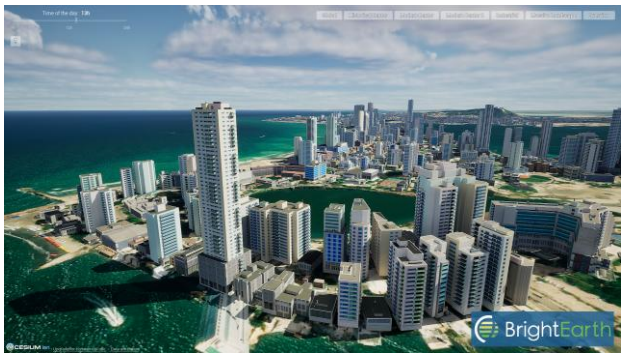
AI-Powered Mapping from Space: Custom 3D Maps on Demand

LuxCarta is developing LxGenAIEarthMapper, a generative AI assistant that transforms satellite imagery into interactive, user-customized maps. Leveraging LLMs and VLMs, the system extracts features like roads, buildings, and vegetation on demand, enabling rapid, semantic-rich 3D mapping through LuxCarta's BrightEarth.ai platform.



From Imagery to Insight – Instantly

Converts raw satellite imagery into detailed 3D geospatial outputs with user-defined semantics and structure.

Interactive Mapping with AI Assistants

Empowers users to extract the exact data they need directly from imagery — fast, flexible, and scalable.

Applications

- **Urban Planning:** Generates accurate, up-to-date maps for infrastructure and development.
- **Telecommunications:** Produces detailed terrain and feature maps for 5G and network planning.
- **Disaster Response:** Enables fast post-event mapping to guide emergency operations.

Key Features

- **LLM + VLM Integration:** Combines language and vision models to understand both image content and user intent.
- **Human-in-the-Loop Refinement:** Keeps expert oversight in place to ensure accuracy and trust in results.
- **Global, On-Demand Coverage:** Generates high-resolution 3D data from any point on Earth's surface.

Team:
LuxCarta (Sacha Lepretre, LuxCarta AI Team)

Technical Experience

Multiple EC2 instances were used for training models on a large amount of image data (6TB+) due to the team's familiarity with server-based training (it was a lift-and-shift operation from on-premises).
They ran into several issues with EC2 instances for model training, including:

- Availability of powerful EC2 instances in the us-east-2 AWS region without an instance reservation (*p5 & g6e* instance families).
- Reserving an EC2 instance and being charged directly for the reservation, even with sufficient credits in the provided AWS account.
- Instances having to shut down due to hardware issues.

As a result of their EC2 experience, the LuxCarta team has started training models using SageMaker as it provides a task-based model training approach, at the expense of a slightly higher cost. On SageMaker, the *ml.p4de.24xlarge* instance provided the resources needed to continue model training.

Overall Experience

Generally positive experience on the development side and using the Amazon SageMaker SDK for training GenAI models. Successfully initiated and managed training jobs through the SageMaker SDK.

Challenges Encountered

- Difficulty streaming large datasets directly from Amazon S3 due to data size constraints and performance bottlenecks.
- Repeated transfer of large datasets to training instances was inefficient and time-consuming for each training job.
- Encountered issues getting instance quotas approved by AWS support, affecting the reliability of training job launches.

Solutions Implemented

- Integrated an Amazon FSx for Lustre file system:
  - Enabled efficient and fast data access during model training.
  - Significantly improved data handling and reduced job startup times.
- Shifted training workloads to the us-east-1 AWS Region:
  - Alleviated instance capacity constraints and improved instance availability, resulting in consistent training operations.

Observations

The Challenge confirms that GenAI for Geospatial is not only powerful for understanding geospatial data, but also for improving global digitalization: "We have built what we believe to be the first Vectorization GenAI engine."

Next Steps

This work enabled the team to create BrightEarth's *LxGenAIEarthMapper v1* for building extraction QC purposes. Next versions will include roads, vegetation, and enabling text assistant capabilities.

Customers

The solution speeds up manual QC by 2 to 4 times, attracting strong customer interest.

Development & DL Experience

LuxCarta worked on training deep learning models and developing the internal pipelines that support and deploy that research. Instances were used, both for development, debugging, and long training runs.

Specifically:

- Visual Studio Code's 'Remote SSH' and 'Dev Containers' features to deploy internal dependencies and build environment to a GPU instance and develop in that environment throughout the project.
- SSH, tmux, and Docker were used in a separate session to run long training runs in a reliable way, with a local-forwarding SSH tunnel from a local machine to the EC2 instance for *TensorBoard* (a webapp that lets the user watch training metrics and examples).

Experience

Once the AWS cloud environment was configured, compute instances ran reliably and without interruption. The Dev Container configuration, previously only used on the team's local servers, carried over to EC2 instances. Performance was very satisfactory considering the remoteness of the instance.

Challenges

There were some failures when connecting VSCode to Dev Containers when SSHing into the instance using the original, shared Linux user account, but it was confirmed those were VSCode issues, not on AWS' side.

Solutions Implemented

The team created a personal account on the instance (to avoid sharing e.g. git user.name and user.email settings as well as the SSH key, which identifies the user to GitHub), set up an SSH key on the GitHub account, and ran VSCode dev container images for all tasks. Since the instance was a standard Ubuntu system, the team encountered no issues during setup or operation.