Synthetix: Pipeline for Synthetic Geospatial Data Generation

Sagar Verma, Siddharth Gupta, Kavya Gupta

Synthetix: Pipeline for Synthetic Geospatial Data Generation

Sagar Verma
Granular AI
sagar@granular.ai

Siddharth Gupta
Granular AI
sid@granular.ai

Kavya Gupta
Granular AI
kavya.gupta100@gmail.com

Abstract

Remote sensing is crucial in various domains, such as agriculture, urban planning, environmental monitoring, and disaster management. However, acquiring real-world remote sensing data can be challenging due to cost, logistical constraints, and privacy concerns. To overcome these limitations, synthetic data has emerged as a promising approach. We present an overview of the use of synthetic data for remote sensing applications.In this regard, we address three conditions that can drastically affect the optimization of computer vision algorithms: lighting conditions, fidelity of the 3D model, and resolution of the synthetic imagery data. We propose a highly configurable pipeline called Synthetix as part of our GeoEngine platform for synthetic data generation. Synthetix allows us to quickly create large amounts of aerial and satellite imagery under varying conditions, given a few samples of 3D objects on real-world scenes. We demonstrate our pipeline's effectiveness by generating 3D scenes from 35 real-world locations and utilizing these scenes to generate different versions of datasets and answer the three questions. We conduct an in-depth ablation study and show that considering different environments and weather conditions increases the reliability and robustness of the deep learning networks.

1 Introduction

Synthetic data mimics the characteristics of real-world remote sensing imagery. It is created using computer graphics techniques, simulation models, or a combination of both. One of the primary advantages of synthetic data is its ability to provide large-scale and diverse datasets, enabling comprehensive training of machine learning algorithms for remote sensing tasks. Synthetic data can simulate various environmental conditions, sensor specifications, and imaging scenarios, allowing researchers and practitioners to assess the performance of algorithms under different settings.

It's important to note that synthetic data has advantages and should be used with real-world data to ensure models can handle the complexities and variations in actual environments. Combining both data types helps balance controlled scenarios and real-world adaptability in computer vision tasks.

Over the last decade, computer vision research and the development of new algorithms have been driven largely by permissively open datasets. Dataset such as ImageNet(2), MS-COCO(10), PAS-CALVOC (3), QFabric(20) among others remain critical drivers for advancement. Convolutional neural networks (CNNs), currently the leading class of algorithms for most vision tasks, require a large amount of annotated observations. However, developing such datasets is often manually intensive, time-consuming, and costly. An alternative approach to manually annotating training data is to create computer-generated images and annotations. After creating realistic 3D environments, one can generate thousands of images at virtually no cost. Such data is effective for augmenting and replacing real data, thus reducing the burden of dataset curation (4). Synthetic datasets continue to be

Preprint. Under review.

developed and have been notably helpful in various domains, including autonomous driving (1; 9), optical flow (11), facial recognition (8), amodal analysis (5) and domain adaptation (14).

Data collection in disasters is tricky, and choosing the right data source is essential. The two most favorable data sources are satellites and UAVs. Satellite sources can suffer from bad weather, which is highly probable during flooding. UAVs can be deployed in such cases but can not cover large areas. Data quality is often poor in post-disaster scenarios due to bad weather conditions, deformation of objects, submerged objects, and clutter. UAVs have more mobility freedom and can capture data closely with high quality. Annotation quantity is constrained due to limited data collected and the fact that annotating post-disaster images compared to pre-disaster takes more time and effort to maintain quality. Satellites can capture data for a particular region for a more extended period than a UAV. However, this can only be achieved if weather conditions are good. On the other hand, UAVs have limited flight time and the region they can capture. Using multiple UAV(s) can remove the cons associated with flight time and the area captured. Annotation quality suffers due to inadequate data quality, which can add more noise to the ground truth. Satellite-based data for both pre/post-disaster scenarios is more challenging to annotate than UAV-based data owing to resolution(16), identifying clutter, and marking object classes with precision. UAVs as a data source can serve better in evaluating flood-based disaster scenarios. Satellite imagery has become integral to the functionality of foundational models, unlocking a myriad of applications across various domains (12; 6).

