HiFiAerial: A Photorealistic Synthetic Dataset with Ground Truth for Dense Prediction in Aerial Imagery

Jack Akers[a], Jeffrey Kerley[a], Derek T. Anderson[a], Andrew R. Buck[a], Daniel Buffum[a], and James M. Keller[a]

[a]Department of Electrical Engineering and Computer Science, University of Missouri, Columbia MO, USA

ABSTRACT

Training Deep Neural Networks (DNNs), particularly modern architectures such as the Transformer, is an incredibly intensive task which requires large quantities of data. However, collecting precise, real-world data can be extremely challenging, if not impossible for certain tasks where dense, per-pixel labels are required. In lieu of real-world data of sufficient quality for these tasks, models trained on high-quality simulated/synthetic data have been shown to outperform models trained on a larger corpus of real-world data, e.g., Depth Anything V2. Despite this, there is a lack of high-quality (photorealistic) publicly-available datasets for aerial contexts, which is necessary for training detection, tracking, and 3D estimation models for unmanned aerial vehicles (UAVs), micro-drones, or other low-to-medium-altitude aircraft. Herein, we present HiFiAerial, an open source collection of simulated image sequences extracted from a diverse selection of photorealistic urban and rural biomes in Unreal Engine 5 at low/medium/high altitudes with both nadir and off-nadir relative viewing angles while traversing random paths. Each sequence is accompanied by a comprehensive set of dense labels (metric depth, object AABBs, etc.), camera poses (location, roll, pitch, yaw), camera intrinsics, and other metadata. While we showcase algorithm performance on this dataset, the broader goal is to empower other researchers and foster community-driven benchmarking experiments. The dataset is available at https://github.com/MizzouINDFUL/HiFiAerial.

Keywords: Unreal Engine 5, simulation, simulated imagery, synthetic imagery, aerial, UAV, drone, image sequence, video, semantic segmentation, optical flow, depth estimation, 3D reconstruction, structure from motion

1. INTRODUCTION

Modern advances in machine learning and artificial intelligence tools have demonstrated the need for massive amounts of data to train increasingly complex models. The big data revolution highlighted the advantage of utilizing large and diverse datasets to build more robust and capable solutions to real-world problems. As the quantity of available data increased thanks to cheaper and more prevalent sensors, a new requirement emerged. Labeled datasets became crucial for training and evaluating accurate models, but they were time-consuming and costly to produce. Semi-supervised and active learning methods seek to mitigate the need for completely labeled data by only labeling some instances or taking advantage of human feedback. Alternatively, synthetically-generated datasets can provide accurate and dense per-pixel labels that would be difficult to acquire through manual labeling.

Aerial imagery is one domain where it is particularly challenging to generate sufficient labeled data. Real-world flights are often restricted by regulations, or they may contain sensitive or proprietary information that limits public availability. Additionally, some applications such as depth estimation require labels that are hard to obtain reliably. This makes simulated environments an attractive option for producing the large amount of labeled data required to train deep neural networks. By training on synthetic data, machine learning models can take advantage of unlimited diversity and unparalleled control over the dataset curation process. Edge cases that

Send correspondence to Jack Akers
E-mail: jdapm8@missouri.edu

are hard to observe in real-world conditions can be explored and used to augment existing datasets to improve overall performance.

The overall lack of publicly available aerial imagery datasets with sufficient labels for training complex machine learning models is the primary motivation for this work. Herein, we present the HiFiAerial dataset consisting of photorealistic simulated image sequences from an aerial platform, generated with Unreal Engine 5. The data is labeled with camera information, pose information, per-pixel depth, and several other per-pixel layers, making it suitable for a variety of computer vision tasks. The flights are collected at low, medium, and high altitudes with a smooth interpolation between randomly-selected waypoints, intended to loosely mimic real-world missions while maximizing data coverage. We discuss the generation process, including both local and cloud-based simulation environments, and provide examples of the data. We hope this dataset will prove to be valuable to the machine learning community and will aid in future work.

2. PREVIOUS WORK

Simulation has emerged as a powerful tool for generating datasets to train and evaluate AI models in areas like computer vision and robotics. An advantage of simulated data is the availability of dense and more accurate ground truth, such as depth maps, object IDs, surface normals, and bounding boxes, which are often challenging or not possible to acquire in real-world settings. Other advantages include the ability to create highly controlled scenarios, large amounts of domain randomized data, rapidly modify object properties and environmental conditions, and generate diverse and customizable datasets. For example, Depth Anything V2[1] made a significant leap in performance by using simulated data. By leveraging high-fidelity synthetic data, including depth maps and semantic segmentation labels, they were able to train models that performed robustly across various real-world scenarios. This method demonstrated that simulated data, when carefully curated, could bridge the gap between simulation and real-world application in depth estimation tasks. In the remainder of this section, we review notable simulation tools and datasets, while discussing their capabilities and limitations.

2.1 Simulation Tools

Several projects focus on simulation frameworks that generate synthetic data with varying levels of realism. For example, ESPADA[2] integrates fully simulated environments with photogrammetry-based scenes, aiming to provide a balance between synthetic control and photorealism. Similarly, Hypersim,[3] developed by Apple, offers a dataset of indoor scenes with rich annotations, which are not video sequences. SceneFlow,[4] another well-known synthetic dataset, provides highly controlled environments where both optical flow and depth data are available, making it useful for stereo and motion estimation tasks. SceneFlow consists of FlyingThings3D (21,818 images of everyday objects flying along randomized 3D trajectories), Monkaa (8591 images from Blender animated short film Monkaa), and Driving (ground images, 4392). Although an improvement, these data sets are not sufficient for training an aerial drone.

