

APPENDIX

A MORE DETAILS OF RENDERING ENGINES AND DATA RESOURCES

In this section, more information about the used rendering engines and data resources are detailed, with several high-quality examples illustrated in Fig. 1.



Figure 1: High-quality examples from different rendering engines and techniques, including several large cities such as Shanghai, Guangzhou, Los Angeles, Osaka, and etc., cover an area of over a hundred square kilometers in total. 3D GS provides five large campus scenes, further enhancing the diversity and realism of the data.

Unreal Engine. UE is a rendering engine capable of providing highly realistic interactive virtual environments. This platform has undergone five iterations, and each version features comprehensive and high-quality digital assets. In UE5, we meticulously select an official sample project named ‘City Sample’, which provides us with a large urban scene covering $25.3km^2$ and a smaller one covering $2.7km^2$. These scenes include a variety of assets such as buildings, streets, traffic lights, vehicles, and pedestrians. Besides, in UE4, we prepare six more high-quality scenes. Specifically, there are two large scenes showcasing the central urban areas of Shanghai and Guangzhou, covering areas of $30.88km^2$ and $58.56km^2$, respectively. The remaining four scenes are selected from AerialVLN (Liu et al., 2023). They have smaller areas for totally about $26.64km^2$. These scenes encompass a wide range of architectural styles, including both Chinese and Western influences, as well as classical and modern designs. Additionally, the UE4 engine allows us to make adjustments in scene time to achieve different appearances of scenes under varying lighting conditions.

AirSim is an open-source simulator, which provides highly realistic simulated environments for UAVs and cars. We integrate the AirSim plugin into UE4 to obtain image data easily from the perspective of a UAV. Since AirSim does not support UE5 and stopped updating in 2022, we use the UnrealCV (Weichao et al., 2017) plugin as an alternative for image acquisition in UE5. To realize a highly efficient data collection in simulated scenes, we modify the UE5 project to a C++ project, integrate the UnrealCV plugin, and package executables for multiple systems like Windows and Linux.

GTA V. It is an open-world game that is frequently used by computer vision researchers due to its highly realistic and dynamic virtual environment. The game features a meticulously crafted cityscape modeled after Los Angeles, encompassing various buildings and locations such as skyscrapers, gas stations, parks, and plazas, along with dynamic traffic flows and changes in lighting and shadows.

Script Hook V is a third-party library with the interface to GTA V’s native script functions. With the help of Script Hook V, we build an efficient and stable interface, which receives the pose information and returns accurate RGB images and lidar data. From the interface, we can control a virtual agent to collect the required data in an arbitrary pose and angle in the game.

Table 1: Different 3D GS Scenes

Campus Name	Images	Area
ECUST (Fengxian Campus)	12008	1.06 km ²
NWPU (Youyi Campus)	4648	0.8 km ²
NWPU (Changan Campus)	23798	2.6 km ²
SJTU (Minhang-East Zone)	20934	1.72 km ²
SJTU (Minhang-West Zone)	9536	0.95 km ²

Google Earth. It is a virtual globe software, which builds a 3D earth model by integrating satellite imagery, aerial photographs, and Geographic Information System (GIS) data. From this engine, we select four urban scenes covering a total area of 53.60km², *i.e.*, Berkeley, primarily consisting of traditional neighborhoods; Osaka, which features a mix of skyscrapers and historic buildings; and two areas with numerous landmarks: Washington, D.C., and St. Louis.

Google Earth Studio is a web-based animation and video production tool that allows us to create keyframes and set camera target points on the 2D and 3D maps of Google Earth. Using this functionality, we can quickly generate customized tour videos by selecting specific routes and angles. In order to efficiently plan the route, we develop a function that automatically draws the flight trajectory in Google Earth Studio according to the selected area and predefined photo interval.

3D Gaussian Splatting. As a highly realistic reconstruction method, hierarchical 3D GS (Kerbl et al., 2024) employs a hierarchical training and display architecture, making it particularly suitable for rendering large-scale areas. Due to these features, we use this method to reconstruct and render multiple real scenes. We utilize the DJI M30T drone as the data collection device, which offers an automated oblique photography mode, enabling us to capture a large area of real-world data with minimal manpower. Practically, we gathered data from five campuses across three universities, which are East China University of Science and Technology, Northwestern Polytechnical University, and Shanghai Jiao Tong University (referred to as ECUST, NWPU, and SJTU). These campus scenes include various types and styles of landmarks, such as libraries, bell towers, waterways, lakes, playgrounds, construction sites, and lawns. The detailed information for the five campuses is presented in Table 1.

