Engineering Challenges for AI-Supported Computer Vision in Small Uncrewed Aerial Systems

Muhammed Tawfiq Chowdhury and Jane Cleland-Huang
Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, Indiana 46556, USA
Email Addresses: mchowdhu@nd.edu, Janeclelandhuang@nd.edu

***Abstract*—Computer Vision (CV) is used in a broad range of Cyber-Physical Systems such as surgical and factory floor robots and autonomous vehicles including small Unmanned Aerial Systems (sUAS). It enables machines to perceive the world by detecting and classifying objects of interest, reconstructing 3D scenes, estimating motion, and maneuvering around objects. CV algorithms are developed using diverse machine learning and deep learning frameworks, which are often deployed on limited resource edge devices. As sUAS rely upon an accurate and timely perception of their environment to perform critical tasks, problems related to CV can create hazardous conditions leading to crashes or mission failure. In this paper, we perform a systematic literature review (SLR) of CV-related challenges associated with CV, hardware, and software engineering. We then group the reported challenges into five categories and fourteen sub-challenges and present existing solutions. As current literature focuses primarily on CV and hardware challenges, we close by discussing implications for Software Engineering, drawing examples from a CV-enhanced multi-sUAS system.**

***Index Terms*—Small Uncrewed Aerial Systems, Computer Vision, Artificial Intelligence**

I. Introduction

Computer Vision (CV) supports many different tasks including object detection, autonomous navigation, and surveillance by small Uncrewed Aerial Systems (sUAS). All of these are critical for the success of diverse missions such as emergency response [1], fire detection [2], parcel delivery [3], and search and rescue [4] missions. However, there are many challenges associated with achieving effective CV on sUAS, many of them are introduced by the significant computational needs of deploying deep-learning algorithms on a highly-resource constrained edge environment, and exacerbated by real-world environment conditions related to weather, terrain, lighting, aerial perspectives, and the constant motion and vibration of the sUAS. These challenges have traditionally been under-explored in the literature, especially in core CV publications, which tend to focus on developing and validating novel algorithmic solutions using static datasets of images, rather than solving the challenges of deploying CV on cyber-physical systems in real-time applications. Without guidelines for how to engineer CV-based, software-intensive sUAS systems, newcomers to the field will inevitably waste time and resources as they learn these lessons the hard way.

To motivate the need for such guidelines, we describe our own missteps as software engineers while deploying CV on sUAS over the past couple of years. Our task was to equip our sUAS to detect and then track people during a search-and-detect mission. We started by experimenting with CV pipelines that were capable of processing a video stream, detecting a person (or people), and raising an alert. We ran an extensive series of experiments on several different Nvidia Jetson models, compared the accuracy of various CV person-detection algorithms and pre-trained models, and selected YOLO V3 and YOLO V4 object detection algorithms. We integrated the CV pipeline into our onboard autopilot, ran extensive simulations until the pipeline worked efficiently, and finally deployed it onto our sUAS environment using an Nvidia Jetson Xavier NX carrier board and an IMX477 camera. However, fitting all of our software and CV modules onto the carrier-board version of Jetson NX was extremely challenging. The Jetson was initially underpowered and quickly became overheated, requiring hardware fixes that included a stepdown transformer and additional airflow through a makeshift cooling system. The gimbal movements of our sUAS were initially misaligned with those in the Gazebo simulator, and the physical placement of the antenna caused interference in the image stream. In addition, during flight, the CV algorithms and autopilot competed for processing cycles, initially causing jerky flight maneuvers. Finally, detection accuracy significantly underperformed in comparison to the results obtained in the pristine, experimentation environment. Each of these problems translated into days, and even weeks, of time-consuming and challenging fixes by our hardware and software engineering teams.

This paper takes a systematic look at these challenges, many of which are directly or indirectly related to the deployment of AI on a resource-constrained edge device. We report on a preliminary systematic literature review (SLR) of CV usage, challenges, and solutions when deployed on sUAS. We label this a preliminary SLR because of the breadth of issues that are covered and the need to take a deeper dive into many of the individual challenges in future work. We address the following research questions:

RQ1: What technical challenges, associated with the deployment of CV on sUAS platforms, are presented in existing literature?

RQ2: What common solutions for addressing these challenges have been proposed?

RQ3: What are the implications of these challenges on the Software Engineering process?

The aim of this study is, therefore, to explore the intersection of Software Engineering and the deep-learning (AI) aspects of deploying CV on limited resource, edge-based sUAS platforms. However, our SLR analysis returned far more information about the CV and hardware-related problems and had little to say about actual Software Engineering challenges at the intersection of CV and sUAS systems design. One of our findings is, therefore, that a clear gap exists in the literature, highlighting the need for more focused work in this emergent area. Despite this lack of prior work, this paper lays important foundations for future exploration through the following contributions:

- It identifies challenges and solutions associated with CV, hardware, and software aspects of deploying CV in sUAS applications, providing fundamental insights for software engineers building systems in this space. The aim is to equip Software Engineers with the knowledge that may help them avoid the kinds of missteps that we experienced due to our initial lack of domain knowledge.
- It offers a simple process model highlighting one of our overarching findings that CV, hardware, and software should be developed and tested in unique workflows, and then integrated incrementally through clearly defined, frequent integration tests that progress rapidly from simulation to the real-world.
- Given the lack of Software Engineering research in this area, it discusses implications for Software Engineering of CV-based sUAS systems with pointers to future work.

The remainder of this paper is laid out as follows. Section II discusses the background information and related work. Section III describes the SLR process including search terms, papers retrieved and analyzed, and the process for identifying challenges and solutions. Sections IV to VIII describe the five challenge areas identified through our preliminary SLR, as well as sub-challenges and potential solutions, and then Section IX discusses the implications of these findings upon Software Engineering practices. Section X describes a case study based on our Drone Response system. Section XI discusses the two primary threats to validity and Section XII presents conclusions and future work.

II. Background and Related Work

An sUAS is a small uncrewed aircraft, and includes all of the onboard and offboard hardware and software components needed for its communication and control. An sUAS can be non-autonomous, semi-autonomous, or fully autonomous. Autonomous sUAS typically carry sensors, such as cameras, in order to perceive the world around them using onboard CV. Their cameras are often mounted on gimbals to control their attitude, comprised of roll, pitch, and yaw.

CV uses machine learning (ML) and deep learning (DL) algorithms to identify different classes of objects in images and/or image streams, with models typically trained, tested and validated using large datasets of images. Once trained, they can be used by software and cyber-physical systems (CPS) to perform activities such as object recognition and depth perception. CV is typically implemented as a pipeline that broadly involves (1) image acquisition, (2) data processing to remove noise, perform frame scaling, and make color corrections, (3) identification of areas of interest using techniques such as segmentation, (4) analysis and recognition, and finally (5) decision making.

A. CV Application Areas for sUAS

CV is used onboard an sUAS to perform many different tasks. We briefly summarize them here in order to provide context for discussing CV-related challenges and solutions throughout the remainder of the paper. One of the most common applications is *object detection* to empower sUAS to perceive their surroundings by identifying objects in a live video stream. [5]. The ability to detect specific types of objects allows sUAS to track moving objects, such as people, and to perform surveillance, obstacle avoidance, path planning based on collision-free trajectories, and other tasks that depend upon the detection of one or more specific classes of objects [6], [7]. CV is also used for autonomous navigation [8] such as autonomous takeoff, landing, and navigating even when obstacles are present [9]. For example, Pulido et al. [10] used image segmentation to support object recognition during navigation, while others used different CV-based approaches for safe landings [11], [12], [13], [14], [15]. In a related area, CV can also be used to help an sUAS track a moving object, such as a person. The sUAS continually monitors the person and then actively generates a trajectory to follow the person whilst avoiding crashing into them [16], [17], [18]. Other common applications of sUAS-based CV are surveillance, monitoring, and inspections, where the sUAS uses aerial image processing [19] to detect events, intruders, and anomalies [20], [21] or to perform tasks such as structural monitoring [22].