2.2 Ground-Level Simulated Datasets

Several datasets focus on ground-level perception, particularly for applications such as depth estimation, optical flow, and semantic segmentation. BlendedMVS[5] is a notable example, using a pipeline that blends real images with 3D simulation to generate synthetic images. This dataset bridges the gap between real and synthetic data, but its reliance on photogrammetry-based reconstructions introduces limitations in terms of diversity and scene variability. Dynamic Replica,[6] developed by Facebook Research, extends static indoor datasets by introducing dynamic stereo sequences, which is valuable for motion analysis. However, like many simulated datasets, it may not perfectly model real-world variations in lighting, material properties, and object dynamics.

Other datasets, such as Sintel,[7] originate from cinematic animation projects, providing high-quality ground truth for depth and optical flow estimation. While it features complex motion and realistic textures, its animation-driven origin means that camera trajectories and object movements may not always follow real-world physical constraints. SYNTHIA[8] is another widely used dataset that offers urban driving scenes with semantic labels, depth maps, and stereo imagery. While its synthetic nature ensures high-quality annotations, its structured city layouts and scripted vehicle/camera movements may not fully capture the stochastic variability of real-world driving. Virtual KITTI 2,[9] an improved re-rendering of the original Virtual KITTI[10] dataset,

has been used in modern depth estimation models like Depth Anything V2. It provides a high-fidelity virtual counterpart to real KITTI[11] data, maintaining camera trajectories consistent with real-world driving scenarios.

2.3 Aerial Simulated Datasets

Aerial data simulation presents unique challenges due to the need for realistic flight dynamics, relative scene viewing conditions (e.g., overhead versus looking at objects on the ground), and scene variability. TartanAir[12] is a dataset specifically designed for aerial robotics and navigation, featuring complex environments with ground-truth depth and pose annotations. Unlike many static datasets, TartanAir incorporates diverse motion patterns, including aggressive camera movements that mimic real-world drone flights. While useful, TartanAir was generated in 2020 using last-generation rendering (UE4). Specifically, TartanAir has quality issues, that make it look more simulated than real and could affect hand crafted algorithms or AI models, with respect to lighting, environmental effects, aliasing, texturing, motion blur, camera effects, and physics accuracy. Furthermore, TartanAir data is more similar to a hobbyist drone flight, e.g., drone racing, than a drone flying higher up and nadir or at a slant angle performing a task like search and rescue.

Indoor Robotics Stereo (IRS)[13] is another dataset that provides stereo imagery for indoor robotics applications, but is more constrained in scope compared to large-scale outdoor aerial datasets. In summary, these are great datasets, but they are becoming out of date, they are limited in volume, and they do not really relate to many aerial challenges.

Last, SimUAV[14] is a notable attempt at generating high-quality simulated UAV imagery. While it provides visually compelling data, one key limitation is the lack of ground-truth depth maps, which restricts its utility for tasks such as depth estimation and 3D reconstruction. Many aerial datasets, including SimUAV, focus on photorealism but may not incorporate structured ground-truth annotations necessary for training robust models.

2.4 Summary

Simulated datasets have played a crucial role in advancing computer vision and robotics by providing large-scale, annotated data for training and evaluation. While simulation offers precise control over scene parameters and sensor outputs, limitations persist in terms of realism, diversity, and annotation completeness. Some datasets, such as Virtual KITTI 2 and SYNTHIA, closely follow real-world trajectories, whereas others, such as Sintel and ESPADA, focus more on high-quality visual realism without strict adherence to physical camera constraints. Additionally, while many datasets provide depth maps, object IDs, and motion data, others, such as SimUAV, lack key ground-truth annotations. Future work should aim to bridge these gaps by developing more diverse and dynamically generated data that better capture the complexities of real-world flights and environments.

3. DATA COLLECTION

At a high level, we utilize Unreal Engine (UE) 5.3 to create and integrate both free and commercially available maps. It is important to note that each pre-existing map and asset comes with specific licensing terms, which may impose restrictions on usage, such as prohibitions on redistribution, integration with generative AI, or limitations based on academic versus industry applications. Additionally, the UE Marketplace, now rebranded as Fab, enforces distinct policies depending on the intended use case. Our approach leverages UE to work with photorealistic scenes rendered in the visible spectrum*, enabling high-fidelity data generation and simulation. Figure 1 presents examples of rural and urban environments within UE5, illustrating varying degrees of realism.

In our prior work,[16–18] we detailed our approach to formally describing scenes and data collections through our scene language (LSCENE) and collection language (LCAP). Our initial study[16] focused on defining the formal language, while the second work[17] aimed to enhance usability by introducing a JSON-based format, outlining the hierarchical structure and metadata extracted. In 2024,[18] we leveraged LLMs to automatically generate JSON scene descriptions from user text prompts. Most recently, in 2025,[19] we transitioned to a strongly typed language, TypeScript, and introduced a multi-agent LLM framework capable of autonomously crawling assets, generating LSCENEs, and performing self-correction. The present work builds upon these foundations, further refining and extending our approach.

*We rely on Infinite Studio,[15] a UE plugin, to render multi spectral imagery beyond the visual spectrum, e.g., near IR, mid wave IR, long wave IR, etc.

Figure 1. Example UE rural (left) and urban (right) photorealistic scenes.