SIBR (Bonopera et al., 2020) viewers is a rendering tool designed for the 3D GS project, enabling visualization of a scene from arbitrary viewpoints. The tool supports high-frame-rate scene rendering and provides various interactive modes for navigation. Building upon SIBR viewers, we developed an HTTP RESTful API that generates RGB images from arbitrary poses, simulating a UAV’s perspective.

B DETAILS OF 3D GS DATA COLLECTION

From the UAV’s perspective, choosing the appropriate shooting altitude poses a dilemma, *i.e.*, if the altitude is too low, the sparse point cloud generated during the initialization of the 3D GS reconstruction will be suboptimal, due to insufficient feature point matches between photos. In contrast, if the altitude is too high, the Gaussian reconstruction will result in an overly coarse training of details. After multiple attempts, the data collection plan using the M30T was determined as follows. For large-scale block scenes, oblique photography is performed at approximately twice the average building height using the default parameters of the M30T’s wide-angle camera, with a tilt angle of -45°. For landmark buildings with heights significantly different from the average height, additional targeted data collection is conducted at twice their height. This altitude setting can, to a certain extent, ensure both higher-quality point cloud initialization and Gaussian splatting training.

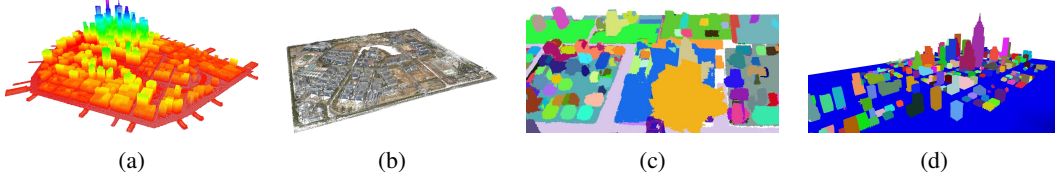


Figure 2: Results of our point cloud acquisition and semantic segmentation. (a) Rasterized sampling point cloud reconstruction. (b) Image-based point cloud reconstruction. (c) Semantic 3D scene segmentation. (d) Point cloud projection and contour extraction.

C UNIFIED INTERFACES

For data collection, we integrate all rendering engines and design three unified interfaces, *i.e.*, the agent movement interface, the lidar data acquisition interface, and the image acquisition interface, allowing an agent to move and perceive the environment within any scene.

- **Agent Movement Interface:** We design a *CoorTrans* module, which implements a customized pose transformation matrix and scaling function to unify all coordinate systems into a meter-based FLU (Front-Left-Up) convention. This interface enables precise agent positioning, ensuring consistency and facilitating automatic trajectory generation.
- **Lidar Data Acquisition Interface:** Lidar data is crucial for scene occupancy perception and essential for trajectory generation. Our platform supports different lidar data acquisition methods, including lidar sensor collection, depth map back-projection, and image feature matching. We develop a unified interface to integrate these methods and leverage the proposed *CoorTrans* module to align all data to the same FLU coordinate system.
- **Image Acquisition Interface:** We integrate HTTP RESTful and TCP/IP protocols to form a unified image request interface, allowing image data to be obtained from any location with flexible resolutions and viewpoints.

D RESULTS OF POINT CLOUD ACQUISITION AND SEMANTIC SEGMENTATION

This section presents results regarding the point cloud acquisition and scene semantic segmentation, as shown in Fig. 2.

Point Cloud Acquisition. OpenFly integrates data resources from different rendering engines. In order to obtain point cloud information from different scenes, we provide two point cloud acquisition methods. 1) For UE and GTAV scenes, we provide a tool that utilizes rasterized point cloud sampling and reconstruction to obtain a global point cloud. The results can be seen in Fig. 2a. 2) For 3D GS scenes, we use COLMAP (Schönberger & Frahm, 2016) to obtain relatively sparse point data from images, as shown in Fig. 2b. Although the point cloud generated by this method is relatively sparse, it provides sufficient coverage information.