Recent advances in deep learning-based algorithms, especially in computer vision, have shown the potential in enabling rapid downstream analysis of such imagery over large areas. Deep learning-based methodologies are well positioned to assist humanitarian efforts due to their scalability, explainability, and robustness to variable conditions (18; 13; 17; 7). In the short term, such models can provide decision-makers with enhanced clarity on ground conditions. Moreover, these models can be extended to support autonomous assistive solutions. One of the critical requirements for generating such deep learning-based solutions is the availability of high-quality training data, which poses a challenge due to the inherent shortage of post-flooding data. Attempting to train with insufficient data can lead to the model learning a poor representation due to the noise in the annotations. Generating new samples from existing data with negligible label noise can enable us to overcome such challenges.

2 Proposed Pipeline

Currently, Synthetix is an off-platform tool, we aim to add to our GeoEngine (19) platform in the future and make it compatible with other features of our platform and datasets (15). Synthetix tool requires the following as input from the user,

- Bounding box of a location anywhere on earth.
- Specify all the satellite image resolutions for which synthetic models should be rendered.
- Specify objects that have to be detected. Currently, we support vehicles, buildings, trees, roads, railways, and HVAC units. HVAC units were added specifically for the challenge.
- Map each selected object to where it should be placed. The locations are natural terrain, road, railway, and rooftop.
- Specify the number of samples per object, the mean size of the object in meters, and the standard deviation of the object size.
- Specify off-nadir angles.
- Specify cloud coverages in percentages.

Once the user provides the above inputs, Synthetix uses the total number of samples that will be generated based on the provided parameters and the total time it will take to generate all the samples. Users can then change some of the parameters or confirm the generation process. The end-to-end pipeline for synthetic data generation is shown in Figure 1. Synthetix utilizes Optics[1] from GeoEngine platform as shown in Figure 2.