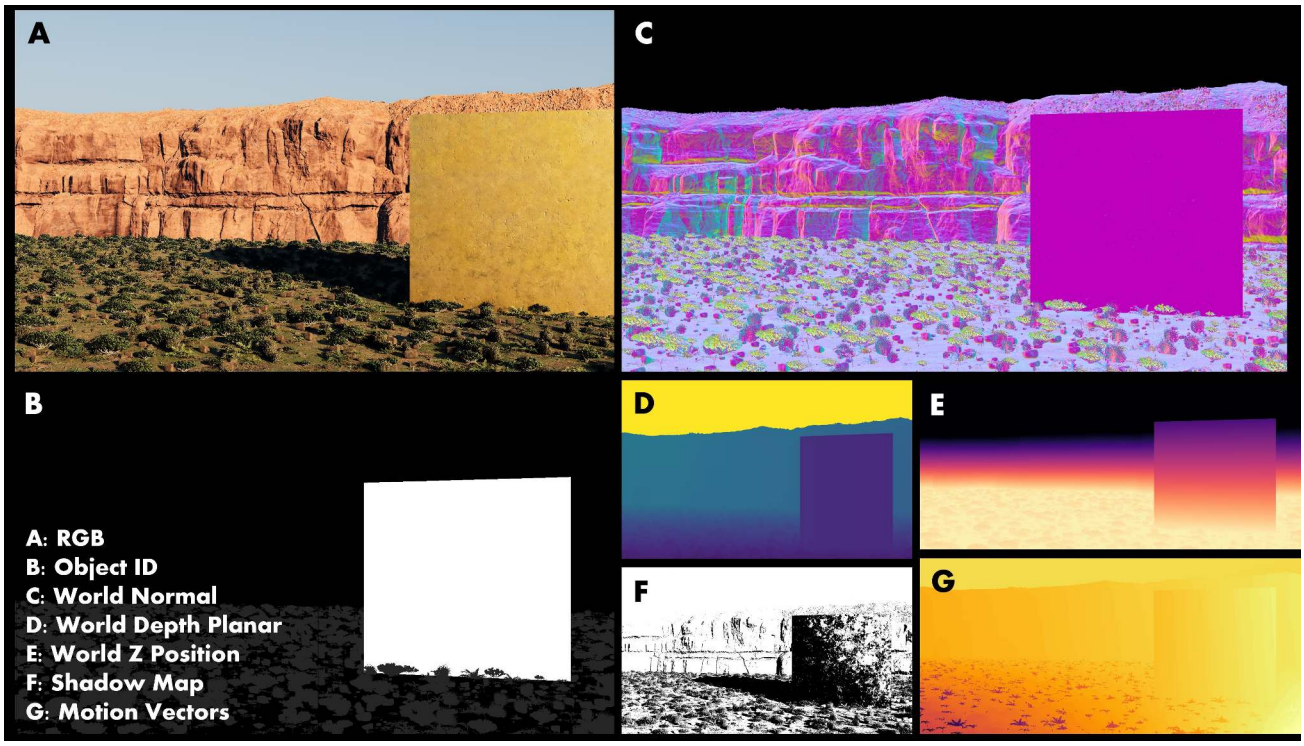

Figure 2. Examples of data layers we generate.

Figure 2 presents example data generated by our tool, encompassing multiple modalities essential for scene understanding and analysis. Specifically, we produce visual spectrum imagery (Figure 2A), integer-based object and instance identifiers (Figure 2B), which must be assigned either manually or via a procedural algorithm, world normals (Figure 2C), world depth in both planar and perspective views (Figure 2D), 3D world position (illustrated as the z-component in Figure 2E), shadow maps (Figure 2F), which are not utilized in the current study, and motion vectors (Figure 2G). Furthermore, we do not provide object IDs, as doing so is not straightforward. Each existing scene would have to first identify a shared ontology, and that would have to be consistently applied to each scene. This does not exist natively in Fab and is beyond the scope of the current article. All exported data is stored as NumPy arrays with `FLOAT32` precision, which is particularly crucial for layers such as depth and 3D position, except for object ID data, which does not necessitate such precision. Additionally, we generate EXR files containing all layers for each image, though their storage footprint can be substantial, often amounting to hundreds of gigabytes even for relatively small datasets. This is due to EXR exports from environments like UE inherently embedding auxiliary information, such as material names, file locations, and related attributes. While these layers are available for download, users must be aware of specific considerations: depth values are stored in centimeters rather than meters, world positions follow an absolute coordinate system that may be relative to $(0, 0, 0)$ or an arbitrary reference point set by a designer or procedural tool, and object dimensions ultimately depend on their creator, potentially resulting in unrealistic metric scales. These nuances, while subtle, can impact downstream applications, such as depth estimation model training.

In addition to data layers, we also export auxiliary but crucial metadata to enhance usability and analysis. Specifically, we generate JSON-formatted files that document both the camera parameters during data collection and detailed per-image content annotations. For instance, the `camera.json` file records the position and orientation of the camera for each captured image, providing essential spatial context. Additionally, the `groundtruth.json` file contains a list of present objects, their unique object IDs, axis-aligned bounding boxes (AABBs), 3D-oriented bounding boxes (OBBs), and other relevant metadata for each image. These structured annotations serve a critical role in understanding the specifics of data collection, supporting evaluation and scoring, and facilitating training pipelines for AI algorithms.

In this work, we specifically utilize environments sourced from Fab,[20] focusing on both urban and rural collections. Notably, these datasets include common objects, particularly people, making them suitable for training or evaluating models such as person detectors or depth prediction networks. While the dataset can also be applied to tasks like semantic segmentation, we do not provide object or instance IDs due to the considerations mentioned earlier. Rather than manually placing objects such as people into the scenes, we leveraged our prior LSCENE framework to automate and randomly distribute them throughout the maps. This approach was adopted based on the assumption that precise contextual placement was not a critical factor for our intended use cases. Figure 3 illustrates an example of LSCENE execution following the user prompt, "a few blue water cans in an open grassland biome at different times of day."