B. Common CNN Algorithms

As our focus is on the challenges of implementing CV on sUAS-based edge computing environments, we also provide a brief summary of common CV algorithms, many of which are based on artificial neural networks which is a branch of artificial intelligence (AI). Convolutional neural networks (CNN) [23] are frequently used by CV algorithms including the following types:

- R-CNN [24] is a two-stage object detector that locates objects in an image using a selective search with feature extraction at a high computational cost.
- Faster R-CNN [25] improved processing speed and accuracy of R-CNN. It takes the entire image as input instead of using a CNN for different regions of the image.
- Mask R-CNN [26] is an extended version of Faster R-CNN with a branch for predicting object mask while simultaneously adding recognizing bounding boxes.
- YOLO [27] is a one-stage object detector that significantly enhances processing speed. Similar to Faster R-

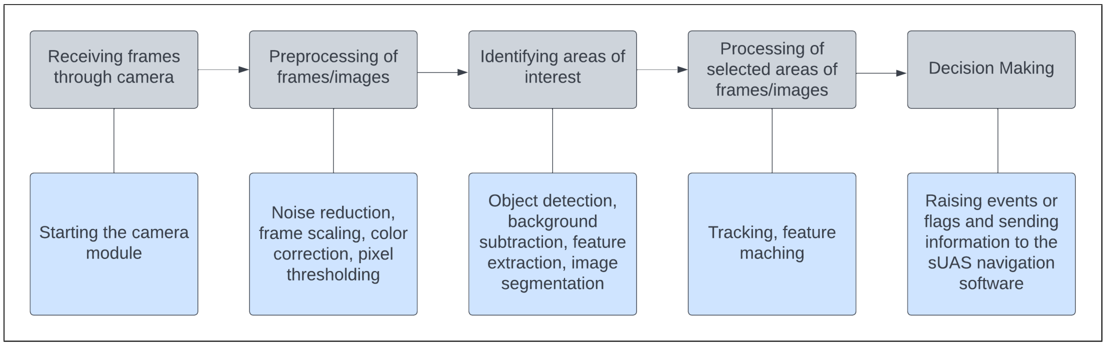

Fig. 1: The general Computer Vision pipeline processes images via a series of steps

CNN, YOLO uses a single feature map to detect objects. However, the image is divided into a grid for performing object searches. There are many versions of YOLO [28], and it has been extensively used in many different applications [29].

Many additional neural networks have been proposed to address issues related to scale variability in aerial images. Yang et al. [30] proposed a three-step pipeline composed of specialized sub-networks. Zhao et al. [31] introduced Mixed YOLOv3-LITE, a lightweight architecture that is suitable for real-time performance. Based on YOLO-LITE, [32] included residual blocks and parallel high-to-low resolution sub-networks for achieving a balance between speed and performance in devices such as non-GPU-based computers.

C. Related Work on CV for sUAS

Researchers have written survey papers on CV algorithms and applications for UAVs and sUAS, including discussions about convolutional neural networks for UAS. Al-Kaff et al. [33], Luo et al. [34], Kanellakis et al. [35], and Liu et al. [36] have all conducted surveys of computer vision applications, including their technical challenges and solutions. Belmonte et al. [37] conducted a survey specifically on CV with UAVs, and Chen et al. [38], Liu et al. [39], and Morton et al. [40] discussed the use of various CNN models and algorithms for CV on sUAS. However, none of these papers considered CV from a software engineering perspective.

III. SLR Methodology and Overview

We performed a preliminary SLR, following the process summarized in Figure 2, in order to address our previously stated research questions.

A. Search Query and Criteria

We initiated the SLR using the following query terms, and executed our search in IEEE, ACM, and Springer digital libraries as well as on Google Scholar:

"UAV" OR "unmanned aerial vehicles" OR "drone"
OR "unmanned aerial system" OR "UAS" OR
"Cyber-physical systems"
AND
"Autonomy" OR "Navigation"
AND
"Computer Vision" OR "Artificial intelligence"

AI-supported computer vision enables the autonomous navigation of sUAS so in our search queries, we included the terms "autonomy" and "navigation" to focus on associated challenges. We used search queries such as "Computer Vision OR Artificial intelligence" to include papers that discussed computer vision or included an AI component, such as deep learning, which was used in computer vision; however, as discussed in exclusion criteria, we ultimately filtered out all papers unrelated to CV. Our initial search returned approximately 1,500 papers. The first author skimmed the titles of these papers and selected 150 papers for which, the titles matched the inclusion and exclusion criteria. The first author then read the abstract of all 150 papers and applied the following filters:

Inclusion Criteria:

- Papers must discuss computer vision for autonomous unmanned aerial systems.
- Papers must describe the AI and/or ML components of the systems.

Exclusion Criteria:

- Papers focused only on recording and/or enhancing aerial images and videos
- Papers focused on designing and/or the technology of a simulator or mechanics of a hardware component to support Computer Vision
- Papers focused on autonomous ground vehicles
- Papers focused on manned aerial systems
- Papers not written in English

This step resulted in 90 papers. The first author then skimmed all of these papers, again applying inclusion and exclusion criteria and selecting the most relevant papers. This produced the final selection of 15 papers. Furthermore, we reviewed each of these papers to identify specific discussions about the CV applications for sUAS and identified any use of ML and DL techniques and algorithms.

We followed an inductive analysis approach whereby we reviewed each paper to identify challenges and solutions and

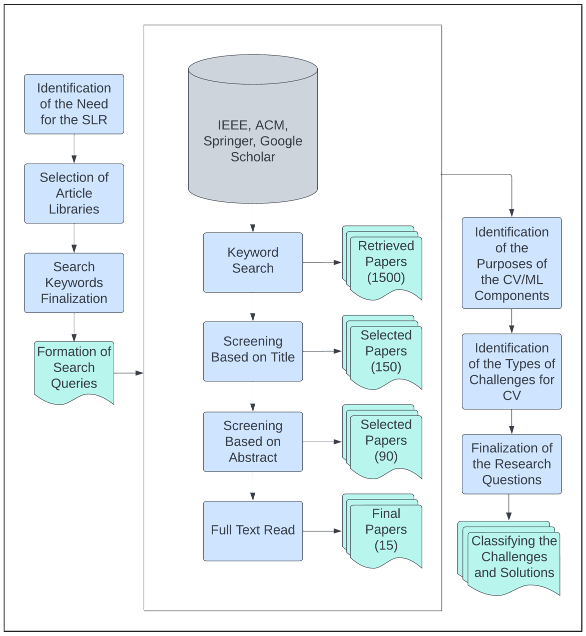

Fig. 2: The Systematic Literature Review process includes paper counts at each step.

tagged each of these with a concept tag. For example, we created 'challenge' tags such as *poor video quality* and *insufficient training data*, and 'solution' tags that included *customized models* and *3D dataset*. We then performed a conceptual card-sorting exercise to group concepts into challenges, and assign solutions to each challenge.

In some cases, where problems were well defined but where the papers did not provide solutions, or where we were aware of additional solutions, we performed a secondary literature search using search terms associated with either the challenge or the known solution to find additional materials. The second author assisted the first author in this process. Papers retrieved from the initial SLR are all shown in Table I and referenced in the text by means of their ID (e.g., P1, P2), whereas all other materials are referenced directly in the text. This entire process resulted in the identification of 14 challenges and 22 solutions. While neither challenges nor solutions are intended to be complete, they provide useful context for informing the software engineering process for CV-imbued sUAS systems.

B. Analysis of SLR Results

Based on our SLR, we identified five major challenges of CV for sUAS. The challenges are:

- TD: Insufficient, Inappropriate Training Data
- QI: Low Quality of Imagery
- EC: Environmental Context
- CM: Computer Vision in Motion
- RC: Resource Constraints of Running AI on a UAV

In the following sections, we describe each of these challenges in more detail, decompose them into sub-challenges, identify solutions, and map each of these back to the papers in which they were discussed. The challenges and sub-challenges

Fig. 3: Three perspectives: A view from the ground, a low-altitude aerial view, and a distant aerial view. The CV model needs to be trained to recognize these diverse perspectives.

are all derived from our SLR; however, in some cases, we have proposed solutions from alternate sources. Our study shows that deploying CV on sUAS introduces challenges that go far beyond those of applying CV to static datasets of videos, or of applying CV on ground-based, stationary, platforms with fewer resource constraints.

