UAVs to handle challenging situations like obstacle collision avoidance, navigation algorithms, and aerial decision-making. UAV are currently being used in multiple applications like military operations such as aerial attacks, traffic monitoring, geo-physics exploration, or navigation purposes. Making sure these important tools return safely is an important part of the warfighting mission. To accomplish this, automatic take-off and landing is a fundamental task for UAV specially in remote, dangerous, or challenging situations. Initially GPS receivers and Inertial sensors such as gyros and accelerometers were widely used for controlling the UAV for automatic landing and takeoff purposes. In (Cho, et al., 2008) a single-antenna GPS receiver along with airspeed sensors are used as a main sensor for UAV takeoff, landing and taxiing. But they already have a predefined path according to which their GPS receiver and sensors receive signals. Any deviation from the preplanned course creates a challenge for the system, limiting its autonomy and applicability in dynamic environments. In addition, GPS receivers are vulnerable to external environments making them a brittle technology to rely upon for UAV landing and navigation in denied environments. With the renewed interest and development in Computer Vision, this technology has shown promise trying to solve the UAV automatic take-off and landing problem. Wenzel et al. in used a Wii remote infrared (IR) camera as the main sensor to track take-off and landing patterns of UAVs (Wenzel, Masselli, & Zell, 2011). However, the use of IR cameras has several limitations such as the system cannot be applied to outdoor flights because of the sensor sensitivity to direct sunlight. In addition, the detection region is limited to 2.5 m due to IR camera range. In Saripalli et al., the authors developed a multi-stage UAV landing algorithm using vision data. Their multi-stage approach includes preprocessing the input data using median filtering and segmentation techniques to find geometrical shapes for the task of landing object recognition and finally state estimation (Saripalli, Montgomery, & Sukhatme, 2002). Though the results were encouraging, the multi-stage process increases the complexity and latency of the whole network. Kwasniewska et al. proposed a method for aircraft detection. The model was able to predict aircraft features such as tail, wings, etc. along with the orientation of the aircraft by experimenting with various deep learning solutions in a low SWaP environment (Kwasniewska, et al., 2022). It used multi-instance satellite imagery characterized by rich background and small spatial resolution of objects, which is analogous to runway detection by an UAV camera system where the runway has a small spatial resolution in the image. The following sections of the paper describe model creation and testing processes for the challenging task of autonomous UAVs Take-off and landing using computer vision techniques and deep neural networks in a low SWaP environment.

METHODS

In this section, the core functionality of the proposed automated ATLC system is described. Specifically, the implemented pipeline allows for runway detection and tracking to improve sensing capabilities of UAVs. In addition, the pipeline is enhanced with an additional neural network aimed at runway re-identification in case of obstacles, such as other aircraft, clouds, or other objects.

The initial ground testing deployment configuration is presented in Fig. 1. First, the ML embedded platform is deployed on the ground and communicates with one selected aircraft. The communication steps are as follows: drone camera sends the video data to the encoder which is hooked up to a network switch in the flight system. The encoder sends out the compressed video data (H.264 compression) together with metadata specifying the time/location of data acquisition (if available) to a multicast IP address and port that is specified (UDP multicast) using MPEG 2 Transport Stream container together with metadata if applicable. A computer that is on that same network switch is connected to the ML embedded platform which has ML accelerator capabilities and AI software. The AI software listens on that specific IP address and port to get the video feed through an Ethernet link from Switch. ML platform decodes, preprocesses, and processes the data and sends predictions with corresponding frame/metadata back to the host using Ethernet.

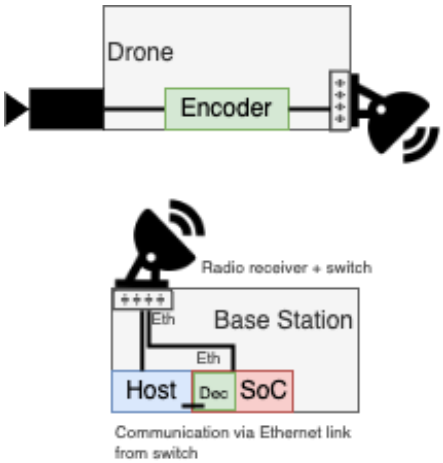

Figure 1. Deployment configuration

Later configurations after testing are completed will include direct deployment of the system on UAV. Ultimately, steps implemented for the application are aimed at performing detection, tracking and re-identification of runways using a combination of Neural Networks (NNs) and Computer Vision (CV) algorithms. The overview of the entire air-based pipeline is shown in Fig. 2.

