Explosive Hazard Pre-Screener Based on Simulated Data with Perfect Annotation and Imprecisely Labeled Real Data

Madeline Kovaleski[a], Aaron Fuller[a], Jeffrey Kerley[a], Brendan Alvey[a], Peter Popescu[a], Derek T. Anderson[a], Andrew Buck[a], James M. Keller[a], Grant Scott[a], Clare Yang[b], Ken E. Yasuda[b], and Hollie A. Ryan[b]

[a]Department of Electrical Engineering and Computer Science, University of Missouri, Columbia MO, USA
[b]U.S. Army DEVCOM C5ISR Center, Fort Belvoir, VA, USA

ABSTRACT

Datasets with accurate ground truth from unmanned aerial vehicles (UAV) are cost and time prohibitive. This is a problem as most modern machine learning (ML) algorithms are based on supervised learning and require large and diverse well-annotated datasets. As a result, new creative ideas are needed to drive innovation in robust and trustworthy artificial intelligence (AI) / ML. Herein, we use the Unreal Engine (UE) to generate simulated visual spectrum imagery for explosive hazard detection (EHD) with corresponding pixel-level labels, UAV metadata, and environment metadata. We also have access to a relatively small set of real world EH data with less precise ground truth – axis aligned bounding box labels – and sparse metadata. In this article, we train a lightweight, real-time, pixel-level EHD pre-screener for a low-altitude UAV. Specifically, we focus on training with respect to different combinations of simulated and real data. Encouraging preliminary results are provided relative to real world EH data. Our findings suggest that while simulated data can be used to augment limited volume and variety real world data, it could perhaps be sufficient by itself to train an EHD pre-screener.

Keywords: machine learning, semantic segmentation, u-net, simulation, unreal engine, pre-screener

1. INTRODUCTION

Obtaining real world datasets from *unmanned aerial vehicles* (UAVs) with quality ground truth (annotation and metadata) for training supervised learning-based *machine learning* (ML) algorithms is time and cost prohibitive. An enormous amount of effort is required to plan and coordinate a data collect. Each collect requires a skilled pilot, charged batteries, flight plans, a working UAV, acceptable weather for flying, etc. In addition, each target emplacement should be well documented with accurate GPS positions and pictures of the surrounding context. For training supervised learning ML models and for scoring, annotations are required. Accurately labeling images of EHs frequently requires the assistance of skilled experts who painstakingly draw *axis aligned bounding boxes* (AABBs) and assign class labels to objects*. Furthermore, the global COVID-19 pandemic made coordinating physically with one another difficult. Meeting in person for data collections, at the moment, requires extra safety precautions in addition to being subject to delays. Each of these requirements and conditions places a bottleneck on the volume of quality annotated data available for training and testing.

Herein, we circumvent the real world data bottleneck by using simulated data to train, evaluate, and understand a ML-based *explosive hazard detection* (EHD) pre-screener. Specifically, our current article makes the following two contributions. First, we explore the use of modern game engine tools with perfect pixel-level ground truth and metadata to train a *neural network* (NN) based semantic segmentation model for EH pre-screening. More specifically, we investigate *cartoon* (TOON) and current generation video *game engine quality* (GEQ) rendering. These datasets possess different levels of visual information and they provide a rich platform to explore

*It is rare to get pixel-level annotations for real world aerial datasets. This is largely due to how much effort is involved and many aerial payloads have high frame rates and image resolutions (thousands of pixels in each dimension).

Table 1: Notations and Acronyms

Acronym	Description
ML	Machine Learning
AI	Artificial Intelligence
GEQ	Game Engine Quality Simulated Data
TOON	Cartoon Simulated Data
RS	Reality Spectrum
EH	Explosive Hazard
EHD	EH Detection
AMA	Altitude Modulated Augmentation

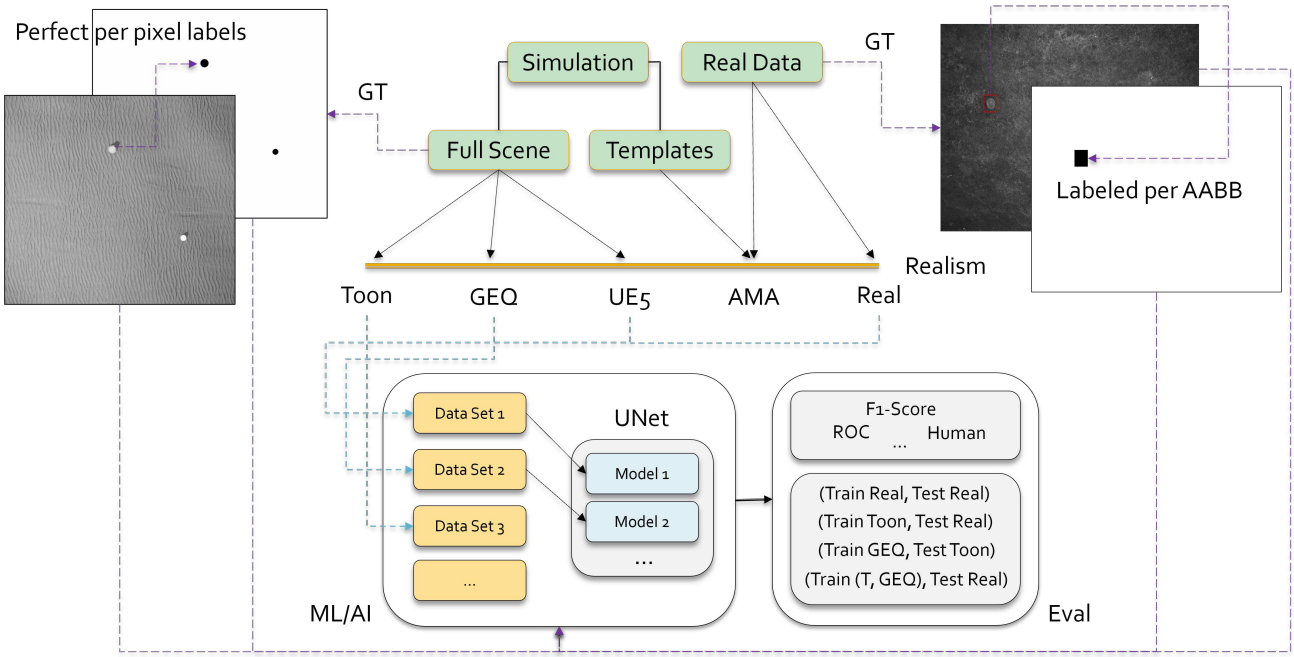

Figure 1: High-level illustration of this article. Simulated data is used to train a per-pixel pre-screener for EHD. While simulation yields perfect per-pixel ground truth, real data has ambiguous AABB labels. The first objective of this article is to explore the performance of neural segmentation networks for EHD pre-screening. The second objective is to explore performance relative to different combinations of perfectly labeled simulation data and less than ideal human labeled real data.

questions surrounding model generalizability. Second, we explore how to combine perfectly labeled simulation data with ambiguously labeled real data. The idea is to determine if simulated data can help a machine learn improved features by discovering how to make better use of ambiguously labeled human data that unfortunately possess both target and background features.