IV. Challenge #1 : Insufficient, inappropriate Training Data

Several papers discussed problems related to training data [2], [41], [15], [18], [9], [12], [42]. As machine learning and deep learning models need large amounts of appropriately representative training, testing, and validation data, the problem of insufficient or inappropriate training data is quite common. It results in poorly trained models which return unsatisfactory object detection rates and false positive detection. This is true for any CV algorithm but is exacerbated in the sUAS domain for reasons discussed below.

A. TD1: Aerial Perspectives

sUAS view the world from a different perspective than ground-based cyber-physical systems. As a result, existing CV models, trained with ground-based images tend to underperform when used on aerial images. To further exacerbate the problem, labeled datasets of images in the public domain (e.g. ImageNet, MS COCO, CIFAR-10 and CIFAR-100) are also badly misaligned with the sUAS' aerial perspective and potential distances. Therefore, as illustrated in Figure 3, sUAS-based CV faces two challenges. The sUAS sees objects from above rather than from the side, and the sUAS is often quite far away from the object it is tasked with identifying.

As a result, CV models trained for ground-based object detection do not work well when deployed on aerial platforms. For example, (P2) discussed the challenge of using sUAS-based CV to detect people and reported that the accuracy of human detection decreases when the sUAS is more than 10 meters above ground level (AGL). This is problematic as sUAS tend to fly at much higher altitudes.

Solutions: Two papers (P2, P9) identified the need for new aerial datasets. (P9) demonstrated the importance of having a dataset that included images taken from diverse altitudes, claiming that the view of a person from 20m AGL is very different from one taken from 100m AGL. There is, therefore, a general need for more publicly available, labeled aerial datasets taken from diverse pitches, distances, and altitudes and validated in diverse settings.

TABLE I: Papers selected from the preliminary SLR with mappings to specific CV Challenges

#	Paper Title	Ref	Challenges
P1	Recognition of a landing platform for unmanned aerial vehicles by using computer vision-based techniques	[10]	EC2
P2	UAV Landing Using Computer Vision Techniques for Human Detection	[11]	TD1, QI1, EC2, CM1, RC3
P3	Autonomous navigation, landing and recharge of a quadrotor using artificial vision	[8]	EC2
P4	Computer Vision based guidance in UAVs: Software Engineering challenges	[43]	QI2, EC2, RC1, RC2
P5	Vision-based UAVs Aerial Image Localization: A Survey	[41]	TD4, EC2
P6	Computer Vision for Fire Detection on UAVs—From Software to Hardware	[2]	TD2
P7	Swaying displacement measure. for structural monitoring using comp. vision and an unmanned aerial vehicle	[22]	EC1, CM2
P8	Vision-based UAV Positioning Method Assisted by Relative Attitude Classification	[42]	QI2
P9	Unsupervised Human Detection with an Embedded Vision Sys. on a Fully Auto. UAV for Search & Rescue Oper.	[18]	TD3
P10	A bio-motivated vision system and artificial neural network for autonomous UAV obstacle avoidance	[9]	EC2
P11	Deep-Learning-Based Aerial Image Classification for Emergency Response App. using Unmanned Aerial Vehicles	[19]	RC3
P12	SafeUAV: Learning to estimate depth and safe landing areas for UAVs from synthetic data	[15]	TD5
P13	A survey of safe landing zone detection techniques for autonomous unmanned aerial vehicles (UAVs)	[12]	TD2, TD3
P14	Timely autonomous identification of UAV safe landing zones	[13]	QI2, RC1
P15	Vision-based UAV Safe Landing exploiting Lightweight Deep Neural Networks	[14]	RC1

B. *TD2: Task Specific Models*

As previously discussed, sUAS perform a wide variety of tasks. Many of these tasks, such as bridge inspections, cable inspections, and fire detection are highly dependent on CV and require task-specific models. For example, (P6) focused on the use of sUAS for wild-fire detection and demarcation, which required a very large number of labeled aerial views showing various aspects of wild-fires including major fires, burned-out areas, brush fires, creeping fires, and general images of vegetation and terrain. Training datasets, therefore, need to broadly cover all aspects of the tasks that CV needs to support. This analysis was supported by (P13), which discussed the challenges of classifying safe landing zones for sUAS when the training and testing datasets of landing zones were not aligned.

Solutions: These findings highlight the need for large, diverse, task-specific training and validation data. The datasets can be collected from the real-world and/or as described by (P6), from existing images collected from the web and labeled appropriately. (P13) proposed a different solution based on using CV models trained to detect and avoid individual obstacles rather than a more holistic scene-based training approach that is possible when larger datasets are available.

C. *TD3: Occluded Views:*

Many objects viewed from above will be occluded in various ways. For example, a human may only be partially visible due to occlusion by other objects (e.g., trees, buildings, other people) or by water if they are partially submerged. This means that the CV needs the ability to recognize various parts of the object (e.g., a human arm, half a body) from multiple viewpoints (i.e., frontal, sideways, backward). Two papers (P9, P13) discussed this challenge and pointed out that a generic person detection model may not work well in this case (P9).

Solutions: There are two solutions to this problem. First, as described by (P9, P13), a model can be trained with occluded images. For example, (P9) developed a new dataset from images of swimmers found on the web, while other datasets collected entirely new sets of occluded images from an sUAS [44], and several others proposed augmenting images to create occlusions (P5).

D. *TD4: Model overfitting*

Model overfitting occurs when ML models perform well on training data but underperform on validation and testing data. This is discussed in (P5) with respect to overfitting for the extraction of semantic features from aerial images using deep learning [45]. Overfitting generally occurs when insufficient training data is available; however, this is particularly problematic in sUAS-based CV, where inadequacies of existing aerial datasets and the cost and effort required to create new ones mean that developers often train CV models (at least initially) using less than ideal datasets.

Solutions: Two complementary solutions were proposed. The first, as in the case of TD1, involves collecting and labeling new aerial datasets. The second is an algorithmic solution that is designed to compensate for non-ideal data during the training process. In (P5), the authors proposed two specific techniques of data augmentation and fine-tuning their CNN architecture to avoid model overfitting. Data augmentation was used to create greater diversity and enlarge the existing training data by vertical, horizontal, and diagonal object flipping, scaling, image shifting, rotation, color jittering, etc. For fine-tuning, they started with a large pre-trained model and performed additional training using images with new classes of objects, in order to improve the performance of object detection on the new classes.

E. *TD5: Lack of High Resolution 3D Datasets*

sUAS operate in 3D space; however, much of the available training data is either 2D, low-resolution 3D, and/or unlabelled. This means that datasets for training sUAS to conduct operations within 3D space, such as landing through trees, are inadequate (P12). Furthermore, learning dynamically in the real world is costly and likely to cause accidents.

Solutions: (P12) addressed this challenge by using the Google Earth application and its 3D reconstructions derived from the real-world to build a virtual dataset [46]. They collected a random set of images of the ground that have uniform elevations between 30 and 90 meters with a tilt angle of 45 degrees. They proposed SafeUAV-Net which is a deep CNN designed for depth estimation using RGB input and used it to train their CV model. They used image segmentation and made a prediction for each pixel of the input image for different categories mentioned in the paper such as horizontal, vertical, and other. For training models, they used Pytorch deep-learning framework [47]. The authors explored two variants of the model running at 35 FPS and 130 FPS respectively and evaluated both on embedded platforms. They both showed good performance.

V. Challenge #2 : Low Quality and Noisy Images

One of the fundamental assumptions of CV algorithms is that the quality of video streams and/or images is sufficient to support an accurate CV system. As a result, problems in the video stream quality can lead to degraded performance. In this challenge, we explore two specific problems related to image quality.

A. QI1: Low Quality Imagery

Inferior sensors lead to low-quality imagery. (P2) reported that sensors that work satisfactorily in ideal lighting conditions are often unable to perform well when operated in outdoor environments where significant variations in lighting conditions can affect image quality. Problems include blurring, over-exposure, and under-exposure. However, due to trade-offs with size, cost, and weight, sUAS often have low-quality sensors and, therefore, tend to underperform (P1).