Scene Semantic Segmentation. To meet the requirements of different data resources and different segmentation granularities, our OpenFly offers three methods for obtaining 3D semantic segmentation of scenes. Here, we present results of segmentation methods other than manual annotation. 1) Fig. 2c illustrates the semantic segmentation results based on the off-the-shelf 3D scene understanding method Octree-Graph (Wang et al., 2024). This method provides more granular results. 2) The result of semantic segmentation via point cloud projection and contour extraction is shown in Fig. 2d. This method leverages high-precision point clouds to achieve instance segmentation for structures like buildings and trees, which directly contact the ground.

E A COMPREHENSIVE INTRODUCTION TO AUTOMATIC TRAJECTORY GENERATION

Leveraging the point cloud map and segmentation tools, OpenFly offers two methods for trajectory generation for different scenes. 1) Path search based on customized action space: First, a global voxel map M_{global} and a bird’s eye view (BEV) occupancy map M_{bev} are constructed from the scene point cloud. Second, the flight altitude is randomly selected within the user-defined height range, and landmarks that are not lower than the height threshold H_τ are chosen as targets. A starting point is selected within the distance range $[r, R]$ from the landmark, ensuring that it is not occupied in both M_{global} and M_{bev} . Then, a point on the line connecting the starting point and the landmark, which is close to the landmark and unoccupied in M_{bev} , is chosen as the endpoint. Third, A collision-free trajectory from the starting point to the endpoint is generated using the A* (Hart et al., 1968) pathfinding algorithm, where the granularity of exploration step size and direction can be adjusted according to the action space. Besides, by repeatedly selecting the endpoint as the new starting point, complex trajectories can be generated. Finally, utilizing OpenFly’s interface, images corresponding to the trajectory points can be obtained. 2) Path search based on grid: Google Map data does not allow image retrieval at arbitrary poses in the space. Thus, we rasterize a pre-selected area and collect images from each grid point in all possible orientations. Starting and ending points are chosen within the grid points to generate trajectories. Corresponding images for these trajectory points are then selected from the pre-collected image set.

F DETAILS OF INSTRUCTION GENERATION

Except for the instruction generation described in the main paper. The remaining process is mainly divided into two parts: landmark feature extraction and sub-instruction fusion. A simplified prompt to the VLM and the corresponding response are probably like this.

- Get Landmark features.

System Prompt: You are an assistant who is proficient in image recognition. You can accurately identify the object in the picture and its characteristics that are different from the surrounding objects. I will give you the three final images you will see. Please focus on the last image and tell me the features of the target building and reply to me in the form of JSON.

User: The target is the nearest prominent landmark to me. Answer me a dictionary like color:–, feature: –, size: –, type: –.

GPT 4o: color: blue, feature: Steel, glass, size: medium size, type: building.

- Instruction Fusion.

System Prompt: You are an assistant proficient in text processing. You need to help me combine these scattered actions and landmarks into a sentence using words with similar meanings and more appropriate words, making them smooth, fluent, and accurate. If the landmarks of adjacent actions are similar or even identical, please use pronouns to refer to them.

User: Multiple sub-instructions.

GPT 4o: Move forward to a high-rise building with a noticeable logo at the top. Then, slightly turn left and go straight to a futuristic tower with a large spherical structure in the middle.

G DATA QUALITY CONTROL

Data Filter. During data collection, it is inevitable that some damaged or low-quality data will be generated. We clean the data in the following situations. 1) We remove damaged images that are produced in generation or transmission. 2) We find that UAVs sometimes appear to pass through the tree models. Therefore, we exclude the trajectories where the altitude is lower than that of the trees. 3) We believe that extremely short or long trajectories are not conducive to model training. Thus, we remove these trajectories, specifically those with fewer than 2 or more than 150 actions.

Instruction Refinement. A known challenge of instruction generation is VLMs’ hallucinations. During the previous instruction generation process, sometimes the same landmark appears across several frames. This results in a VLM generating similar captions for the repeated observations

of a landmark, increasing the complexity of the final instruction and introducing ambiguity due to duplication.