[1] https://engine.granular.ai/organizations/granular/image-sets

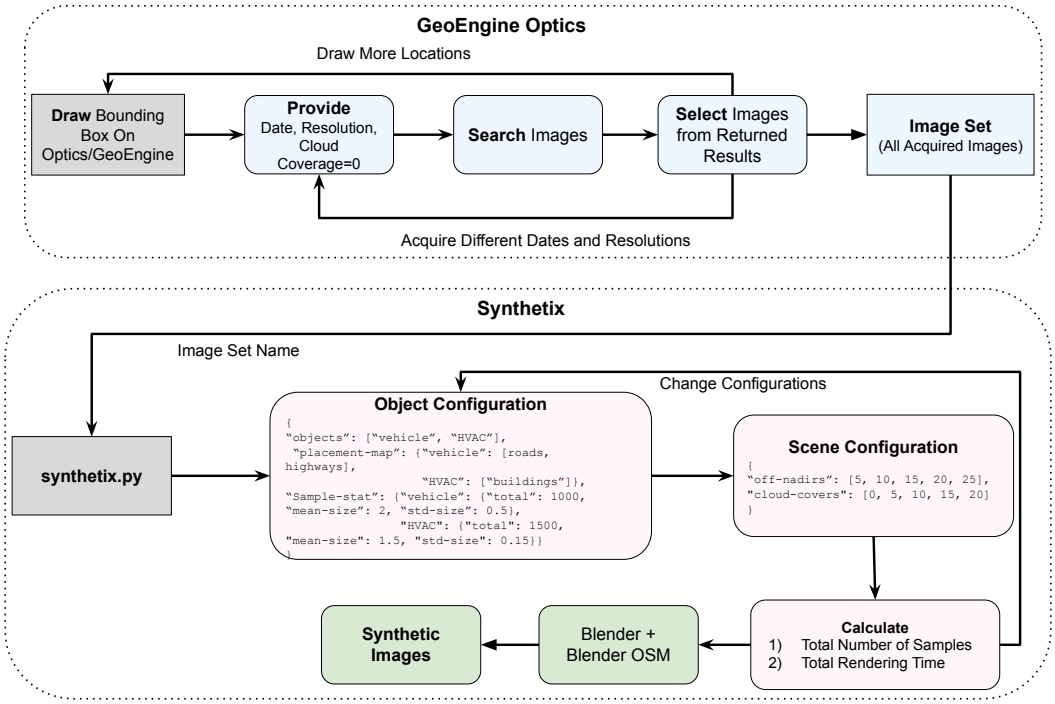

Figure 1: **Synthetix Pipeline:** Upper block shows image acquisition on GeoEngine Optics, and the lower block shows using synthetix.py with object and scene configuration to generate synthetic data for the acquired images from Optics.

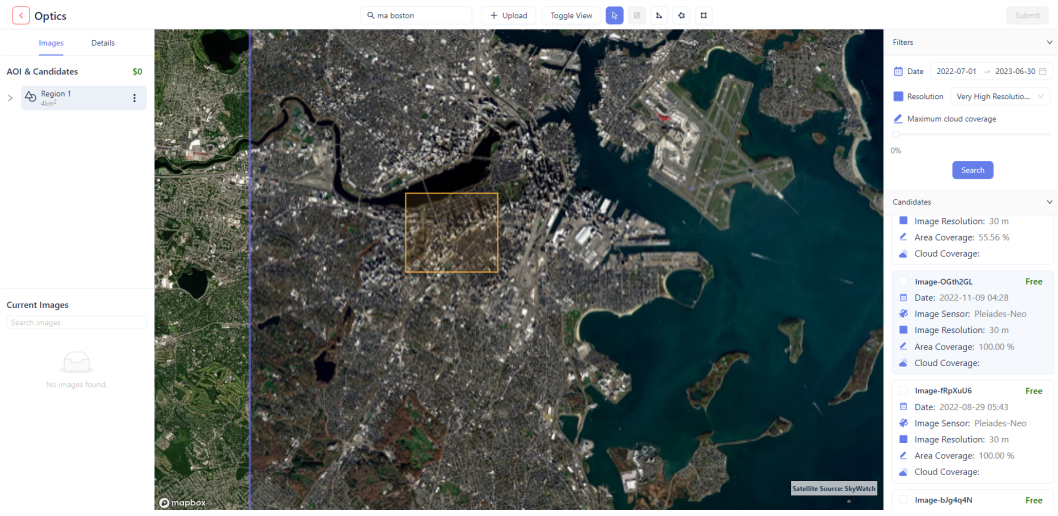

Figure 2: Acquiring satellite images with zero cloud cover and highest resolution on Optics.

3 Results

For all our data generation and deep learning training experiments we utilize following:

3.1 System Information

We use Microsoft Windows 11 Pro (Version 10.0.22621 Build 22621) running on an AMD Ryzen 9 3900 12-Core Processor, 3094 Mhz, 12 Cores processor, and 64 GB RAM for synthetic data generation. The system is also equipped with an NVIDIA RTX 3060 12GB GPU. For synthetic data generation, we use Blender 3.5.1 with Blender-OSM add-on. We use Blender Python API to connect Python 3.7 to Blender. Our data generation pipeline entirely runs on Python.