Figure 1 is a high-level illustration of the proposed article and Table 1 is notation. In Section 2, we summarize related work. Section 3 details our pre-screener, Section 4 discusses simulated datasets, Section 5 documents real world datasets, and Section 6 discusses experiments and results.

2. RELATED WORK

2.1 Explosive Hazard Detection

EHD is a real world challenge that is not going away. EHs are not trivial to detect as they vary with respect to factors like size, shape, composition, material, and context in/across environments and environmental conditions. The task of detecting EHs is one better suited for a UAV than a precious human being. UAVs are replaceable and they offer a way to keep humans at a safe standoff distance. They can be equipped with high resolution

cameras observing different portions of the electromagnetic spectrum as well as position sensors (e.g., GPS and IMU). However, detecting EHs from an aerial platform is not a solved nor trivial problem. For example, imagery collected by a UAV at an altitude of 30 meters looking straight down (nadir) looks vastly different from a UAV at a lower altitude looking at a different pitch. The point is, UAV-based EHD has great potential, but it is a complex technological task that is full of many sub-challenges.

While our current article is focused on UAVs, a number of technologies have been explored to date. An early and well-known technique is the so-called "metal detector", which can be used to detect metal buried in the ground. However, one limitation with this form of detection is that threats that contain low amounts of metal may go undetected. Increasing the sensitivity of the device does not necessarily counteract this, as the number of false alarms would likely dramatically increase. To increase the robustness of detection, many different combinations of sensing methods have and are being explored, such as *infrared* (IR), *ground penetrating radar* (GPR), *electromagnetic induction* (EMI), and *hyperspectral imaging* (HSI), to name a few. The two predominant approaches to date for detecting explosives is vehicle-mounted detectors and hand-held detectors. While the latter is predominantly used in a downward looking fashion, the prior comes in a multitude of forms, e.g., forward looking,[1] downward looking,[2] and even side looking.[3] The reader can refer to[4] for a recent review of computational intelligence algorithms in EHD. Herein, we focus on UAVs, which can operate in each of the above modalities. This method of data collection has the potential to help search areas faster, especially in the case of a swarm of UAVs, and with behaviors to facilitate dynamically interrogation of regions of interest.

On a final note, it is important to note that the signatures being exploited here derive from line of sign spatial features, the targets under consideration are surface-deployed, and the solution space being worked is *electro-optical/infrared* (EOIR) against line-of-sight EH only.

2.2 Simulation

We are not the first to use game engines to train and evaluate ML/AI techniques. In 2015, Gaidon et al. used Unity to make a *VIRTUAL KITTI* (VKITTI) autonomous car non-photorealistic dataset.[5] In 2015, Chen et al.[6] used *The Open Racing Car Simulator* (TORCS)[7] to train a *deep NN* (DNN) to drive (again, non-photorealistic imagery). In 2017, Martinez et al. used the video game Grand Theft Auto to generate high quality rendered imagery for training, testing, and enhancing DL for self-driving cars.[8] In 2018, Martinez et al. proposed UnrealROX,[9] which used UE4 to create realistic looking indoor scenes for robots to interact with objects in simulation. In 2018, Muller et al. explored Sim4CV[10] in UE4 for generic CV research. In 2020, Drouin et al.[11] used real ortho-photos to simulate aerial data collections, and in 2021, Nouduri et al.[12] generated synthetic views from a dense 3D point cloud. In Alvey, et al.,[13] we proposed UE and AirSim based frameworks and workflows, with open source code[†] and training videos[‡], for accelerating computer vision and unmanned low-altitude aerial vehicle research. Examples were highlighted for object detection, passive ranging, context-driven fusion,[14] and augmented reality. While Sim4CV is the closest to our current article and previous UE-based work,[13] the content used and imagery produced is not photorealistic, no detailed workflows are outlined, no supplemental online training is available, and results are simulator-focused vs. real world and simulation cross-validated.

In,[15] see Figure 2, we previously put forth a process called *altitude modulated augmentation* (AMA). AMA uses UE to render images of objects in different contexts, allowing us to render target EHs in different object poses, camera relative poses, and solar positions (which impact illumination and shadows). As Figure 2 shows, the result in a false color RGB image where the red channel is the grayscale (monochromatic) image, green is a per-pixel map of object locations, and the blue channel is a per-pixel map of shadow locations and values[§]. These images, which we refer to as "templates", are then inserted into real low-altitude aerial data. The object and shadow layers are used as alpha transparency maps. AMA uses the platform/sensor metadata (drone altitude, pose, etc.) to decide where and how to insert templates into the real data (see[15] for more details). The goal of AMA is to use simulation to render novel contexts of objects that are not observed in an existing data collection.

[†] https://github.com/MizzouINDFUL/UEUAVSim

[‡] https://bit.ly/MizzouINDFUL

[§] If additional per-pixel information needs to be stored, e.g., RGB color, then different UE targets, e.g., custom depth, custom stencil, render targets, etc., can be used. The reader can learn more by reading about custom UE materials, post processing effects and materials, and the Movie Render Queue.