Solutions: Selecting appropriate sensors is critical for improving the quality of input for real-time applications. (P2) discussed the importance of a good camera/sensor in object detection quality and proposed several types of cameras and observed that infrared cameras work well in all lighting conditions. They experimented with three different cameras (i.e., iPhone 6S Plus, DJI Phantom 4, Raspberry Pi NoIR Camera V2) and observed that each of them had a sweet spot with respect to distance. This of course can be computed based on the camera's specifications. Clear trade-offs exist between cost, weight, size, power consumption, and resolution. For embedded platforms, CSI cameras are preferred over USB cameras as they transfer data faster.

B. QI2: Obscured Images:

Noise in images caused by environmental conditions, such as fog, rain, bright sunlight, atmospheric disturbance, and low-altitude wind sheer (P4, P14), as well as electrical interference from the equipment on the sUAS (P8), and vibration from sUAS motion can all negatively impact image quality, resulting in poor CV outcomes. Figure 4 depicts the electrical interference and sun glare problem in images captured by the camera in our own sUAS.

Solutions: Issues such as electrical interference and vibration can be partially resolved through careful placement of components and wiring on the sUAS and the use of vibration dampers. In addition, the CV pipeline can be augmented with additional pre-processing steps aimed at removing specific types of noise such as fog or glare (P4). This can be performed in real-time using libraries such as OpenCV [48] to improve the performance of vision algorithms.

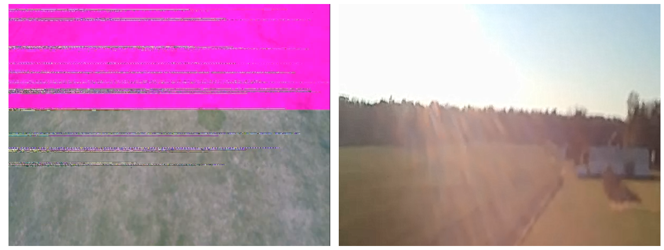

Fig. 4: Electrical interference and sun glare

VI. Challenge #3: Environmental Context

CV solutions are tasked with detecting and identifying objects within the context of real-world scenes. They accomplish this by extracting global and local features. Global features describe the overall image and include attributes such as contour representations, textures, and shape descriptors, whereas local features represent key-points within an image such as an edge or point.

A. EC1: Image Feature Sensitivity

(P7) explored issues related to both global and local features. Imagery collected from an sUAS often contains many different overlapping objects and rich background contexts, which makes feature analysis and extraction quite challenging and can ultimately lead to reduced CV accuracy in real-world sUAS deployments. The problem impacts both local and global features; however, local features tend to have good *viewpoint invariance*, meaning that objects can be recognized regardless of their viewing angle, while global features have limitations in densely populated areas with large texture repeatability, and are also sensitive to viewpoint changes.

Solutions: Many researchers have proposed algorithmic solutions for solving this general problem, which exists in many domains and contexts. However, we focus on one example. (P5) explored the issue for sUAS live video stream and proposed dividing images into groups of pixels (i.e. 10x10 pixels, 7x7 pixels, etc.) called patches, and then extracting features from each of the patches instead of from the whole image. Patch size needs to be small enough to ensure that the viewpoint is able to highlight local features, whilst large enough to also detect global features. They made other suggestions too concerning the treatment of colors and shapes in order to improve object detection accuracy [49].

B. EC2: Weather and Daylight Conditions

sUAS need to fly in diverse weather conditions, however, CV algorithms perform differently under different conditions, creating problems of system reliability. For example, (P4) discussed the weather-related impact of wind speed, cloud and haze, and lighting, which in turn translates to different levels of CV performance across different weather conditions and seasons. (P3) reported varying performance for the same CV systems when deployed indoors vs. outdoors.

In general, CV performs better in the summer under sunny conditions than in dark winter days with low sunlight. In low lighting conditions, there is a tendency for higher false positive rates due to the lack of details in an image. (P2) analyzed the performance of a vision system using the SSD-MN-V2 model for different phases of the day and showed that the performance of the system was lower in the morning compared to the afternoon lighting conditions.

On the other hand, if a camera faces the sun on a bright sunny day, the resulting glare can cause complete failure of a vision system as most CV algorithms and models cannot extract useful information from extremely glared regions. The issues related to variations in lighting were discussed in several papers (P1, P2, P5, P10). From the Software Engineering perspective, smart solutions are needed to reposition the sUAS to avoid glare and other similar noise.

Solutions: Proposed solutions were quite diverse. (P4) proposed augmenting the sUAS with fog lights to improve lighting; however, this approach clearly has distance and weather-related limitations. (P14) recommended an algorithmic solution based on using a feature-based algorithm such as Scale-invariant feature transform (SIFT) [50] to counteract the variations in lighting in various environments. Finally, (P3) proposed adaptive CV algorithms, able to self-adapt according to the current lighting conditions.

VII. Challenge #4: CV in Motion

sUAS-based computer vision applications face additional challenges caused by the motion of the sUAS. Problems include vibration and sudden jerky movements of the sUAS caused by wind and/or turns. This can impact individual images but is particularly problematic when CV is used over a sequence of images, for example, to track a moving object such as a person or a vehicle, or to circle an object during a surveillance activity. It also impacts CV-related challenges such as accurate geolocation of an object, which requires alignment of image frames with sUAS position and attitude (yaw, pitch, and roll) at the time that the image was taken. We discuss each of these challenges in turn.

A. CM1: Image Blurring Caused by Vibration and Jerk

At the most basic level, vehicular movement caused by vibration, wind, or abrupt vehicular motions can cause image degradation such as blurred images (P2), [51].

Solutions: (P4) recommended applying pre-processing algorithms to enhance the quality of images before the primary algorithms process them. They also recommended the use of feature-based vision algorithms similar to the algorithms mentioned in the solution of CM1. Features in an image can be defined as important properties in an image such as edges, corners, texture, etc. When images are blurred,preprocessing methods such as sharpening images are useful techniques.

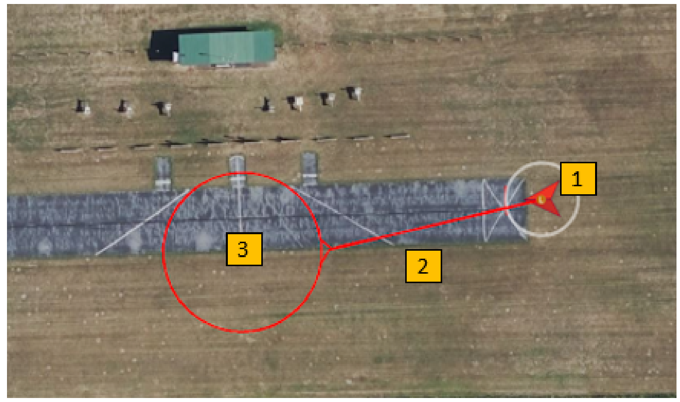

Fig. 5: During a live test with physical sUAS using Drone Response, the sUAS took off at [1], detected the person at [2], but miscalculated the person's position, and, therefore, instead of circling [2], it circled an empty space [3]. The problem was its initial failure to match the figure to the correct timeframe of the flight log data.

B. CM2: CV-Based Geolocation of Objects from an sUAS

Several of our papers discussed aerial surveys (P4) and object detection during landing (P2, P3). In these cases, we are interested in either computing accurate coordinates of the objects or accurately determining the relative direction of the objects from an sUAS. The challenge is that the sUAS needs to utilize its CV to geolocate the object whilst it is itself in motion. It accomplishes this by first detecting the targeted object in the image, secondly extracting the position of the object with respect to its pixel coordinates in the image, and then geolocating it either relative to the sUAS or in absolute coordinates by considering the attitude (yaw, pitch) of the gimbal carrying the camera, and the absolute attitude of the sUAS at the time the frame was taken. Given an image frame, the challenge is to account for the movement of the sUAS in determining the true position of the targeted object (P7). Computing the position of an object based on the current position of the sUAS rather than its position at the time the frame was taken (if only milliseconds different) can lead to incorrect geolocation of the object. Figure 5 shows a live test with sUAS.