To mitigate this challenge, we utilize the NLTK library (Bird, 2006) to simplify the instruction by detecting and merging similar descriptions. Specifically, a syntactic parse tree is first constructed to extract all landmark captions using a rule-based approach. Then, a sentence-transformer model is employed to encode the extracted landmark captions into embedding vectors. Their similarities are computed with dot product, and high-similarity captions are then identified and replaced with referential pronouns (*e.g.*, “it,” “there,” *etc.*). For example, a generated instruction with redundant information is “... make a left turn toward a **medium-sized beige building marked by a signboard reading CHARLIE’S CHOCOLATE**. Continue heading straight, passing a **medium-sized gray building with a prominent rooftop billboard displaying Charlie’s Chocolate** ...”. After simplification, a more concise sentence is obtained, *i.e.*, “... make a left turn toward a **medium-sized beige building marked by a signboard reading CHARLIE’S CHOCOLATE**. Continue heading straight, passing it ...”, demonstrating the effectiveness of this post-processing technique.

Manually Check. At the same time, we built a data inspection platform to provide instructions, action sequences, and corresponding images to human examiners. If an instruction describes all the actions and landmarks in a trajectory well, it is considered qualified. We randomly select 3K samples from the entire dataset according to the data distribution. After manually inspecting these samples, we find that they reach a high qualification rate of 91%. There is some ambiguity in the description of some landmarks in the remaining data, making it likely that these landmarks are not easily distinguishable from the surrounding environment. However, the examiners consider this not entirely unacceptable. In summary, most of the generated data feature good quality for the aerial VLN task.

H MORE DATASET ANALYSES

Following previous studies (Liu et al., 2023; Wang et al., 2025; Fan et al., 2022), we conducted a statistical analysis of the linguistic phenomena using 25 randomly selected instructions and compared the results with other VLN datasets, as detailed in Table 2. The analysis shows that the generated instructions exhibit rich linguistic phenomena such as ‘Reference’ and ‘Comparison’. Notably, our dataset is not the most complex one, since we believe that instructions in VLN tasks should be more aligned with real-life scenarios, rather than emphasizing length and complexity. This cognition is consistent with that of REVERIE (Qi et al., 2020). Our instructions avoid overly lengthy and unrealistic expressions to some extent, making them more practical to command UAVs.

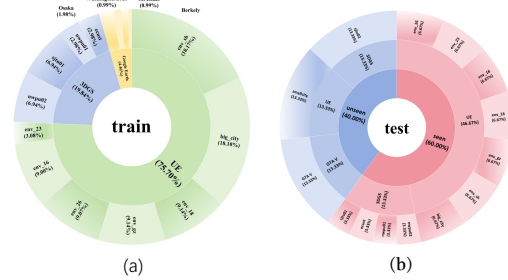


Figure 3: The distribution of the data volume in different scenes under the Train and Test sets. (a) Train set distribution. (b) Test set distribution.

Fig. 3 (a) shows the data distribution of the train set, where 7 UE scenes account for 75.7% of the total 100K data, 4 3D GS scenes account for nearly 20% of the total amount, and Google Earth data accounts for 4.46%. Fig. 3 (b) presents the data distribution of the test set, where the seen data and unseen data account for 60% and 40%, respectively.

I QUALITATIVE EXPERIMENTAL RESULTS

Fig. 4 presents a qualitative result in a UE scene, where our OpenFly-Agent successfully navigates to the destination according to the instruction. It presents a powerful capability in perceiving environments and aligning observations with complex instructions. Fig. 5 presents another successful aerial VLN example in a 3D GS scene. The image style, flight heights, and viewpoints are significantly different from UE’s scenarios. In this case, our OpenFly-Agent exhibits robustness to handle data with great diversity. In addition, Fig. 6 shows two failure cases, where our model sometimes fails to identify the landmark or output actions with proper amplitudes. To further verify the supe-