3.2 Deep Learning Training

We utilize a variety of hardware for model training based on availability and model requirements. Our local workstation is an Ubuntu 22.04 LTS system running on dual cpu Intel(R) Xeon(R) Silver 4208 CPU @ 2.10GHz, 16 cores processors, and 96 GB RAM. The system is equipped with two NVIDIA RTX 3090 24GB GPUs. We also utilize AWS EC2 instances with GPU. All the AWS instances use Deep Learning AMI GPU PyTorch 2.0.0 (Ubuntu 20.04) 20230530. We use three different types of GPUs based on model and dataset sizes:

- g3.16xlarge with 4 x M60 GPUs (4x8 GB), 64 cores, and 488 GB RAM.
- p3dn.24xlarge with 8 x V100 GPUs (8x32 GB), 96 cores, and 768 GB RAM
- p4d.24xlarge 8 x A100 GPUs (8x40 GB), 96 cores, and 1152 GB RAM

For all our experiments, we use GeoLibs v0.0.14 [2], a multipurpose geospatial model training library by Granular AI, which uses PyTorch 2.0.0+cu117 and works on Python 3.7 onwards.

4 Datasets

Using Synthetix, we generate different datasets to evaluate the pipeline. Table 1 shows details about five datasets generated by setting off-nadir angle, sun angle, cloud cover, quality of HVAC unit 3D model, and satellite image resolution. We first start with identifying 35 real-world locations. These locations were chosen from Brockton, Boston, Corpus Christi, Houston, Missouri City, Alvin, Lamesa, Framingham, San Diego, and Tucson. We then acquire 146 high-resolution Vexcel images (20cm) between 2019 and 2021. We also acquire WorldView-3 (30cm), Pléiades Neo (30cm), SkySat (50cm), and TripleSat (80cm) images for the exact locations and same dates. All locations cover a 4km-square area. We make sure to download images without any cloud cover to have perfect images to control cloud cover and sun angle in the synthetic data generation phase. Figure 3 shows an image acquired from WorldView-3 for Cambridge, Boston location.

- To make the "Nadir Dataset," we fixed the light source from the z-axis and used only 20cm resolution images. We use high-quality textured HVAC units and place them randomly on roofs. We utilize Blender-OSM to generate shapes for each building and project satellite images on top of all the buildings for realism. No artificial clouds were simulated in this case.
- "Sun Angle Dataset" requires putting a light source in the Blender scene at three different angles (30, 60, and 90) degrees.
- In the case of the "Cloud Cover Dataset," we keep everything else fixed and introduce clouds in the scene at an altitude of 500m. The number of samples generated and rendering time can be seen on the Table 1.
- In the case of "Resolution Dataset," we use images acquired from four different sources as shown in Figures 4 and 5. It takes less rendering time as we go for lower resolution images as the image size and the polygon size also decreases. We gain the speedup by removing very small polygons (objects) like vehicles from the scene to speed up the rendering. We also adjust the size of the HVAC unit object as per the image's resolution.

[2] https://pypi.org/project/geolibs/

Dataset Name	Off-Nadir Angle	Sun Angle	Cloud Cover (%)	3D Quality	Resolution (cm)	Images	Render Time (Hours)
Nadir Dataset	0	90	0	Textured	20	315	3.2
	27	90	0	Textured	20		
	54	90	0	Textured	20		
Sun Angle Dataset	0	90	0	Textured	20	315	3.9
	0	60	0	Textured	20		
	0	30	0	Textured	20		
Cloud Cover Dataset	0	90	0	Textured	20	420	5.8
	0	90	5	Textured	20		
	0	90	10	Textured	20		
	0	90	15	Textured	20		
Resolution Dataset	0	90	0	Textured	20	420	4.1
	0	90	0	Textured	30		
	0	90	0	Textured	50		
	0	90	0	Textured	80		
Fidelity Dataset	0	90	0	Textured	20	315	2.4
	0	90	0	Image Plane	20		
	0	90	0	Colored Cube	20		

Table 1: Different datasets generated using our platform Synthetix.

Figure 3: Image set on GeoEngine platform showing a 30cm resolution image from Cambridge, Boston.

- Regarding fidelity of the HVAC unit 3D models, we use a highly detailed textured model of the HVAC unit, a cube with an image plane showing images of the HVAC unit, and just a colored cube that depicts the HVAC unit as a box, shown in 6.