Solutions: (P7) discussed the use of sUAS with CV to accurately geolocate buildings in order to measure their degree of sway. They adopted a technique to translate an image between the camera reference coordinate system (i.e. three-dimensional XYZ coordinate system) and the sUAS body reference coordinate system to which the camera is attached and then used key-points, referring to specific shapes and illuminations, in the scene and compared consecutive image frames. Algorithms included the Scale-invariant Feature Transform (SIFT) [52], Binary Robust Invariant Scalable Key-points (BRISK) [53], Speed-up Robust Feature (SURF) [54]. An alternate, geometric approach, (adopted in our system) involves matching the sUAS flight data (i.e., sUAS and gimbal attitude) with the exact time

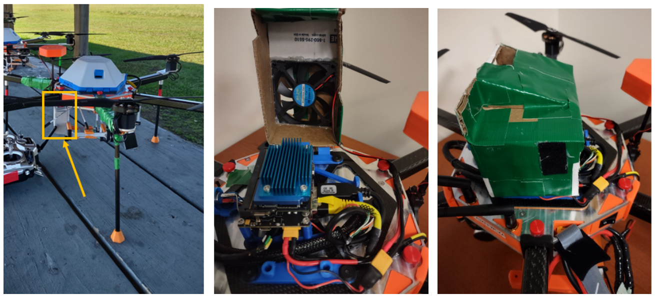

Fig. 6: Hardware was augmented to address problems related to [RC1, RC2]. A stepdown transformer shown in the leftmost image was added to provide sufficient power to the Jetson to support the CV algorithms [RC2], and a new temporary cover that included a fan in the middle image was constructed and used to replace the poor airflow in the original cap [RC1]. The rightmost image shows the ventilation system for the Jetson.

frame in which an image is taken, thereby performing more accurate geolocation computations based on the actual position of the sUAS.

VIII. Challenge #5: Resource Constraints and Conflicts

Deploying CV on an sUAS can be challenging due to processing intensive CV algorithms matched with limited computational resources. We summarize these problems under two key issues.

A. RC1: Power Limitations and Overheating

Keeping the sUAS in the air requires significant power, typically from LIPO batteries, which provide a flight time of anywhere from about 15 to 40 minutes on an average sUAS. While additional batteries can be added, the increased weight makes the sUAS heavier and causes it to drain power even faster (P14). At the same time, CV algorithms can rapidly draw down the available power and cause overheating (P4). This is especially problematic while running computationally intensive algorithms such as deep neural networks (DNNs) or CNNs-based algorithms, such as YOLO, which require significant resources from the CPU (P15), RAM, and GPU and therefore, draw excessive power and/or easily overheat the embedded platforms, causing unexpected slowdowns or even shutdowns. Some solutions to these problems from the Drone Response system are depicted in Figure 6.

Solutions: The problem can be tackled from both a hardware and software perspective. First, hardware can be modified if it is needed to ensure that sufficient power is available (e.g., adding a stepdown transformer) and that the processor is properly ventilated. (P4) suggested that the camera should be placed into sleep mode or power-save mode when not in use. (P15) and (P10) proposed the use of lightweight deep neural networks to reduce processing requirements, and (P15) implemented a lightweight version of the MobileNetV2+PSPNet CNN, while (P10) proposed down-scaling input images to increase processing speed and reduce computational needs.

B. RC2: Resource Conflicts

In systems with a single onboard processor, multiple system functions will compete for cycle time. (P4) discussed CV system failures on sUAS due to memory (RAM) overflow as a single embedded platform often needs to support multiple systems including CV, navigation, and higher-level mission planning. Ideally, the image processing time should match the capture rate of the camera, but due to limited resources, CV systems running on an sUAS often suffer from low frames per second (FPS) (P4).

Solutions: Software Engineers need to understand these constraints and design a solution that allows multiple systems to run synchronously on the available space and works effectively at a low frame rate. Ideally, the frame rate can be throttled up and down depending on the current task. When this is unacceptable, the system hardware and architecture need to support a distributed solution in which systems are isolated across two or more embedded platforms. (P11) recommended powerful and energy-efficient embedded systems for the deployment of computer vision systems in UAVs. The authors used Odroid XU4 embedded system for their work.

C. RC3: Weight and Space Constraints

As discussed for RC1, adding additional payload comes at the price of reduced flight time. (P11) discussed the limitation of sUAS size and weight upon its payload. Each sUAS has a maximum payload capacity, above which it cannot reliably maintain flight. This is a hard constraint for a given sUAS. Furthermore, embedded systems are often restricted in terms of space available for installing software and its associated libraries (P2). This is especially the case when platforms are run on carrier boards, and even though boards typically have extension slots, these may be better suited for storing videos than for running core programs.

Solutions: (P2) recommended converting CV models to a common format to avoid using new libraries. In our own experience, we have loaded autopilot and CV libraries into very constrained spaces through careful installation routines and added and removed libraries and features in very specific orders so as to remain within the space constraints.

IX. Implications for Software Engineering

In this section, we summarize some of the key findings from the SLR and consider their implications for the Software Engineering and Safety Engineering process. Notably, as depicted in Table II, many of the proposed solutions are targeted at the CV pipeline and/or the system hardware with very little discussion about the Software Engineering aspects of the system. In prior work, several authors have discussed general safety issues related to sUAS (e.g., [55], [56], [57]), but few have addressed safety concerns directly related to the use of CV onboard sUAS. Some exceptions include work by Abraham et al. [58], which explored the role of humans-in-the-loop when CV confidence was low, and Lutz et al., who used

TABLE II: Proposed solutions mapped back to the Sub-challenges that they are designed to address

#	Solutions	Type			Identified Challenges															
					Data					Image		Environ.		Motion		Resource		
		CV	HW	SE	TD1	TD2	TD3	TD4	TD5	QI1	QI2	EC1	EC2	CM1	CM2	RC1	RC2	RC3
1	Create image datasets from internet images	●			○	○	○	○	○									
2	Collect image datasets using physical sUAS	●			○	○	○	○	○									
3	Create 3D datasets from 3D maps & simulations	●			○	○	○	○	○									
4	Dataset diversity wrt altitudes, pitches, & distance, etc.	●			○	○	○	○	○									
5	Include occluded images in datasets	●			○	○	○	○	○									
6	Augment existing datasets by image augmentation	●			○	○	○	○	○									
7	Fine-tuning existing models with additional classes	●			○	○	○	○	○									
8	Use a good camera with fast communication protocol		●							○								
9	Pre-process images to remove noise	●									○			○				
10	Use self-adaptive CV algorithms	●											○					
11	Use robust key-point based algorithms	●													○			
12	Balance between local and global feature	●										○						
13	Avoid electrical interference through configuration		●								○							
14	Use fog lights in low visibility conditions		●								○							
15	Put camera in sleep mode when not in use to save power		●	●												○		
16	Use lightweight version of CNN for embedded platforms	●	●													○		
17	Select CV models compatible with current libraries	●	●															○
18	Downscale images to reduce processing	●															○	
19	Match the sUAS flight data with timestamped images			●											○			
20	Improve the cooling and heating systems		●													○		
21	Use powerful and energy-efficient embedded systems		●														○	
22	Discard old sensor messages			●										○				

obstacle analysis to identify specific sUAS safety concerns, including CV-related ones [59].

However, CV-based sUAS systems are complex to develop and their safety-critical nature warrants a rigorous Software Engineering process. Based on findings from the SLR we, therefore, make the following observations:

- **System requirements** need to be clearly specified, especially with respect to the expected operating environment (e.g., wind, precipitation), and the purpose of the application (e.g., search-and-detect). Specific end goals of the system under development may lead to different hardware, software, and CV design decisions. Architecturally significant requirements that impact both CV outcomes and more general mission outcomes need to be identified, prioritized, and evaluated with respect to their specific tradeoffs. Examples highlighted in the SLR included tradeoffs between hardware resource constraints (RC1-RC3), sophistication and accuracy of CV algorithms (QI1, QI2, EC1, EC2, CM1, CM2)), and the selection and/or development of datasets and CV models (TD1-TD5).
- Delivering CV as a **deep-learning solution** on an edge-device requires systematic Software Engineering effort across *three distinct workflows* as depicted in Figure 7. Results from the SLR indicated five categories of challenges. Two of them (TD, QI) relate directly to the CV pipeline, whilst the other three (CM, EC, RC) cross-cut CV, hardware, and software engineering issues. The workflows include (1) composing the CV pipeline, including acquiring datasets, training models, and selecting core algorithms and preprocessing steps (green), (2) building and configuring the physical hardware (gray), and (3) designing, developing, and validating the software infrastructure, such as control and integration software (blue). Understanding the individual challenges of each workflow enables the engineering team including data scientists, hardware engineers, and software engineers to identify and tackle development risks in parallel throughout the project. Components for each of the three workflows need to be tested independently before being integrated into the system. Test examples include:
  - *CV Workflow:* The trained CV model must be validated for accuracy against diverse images, and the overall CV pipeline must be evaluated for accuracy against datasets that match the targeted end-use [TD1-TD5].
  - *Hardware Workflow:* Physical components such as the gimbal controls and movements must be tested for accuracy and responsiveness, for example, to ensure that an angular command moves the gimbal to the correct position [CM2, RC1-RC5].
  - *Software Engineering:* All CV-related software features must be validated. Examples include the ability to activate and deactivate the camera to save power [RC1] or to interpolate the flight data to correctly align an image frame with the position and attitude of the sUAS at the time the frame was captured [CM2]. Navigation software should be designed robustly for handling real-time requirements. For instance, if sensor messages enter the message queue too quickly and the software does not process them fast enough, messages may accumulate. This may result in delayed processing of sensor messages and jerky motion of the sUAS. There are many ways to solve this problem depending on specific message type. Solutions include reducing the polling rate of sensor data, use of priority queues, retaining one, and only one, message for certain types of messages, or discarding old messages when a maximum threshold is reached.

- **Integration Tests** must start early and be executed incrementally. Surprises are inevitable when working with embedded CV in real-world settings. Therefore, it is essential to integrate and validate system-wide functionality as early as possible. In general, it is a good practice to integrate across the workflows in simulation first, and then to progress rapidly and incrementally to deployment tests on physical sUAS. Examples of integration tests include:
  - *Integrating CV with sUAS Software Control System:* Perform functional tests in simulation to validate that the sUAS can effectively use its CV features to detect and track a person in realistic conditions [TD1-TD5, QI1, QI2, EC1, EC2].
  - *Whole System Integration:* Conduct real-world tests in diverse weather and lighting conditions [CM1, CM2]. Validate the accuracy of CV in the real-world, responsiveness and latency of the CV pipeline, and overall synchronization between the CV pipeline, the hardware components (eg., gimbal) [RC1-RC3], and the general software system.
- **Safety Analysis** is essential before deploying any sUAS system on real-world missions. The analysis needs to consider risks at the integration of CV and other aspects of the system and to identify specific hazards, propose and assess potential mitigations, and implement them into the system. Approaches such as Failure Mode, Effects & Criticality Analysis (FMECA), Fault Tree Analysis, or Safety-Cases [60] can be employed. While an in-depth discussion of safety analysis was outside the scope of our SLR; the individual challenges and solutions reported in this paper provide initial foundations for exploring CV-related integration risks. Examples include the impact of false positives or false negatives in the core CV detection algorithms [TD1-TD5], resource contentions between core control software and CV which could impact the safe flight of the sUAS [RC1-RC3], and problems in synchronizing CV data and the sUAS flight data resulting in incorrect geolocation computations leading to unsafe flight paths [EC1, EC2].

X. CV Deployment in Drone Response

We deployed CV in our own Drone Response system [61], [62], [63] in both a simulated and physical-world environment.

A. Background of Drone Response System

The Drone Response system can be deployed and tested in both simulation and physical world environments without any changes to the code. It is built over the PX4 open source autopilot system, and uses YOLO V3 and YOLO V4 for CV-based object detection of persons. For simulation, we used the Gazebo simulator with a high-fidelity physics engine. The typhoon480 drone in Gazebo comes with a simulated camera and gimbal which communicates with our CV pipeline over the UDP protocol. For the physical world, we deployed Drone Response on a Hexacopter equipped with an mRo Control Zero F7 flight controller, and a Nvidia Jetson NX carrier module on which we deployed the Drone Response

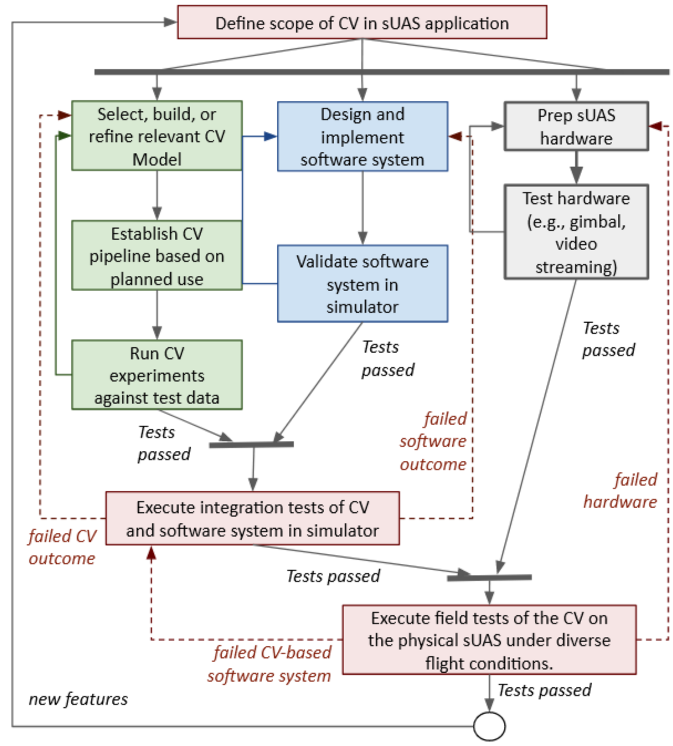

Fig. 7: Engineering a CV-enhanced sUAS requires concurrent development of the CV pipeline, the software system that controls the sUAS and integrates the CV, and the hardware components. Each part must be developed and tested individually and then systematically integrated, with system tests progressing from simulation to physical field tests.

autonomous pilot and the CV pipeline. CV was supported by an IMX477 CSI camera controlled using a 3-axes rotation gimbal. Finally communication between the sUAS and ground station was supported by mesh radio.

B. Experiments using Drone Response System

We faced numerous challenges, as discussed in this paper, when deploying the CV pipeline on a physical drone. First of all, the Jetson Xavier NX' carrier board has an eMMC module with under 16 GB of storage; therefore, in order to install all of the essential CV libraries on the eMMC module, we had to manually remove packages such as LibreOffice from the Jetpack operating system. Second, while running the CV pipeline using the GPU of the Jetson, the Jetson drew too much current and kept shutting down. We used a stepdown transformer to reduce voltage and increase the current. However, when running the CV pipeline, it overheated causing the GPU the throttle down the Jetson making the hexacopter unstable. We designed a cover with a fan to address this issue. Based on our observation, this issue persists irrespective of the ambient temperature around the Jetson, meaning that proper ventilation is needed even in the winter.

In our experiments, different computer vision algorithms delivered different levels of performance for aerial images of people, while running on the NX carrier model. YOLO V3 processed 8 frames per second with confidence scores close to 40%, while YOLO V4, using the model pre-trained on the MS COCO dataset, processed only 4 frames per second but achieved a confidence score of over 90%.

Using our Drone Response System, we flew the hexacopter in a circle around a detected person using the person's calculated GPS location. In one of the experiments, the drone circled in the wrong place, and we later discovered that possibly due to low lighting conditions just prior to sunset, a wooden pillar was incorrectly detected and labeled as a person. This is illustrated in Figure 8, and highlights the importance of having high-quality sensors, a well-trained model for aerial object detection, and the need for adequate lighting for accurate object recognition.

In bright sunlight, the same algorithm and model (YOLO V4 with a model pre-trained on the MS COCO dataset) showed good performance. Figure 9 shows a person detection with 93.88% confidence score on a bright sunny day.