Finally, it is important to highlight the key differences between this work and our prior articles, particularly in terms of process modifications. Notably, this study required only minor adjustments to our existing workflows. In the next section, we describe how we leverage a cloud-enabled platform for batch data collection, as opposed to extending our tool to iterate over maps sequentially. This shift enables multi-processing, significantly improving efficiency compared to serial data collection. Additionally, the primary modification involved our LCAP, specifically to facilitate the collection of aerial data. To achieve this, we defined a volume within the map with a known altitude range, within which random waypoints were selected. A "random walk" approach was then used to navigate through these waypoints, where the user (ourselves, in this case) specifies a "step distance" (displacement delta) and an "allowable orientation change," such as a pitch variation of ± 10 degrees around an initially defined context. This methodology allows us to simulate different flight behaviors, including nadir, slant-angle, and horizon flights, while introducing a degree of natural positional and orientational deviation similar to real-world conditions affected by factors like GPS error and wind. However, it is important to note that we do not explicitly simulate errors; rather, we collect precise data within the accuracy bounds of our collection tool (UE). It is left to the reader to introduce additional randomness or structured errors post-collection, such as incorporating specific GPS inaccuracies to study their effects on downstream applications.

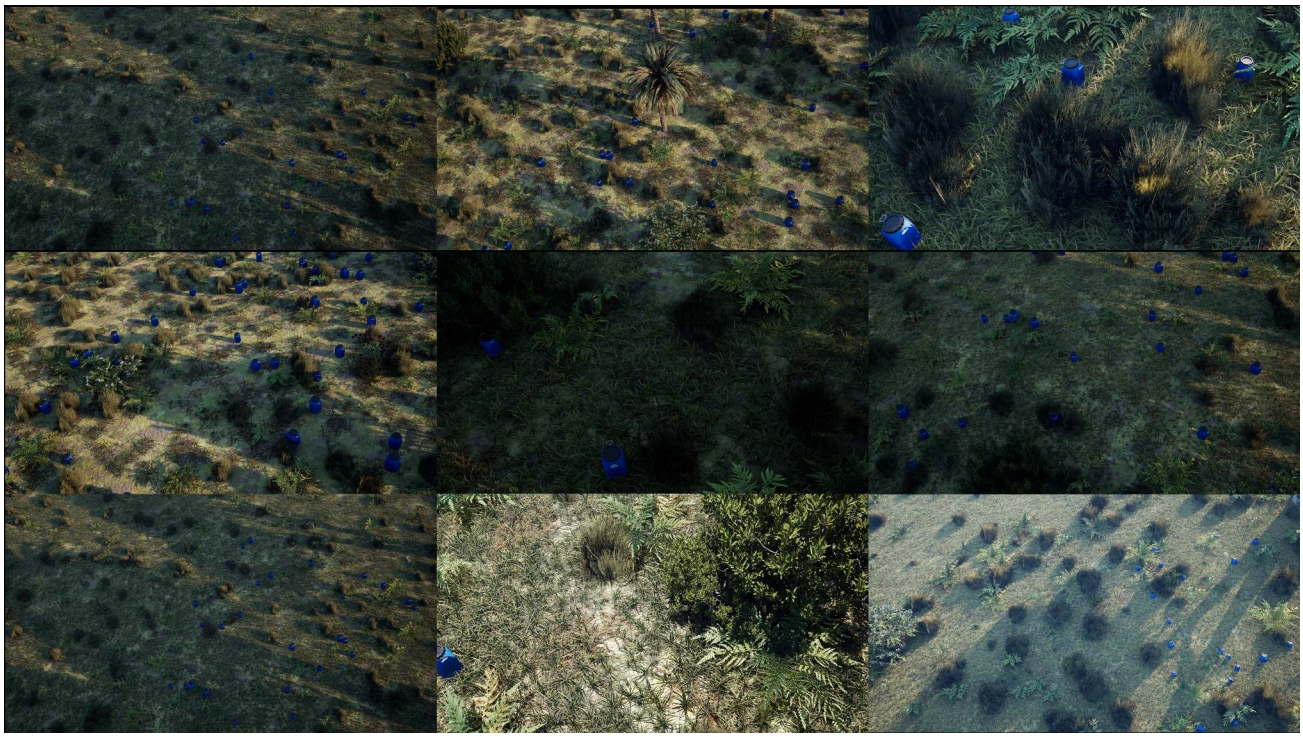

Figure 3. Example outputs of LSCENE for the prompt, "a few blue water cans in an open grassland biome at different times of day."