The input data for the system is produced by the imaging camera placed on UAVs (nose camera) in 720p resolution. Data may be acquired in a visible spectrum as well as an infrared spectrum, however, at this step we cover visible spectrum only and will propose a solution for both spectrums in future work. Acquired images must be decoded and converted to the RGB color space. Then data is resized to match the model input feed and normalized using mean and standard distribution of the training data. In the first step the feature extraction portion (backbone) of the detection model extracts representations required for object localization. Then, the remaining portion of the detection model produces coordinates of identified objects together with their category and confidence threshold. Since our problem at this stage is a binary detection problem (one category - runway), we are only interested in a single class output. Usually, detection models require some post-processing operations to convert produced outputs to top left and right bottom points representing the position of objects, filter out irrelevant bounding boxes, and map coordinates back to the original input. At this step, we have information about object coordinates, however, we are also interested in object tracking and re-identification in case of object occlusion, disappearance from the view or unusual motion patterns. Thus, follow on processing steps are required for the imagery.

Inspired by the Deepsort algorithm (Wojke, Bewley, & Paulus, 2017), we decided to apply a similar concept in our case and use a combination of CV-based tracking using Kalman filtering concurrently with NN. These two pieces are responsible for extracting object embeddings that could be then compared with all stored object profiles to restore the correct ID of the object. After we detect an object from a frame, we crop the object using the co-ordinates predicted by the CenterNet object detection model. The cropped image containing the object is then fed to Resnet model which produces the object features. These object features are mapped to an object identity in the form of map, so these features correspond to object 1. Now when another object is detected in the next frame, if the cosine similarity between features produced by this object is same as that produced by the object present in previous frame, it's the same object. This provides the ability to track objects historically throughout the frames for later more complex autonomy enabled decision making.

As shown on Figure 2, such a pipeline is complex and requires multiple specialized backends to handle all processing steps: input must be decoded requiring on-board Encoder/Decoder capabilities, pre/post processing and CV-based

tracking. These steps are usually done higher precision and must be run on DSP-like engines or specialized image processing computes which can require significant compute power. Moreover, the pipeline uses multiple concurrent neural networks that should be supported by the ML accelerators. In all to accomplish the task of runway detection, there are many complex power-hungry steps. However, SWaP limitations on the UAV dictate a lightweight solution. As a result, the special processing pipeline described below is utilized to reduce the model complexity to ensure high performance, low latency, and small SWaP of the AI/ML models developed for runway detection.

Model Development

The first step was to select a model used for detection of the runway. We decided to not utilize the segmentation model that would provide pixel-level class annotations as it's more computation heavy and would increase power consumption of the implemented pipeline. In addition, when analyzing data, we concluded that bounding box information is sufficient for the landing procedure, as UAVs are looking for the center of the landing zone, and precise detection of lines is not required. In the future we will perform more experiments with other models as well, including landmark detection models that could give us a non-rectangular landing area. Instead, a detection model was used which provides the bounding box location of the object of interest. One of the most important factors in deep neural network training is a sufficient amount of high-quality data used for optimization of weights. As such, in this section the data collection, cleaning and preparation process are steps explained, then followed by the model design and training details.

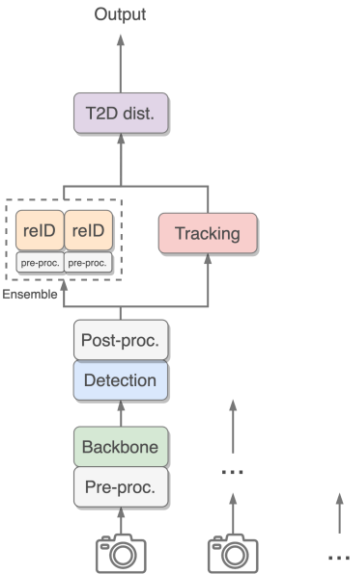

Figure 2. Overview of the proposed pipeline

Data collection
Training data for the model is collected from the imaging system on the UAV at 720p resolution. The drone was flown in varying weather conditions and different times of day to improve the dataset quality and variety. Videos recorded by the camera placed on the drone were saved using MPEG encoding. In total there were around 14 videos with different lengths ranging from 15 to 30 minutes. A video has RGB as well as IR color space frames, and they switch the color space in the same video instantly.