In all the datasets, in every image, we generate at max 1000 random HVAC units with a mean size of 2.5m and a standard deviation of size of 0.35m. We also set the spacing of at least 4m between HVAC units. HVAC units are only placed on roofs. At the moment, we have not made any logic around excluding non-planar roofs. In total, we generate 435K HVAC units across all the datasets.

5 Experiments and Results

We split each of the five datasets into training and validation sets with a ratio of 80:20. We then train Mask-RCNN to understand the utility of synthetic data and answer the three questions.

5.1 Inclusion of shadows and varying the sun angle

It can be observed in Table 2 that when a model trained on a particular sun-angle and inferenced on the validation set of the same sun-angle it performs better than when it is inferenced on the validation

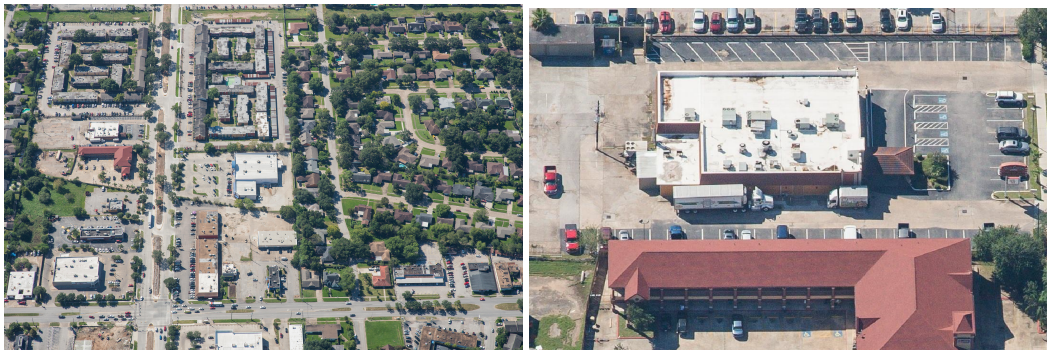

Figure 4: **Houston, Texas.** Left: 20cm resolution image. Right: Zoomed into a roof with HVAC units.

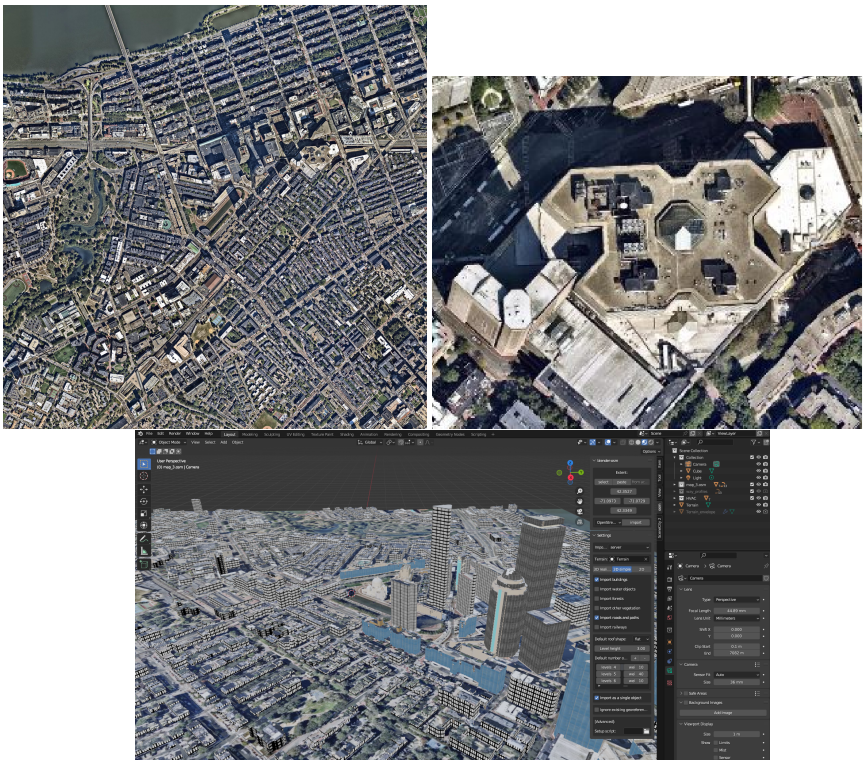

Figure 5: **Cambridge, Boston, MA.** Left: 50cm resolution image. Right: Zoomed into a roof with HVAC units. Bottom: 3D model of this image in Blender viewport.