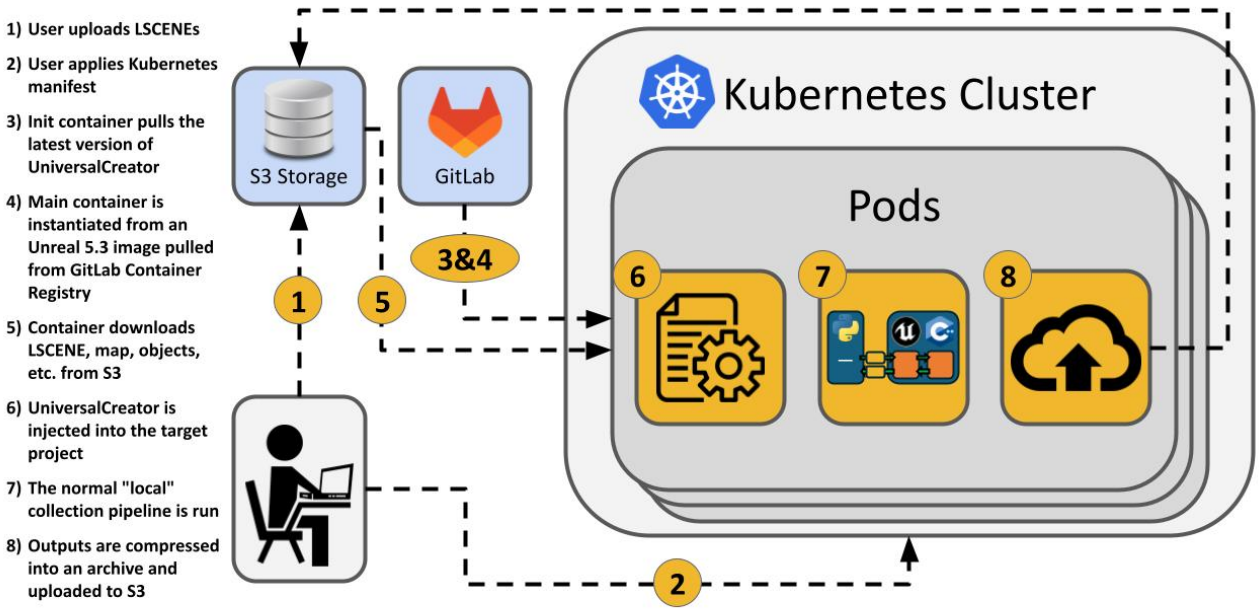

Figure 4. Automated collection pipeline as deployed on the National Research Platform (NRP) Nautilus HyperCluster.

4. AUTOMATED CLOUD-ENABLED METHODOLOGY

To adapt our simulation methodology to run at scale and accommodate our numerous desired environments and variations, we leverage the power of the National Research Platform (NRP) Nautilus HyperCluster. This provides us with an otherwise impossible quantity of simultaneously-available compute resources, including high-power GPUs. This cluster is managed by Kubernetes, a system where all of our software must be deployed as one or more "pods" (ephemeral environments containing one or more Docker containers with shared storage and network resources). We do this by writing a manifest in JSON or YAML which specifies our desired hardware requirements, external storage mounts, and software images. This introduces some new engineering complexities which we must address. See Figure 4.

The software that runs inside of a Kubernetes pod is supposed to be pre-packaged within a container image. Fortunately, there is an official Docker Image for Unreal Engine made available by Epic Games on the GitHub Container Registry. This base image includes the engine itself, along with the NVIDIA drivers required to use GPU acceleration, but obviously does not include any of our custom software or other dependencies. The idiomatic approach to fixing this is to extend this image to create a new, custom image with our full software suite bundled together and ready to run. However, the base image is already more than 18 GB in size, and we found it prohibitively slow to rebuild that image every time we needed to make an update to our software. So instead, we use an "init container," a Kubernetes feature which allows us to do some setup work in a separate container as the pod is initializing and before our main container is launched. Our init container runs a lightweight (approximately 32 MB) image which has git preinstalled to pull the very latest version of our software directly from GitLab version control. This repository contains all of our software, alongside a small set of scripts which are used by the main container to "patch" itself at runtime to be ready-to-use. These patches include installing and starting an X virtual frame buffer (seemingly required for the Unreal Editor to launch properly), installing rclone (required for transferring data to/from S3 storage), installing any Python libraries used by our API client, and transferring ownership of all Kubernetes storage mounts from "root" to "ue4" (the default user included within the base container provided by Epic Games).

With these images and scripts prepared, we are now ready to assemble our bulk generation workflow. An S3-compatible storage bucket is staged with all of the project files for each of the environments that we will be collecting from. In a separate bucket, we have a set of LCAP/LSCENE JSON files defining every experiment that we wish to perform. This includes an axis-aligned bounding box specifying the region that the simulated drone flight should fly within, the rotation of the camera (expressed as a uniform distribution rather than a single value), the FOV/resolution of the camera, as well all required information about our map and the objects that we want to be added to it. These JSON files could be easily written manually for a small number of experiments, but in our case we instead used a Python script to quickly generate all of our desired variations.

We then need one or more Kubernetes manifests to define all of the pods required to run these experiments. In our case, we use exactly one pod per experiment. This ensures a clean, reproducible environment for each experiment and prevents any unforeseen side-effects from carrying over from a prior experiment. Within this manifest, we must pass CLI parameters to each pod so that it knows which Unreal project (environment) to use and which LSCENE/LCAP file to use. We also specify a destination for the output imagery to be copied to once the simulation terminates. In our case, we used yet another S3 bucket with subdirectories for each experiment. This allows for performant retrieval of the results from outside of the cluster.

5. DATA COLLECTION ENVIRONMENTS

As outlined in Section 1, the primary purpose of this dataset is to provide an aerial dataset that contains sufficient quantity and variety to accurately train and evaluate a machine learning model to perform dense prediction tasks against a high-fidelity ground truth. To this end, we collect from multiple simulated environments. The careful selection of these environments allows for a tailored balance of foliage and man-made structures. In total, we utilize six different environments featuring a mix of urban and rural settings. Although not exhaustive, this provides a variety of backgrounds and terrain features upon which various objects such as people and vehicles can be placed using LSCENE.