Data exploration
Most of the videos start with the drone already in the air, it flies in the hills for some time, lands on the runway, flies back in the sky, lands again, then taxis on the runway. The drone takes around 1-2 minutes to complete the land and fly loop. So, the amount of time where the runway is visible in the video is much less than the amount of time in the sky. The number of times the drone lands on the runway ranges from 1 to 5 in different videos. The landing of the drone was captured from multiple angles, different weather conditions like sunny, cloudy etc. and different daylight conditions as presented in Fig. 3. This diversity helps make models robust in practical situations.

Data preparation
First, we extracted images from videos and created a robust dataset by including images with and without runway visibility. An external tool was used to label the images marking the (x,y) coordinates of the runway covering the whole runway in a quadrilateral shape. We self-annotated the four corner coordinates of the runway to create the ground truth of runway location for the detection model. In total, around 400 frames were extracted from the collected videos by taking around 50 frames per landing. After initial model training, the accuracy was not sufficient (0.07 mAP), even though the selected model (Faster-RCNN) was one of the most accurate since it is a two-stage model tuned to identify small-scale objects in the frame. To improve the model performance, the data set was curated by removing images with the drone on the runway, which potentially confused the model whose goal is airborne runway detection, resulting in 293 images. Out of that 210 were split as training and 83 were kept for testing. After removing the images taken on the ground, the accuracy of the model significantly improved to mAP equal 0.305.

Figure 3. Input images from the dataset

Model Training

After the satisfactory results achieved by the two-stage model, we decided to explore more compact object detection architectures that would allow for improving performance of the pipeline even further. The CenterNet model was selected due to its efficient fully convolutional architecture that eliminates the need for using costly non-max suppression functions. During initial model training, it failed on corner cases such as runway too far, runway not clearly visible, runway not present in image center, images too blurry, and runway at an angle w.r.t Y-axis. Further data set curation was conducted by adding 100 additional images with corner cases. This helped improve the accuracy of the model. Achieved results are presented in the next section. The training environment setup for the CenterNet was a Linux (Ubuntu 18.04) based GPU server and the parameters specified in Table 1. For the re-identification portion of the pipeline the off the shelf bag of Tricks re-identification model was used (Wojke, Bewley, & Paulus, 2017). Tracking was performed using the Kalman filtering algorithm.

Table 1. Training parameters for the object detection model

Parameters	**Values**
batch_size	32
learning rate	0.000125
num_epochs	140
lr_step	[90, 120]
arch	centernet with dla34 backbone
num_classes	1
reg_loss	l1

Model Optimization for Deployment

To ensure the workload meets SWaP constraints, the model must be properly prepared and modified without degradation in behavior or prediction accuracy. Performed optimizations are threefold: 1) a series of model optimization passes to generate simplified and more efficient computational graph 2) the numerical precision of the model is reduced to integer arithmetic only 3) to alleviate the costs of model sizes with minimal impact on its performance, the weights are represented in a more efficient format using the model compression technique. The optimization passes and quantization techniques are described in detail in the following subsections:

Optimization passes

Optimization pass is a process of applying a series of graph modifications, so that the performance metrics, such as latency, memory footprint, or power consumption are improved, which allows for meeting SWaP constraints. Graph modifications can include standard optimizations as well as machine learning-specific optimizations (Boehm, et al., 2018). In order to simplify the process of applying and managing such optimizations, we base our software toolchain on Tensor Virtual Machine (TVM) (Apache, 2022). TVM contains a series of optimization passes including constant folding, dead code elimination, operator layout alteration, operator fusion, buffer handling, and loop transformation that we can leverage. In addition, we also develop a set of custom passes to further optimize the network and prepare it specifically for low-SWaP UAV platforms. The software toolchain is designed in such a way that all passes are performed automatically without the need for user's intervention to ease the process of model porting, integration, and deployment. Table 2 presents examples of introduced passes with a brief description of their functionality. Please refer to (Apache, 2022) for a more detailed summary.