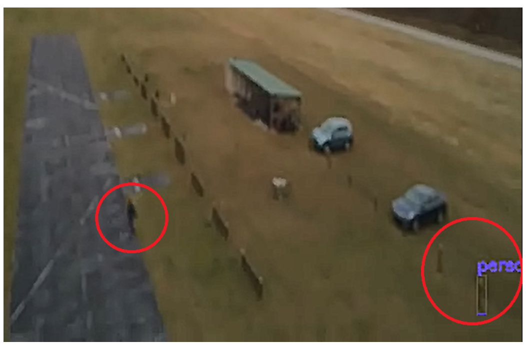

Fig. 8: False positive detection of a person in low light

Fig. 9: Person detection with aerial view

XI. Threats to Validity

We highlight two particularly important threats to validity. The primary threat is in the scope of the SLR and the selection of search terms. Omissions of key search terms may well have led to missing important papers with additional challenges and more diverse solutions. For this reason, we refer to our work as a preliminary study that was useful for identifying the primary types of challenges. Further work is needed to identify a more complete set of known solutions. Secondly, the SLR returned more information about the CV pipeline and resource challenges without providing deep insights into the associated Software Engineering challenges. Where insights were provided, we have included them in our findings; however, the implications for Software Engineering were based on an introspective analysis of the development of Drone Response system including an informal mapping to the findings from the preliminary SLR. We have not yet run an exhaustive set of experiments with CV on Drone Response and, therefore, our findings are not intended to be exhaustive.

XII. Conclusion and Future Work

To address our research questions, our study identified five distinct areas of challenges related to deploying CV on sUAS. These included data collection and model training, quality of collected imagery, environmental contexts, the impact of sUAS motion, and edge-based resource constraints. From these, we identified a total of 14 sub-challenges and 22 associated solutions, which we summarized in Table II.

However, one surprising outcome of our SLR was the lack of emphasis on Software Engineering. The papers discussed many CV-related features, but we found limited information about actual Software Engineering of the products. As discussed in the introduction, our own lack of knowledge at the intersection of CV and Software Engineering resulted in several missteps that impacted the development process. As a preliminary exploration of the role Software Engineering plays in the process of developing a CV-imbued sUAS system, we, therefore, discussed Software Engineering practices for addressing several of the identified challenges. These proposed practices were designed to support a more holistic approach for engineering the CV-imbued sUAS system in a way that considered all three aspects of CV, hardware, and Software Engineering, and which lay the foundation for future work.

In future work, we will extend the SLR to consider specific challenges and solutions in greater depth through extended literature reviews and through evaluating proposed solutions in a more structured experimental environment. Finally, we intend to conduct a focused investigation on safety-related aspects of CV deployment in sUAS. We will conduct more experiments in the future to further validate our findings.

Acknowledgments

The work described in this paper was partially funded by the US National Science Foundation (NSF) under grant # CNS:1931962. It was also partially supported by the Intelligence Advanced Research Projects Activity (IARPA), via *2022-21102100003*. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied of IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

[1] C. Kyrkou and T. Theocharides, “Deep-learning-based aerial image classification for emergency response applications using unmanned aerial vehicles,” in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019, pp. 517–525.

[2] S. S. Moumgiakmas, G. G. Samatas, and G. A. Papakostas, “Computer vision for fire detection on uavs—from software to hardware,” *Future Internet*, vol. 13, no. 8, p. 200, 2021.

[3] A. Li, M. Hansen, and B. Zou, “Traffic management and resource allocation for uav-based parcel delivery in low-altitude urban space,” *Transportation Research Part C: Emerging Technologies*, vol. 143, p. 103808, 2022.

[4] S. Yeong, L. King, and S. Dol, “A review on marine search and rescue operations using unmanned aerial vehicles,” *International Journal of Marine and Environmental Sciences*, vol. 9, no. 2, pp. 396–399, 2015.

[5] K. P. Valavanis, “Advances in unmanned aerial vehicles: state of the art and the road to autonomy,” 2008.

[6] D. Cazzato, C. Cimarelli, J. L. Sanchez-Lopez, H. Voos, and M. Leo, “A survey of computer vision methods for 2d object detection from unmanned aerial vehicles,” *Journal of Imaging*, vol. 6, no. 8, p. 78, 2020.

[7] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to autonomous mobile robots*. MIT press, 2011.

[8] F. Cocchioni, A. Mancini, and S. Longhi, “Autonomous navigation, landing and recharge of a quadrotor using artificial vision,” in *2014 international conference on unmanned aircraft systems (ICUAS)*. IEEE, 2014, pp. 418–429.

[9] M. Pethő, Á. Nagy, and T. Zsedrovits, “A bio-motivated vision system and artificial neural network for autonomous uav obstacle avoidance,” in *2020 3rd International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*. IEEE, 2020, pp. 632–637.

[10] J. A. Garcia-Pulido, G. Pajares, S. Dormido, and J. M. de la Cruz, “Recognition of a landing platform for unmanned aerial vehicles by using computer vision-based techniques,” *Expert Systems with Applications*, vol. 76, pp. 152–165, 2017.

[11] D. Safadinho, J. Ramos, R. Ribeiro, V. Filipe, J. Barroso, and A. Pereira, “Uav landing using computer vision techniques for human detection,” *Sensors*, vol. 20, no. 3, p. 613, 2020.

[12] M. S. Alam and J. Oluoch, “A survey of safe landing zone detection techniques for autonomous unmanned aerial vehicles (uavs),” *Expert Systems with Applications*, vol. 179, p. 115091, 2021.

[13] T. Patterson, S. McClean, P. Morrow, G. Parr, and C. Luo, “Timely autonomous identification of uav safe landing zones,” *Image and Vision Computing*, vol. 32, no. 9, pp. 568–578, 2014.

[14] C. Symeonidis, E. Kakaletsis, I. Mademlis, N. Nikolaidis, A. Tefas, and I. Pitas, “Vision-based uav safe landing exploiting lightweight deep neural networks,” in *2021 The 4th International Conference on Image and Graphics Processing*, 2021, pp. 13–19.

[15] A. Marcu, D. Costea, V. Licaret, M. Pîrvu, E. Slusanschi, and M. Leordeanu, “Safeuav: Learning to estimate depth and safe landing areas for uavs from synthetic data,” in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018, pp. 0–0.

[16] J. Pestana, J. L. Sanchez-Lopez, S. Saripalli, and P. Campoy, “Computer vision based general object following for gps-denied multirotor unmanned vehicles,” in *2014 American Control Conference*. IEEE, 2014, pp. 1886–1891.

[17] J. Pestana, J. L. Sanchez-Lopez, P. Campoy, and S. Saripalli, “Vision based gps-denied object tracking and following for unmanned aerial vehicles,” in *2013 IEEE international symposium on safety, security, and rescue robotics (SSRR)*. IEEE, 2013, pp. 1–6.

[18] E. Lygouras, N. Santavas, A. Taitzoglou, K. Tarchanidis, A. Mitropoulos, and A. Gasteratos, “Unsupervised human detection with an embedded vision system on a fully autonomous uav for search and rescue operations,” *Sensors*, vol. 19, no. 16, p. 3542, 2019.

[19] C. Kyrkou and T. Theocharides, “Deep-learning-based aerial image classification for emergency response applications using unmanned aerial vehicles.” in *CVPR Workshops*, 2019, pp. 517–525.

[20] E. Semsch, M. Jakob, D. Pavlicek, and M. Pechoucek, “Autonomous uav surveillance in complex urban environments,” in *2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, vol. 2. IEEE, 2009, pp. 82–85.

[21] J. Nikolic, M. Burri, J. Rehder, S. Leutenegger, C. Huerzeler, and R. Siegwart, “A uav system for inspection of industrial facilities,” in *2013 IEEE Aerospace Conference*. IEEE, 2013, pp. 1–8.

[22] T. Khuc, T. A. Nguyen, H. Dao, and F. N. Catbas, “Swaying displacement measurement for structural monitoring using computer vision and an unmanned aerial vehicle,” *Measurement*, vol. 159, p. 107769, 2020.

[23] K. O’Shea and R. Nash, “An introduction to convolutional neural networks,” *arXiv preprint arXiv:1511.08458*, 2015.

[24] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[25] R. Girshick, “Fast r-cnn,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[26] K. He, G. Gkioxari, P. Dollar, and R. Girshick, “Mask r-cnn,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[27] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[28] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” *arXiv preprint arXiv:1804.02767*, 2018.

[29] M. Ju, H. Luo, Z. Wang, B. Hui, and Z. Chang, “The application of improved yolo v3 in multi-scale target detection,” *Applied Sciences*, vol. 9, no. 18, p. 3775, 2019.