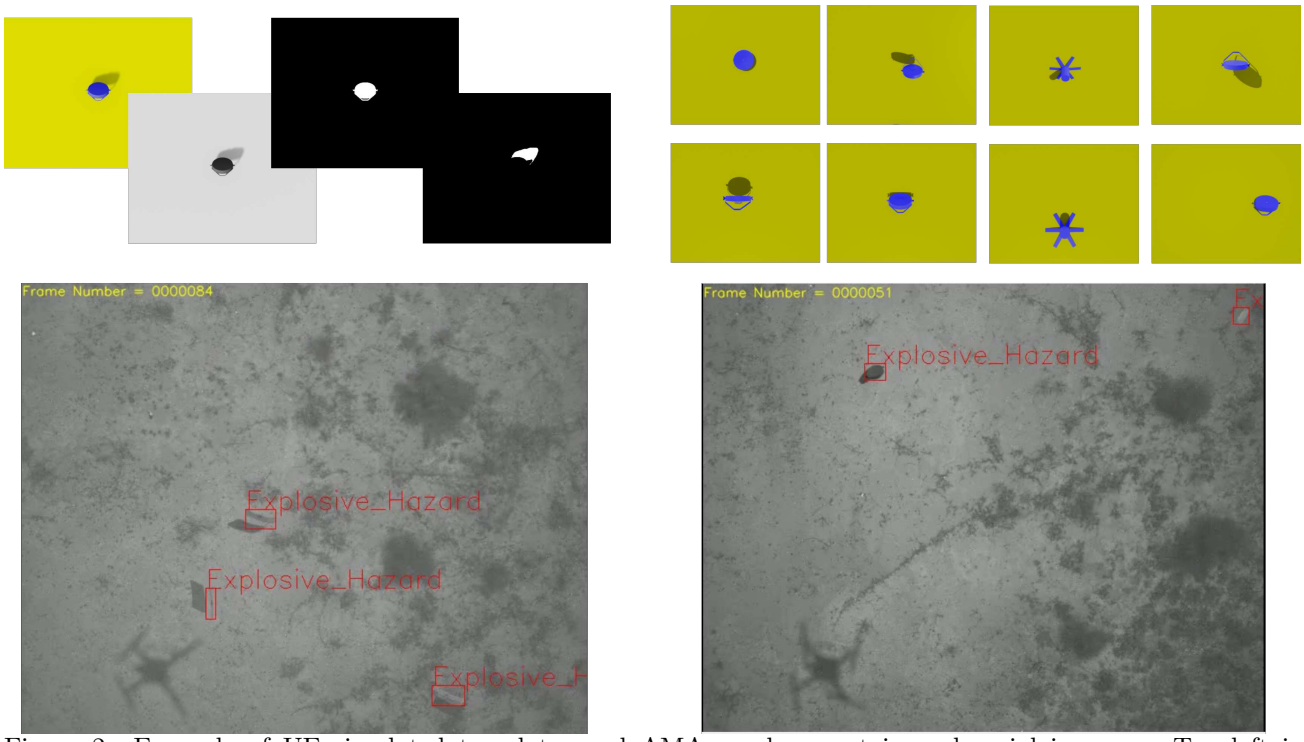

Figure 2: Example of UE simulated templates and AMA emplacement in real aerial imagery. Top left is a simulated false color RGB image (R=grayscale image, G=object pixels, B=shadow pixels) and individual channels. Top right is simulated objects at different camera poses and sun positions. Bottom left and right is AMA placing the above simulated templates into real aerial data (EHs highlighted as red AABBs).

AMA can be regarded as a type of data augmentation. As shown in,[15] the performance of AMA is notably higher than what is currently achieved using full image simulated data. This should be expected as AMA aims to use existing real world data for a specific environment. It is hypothesized that AMA is ideal for scenarios like training or transferring an AI/ML model to a new operating environment using only a small collection of non-target background data. This is an attempt to reduce the expense (cost, time, etc.) of acquiring training data required for a new environment. While useful, AMA has limitations. For example, AMA requires background (non-target) data from the new environment. Furthermore, AMA only enriches existing data with respect to the object. It does not help with variables like time of day, environmental differences, emplacement context (e.g., occlusion), etc. This is a goal of the current article. We wish to use simulation to increase our volume and variety of training data with respect to a richer set of environments, environmental conditions, and platform (drone and camera) contexts. Furthermore, we desire supervised data with complete pixel-level labels.

3. EHD PRE-SCREENER

In this section we discuss our EHD pre-screener. In general, a pre-screener is a process (algorithm, feature, etc.) that nominates *alarms* (candidate targets) in data. Ideally, a pre-screener should have near a 100% *positive detection rate* (PDR) which sometimes comes at the expense of a relatively high *false alarm rate* (FAR). Strong classifiers are usually not considered pre-screeners. Most pre-screeners are relatively lightweight, can be run efficiently, and ideally result in as high of a PDR as possible; perhaps a rate comparable to that of a human.

Herein, we explore the U-Net[16] neural net architecture as an EHD pre-screener. While U-Nets have been extensively used for image semantic segmentation, they have also become the workhorse for problems like passive ranging, deep diffractive neural networks and holography, and beyond. We selected U-Nets for the following reasons. First, they can be used for segmentation at a per-pixel level, something not possible via detection

and localization networks like YOLO[17]¶. Second, a U-Net can be engineered to model behaviors similar to attention in humans.[19] This is in effect what we are trying to achieve: an algorithm that queues AOIs requiring further analysis; which is also supported indirectly in networks like YOLO through cost function encouragement ("objectness"). Third, a U-Net can achieve a fair amount with its limited parameters due to weight sharing and skip connections. A U-Net can be reduced in number of parameters, then trained and used computationally on a limited resource embedded device like an NVIDIA Jetson.

While many variants exist, in general the U-Net architecture consists of a downsampling path to extract features (the "what"), and an upsampling path to identify target class locations in an image (the "where"). These paths consist of blocks connected by skip connections, allowing for more detailed image construction across scale from encoder to decoder. The paths are symmetric, meaning that for each encoder convolution that decreases in resolution, there is a corresponding decoder convolution that increases the resolution, giving the U-Net its name in the way the visualization of the architecture mirrors the letter "U".

The U-Net used herein was pulled from an existing codebase for semantic segmentation‖. Our network was trained on grayscale image datasets where the weights were updated based on a Dice Loss function and a Cosine Annealing learning rate scheduler. A per-pixel ground truth was provided for each training image, where zero indicated not a target and one indicated target. The upsampling and downsampling paths of the U-Net both consisted of five stages, respectively.

4. SIMULATED DATA - DATASET 1 (DS1)

As discussed above, this article is focused on the use of simulated data. The goal is to have the ability to alter object, environment, and platform/sensor variables that drive ML/AI and the task at hand (e.g., EHD). While different simulators could have been used, e.g., Unity,[20] VANE and ANVEL,[21] etc., we focus on UE for several reasons. It has proven longevity (decades of R&D), wide integration of assets (3D models, textures, etc.), simplicity of use and excellent online documentation. UE also supports high quality global illumination and realtime rendering capabilities (see NVIDIA *deep learning super sampling* (DLSS)). Historically, UE was created for video games, but it has found uses in film,[22] computer graphics,[23] architecture,[24] and beyond. Most importantly, high fidelity custom dynamic scenes can be conveniently purchased or produced manually in little time. It is also easy to mix content, manually or via scripting, to vary or cater to niche applications where scenarios are costly, hard, or not practical to obtain. UE is also constantly improving and making free content available, e.g., Quixel megascans 3D scanned real world objects and materials[25] and free military 3D objects,[26] as well as supporting tools for procedural content generation, e.g., large scale terrain generation with Instant Terra (UE plugin[27]) and Houdini for object/geometry, texture, terrain, animation, city generation and beyond (see UE Houdini plugin[28]). Furthermore, UE has a rich support for models, materials, effects, and more on their UE Marketplace,[29] including the artistic shaders we used herein (cel shader[30] and artistic pen and paper shader[31]).

4.1 Generated Data

The reader can refer to our 2022 SPIE article[32] for details about our procedural framework and workflow in UE to generate EH environments and automated data collections. In this section, we briefly summarize our simulated datasets. DS1 (see Table 2) is made up of three sub-datasets, which are detailed next.

Manually Created UE EHD GEQ Data In 2021, we produced an initial dataset of 720 images in the UE (see Figure 3) to train object detection and localization algorithms[15] and explore XAI.[18] This dataset is rasterization rendering based, versus global illumination or ray traced, and textures and objects are relatively low detail, e.g., no high 3D model poly counts or *physically-based rendering* (PBR) textures (albedo, roughness, normal maps, specular, etc.). The reader can see in Figure 3 that the result is similar to last generation video games. Details like bill boarded foliage and simple coloring, illumination, and shadow are present.

¶The reader can refer to[14,15,18] for our prior works using simulation for training and characterizing localization and detection AABB detectors (specifically YOLO) vs. segmentation.

‖Semantic segmentation network library used herein can be found at https://github.com/qubvel/segmentation_models.pytorch/blob/master/docs/index.rst and https://smp.readthedocs.io/en/latest/

Table 2: Simulated Dataset 1 (DS1)

Variable	Description
Number of Images	8720 (8000 images from 2022 and 720 from 2021 collections)
Environments	arid
Foliage	rocks, bushes, trees, stumps, grass, flowers, cacti, branches
Background Texture	sand, beach, grass, rocky
Clutter	vehicles, signs, fences, tires, helicopters, tents, small buildings, crates, barrels, cones, umbrellas, trashbags, cans of soup
Drone Altitude	$[20, 40]$ meters
Drone Pose	Nadir with random pitch perturbation (± 5 °), random yaw, no roll

Procedurally Created UE EHD GEQ Data In 2022, we generated new UE EHD GEQ imagery with our procedural UE algorithm,[32] see Figure 4. Improvements to our 2021 imagery are as follows. 3D models were more detailed (higher geometry count), textures were PBR-based, and global illumination was used. From a quality standpoint, object shape, texture, color, illuminations, and shadows are much more photo-realistic. A total of 4,000 images were generated from four procedural scenes. Two scenes were captured around solar noon (2,000 images), one was captured in the early morning (1,000 images) and one in the late afternoon (1,000 images). The reason for these collects was to sample variation with respect to lux and shadows.