Table 2. Examples of applied optimization passes

Pass Type	Description
MergeComposite	Merge a series of subsequent operators into one to reduce computations, examples include a conv2D-add-relu sequence
Decomposition	Decompose an operator into a sequence of simple operations to simplify implementation and scheduling
RemoveRedundant	Evaluate if the operator is redundant and reduce if it doesn't change the model behavior
RemoveUnusedFunctions	Remove unused global relay functions in a relay module.
DeadCodeElimination	Remove expressions that do not have any users (dead code).
DynamicToStatic	If possible, convert dynamic operators to static versions
ConvertLayout	Convert channel first data layout to channel last
CanonicalizeOps	Canonicalize special operators to basic operators
SimplifyInference	Simplify the data-flow graph for inference phase
FoldConstant	Fold the constant expressions to their corresponding operators
PartitionIRModel	Partition the graph to achieve a desired performance/accuracy tradeoff

Quantization

Quantization is a process of reducing the number of bits that represent weights and activations of a neural network to reduce memory and power consumption and thus create more efficient applications. So far, the predominant numerical format used for model training was 32-bit or 16-bit floating point. Recently other numerical precision has also become popular, such as 16-bit brain floating point. However, the desire for even more reduced bandwidth and compute requirements has driven the progress of integer only arithmetic for the inference (Benoit, et al., 2018). We also take advantage of this technique and introduce fully automated post training linear quantization of neural networks. In linear quantization, a float value is converted into an integer format by multiplying it with a numeric constant (the scale factor). The post training mode means that no changes to the training pipelines are required, as the entire optimization is performed on the frozen model. In the simplest form, the quantization can be represented as:

$$x_q = round(x_f s), \qquad \text{Eq. 1}$$

where x_q is the quantized value, $round$ is the rounding operation, x_f is the floating-point original value and s is the scale factor. As can be seen in Eq. 1, an important factor in quantization is selection of the scale factor that directly corresponds to the quantization range. To simplify the computation, instead of mapping the exact minimum and maximum value of the float range to the quantized range, we use the maximum absolute value from the minimum and maximum values, so that the utilized range is always symmetric as presented in Fig. 4. In addition, there is no need for zero-point correction, as it's always zero.

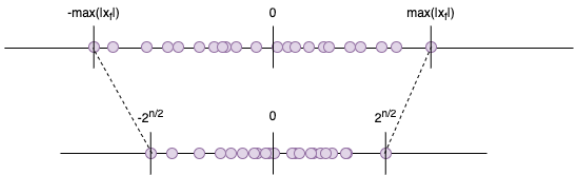

Figure 4. Symmetric quantization

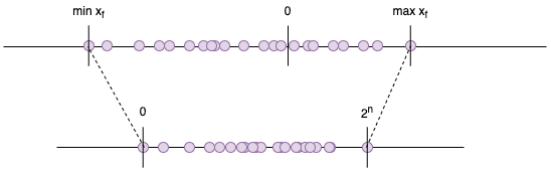

Figure 5. Asymmetric quantization

On the other hand, some models use non-normal, skewed distributions and the use of symmetric ranges may lead to big errors, as significant dynamic range will be dedicated to never seen values. In this case, better utilization of the quantized range is required. This quantization mode is known as asymmetric because we exactly map the minimum and maximum values from the float range to the quantized range (see Fig. 5). However, such an approach is more difficult in implementation, because zero-point correction is needed which requires additional logic in hardware, then Eq. 1 changes to:

$$x_q = round(x_f s - min_{x_f} s), \quad \text{Eq. 2}$$

where x_q is the quantized value, $round$ is the rounding operation, x_f is the floating-point original value, s is the scale factor, and $min_{x_f} s$ is the zero point, denoted as zp. To further optimize computation and reduce power consumption we utilize integer multiplication followed by bitwise shift operation instead of multiplication with the floating-point scale factor $s \approx \frac{A}{2^n}$, where A is the integer multiplier and n is the number of bits used in the bitwise shift operation. Given the specific hardware implementation and m indicating the maximum number of bits, we can define the upper limit for A:

$$2^n s \leq 2^m - 1 \quad \text{Eq. 3}$$

From this, we can define the number of bits (n) in the bitwise shift operation as:

$$2^n = \lfloor \frac{2^m - 1}{s} \rfloor \quad \text{Eq. 4}$$

$$n = \lfloor log_2(\frac{2^m - 1}{s}) \rfloor \quad \text{Eq. 5}$$

Applied quantization leads to sparse and redundant information that we exploit to further reduce tensor storage in memory. All these processing steps combined help reduce the model complexity and power requirements with minimum accuracy reductions, better allowing deployment on SWaP constrained UAVs.