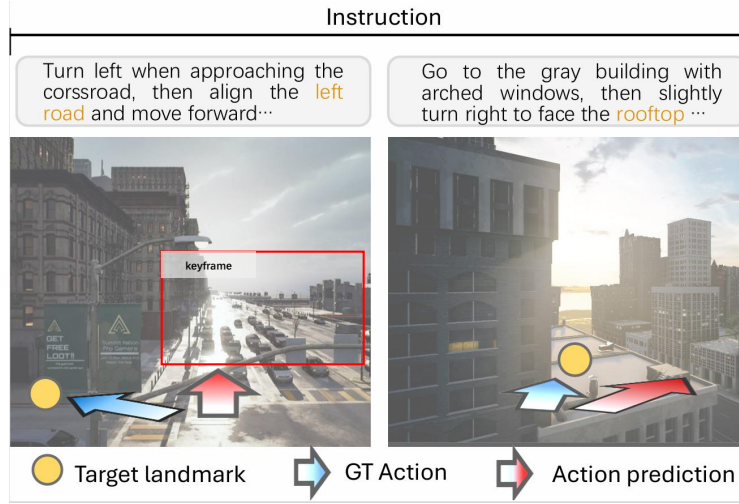


Figure 6: Illustration of failure cases. Sometimes our model may misclassify key landmarks or output wrong actions. The red bounding box represents incorrect landmark locations.

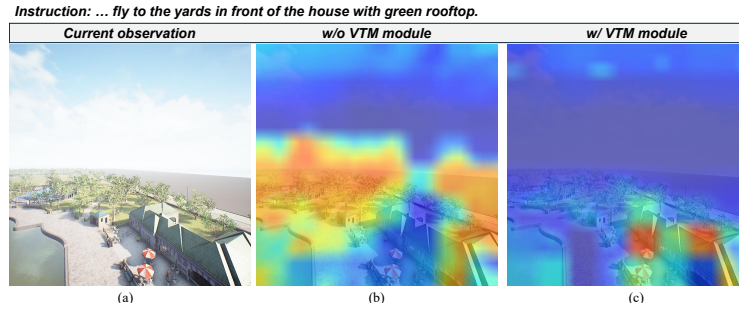


Figure 7: Visualization of attention maps of current observation patches. a) Current observation. b) The attention map without the VTM module and c) the attention map with the VTM module, reflecting token merging strategy matters in avoiding diluting attention of current observation.

J USE OF LLMs

In this work, we employ large language models (LLMs) to automatically identify and correct grammatical errors, thereby improving the overall fluency and readability of the generated text.

REFERENCES

- Steven Bird. Nltk: the natural language toolkit. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, pp. 69–72, 2006. 5
- Sebastien Bonopera, Peter Hedman, Jerome Esnault, Siddhant Prakash, Simon Rodriguez, Theo Thonat, Mehdi Benadel, Gaurav Chaurasia, Julien Philip, and George Drettakis. sibr: A system for image based rendering, 2020. 2
- Yue Fan, Winson X. Chen, Tongzhou Jiang, Chun ni Zhou, Yi Zhang, and Xin Wang. Aerial vision-and-dialog navigation. *ArXiv*, abs/2205.12219, 2022. 5
- Peter E Hart, Nils J Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968. 4

-
- Bernhard Kerbl, Andreas Meuleman, Georgios Kopanas, Michael Wimmer, Alexandre Lanvin, and George Drettakis. A hierarchical 3d gaussian representation for real-time rendering of very large datasets. *ACM Transactions on Graphics (TOG)*, 43(4):1–15, 2024. 2
- Shubo Liu, Hongsheng Zhang, Yuankai Qi, Peng Wang, Yanning Zhang, and Qi Wu. Aerialvln: Vision-and-language navigation for uavs. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pp. 15338–15348, 2023. 1, 5
- Yuankai Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. REVERIE: remote embodied visual referring expression in real indoor environments. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, 2020. 5
- Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 3
- Xiaoda Wang, Kaiqiao Han, Yuhao Xu, Xiao Luo, Yizhou Sun, Wei Wang, and Carl Yang. Simulator and experience enhanced diffusion model for comprehensive ecg generation, 2025. URL <https://arxiv.org/abs/2511.09895>. 5
- Zhigang Wang, Yifei Su, Chenhui Li, Dong Wang, Yan Huang, Bin Zhao, and Xuelong Li. Open-vocabulary octree-graph for 3d scene understanding. *CoRR*, abs/2411.16253, 2024. 3
- Qiu Weichao, Zhong Fangwei, Zhang Yi, Qiao Siyuan, Xiao Zihao, Kim Tae, Wang Yizhou, and Yuille Alan. Unrealcv: Virtual worlds for computer vision. *ACM Multimedia Open Source Software Competition*, 2017. 1