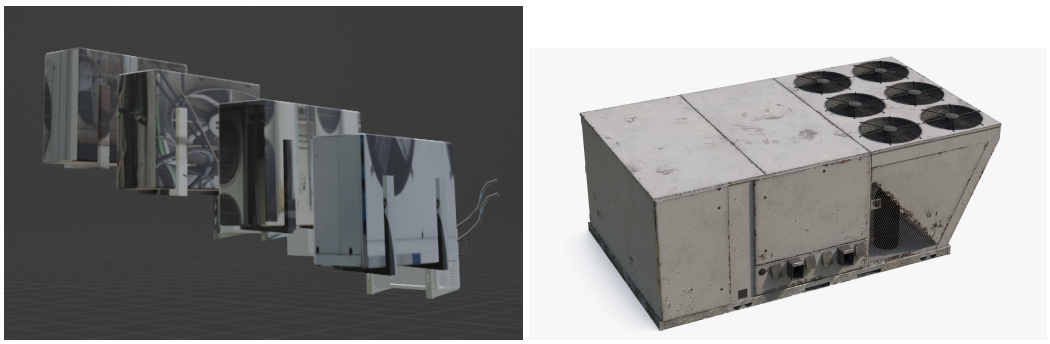

Figure 6: Left: Planar image HVAC model. Right: High quality HVAC model.

Model	Training sun-angle	Validated On								
		90			60			30		
		mAP	mAP50	AR	mAP	mAP50	AR	mAP	mAP50	AR
Mask-RCNN	90	**56.7**	**72.8**	**61.2**	55.2	71.1	59.2	49.2	65.3	50.3
	60	53.2	68.4	54.1	**56.1**	**70.8**	**57.8**	42.1	50.3	43.1
	30	49.2	57.8	52.3	51.4	62.3	53.5	**53.5**	**61.8**	**55.5**

Table 2: The effect of models trained on images with one sun-angle when infered on another. mAP is mean average precision and AR is average recall.

set of other two sun-angles. At the same time as the light exposure goes, the models' accuracy also decreases. This indicates that the sun angle matters when we want to cover all base variations.

5.2 Fidelity of the 3D model

Our experiments found that when low-fidelity HVAC models are used, the models fail to discern between roof and HVAC units. This is because the roof color matches the dark-colored cube in many cases. The model cannot differentiate between other generic objects and the HVAC unit without any realistic texture. In the case of 3D models where cube faces are covered with HVAC images, the model can learn as it can see fans and wires in the image and use those to differentiate visual cues. In the case of detailed, realistic, high fidelity HVAC units, the model performed quite well.

5.3 Resolution of the synthetic imagery data

In the case of resolution experiments, we found that as we increase the resolution of satellite images, we get bigger and better quality HVAC unit 3D models; we are also able to place them accurately on roof buildings at the highest resolution (20cm). Due to this, the ability to detect them increases, and the Mask-RCNN gives the highest mAP.

6 Conclusion

This article covers a synthetic geospatial data generation tool called Synthetix by Granular AI. We have described the pipeline in detail and shown how it utilizes our existing platform to get satellite imagery. Then, using a single Python file and a few configurations, it generates many synthetic scenes with many variations in weather, environment, lighting, and other conditions. We demonstrate the usability of synthetic data generation and our pipeline by answering three main questions in vision algorithms. To overcome the challenges of unseen scene conditions, it is best to utilize Synthetix to generate as many samples as possible to pre-train deep-learning models. We want to extend this work to include more objects and shapes and completely platformize the Synthetix tool on GeoEngine.