RESULTS AND DISCUSSION

After performing all graph optimizations, the performance of the solution was evaluated by computing the mean average precision score (mAP) for different runway sizes in the image and for different post processing pipeline cases. The mAP score is the average under the precision recall curve. It is a weighted way to describe model performance both in terms of precision and recall based off of different intersection over union (IoU) threshold detection settings. It is one of the standard accuracy metrics in computer vision applications. By varying the amount of runway visible, we can gauge how well the model can detect runways at far away distances as the UAV approaches. In addition,

Figure 6. Examples of images with detected runway

results below show accuracy when subsequent error reducing steps were applied to models to reduce accuracy loss from the quantization process described above.

Qualitative results of the runway detector are shown in Fig. 6. Initial inspection shows that the model can detect the runway from various camera distances. For quantitative evaluation, different potential deployment platforms were compared by evaluating performance and accuracy of custom runway detection models optimized for edge deployment using the pipeline described above.

Table 3. Initial object detection model accuracy by numerical precision and runway size

				Raw fp32	SiMa.ai INT8	SiMa.ai INT8, [CASE I]	SiMa.ai INT8, [CASE II]
Average Precision (AP)	IoU=0.50:0.95	area= all	maxDets=100	0.405	0.366	0.369	0.400
Average Precision (AP)	IoU=0.50	area= all	maxDets=100	0.952	0.851	0.851	0.927
Average Precision (AP)	IoU=0.75	area= all	maxDets=100	0.328	0.352	0.338	0.385
Average Precision (AP)	IoU=0.50:0.95	area= small	maxDets=100	0.300	0.242	0.274	0.244
Average Precision (AP)	IoU=0.50:0.95	area=medium	maxDets=100	0.542	0.565	0.551	0.563
Average Precision (AP)	IoU=0.50:0.95	area=large	maxDets=100	0.329	0.299	0.299	0.342
Average Recall (AR)	IoU=0.50:0.95	area= all	maxDets=1	0.496	0.490	0.490	0.494
Average Recall (AR)	IoU=0.50:0.95	area= all	maxDets=10	0.496	0.495	0.495	0.495
Average Recall (AR)	IoU=0.50:0.95	area= all	maxDets=100	0.496	0.495	0.495	0.495
Average Recall (AR)	IoU=0.50:0.95	area= small	maxDets=100	0.395	0.352	0.386	0.352
Average Recall (AR)	IoU=0.50:0.95	area=medium	maxDets=100	0.603	0.617	0.597	0.617
Average Recall (AR)	IoU=0.50:0.95	area=large	maxDets=100	0.426	0.436	0.436	0.436

Table 3 shows initial model quantitative results (mAP values) on the evaluation dataset for the fp32 model and the pipeline quantized int8 model. The table shows accuracy of the runway detection model in standard computer vision COCO metrics, where Average Precision (AP): (IoU=0.50:0.95) is the primary metric. The IoU setting controls the

threshold for a runway being counted as detected by the model against the ground truth label. For example, if the IoU is 0.50 then only 50% of the pixels in the model prediction and the ground truth need to match for it to be a detection. The higher the IoU the tighter the detection standards and the more accurate the model must be at locating the runway to meet them. Ultimately, higher mAP values mean the model detects runways better in captured frames.

Testing results show, average testing accuracy for an IoU range of (IoU=0.50:0.95) are 0.405 for the raw fp32 model & 0.366 for the raw int8 model. However, applying the quantization and optimization process in case I and II brings the int8 score more closely in line with the fp32. Ultimately, fp32 has benefits like high accuracy, but brings drawbacks like high latency, high memory, high power consumption, and lower throughput. Whereas the int8 model has benefits like low latency, lower memory, lower power consumption, higher throughput but has slightly less accuracy. Results show that for applications such as SWaP constrained platforms the minimal accuracy tradeoff for the quantized int8 model could help ensure promising AI/ML technology makes it into the hands of the warfighter.

Taking a deep dive into the process, much of the model accuracy loss results from quantizing a layer to run in int8 precision due to the conversion from fp32 to int8. This accuracy loss depends on various factors like the type of quantization, distribution of the inputs to the layer and distribution of weights of the layer etc. To find the layer that introduces the maximum quantization error and mitigate that error, per layer analysis is used. This allows the model to be optimized for maximum accuracy after passing through the pipeline, minimizing any loss from the fp32 to int8 conversion process. To analyze a layer, it is run in fp32 precision i.e., using fp32 input and fp32 weights. Then the same layer is run in int8 precision, using int8 input quantized from fp32 output of previous layer and quantized int8 weights of the current layer. To gauge the impact on accuracy for a layer, MAE (Mean Absolute Error), MSE (Mean Squared Error), PSNR (Peak Signal to Noise Ratio) is calculated between these two outputs.