Each base environment can be modified to appear differently with various controls and parameter settings. Through the use of the Ultra Dynamic Sky[21] plugin, which includes Ultra Dynamic Weather, we can easily control environmental variables such as fog density and time of day, using the real-world positions of the Sun, Moon, and stars, in addition to volumetric clouds, accurate cloud shadows, etc. to generate impressively photorealistic results as a drop-in augmentation to an existing environment. By using combinations of these variations, we drastically expand the variety of scenes in our dataset.

Our first publicly released dataset contains simulated flights from six environments with a mix of rural and urban terrain. Each map is sampled at three distinct altitudes above ground level, three different times of day, two different fog levels, and two different camera pitches (nadir and 45 degrees). This results in **216 total simulated flights**. Each flight lasts for 1000 frames, thus the combined dataset contains **216,000 total frames** in our first release. A summary of these variations is given in Table 1.

Table 1. Breakdown of experimental conditions and total combinations.

Factor	**Variations**
Environments	6 distinct environments (rural and urban)
Altitudes	3 (Low, Medium, High)
Time of Day	3 (Morning, Noon, Night)
Fog Level	2 (Clear, Foggy)
Camera Angle	2 (Nadir, 45-degree)
Total Combinations	$6 \times 3 \times 3 \times 2 \times 2 = 216$
Total Frames	$216 \times 1000 = 216,000$

All simulated flights include randomly-placed people and vehicles to serve as ground-truth for object detection. The flight pattern is a smooth interpolation between a random selection of waypoints inside of a 2D bounding box with the z-coordinate set to a fixed height above ground level. These choices are simple and sufficient for our current use, but more variety may be worthwhile to investigate for future work.

6. DEPTH AND OBJECT DETECTION AND LOCALIZATION EXAMPLES

To demonstrate the viability of this methodology for producing data of sufficient quality for inference evaluation, we ran some of our data samples through popular algorithms for object detection, dense depth prediction, and semantic segmentation. Figure 5 shows four sample images from the HiFiAerial dataset. Each sample is shown as a color image with ground-truth bounding boxes for certain objects placed in the scene. Here, we focus only on people and vehicle object types. These examples show the different types of backgrounds and relative camera angles used in the dataset.

One potential application of this dataset is for object detection. The YOLOv12 detector[22] is a state-of-the-art model in this domain that uses an attention mechanism to improve over previous methods. Figure 6 shows the sample images from Fig. 5 with declarations from YOLOv12-X declarations instead of ground-truth. These results are obtained without specific fine-tuning on the HiFiAerial dataset and show promising performance out of the box. The labels are the default labels provided by YOLO and could be adapted for specific applications.

Another possible application of the HiFiAerial dataset is depth estimation. The Depth Anything V2 model[1] is trained to estimate relative depth from a single input image. Figure 7 shows the output of the model compared to the ground-truth depth provided by Unreal Engine. It should be noted that although relative depth prediction is quite good, accurate metric depth prediction is still challenging, particularly from many of the aerial perspectives in our dataset.

One last application we consider is object segmentation. The Segment Anything model[23] provides object IDs for identified objects in an image. Figure 8 shows the output of the model for the images in Fig. 5. Each object is shown with a random color and can be masked out to identify only a particular instance. Due to the complexity of providing accurate ground-truth segmentation labels, we do not currently include these in the HiFiAerial dataset, although future iterations of the dataset may endeavor to add this.

Figure 5. Original color images with ground truth bounding boxes overlayed.

Figure 6. Color images with YOLOv12-X declarations overlayed.