References

[1] Emanuele Alberti, Antonio Tavera, Carlo Masone, and Barbara Caputo. Idda: A large-scale multi-domain dataset for autonomous driving. *IEEE Robotics and Automation Letters*, 2020.

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[3] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The Pascal Visual Object classes (voc) challenge. *IJCV*, 2010.

[4] Shubham Goswami, Sagar Verma, Kavya Gupta, and Siddharth Gupta. Floodnet-to-floodgan: Generating flood scenes in aerial images. 2022.

[5] Yuan-Ting Hu, Hong-Shuo Chen, Kexin Hui, Jia-Bin Huang, and Alexander G Schwing. Sail-vos: Semantic amodal instance level video object segmentation-a synthetic dataset and baselines. In *CVPR*, 2019.

[6] Hitesh Jain, Sagar Verma, and Siddharth Gupta. Investigating large vision model training challenges on satellite datasets. In *InGARSS 2023*, 2023.

[7] Benjamin Kiefer, Matej Kristan, Janez Perš, Lojze Žust, Fabio Poiesi, and Others Andrade. 1st workshop on maritime computer vision (macvi) 2023: Challenge results. In *WACVW*, 2023.

[8] Adam Kortylewski, Bernhard Egger, Andreas Schneider, Thomas Gerig, Andreas Morel-Forster, and Thomas Vetter. Analyzing and reducing the damage of dataset bias to face recognition with synthetic data. In *CVPR Workshops*, 2019.

[9] Wei Li, CW Pan, Rong Zhang, JP Ren, YX Ma, Jin Fang, FL Yan, QC Geng, XY Huang, HJ Gong, et al. Aads: Augmented autonomous driving simulation using data-driven algorithms. *Science robotics*, 2019.

[10] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.
[11] Simon Meister, Junhwa Hur, and Stefan Roth. Unflow: Unsupervised learning of optical flow with a bidirectional census loss. In *AAAI*, 2018.
[12] Akash Panigrahi, Sagar Verma, Matthieu Terris, and Maria Vakalopoulou. Have Foundational Models Seen Satellite Images? In *IGARSS*, 2023.
[13] M. Papadomanolaki, S. Verma, M. Vakalopoulou, S. Gupta, and K. Karantzalos. Detecting urban changes with recurrent neural networks from multitemporal sentinel-2 data. In *IGARSS*, 2019.
[14] Swami Sankaranarayanan, Yogesh Balaji, Arpit Jain, Ser Nam Lim, and Rama Chellappa. Learning from synthetic data: Addressing domain shift for semantic segmentation. In *CVPR*, 2018.
[15] Hal Shin, Natanael Exe, Ujwal Dutta, Tanka Raj Joshi, Sagar Verma, and Siddharth Gupta. Europa: Increasing accessibility of geospatial datasets. In *IGARSS*. IEEE, 2022.
[16] Matthieu Terris and Sagar Verma. Investigating model robustness against sensor variation. In *IGARSS*, 2023.
[17] Sagar Verma and Kavya Gupta. Post Wildfire Burnt-up Detection using Siamese UNet. In *ECML PKDD* , 2023.
[18] Sagar Verma, Siddharth Gupta, and Kavya Gupta. Aligning Geospatial AI for Disaster Relief with The Sphere Handbook. 2022.
[19] Sagar Verma, Siddharth Gupta, Hal Shin, Akash Panigrahi, Shubham Goswami, Shweta Pardeshi, Natanael Exe, Ujwal Dutta, Tanka Raj Joshi, and Nitin Bhojwani. Geoengine: A platform for production-ready geospatial research. In *CVPRD*, pages 21416–21424, June 2022.
[20] Sagar Verma, Akash Panigrahi, and Siddharth Gupta. Qfabric: Multi-task change detection dataset. In *CVPR*, 2021.