Looking again at Table 3, [CASE I] shows that when the convolutional layer producing width, height values of the bounding boxes is kept in fp32, the accuracy improves. Even better accuracy is achieved [CASE II] when the output of the convolutional layer producing the confidence score is kept in int32 precision. This shows that by selectively optimizing portions of the model the best of accuracy and low SWaP deployment considerations can be baked in.

Overall, the results in Table 3 show that the optimized int8 model has comparable performance to the fp32 model, with lower latency, power consumption, and memory required. These factors help in making the int8 model more deployable in low SWaP environments such as legacy UAVs. Thus, it helps in improve real-time performance of the UAV system in warzone areas providing accurate and quick results required for take-off and landing of drones on the battlefield.

CONCLUSIONS AND FUTURE WORK

Research conducted for this paper proved the feasibility of an autonomous UAV system using deep neural networks for takeoff and landing, detection of objects of interest such as runways in a low SWaP deployment environment. The introduced heterogeneous image processing pipeline allowed for accurate detection and tracking of runways. In addition, the proposed custom graph optimizations showed that accuracy of predictions is not affected while significantly improving the efficiency of the solution, which was shown in the accuracy, throughput, and power consumption benchmarks. Based on the conducted analysis, it has been proved that the proposed solution is suitable for SWaP constrained platforms, such as UAVs.

The presented work is an important step towards enabling development of the fully automated ATLC systems using relatively low-resolution image inputs, while making sure the end-to-end pipeline meets critical SWaP constraints needed for military platforms. As has been shown in the performed analysis, hardware software co-design is very important to ensure low latency and high accuracy of the deployed models. However, DoD is increasingly moving toward deploying autonomous AI/ML systems and even more precise and efficient systems will be needed. To make sure the proposed solution meets requirements when deployed in production, it’s important to further verify it in practice under real life military centric edge and low SWaP environments. Most hardware is developed for perfect lab conditions and the commercial world in mind. We will expand our analysis and in future work will focus on showing an example of real-life deployment that will help accelerate deployment of promising tech into warfighters' hands for the 21st century battle.

Future work will also focus on improving performance of the ATLC system, fusion of input from more sensors in the UAV for more robust navigation and detection in more challenging environments. In addition, more advanced analytics will also be added to the core functionality to further improve sensing capabilities. Some of the use cases include calculation of a distance to the runway from camera inputs only, recognizing activity of other objects in the scene, running predictive models concurrently with computer vision pipelines, etc.

REFERENCES

Ahamed, A., Tariq, U., & Nusir, M. (2019). Real-Time Methodology for Improving Cyber Security in Internet of Things Using Edge Computing During Attack Threats. *2019 International Conference on Smart Systems and Inventive Technology (ICSSIT).* IEEE.

Akbar, J., Shahzad, M., Malik, M., UL-Hasan, A., & Shafait, F. (2019). Runway Detection and Localization in Aerial Images using Deep Learning.

Apache. (2022, June 10). *TVM*. Retrieved from https://tvm.apache.org/docs/reference/api/python/relay/transform.html

Benoit, J., Skirmantas, K., Chen, B., Zhu, M., Tang, M., Howard, A., . . . Kalenichenko, D. (2018). Quantization and training of neural networks for efficient integer-arithmetic-only inference. *Proceedings of the IEEE conference on computer vision and pattern recognition.* IEEE.

Boehm, M., Reinwald, B., Hutchison, D., Sen, P., Evfimievski, A., & Pansare, N. (2018). On Optimizing Operator Fusion Plans. *Proceedings of the VLDB Endowment*.

Cameron, L. (2018). *Internet of Things Meets the Military and Battlefield.* IEEE.

Carvalho, A., O'Mahony, N., Krpalkova, L., Cambell, S., Walsh, J., & Doody, P. (2019). Edge Computing Applied to Industrial Machines. *Procedia Manufacturing*.

Chen, C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, L. (2018). DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence.*

Cho, A., Kim, J., Lee, S., Kim, B., Park, N., Kim, D., & Kee, C. (2008). Fully automatic taxiing, takeoff and landing of a UAV based on a single-antenna GNSS receiver. *IFAC Proceedings Volumes*.