Procedurally Created UE EHD TOON Data Figure 5 shows example procedurally generated[32] cartoon (TOON) UE imagery from our 2022 collect. Herein, we consider TOON to be simplified object geometry (thus shape), overly saturated color, and stylized or simplified texture. Overall, it is abstract in visual appearance. Our procedurally generated imagery is similar in style to a cartoon like Garfield or whimsical and fairy tale like the Legend of Zelda. Our intention is to produce imagery that places important features like shape, color, and contrast center stage. While this information is present in real imagery, it is often overshadowed by higher frequency texture information. A total of 4,000 images were generated for two solar noon sub-datasets (2,000 images), one early morning sub-dataset (1,000 images), and a late afternoon sub-dataset (1,000 images).

All procedurally generated data was subject to the general guidelines outlined in Table 2. This consisted of arid and forested locations, common foliage in those environments, related background (aka ground) textures, and clutter common to those environments and our task at hand. Our three EH objects were pseudo-randomly placed in a dense respect across our maps. The collection platform (camera on a drone) was pseudo-randomly varied within operating conditions similar to our real world data collections. Specifically, yaw was randomized, roll was fixed, and pitch was slightly varied around nadir. Overall, our goal was not to generate simulated datasets in support of a specific EHD scenario. For example, a data collection in a specific location at a specific time of day with respect to particular occlusion conditions, materials, etc. Our simulated 2021 and 2022 datasets are a random collection of data across related environments, flight conditions, and with respect to initial EH objects of interest. In future work, we will explore restricting the procedural simulation dataset variables to help train and/or test to answer questions of interest. One main objective of this paper is to see if a wide net of random aerial EH variable sampling works relative to training a pre-screener.

5. REAL DATA - DATASET 2 (DS2)

Our real dataset consists of images with EHs collected from a low-altitude UAV. This dataset has eight flights, referred to as runs hereafter. Four of the runs were collected on one day, three a month later, and the remaining three were collected some months later. Overall, we have an extremely limited amount and variety of labeled training and test data as we begin to explore aerial-based EHD. To make matters more complicated, the drone altitude, location, month, time of day, objects, object emplacements, and other variables changed across runs. This poses great complexity with respect to performing cross validation. Most of these collections are very different. This is one of the main factors driving our exploration of simulation. It is extremely hard, if not a bottleneck, to design and collect a wide range of data collections with respect to our combinatorial explosion

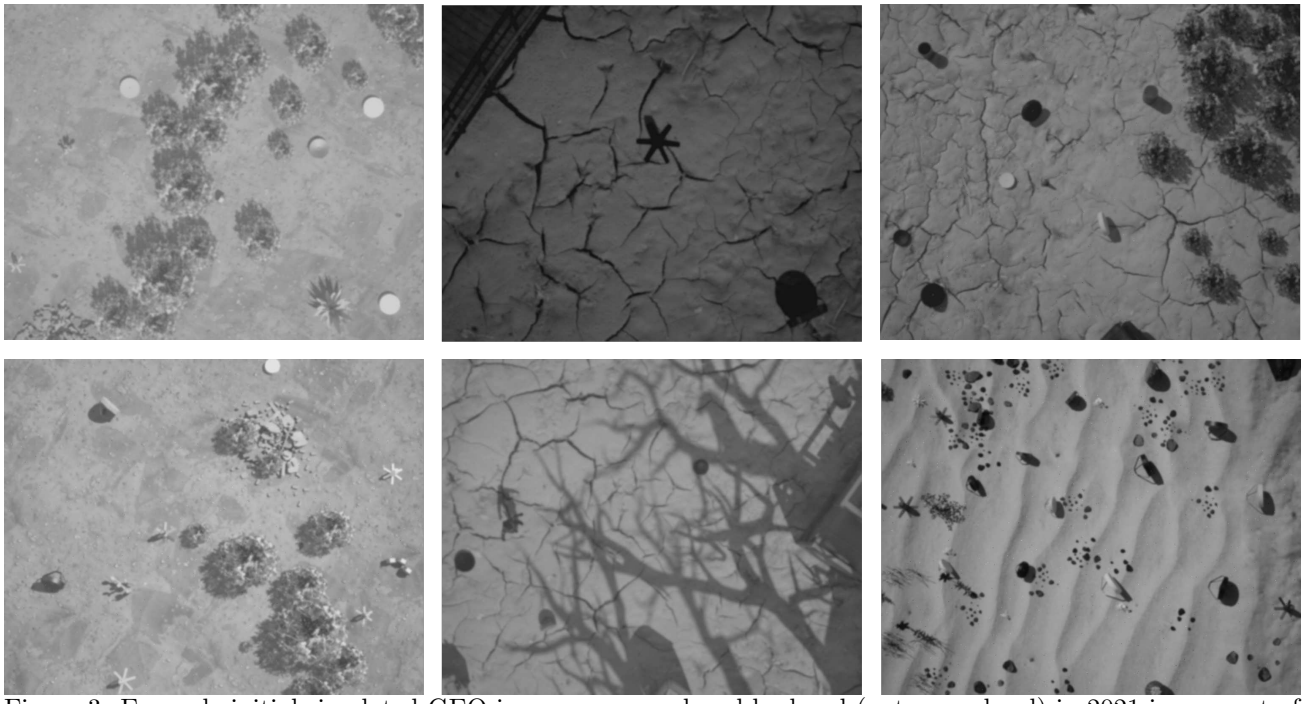

Figure 3: Example initial simulated GEQ imagery we produced by hand (not procedural) in 2021 in support of past object detection, localization, and XAI research.[15,18]

Figure 4: Example 2022 GEQ data – top is RGB, bottom is corresponding grayscale – produced procedurally[32] in support of this article. Compared to our old simulated imagery (see Figure 3), this new data has more realistic objects, textures, illumination, shadows, and the overall quality is more photo-realistic.