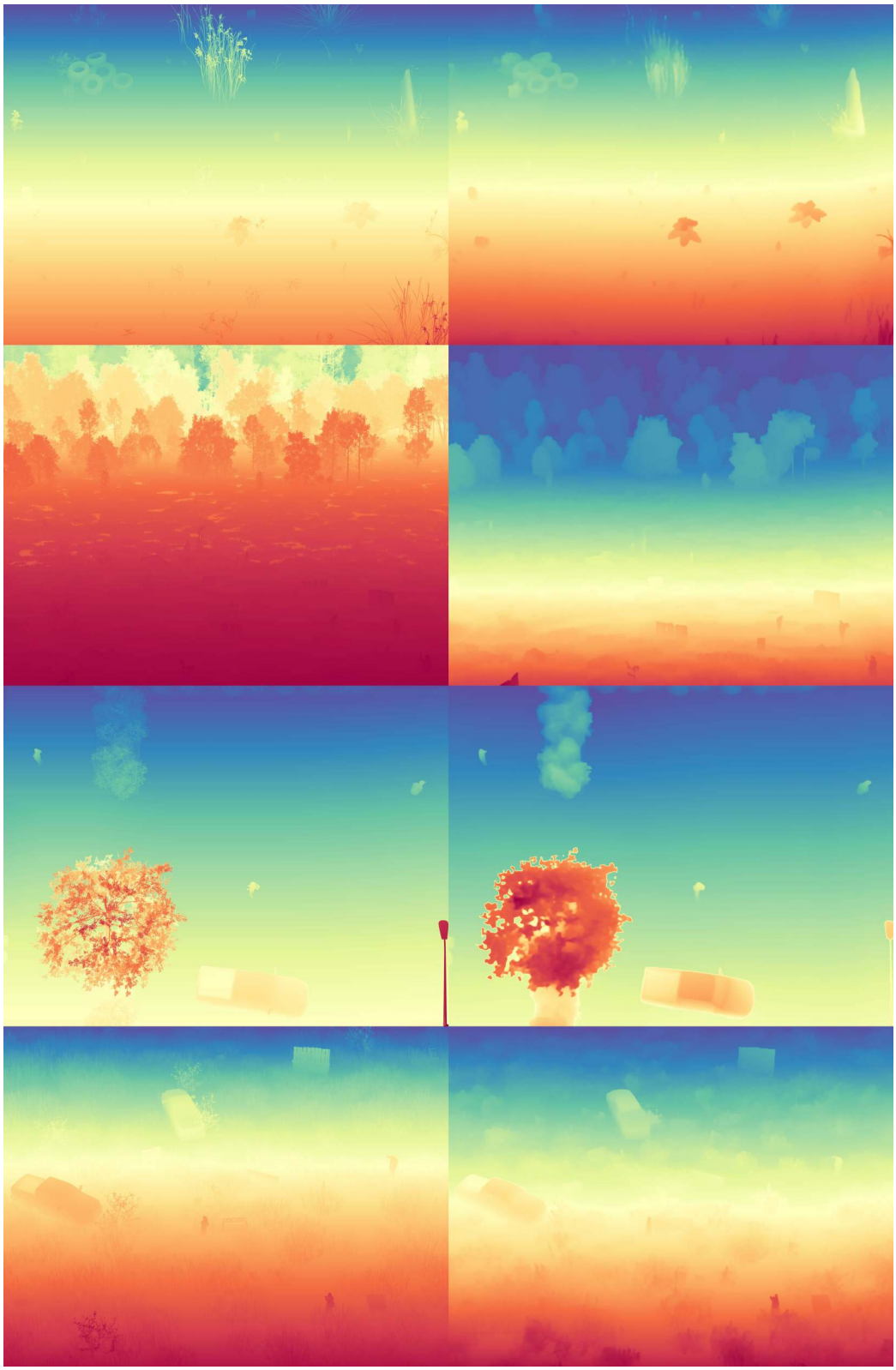

Figure 7. Depth maps derived from the color images shown in Figure 5. Left is ground truth produced by Unreal Engine; right is Depth Anything V2 prediction.

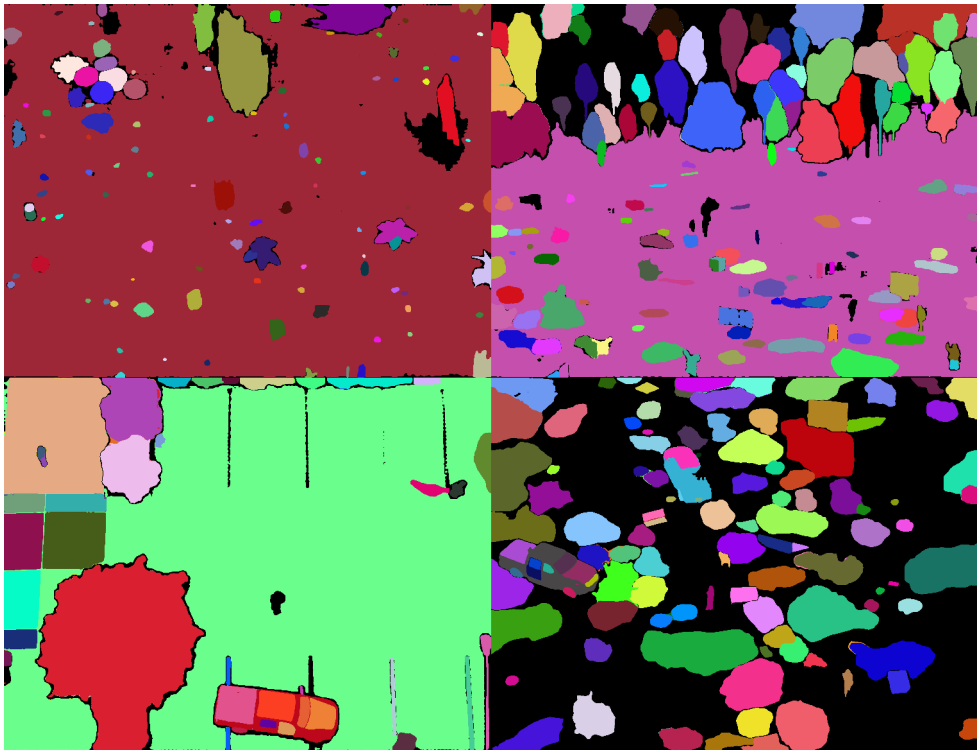

Figure 8. Segmentation maps derived from the color images shown in Figure 5 using Segment Anything. Random colors are applied to each mask.

7. CONCLUSION

To conclude, there has previously been an insufficient availability of publicly-available aerial imagery datasets of sufficient quality, both in photorealism and in ground-truth precision. With this important domain being underrepresented, many state-of-the-art models suffer reduced performance in these contexts, particularly at high altitudes with near-nadir views.

Given the extreme challenges associated with collecting this sort of data in the real world, we turn to high-fidelity simulation as a scalable alternative with vastly more precise ground-truth labeling. After demonstrating the viability of this technology on a local machine, we scale the project up to leverage cloud computing resources to generate imagery in parallel to quickly produce data in quantities that would otherwise be overwhelmingly impractical to generate manually on a single machine.

Herein, we present HiFiAerial, a synthetic aerial dataset of unprecedented quality for a diverse array of computer vision tasks, including object detection and dense prediction. We demonstrate that this type of data can be affordably generated at large scale by leveraging cloud computing to run a modern render engine with an incredible level of fidelity. It is our hope that this data, and the methodology used to create it, will help satiate the demand for high-fidelity ground truth for training machine learning models to perform dense prediction and object detection tasks from aerial contexts where data is otherwise not publicly available in sufficient quantity or quality.