Dahad, N. (2020). *'Connected Embedded Intelligence' Requires a Smarter Edge.* Retrieved from EE Times: https://www.eetimes.eu/connected-embedded-intelligence-requires-a-smarter-edge/

Danilchenko, D., & Segal, M. (2021). An Efficient Connected Swarm Deployment via Deep Learning. *16th Conference on Computer Science and Intelligence Systems (FedCSIS).* IEEE.

Deltek. (2022). *Federal Artificial Intelligence Landscape.* Herndon.

Fortune Business Insights. (n.d.). *Artificial Intelligence Market.* Retrieved from https://www.fortunebusinessinsights.com/industry-reports/artificial-intelligence-market-100114

Friedl, K. E. (2018). Military applications of soldier physiological monitoring. *Journal of Science and Medicine in Sport*.

Gupta, D., Artacho, B., & Savakis, A. (2022). HandyPose: Multi-level framework for hand pose estimation. *Pattern Recognition*.

IEC. (2017). *Edge Intelligence.* Retrieved from https://storage-iecwebsite-prd-iec-ch.s3.eu-west-1.amazonaws.com/2019-09/content/media/files/iec_wp_edge_intelligence_en_lr.pdf

Iriarte, M. (2019, April 15). Military UAVs tackle performance issues under SWaP-driven designs. Retrieved from https://militaryembedded.com/unmanned/isr/military-uavs-tackle-performance-issues-under-swap-driven-designs

Javorsek, D. (2019). Air Combat Evolution (ACE). *DARPA-BAA-HR001119S0051*, (pp. 1-24).

Kalamkar, D., Mudigere, D., Mellempudi, N., Das, D., Banerjee, K., Avancha, S., . . . Yang, J. (2019). A study of BFLOAT16 for deep learning training. arXiv preprint arXiv:1905.12322 .

Krizhevsky, A., Sutskever, I., & Hinton, G. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems.*

Kwasniewska, A., Chougule, O., Kondur, S., Alavuru, S., Nicolas, R., Gamba, D., . . . MacAllister, A. (2022). AI-Based Rotation Aware Detection of Aircraft and Identification of Key Features for Collision Avoidance Systems. SAE.

Li, B., & Wu, Y. (2020). Path Planning for UAV Ground Target Tracking via Deep Reinforcement Learning. *IEEE Access.* IEEE.

Pan, X., Desbarats, P., & Chaumette, S. (2019). A Deep Learning based Framework for UAV Trajectory Pattern Recognition. *2019 Ninth International Conference on Image Processing Theory, Tools and Applications (IPTA).* IEEE.

Pandey, S., Kim, K., Alsenwi, M., Kyaw Tun, Y., Han, Z., & Seon Hong, C. (2021). Latency-Sensitive Service Delivery With UAV-Assisted 5G Networks. *IEEE Wireless Communications Letters* . IEEE.

Parikh, S., Dave, D., Patel, R., & Doshi, N. (2019). Security and Privacy Issues in Cloud, Fog and Edge Computing. *Procedia Computer Science*.

Qiu, M., Kung, S.-Y., & Gai, K. (2020). Intelligent security and optimization in Edge/Fog Computing. *Future Generation Computer Systems*.

Quadintel. (n.d.). *Ai In Defense Market Size, Share & Trends Analysis - Global Opportunity Analysis And Industry Forecast 2030.*

Saripalli, S., Montgomery, J., & Sukhatme, G. (2002). Vision-based autonomous landing of an unmanned aerial vehicle. *Proceedings 2002 IEEE International Conference on Robotics and Automation.* IEEE.

Shaf, J. (2020). The future of computing beyond Moore's law. *Philosophical Transactions of the Royal Society A 378.2166*.

Wenzel, K. E., Masselli, A., & Zell, A. (2011). Automatic take off, tracking and landing of a miniature UAV on a moving carrier vehicle. *Journal of intelligent & robotic systems*.

Wojke, N., Bewley, A., & Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. *IEEE international conference on image processing.* IEEE.

Yang, H., Li, G., Sun, G., Meng, X., Yu, H., Xu, W., . . . Ying, X. (2021). Dispersed Computing for Tactical Edge in Future Wars: Vision, Architecture, and Challenges. *Wireless Communications and Mobile Computing*.