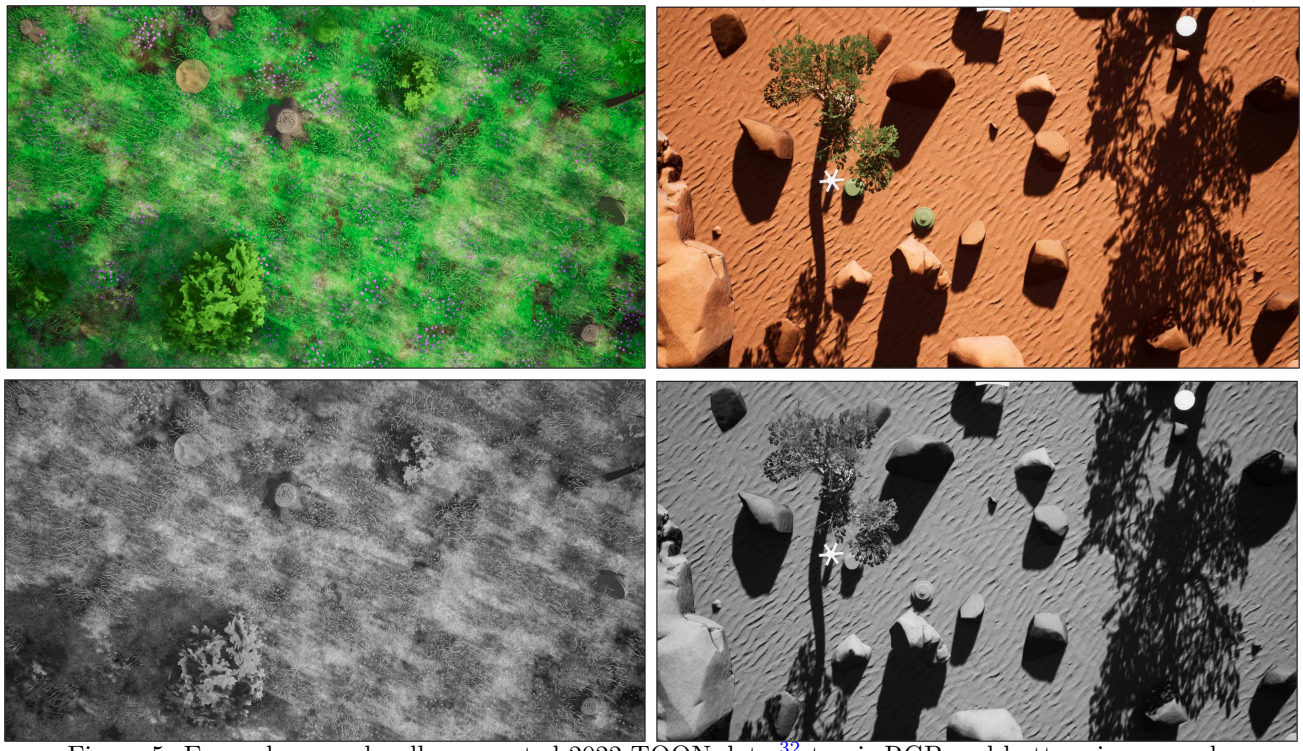

Figure 5: Example procedurally generated 2022 TOON data;[32] top is RGB and bottom is grayscale.

of object, environment, and platform/sensor EH variables**. Even if this data is collected, its an even larger undertaking to label it and provide ground truth. As a result of this real world predicament, we restrict our analysis herein to the following. We use seven runs for training and one run is left out for testing. This run (operating condition) was identified in conjunction with our collaborating funding source. The other runs are primarily of interest with respect to exploring how much we can vary these operating conditions.

Overall, a variety of EH targets exist in our dataset. Figure 7 shows examples of the four EH objects analyzed herein. Object 1 and Object 4 are mainly driven by shape, which is somewhat unique, and one is a smaller version of the other. Object 2 is driven by unique shape and Object 3, the most simple, has a very general shape, is not unique, and can easily be confused with other image content. In this article, we limit the scope of our analysis to grayscale vs. RGB imagery. That is, at this moment in time we are not considering spectral information. The resolution of the imagery explored herein is 640 x 512.

6. EXPERIMENTS AND PRELIMINARY RESULTS

This section outlines three experiments. Our objective is to learn from these preliminary results and develop intuition regarding best practices as we move forward with low-altitude aerial-based EHD. For example, should we primarily continue to request and wait on additional real world datasets to train our algorithms? Do we need to continue to refine our simulation before it can be used to train an ML/AI algorithm directly for real world data? Is our current simulation already sufficient, and if so what is the role of real world data? There are many intriguing questions in the intersection of these spaces. The point is, we are looking to better understand how well each of these processes are working individually and what utility is there in combining them.

Experiment 1 (E1): Baseline: Train on Real, Test on Real: The objective of E1 is to establish a baseline. What can be achieved using our limited variety and volume raw data?

**Example EH variables include the following. Object: type, size, pose, texture, contrast, occlusion, etc. Platform/sensor: camera/drone relative pose, altitude and standoff (number of pixels on target), speed, image resolution, focal length, spectrum, etc. Environment: season, time of day, region, foliage, clutter, etc.

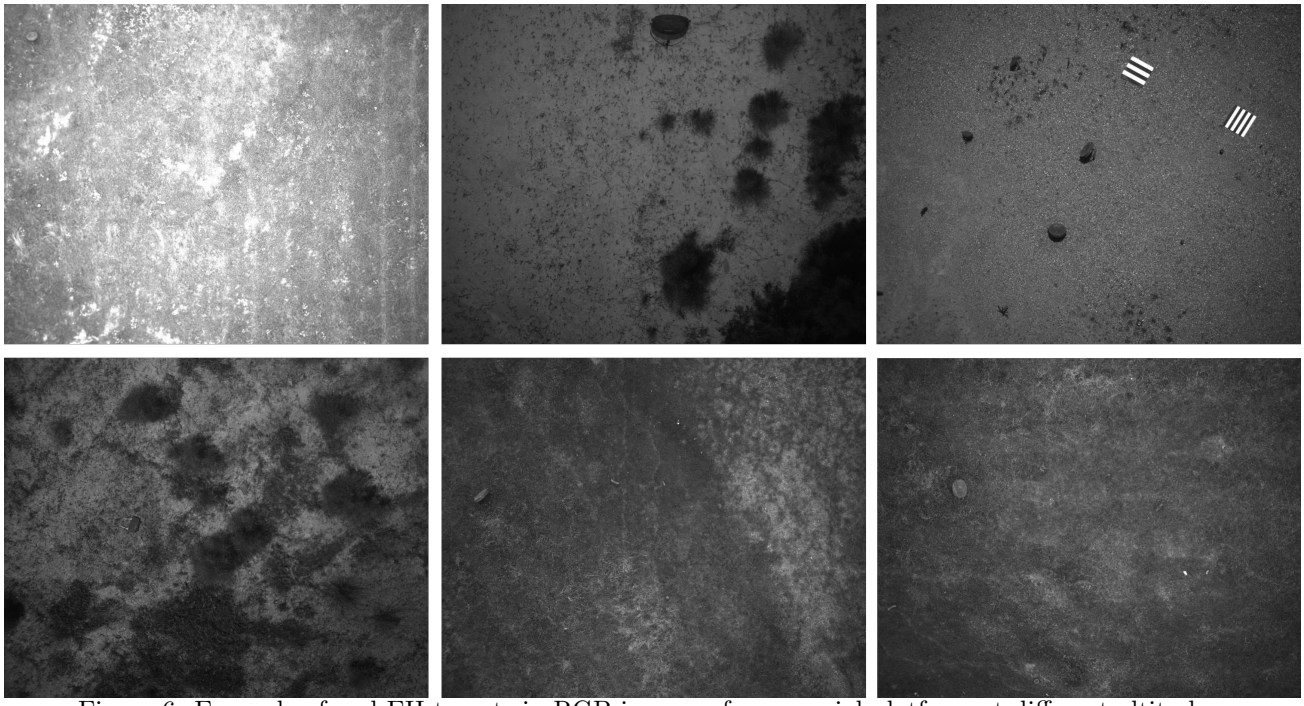

Figure 6: Example of real EH targets in RGB imagery for an aerial platform at different altitudes.