[30] F. Yang, H. Fan, P. Chu, E. Blasch, and H. Ling, “Clustered object detection in aerial images,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 8311–8320.

[31] H. Zhao, Y. Zhou, L. Zhang, Y. Peng, X. Hu, H. Peng, and X. Cai, “Mixed yolov3-lite: a lightweight real-time object detection method,” *Sensors*, vol. 20, no. 7, p. 1861, 2020.

[32] R. Huang, J. Pedoeem, and C. Chen, “Yolo-lite: a real-time object detection algorithm optimized for non-gpu computers,” in *2018 IEEE International Conference on Big Data (Big Data)*. IEEE, 2018, pp. 2503–2510.

[33] A. Al-Kaff, D. Martin, F. Garcia, A. de la Escalera, and J. M. Armingol, “Survey of computer vision algorithms and applications for unmanned aerial vehicles,” *Expert Systems with Applications*, vol. 92, pp. 447–463, 2018.

[34] B. Luo, X. Wang, and Z. Zhang, “Application of computer vision technology in uav,” in *Journal of Physics: Conference Series*, vol. 1881, no. 4. IOP Publishing, 2021, p. 042052.

[35] C. Kanellakis and G. Nikolakopoulos, “Survey on computer vision for uavs: Current developments and trends,” *Journal of Intelligent & Robotic Systems*, vol. 87, pp. 141–168, 2017.

[36] Y.-c. Liu and Q.-h. Dai, “A survey of computer vision applied in aerial robotic vehicles,” in *2010 International Conference on Optics, Photonics and Energy Engineering (OPEE)*, vol. 1. IEEE, 2010, pp. 277–280.

[37] L. M. Belmonte, R. Morales, and A. Fernández-Caballero, “Computer vision in autonomous unmanned aerial vehicles—a systematic mapping study,” *Applied Sciences*, vol. 9, no. 15, p. 3196, 2019.

[38] C. Chen, J. Zhong, and Y. Tan, “Multiple-oriented and small object detection with convolutional neural networks for aerial image,” *Remote Sensing*, vol. 11, no. 18, p. 2176, 2019.

[39] T. Liu and A. Abd-Elrahman, “Deep convolutional neural network training enrichment using multi-view object-based analysis of unmanned aerial systems imagery for wetlands classification,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 139, pp. 154–170, 2018.

[40] T. Liu, A. Abd-Elrahman, J. Morton, and V. L. Wilhelm, “Comparing fully convolutional networks, random forest, support vector machine, and patch-based deep convolutional neural networks for object-based wetland mapping using images from small unmanned aircraft system,” *GIScience & remote sensing*, vol. 55, no. 2, pp. 243–264, 2018.

[41] Y. Xu, L. Pan, C. Du, J. Li, N. Jing, and J. Wu, “Vision-based uavs aerial image localization: A survey,” in *Proceedings of the 2nd ACM SIGSPATIAL International Workshop on AI for Geographic Knowledge Discovery*, 2018, pp. 9–18.

[42] S. Zhang, J. Li, C. Yang, Y. Yang, and X. Hu, “Vision-based uav positioning method assisted by relative attitude classification,” in *Proceedings of the 2020 5th International Conference on Mathematics and Artificial Intelligence*, 2020, pp. 154–160.

[43] S. Bhat, V. Malagi, K. Rangarajan, and R. Babu, “Computer vision based guidance in uavs: software engineering challenges,” *ACM SIGSOFT Software Engineering Notes*, vol. 35, no. 6, pp. 1–6, 2010.

[44] A. M. R. Bernal and J. Cleland-Huang, "Hierarchically organized computer vision in support of multi-faceted search for missing persons," in *2023 IEEE 17th International Conference on Automatic Face and Gesture Recognition (FG)*. IEEE, 2023, pp. 1–7.

[45] S. Workman, R. Souvenir, and N. Jacobs, "Wide-area image geolocalization with aerial reference imagery," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3961–3969.

[46] L. Kumar and O. Mutanga, "Google earth engine applications since inception: Usage, trends, and potential," *Remote Sensing*, vol. 10, no. 10, p. 1509, 2018.

[47] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[48] I. Culjak, D. Abram, T. Pribanic, H. Dzapo, and M. Cifrek, "A brief introduction to opencv," in *2012 proceedings of the 35th international convention MIPRO*. IEEE, 2012, pp. 1725–1730.

[49] R. WANG and Z. ZHU, "Sift matching with color invariant characteristics and global context," *Opt. Precision Eng*, vol. 23, no. 1, pp. 295–301, 2015.

[50] J. Markel, "The sift algorithm for fundamental frequency estimation," *IEEE Transactions on Audio and Electroacoustics*, vol. 20, no. 5, pp. 367–377, 1972.

[51] X. Wang, B. Luo, and Z. Zhang, "Application of uav target tracking based on computer vision," in *Journal of Physics: Conference Series*, vol. 1881, no. 4. IOP Publishing, 2021, p. 042053.

[52] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[53] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *2011 International conference on computer vision*. Ieee, 2011, pp. 2548–2555.

[54] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Computer vision and image understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[55] M. Vierhauser, M. N. A. Islam, A. Agrawal, J. Cleland-Huang, and J. Mason, "Hazard analysis for human-on-the-loop interactions in suas systems," in *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2021, pp. 8–19.

[56] R. A. Clothier and R. A. Walker, "The safety risk management of unmanned aircraft systems," *Handbook of unmanned aerial vehicles*, pp. 2229–2275, 2015.

[57] A. Chhokra, N. Mahadevan, A. Dubey, and G. Karsai, "Qualitative fault modeling in safety critical cyber physical systems," in *Proceedings of the 12th System Analysis and Modelling Conference*, 2020, pp. 128–137.

[58] S. Abraham, Z. Carmichael, S. Banerjee, R. VidalMata, A. Agrawal, M. N. Al Islam, W. Scheirer, and J. Cleland-Huang, "Adaptive autonomy in human-on-the-loop vision-based robotics systems," in *2021 IEEE/ACM 1st Workshop on AI Engineering-Software Engineering for AI (WAIN)*. IEEE, 2021, pp. 113–120.

[59] J. Chen, H. Guo, P. Liu, and Y. Wang, "The summary on atmospheric disturbance problems in the motion imaging of high resolution earth observation system," in *Proceedings of 2011 International Conference on Electronic & Mechanical Engineering and Information Technology*, vol. 8. IEEE, 2011, pp. 3999–4003.

[60] E. Denney, G. Pai, and I. Whiteside, "Modeling the safety architecture of uas flight operations," in *Computer Safety, Reliability, and Security: 36th International Conference, SAFECOMP 2017, Trento, Italy, September 13-15, 2017, Proceedings 36*. Springer, 2017, pp. 162–178.

[61] J. Cleland-Huang, A. Agrawal, M. N. A. Islam, E. Tsai, M. V. Speybroeck, and M. Vierhauser, "Requirements-driven configuration of emergency response missions with small aerial vehicles," in *SPLC '20: 24th ACM International Systems and Software Product Line Conference, Montreal, Quebec, Canada, October 19-23, 2020, Volume A*, 2020, pp. 26:1–26:12. [Online]. Available: https://doi.org/10.1145/3382025.3414950

[62] M. N. A. Islam, M. T. Chowdhury, A. Agrawal, M. Murphy, R. Mehta, D. Kudriavtseva, J. Cleland-Huang, M. Vierhauser, and M. Chechik, "Configuring mission-specific behavior in a product line of collaborating small unmanned aerial systems," *J. Syst. Softw.*, vol. 197, p. 111543, 2023. [Online]. Available: https://doi.org/10.1016/j.jss.2022.111543

[63] A. Agrawal, S. J. Abraham, B. Burger, C. Christine, L. Fraser, J. M. Hoeksema, S. Hwang, E. Travnik, S. Kumar, W. J. Scheirer, J. Cleland-Huang, M. Vierhauser, R. Bauer, and S. Cox, "The next generation of human-drone partnerships: Co-designing an emergency response system," in *CHI '20: CHI Conference on Human Factors in Computing Systems, Honolulu, HI, USA, April 25-30, 2020*, 2020, pp. 1–13. [Online]. Available: https://doi.org/10.1145/3313831.3376825