ACKNOWLEDGMENTS

Computing resources for this research have been supported by the NSF National Research Platform and by NSF OAC Award #2322218 (GP-ENGINE).

REFERENCES

[1] L. Yang, B. Kang, Z. Huang, Z. Zhao, X. Xu, J. Feng, and H. Zhao, "Depth anything V2," 2024. [Online]. Available: https://arxiv.org/abs/2406.09414

[2] R. Lopez-Campos and J. Martinez-Carranza, "ESPADA: Extended synthetic and photogrammetric aerial-image dataset," *IEEE Robotics and Automation Letters*, pp. 1–1, October 2021.

[3] M. Roberts, J. Ramapuram, A. Ranjan, A. Kumar, M. A. Bautista, N. Paczan, R. Webb, and J. M. Susskind, "Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding," in *International Conference on Computer Vision (ICCV) 2021*, 2021.

[4] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, arXiv:1512.02134. [Online]. Available: http://lmb.informatik.uni-freiburg.de/Publications/2016/MIFDB16

[5] Y. Yao, Z. Luo, S. Li, J. Zhang, Y. Ren, L. Zhou, T. Fang, and L. Quan, "BlendedMVS: A large-scale dataset for generalized multi-view stereo networks," *Computer Vision and Pattern Recognition (CVPR)*, 2020.

[6] N. Karaev, I. Rocco, B. Graham, N. Neverova, A. Vedaldi, and C. Rupprecht, "DynamicStereo: Consistent dynamic depth from stereo videos," *CVPR*, 2023.

[7] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *European Conf. on Computer Vision (ECCV)*, ser. Part IV, LNCS 7577, A. Fitzgibbon et al. (Eds.), Ed. Springer-Verlag, Oct. 2012, pp. 611–625.

[8] D. Hernandez-Juarez, L. Schneider, A. Espinosa, D. Vazquez, A. M. Lopez, U. Franke, M. Pollefeys, and J. C. Moure, "Slanted stixels: Representing san francisco's steepest streets," in *British Machine Vision Conference (BMVC), 2017*, 2017.

[9] Y. Cabon, N. Murray, and M. Humenberger, "Virtual KITTI 2," 2020.

[10] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig, "Virtual worlds as proxy for multi-object tracking analysis," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2016, pp. 4340–4349.

[11] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3354–3361.

[12] W. Wang, D. Zhu, X. Wang, Y. Hu, Y. Qiu, C. Wang, Y. Hu, A. Kapoor, and S. Scherer, "TartanAir: A dataset to push the limits of visual SLAM," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[13] Q. Wang, S. Zheng, Q. Yan, F. Deng, K. Zhao, and X. Chu, "IRS: A large naturalistic indoor robotics stereo dataset to train deep models for disparity and surface normal estimation," 2021. [Online]. Available: https://arxiv.org/abs/1912.09678

[14] C. Rui, G. Youwei, Z. Huafei, and J. Hongyu, "A comprehensive approach for UAV small object detection with simulation-based transfer learning and adaptive fusion," 2021.

[15] "Infinite Studio," https://infinitestudio.software/, (Accessed: 25 March 2023).

[16] J. Kerley, A. Fuller, M. Kovaleski, P. Popescu, B. Alvey, D. T. Anderson, A. Buck, J. M. Keller, G. Scott, C. Yang, K. E. Yasuda, and H. A. Ryan, "Procedurally generated simulated datasets for aerial explosive hazard detection," in *Chemical, Biological, Radiological, Nuclear, and Explosives (CBRNE) Sensing XXIII*, J. A. Guicheteau and C. R. Howle, Eds., vol. 12116, International Society for Optics and Photonics. SPIE, 2022, p. 1211611. [Online]. Available: https://doi.org/10.1117/12.2618798

[17] J. Kerley, D. T. Anderson, B. Alvey, and A. Buck, "How should simulated data be collected for AI/ML and unmanned aerial vehicles?" in *Synthetic Data for Artificial Intelligence and Machine Learning: Tools, Techniques, and Applications*, C. L. Howell, K. E. Manser, and R. M. Rao, Eds., vol. 12529, International Society for Optics and Photonics. SPIE, 2023, p. 125290J. [Online]. Available: https://doi.org/10.1117/12.2663717

[18] J. Kerley, D. T. Anderson, A. R. Buck, and B. Alvey, "Generating simulated data with a large language model," in *Synthetic Data for Artificial Intelligence and Machine Learning: Tools, Techniques, and Applications II*, K. E. Manser, C. L. Howell, R. M. Rao, and C. D. Melo, Eds., vol. 13035, International Society for Optics and Photonics. SPIE, 2024, p. 1303508. [Online]. Available: https://doi.org/10.1117/12.3013460

[19] J. Kerley, D. T. Anderson, and B. Alvey, "LLM-Based TypeScript Generation and Asset Management for Procedural Synthesis of Scenes and Data for AI." SPIE, 2025.

[20] "Unreal Engine Fab," https://www.fab.com/, (Accessed: 20 March 2025).

[21] "Ultra Dynamic Sky," https://www.fab.com/listings/84fda27a-c79f-49c9-8458-82401fb37cfb, (Accessed: 2 April 2025).

[22] Y. Tian, Q. Ye, and D. Doermann, "YOLOv12: Attention-centric real-time object detectors," 2025. [Online]. Available: https://arxiv.org/abs/2502.12524

[23] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment anything," *arXiv:2304.02643*, 2023.