Experiment 2 (E2): Simulation: Train on Sim, Test on Real: The objective of E2 is to measure how well a simulated data trained U-Net works on real world test data. Our simulated data has the advantage that it has a much greater number of looks on target; altitudes, poses, illuminations, shadows, textures, etc. Its disadvantage is that it is synthetic and it could possess rendering artifacts (false features). Our synthetic data is a combination of arbitrarily constructed maps and procedurally random maps.

Experiment 3 (E3): Sim+Real: Train on Sim and Real, Test Real: The objective of E3 is to determine if synthetic data enhanced real data.

Figure 7 shows ROC curve results. Accompanying qualitative imagery – input imagery and U-Net results – are shown in Figure 8. The user can observe the following trends. The real data U-Net performs the worst. The second best model is Sim. It is important to note that the Sim U-Net was not fine tuned for our real data, i.e., specific site, time of day, textures, etc. Furthermore, our simulated objects are abstract representations of most of the real targets. They have the general shape and properties, but they are not scanned models nor specific instances of these objects. In particular, Object 2 was created in Blender using 1 cylinder and 7 rectangular boxes. Our Sim data also has different properties such as intensities, contrast, platform motion, sensor noise, etc. Regardless, the Sim U-Net is able to outperform the real data model.

The top performing model is Real+Sim. We do not have a mechanism to evaluate what data-driven features were learned; something generally associated with the left half of the U-Net. Nor do we have a way of measuring if these features are simulation image specific features, real data specific features, or features applicable to both. The next question is the right half of the U-Net. How are these features combined with respect to reasoning ({target, not target}) and producing the final segmentation image? We can only conjecture at this moment in time, but it appears that improved features were learned and/or the act of having both simulated and real data was useful with respect to the machine learning how to use these features on both sets of data relative to predicting and segmenting objects with similar shape and contrast. The Real+Sim model rises quickly and plateaus at less than 100% PDR due to EH objects at the edge of our imagery (see Object 1 in Figure 7).[††]

[††]Alarms near the edge an image are often discarded in detection systems as there is not enough information to reliably detect and discriminate them from clutter.

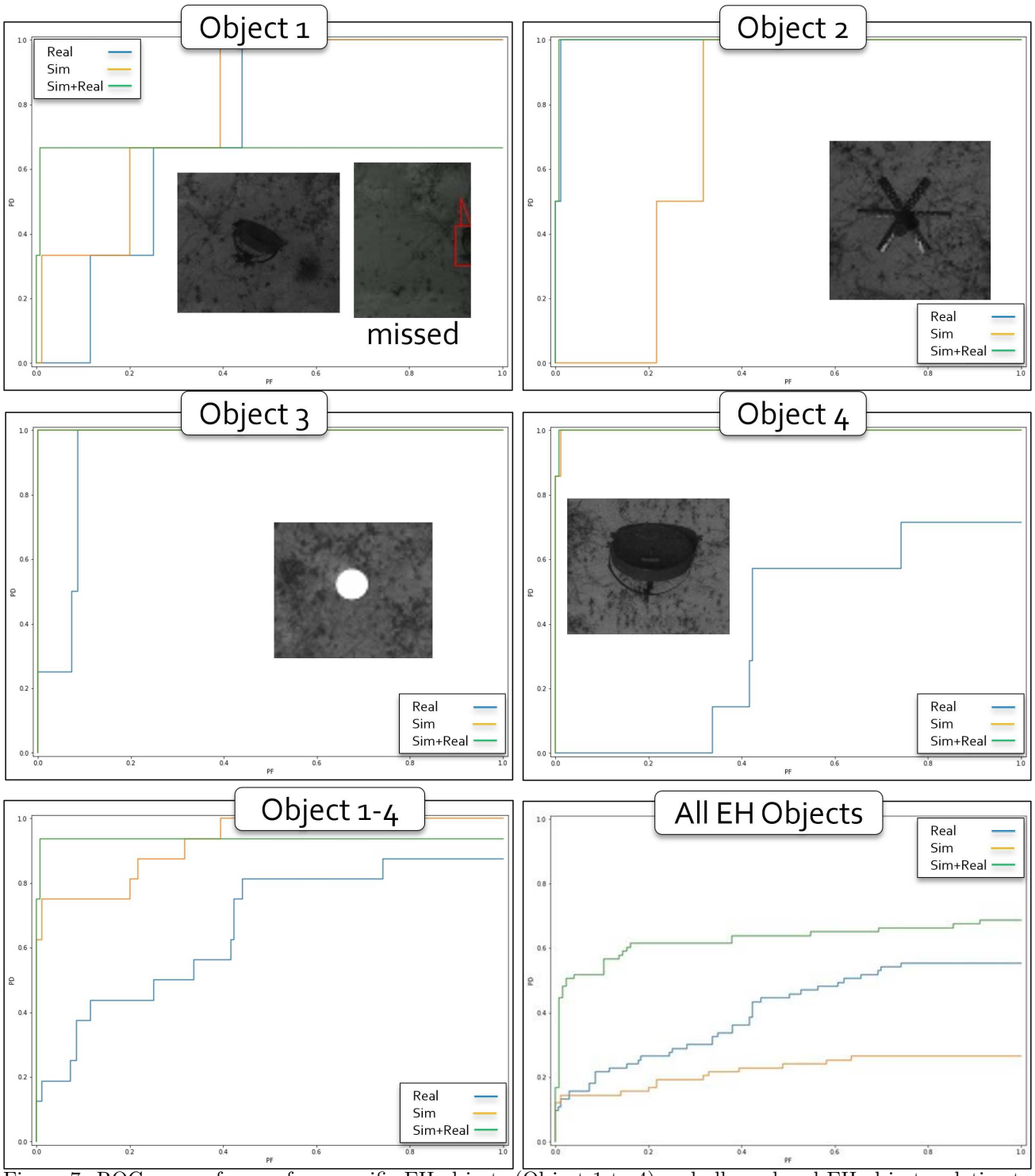

Figure 7: ROC curves for our four specific EH objects (Object 1 to 4) and all emplaced EH objects relative to our trained models; real data only trained U-Net, Sim only U-Net, and Real+Sim U-Net. Examples chips are shown in each ROC and an example Object 1 missed detection is shown. This detection was missed because it is at the edge of our image. Should it have been labeled and be included in scoring?

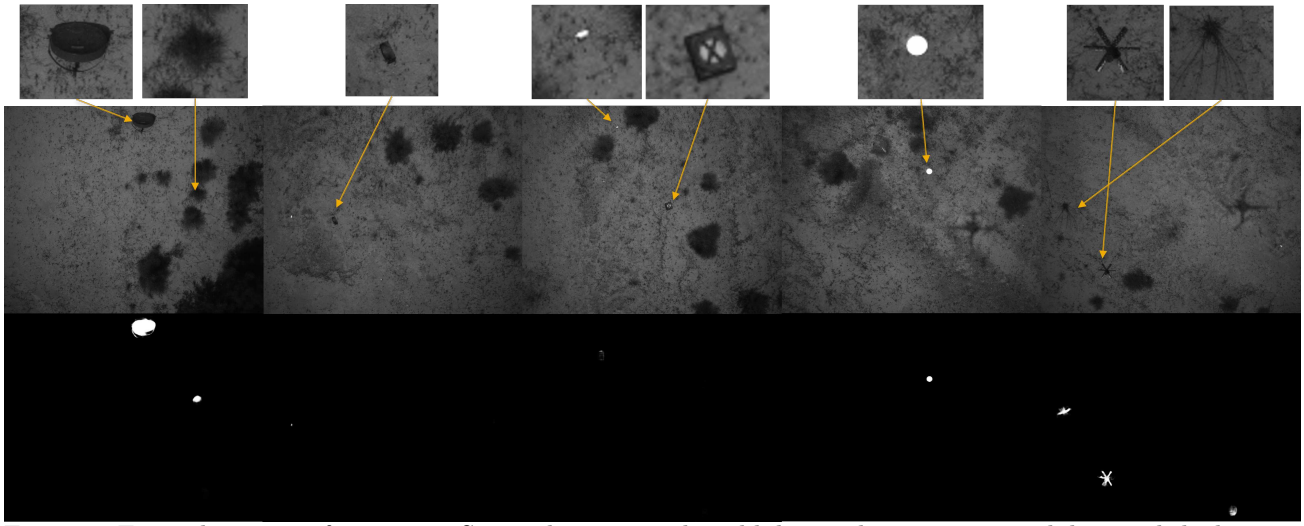

Figure 8: Example outputs for train on Sim and test on real world data. The top row is real data and the bottom row are Sim trained U-Net outputs. Example targets and FAs chips are shown.

Figure 7 also reports a ROC curve relative to scoring against all EH objects present in our data. It can be a challenge to interpret such a ROC as a good majority of these targets have too few of pixels on target and some are widely different that our four studied EH objects. We show this ROC because some of our additional EH objects, beyond Object 1 to Object 4, have similar shape and features. It is therefore interesting to observe the pre-screeners performance on this harder problem; generalizing to similar EHs. The reader can see that the system has a relatively low FAR, as the ROC curve climbs quickly, beyond Objects 1 to 4. This suggests that the combination of real and simulated data is able to enhance our U-Net and help learn more generalizable features and combination logic.

We do find it interesting, but not alarming, that in the case of all available EH objects the Sim performs the worst. Based on manual and visual inspection, we believe this can be explained as follows. The Sim data possesses just the four mentioned objects. At that, it sees them in a range of conditions, but not conditions (number of pixels on target, occlusion, etc.) consistent with the real data. It is not alarming that a real model will find a better way to make use of similar collect conditions and a specific environment. In future work we will explore if this gap can be closed with respect to a closer modeling of the environment, objects, and platform operating conditions, and/or a closer modeling of specific image statistics like intensities, contrast, etc.

Last, the reader can observe the example U-Net input (top) and output (bottom) images in Figure 8 to get a visual understanding of how well the network learns to predict our objects and their respective shapes. It should be noted that all real datasets had the human AABB labels were converted to rectangular pixel ground truth. However, the final predictions are not drawn by the U-Net as rectangles, which was the case for a U-Net trained only on real data. Instead, the Real+GEQ model has learned features that help us identify objects and ultimately more refined localization predictions. Figure 8 also shows the reader typical FAs. The Real+Sim model has relatively few FAs. However, the real model makes a greater number of mistakes on similar size, shape, and/or contrast regions, e.g., bushes, wild grass, tires, etc. An advantage of the simulation data is that it contains a greater number of close confusers.

7. CONCLUSIONS AND FUTURE WORK

In this article, we present preliminary evidence that a U-Net can be trained using simulated data from the Unreal Engine in the visual spectrum to recognize explosive hazards from a low-altitude aerial drone. Furthermore, we show that this performance is better than only using available limited volume and variety domain data. Ultimately, as one might anticipate, the combination of this data led to the best performance. However, it should be noted that the signatures being exploited here derive from line of sign spatial features, the targets

under consideration are surface-deployed, and the solution space being worked is electro-optical/infrared against line-of-sight EH only.

We are aware that this article is preliminary and it does not prove that these processes will work reliably and scale. The article is focused on a niche domain and small scale experimentation. We have also not provided any quantitative metrics beyond ROC curves to help understand what the U-Net learned. We have also not demonstrated to what degree TOON versus GEQ data, and what simulated object, environment, and platform variables are important.

There is a large body of future work ahead of us. Examples include focused experiments to assess: performance in light of non-deterministic U-Net training (what observations are real and not just a result of a single trained model); trends when/if greater amounts of real data are available (is there a trade off point where simulated data is of little to no use); to what degree does data sampled from the simulation reality spectrum impact performance. Furthermore, experiments should be performed to assess performance relative to controlled variation of underlying domain EH variables: object (e.g., size, pose, etc.), environment (e.g., time of day, occlusion, etc.), platform/sensor (altitude, focal length, etc.). This is something we are working on, but the initial framework had to be stood up. In Section 6, we also highlighted a number of questions that need more formal structured analysis. For example, did we learn simulator specific rendering features, real data image features, and/or combined or refined features. Furthermore, what *logic* was learned with respect to how to use these features and is it of benefit to present different visual abstractions of data during AI/ML training? The reader can also refer to our article on procedural generation of data in UE for a wealth of future work questions focused on the act of how to generate simulated data for a single dataset or closing the learning loop.[32]

We also did not make use of synthetically generated objects (aka templates) inserted into real data. In Alvey, et al.,[15] summarized in Section 2.2, we presented AMA. AMA had a much higher performance over real and fully simulated data relative to the YOLO object detection and localization algorithm. However, the data used in that article is our limited 720 2021 GEQ images. In future work, we need to assess if its more valuable to continue template insertion into real data for EHD or if full image simulated data is better. Clearly, AMA is limited with respect to what background imagery is available, whereas simulation can model different environments, environmental conditions, and platform contexts. While we anticipate that a combination will lead to the best performance, what simulated data should we produce?

In summary, we are encouraged by the preliminary results in this work and excited about future directions. At the time that this research was performed, the lead author was a senior in high school and the second and third authors were undergraduates in college. The point being, all of the tools and processes presented herein do not require a decade of education and a Ph.D.. Technology, specifically relatively free or low cost content, rendering tools, computational resources, and the ability to interface these technologies has the potential to unlock a rich simulated future. While simulation has been around for a long time, we are perhaps at a convergence point where it will exponentially flourish. Having possession of the truth and ability to alter things in controlled fashions is key towards our understanding of how things work and will aid our research into reliable and explainable AI/ML for domains like aerial-based EHD.

8. ACKNOWLEDGMENTS

This research is funded by ARO grant number W911NF1810153 to support the U.S. Army DEVCOM C5ISR Center. This research would not be possible without our collaborators.

REFERENCES

[1] Anderson, D. T., Stone, K. E., Keller, J. M., and Spain, C. J., "Combination of anomaly algorithms and image features for explosive hazard detection in forward looking infrared imagery," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **5**(1), 313–323 (2012).

[2] Gader, P. D., Mystkowski, M., and Yunxin Zhao, "Landmine detection with ground penetrating radar using hidden markov models," *IEEE Transactions on Geoscience and Remote Sensing* **39**(6), 1231–1244 (2001).

[3] Dowdy, J., Brockner, B., Anderson, D. T., Williams, K., Luke, R. H., and Sheen, D., "Voxel-space radar signal processing for side attack explosive ballistic detection," in [*Detection and Sensing of Mines, Explosive Objects, and Obscured Targets XXII*], Bishop, S. S. and Isaacs, J. C., eds., **10182**, 421 – 435, International Society for Optics and Photonics, SPIE (2017).

[4] Havens, T., Anderson, D. T., Stone, K. E., Becker, J., and Pinar, A. J., "Computational intelligence methods in forward-looking explosive hazard detection," in [*Recent Advances in Computational Intelligence in Defense and Security*], (2016).

[5] Gaidon, A., Wang, Q., Cabon, Y., and Vig, E., "VirtualWorlds as proxy for multi-object tracking analysis," in [*2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*], 4340–4349 (2016).

[6] Chen, C., Seff, A., Kornhauser, A., and Xiao, J., "DeepDriving: Learning affordance for direct perception in autonomous driving," in [*2015 IEEE International Conference on Computer Vision (ICCV)*], 2722–2730 (2015).

[7] Wymann, B., Dimitrakakisy, C., Sumnery, A., and Guionneauz, C., "TORCS: The open racing car simulator," (2015).

[8] Martinez, M., Sitawarin, C., Finch, K., Meincke, L., Yablonski, A., and Kornhauser, A., "Beyond Grand Theft Auto V for training, testing and enhancing deep learning in self driving cars," (2017).

[9] Martinez-Gonzalez, P., Oprea, S., Garcia-Garcia, A., Jover-Alvarez, A., Orts-Escolano, S., and Garcia-Rodriguez, J., "UnrealROX: An extremely photorealistic virtual reality environment for robotics simulations and synthetic data generation," *ArXiv e-prints* (2018).

[10] Müller, M., Casser, V., Lahoud, J., Smith, N., and Ghanem, B., "Sim4CV: A photo-realistic simulator for computer vision applications," *Int. J. Comput. Vision* **126**, 902–919 (Sept. 2018).

[11] Drouin, M., Fournier, J., Boisvert, J., and Borgeat, L., "Modeling and simulation framework for airborne camera systems," in [*ICPR Workshops*], (2020).

[12] Nouduri, K., Gao, K., Fraser, J., Yao, S., AliAkbarpour, H., Bunyak, F., and Palaniappan, K., "Deep realistic novel view generation for city-scale aerial images," in [*2020 25th International Conference on Pattern Recognition (ICPR)*], 10561–10567 (2021).

[13] Alvey, B., Anderson, D. T., Buck, A., Deardorff, M., Scott, G., and Keller, J. M., "Simulated photorealistic deep learning framework and workflows to accelerate computer vision and unmanned aerial vehicle research," in [*2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*], 3882–3891 (2021).

[14] Deardorff, M., Alvey, B., Anderson, D. T., Keller, J. M., Scott, G., Ho, D., Buck, A., and Yang, C., "Metadata enabled contextual sensor fusion for unmanned aerial system-based explosive hazard detection," in [*SPIE*], (2021).

[15] Alvey, B., Anderson, D. T., Keller, J. M., Buck, A., Scott, G., Ho, D., Yang, C., and Libbey, B., "Improving explosive hazard detection with simulated and augmented data for an unmanned aerial system," in [*Detection and Sensing of Mines, Explosive Objects, and Obscured Targets XXVI*], Bishop, S. S. and Isaacs, J. C., eds., **11750**, 76 – 90, International Society for Optics and Photonics, SPIE (2021).

[16] Ronneberger, O., P.Fischer, and Brox, T., "U-net: Convolutional networks for biomedical image segmentation," in [*Medical Image Computing and Computer-Assisted Intervention (MICCAI)*], *LNCS* **9351**, 234–241, Springer (2015). (available on arXiv:1505.04597 [cs.CV]).

[17] Jocher, G., Stoken, A., Borovec, J., NanoCode012, ChristopherSTAN, Changyu, L., Laughing, tkianai, yxNONG, Hogan, A., lorenzomammana, AlexWang1900, Chaurasia, A., Diaconu, L., Marc, wanghaoyang0106, ml5ah, Doug, Durgesh, Ingham, F., Frederik, Guilhen, Colmagro, A., Ye, H., Jacobsolawetz, Poznanski, J., Fang, J., Kim, J., Doan, K., and Yu, L., "ultralytics/yolov5: v4.0 - nn.SiLU() activations, Weights & Biases logging, PyTorch Hub integration," (Jan. 2021).

[18] Alvey, B., Anderson, D., Yang, C., Buck, A., Keller, J., Yasuda, K., and Ryan, H., "Characterization of deep learning-based aerial explosive hazard detection using simulated data," in [*SPIE*], (2021).

[19] Li, C., Tan, Y., Chen, W., Luo, X., Gao, Y., Jia, X., and Wang, Z., "Attention unet++: A nested attention-aware u-net for liver ct image segmentation," in [*2020 IEEE International Conference on Image Processing (ICIP)*], 345–349 (2020).

[20] "Unity." https://unity.com/. (Accessed: 1 March 2021).

[21] Rohde, M. M., Crawford, J., Toschlog, M. A., Iagnemma, K., Kewlani, G., Cummins, C. L., Jones, R. A., and Horner, D. A., "An interactive physics-based unmanned ground vehicle simulator leveraging open source gaming technology: progress in the development and application of the virtual autonomous navigation environment (vane) desktop," in [*Defense + Commercial Sensing*], (2009).

[22] "Storytelling reimagined," (2021).

[23] "Real-time ray tracing," (2021).

[24] "UE Architecture." https://www.unrealengine.com/en-US/solutions/architecture. (Accessed: 1 March 2021).

[25] "Quixel." https://quixel.com/. (Accessed: 1 March 2021).

[26] "Free Vigilante UE Models." https://www.unrealengine.com/marketplace/en-US/profile/Vigilante?count=20&sortBy=effectiveDate&sortDir=DESC&start=0. (Accessed: 1 March 2021).

[27] "Instant Terra Unreal Engine Plugin." https://www.unrealengine.com/marketplace/en-US/item/0c08f0ce7ac04724931565b2386012d0. (Accessed: 1 March 2021).

[28] "Houdinin Unreal Engine Plugin." https://www.sidefx.com/products/houdini-engine/plug-ins/unreal-plug-in/. (Accessed: 1 March 2021).

[29] "Unreal Marketplace." https://www.unrealengine.com/marketplace/en-US/store. (Accessed: 1 March 2021).

[30] "Cel Toon Outline Post Process Material." https://www.unrealengine.com/marketplace/en-US/item/65cd54d6adcc4b47a6c383d579beda4c. (Accessed: 1 March 2021).

[31] "Post Process Toolkit." https://www.unrealengine.com/marketplace/en-US/item/94cebc0ec72945899a494cf724f96293. (Accessed: 1 March 2021).

[32] Kerley, J., Fuller, A., Kovaleski, M., Popescu, P., Alvey, B., Anderson, D., Buck, A., Keller, J. M., Scott, G., Yang, C., Yasuda, K., and Ryan, H., "Procedurally generated simulated datasets for aerial explosive hazard detection," in [*SPIE*], (2022).