OpenSatMap: A Fine-grained High-resolution Satellite Dataset for Large-scale Map Construction

Hongbo Zhao[1,3*] **Lue Fan**[1,3*†] **Yuntao Chen**[2*] **Haochen Wang**[1,3*] **Yuran Yang**[4,5*]
Xiaojuan Jin[1] **Yixin Zhang**[4] **Gaofeng Meng**[1,2,3] **Zhaoxiang Zhang**[1,2,3†]
[1]Institute of Automation, Chinese Academy of Sciences (CASIA)
[2]Centre for Artificial Intelligence and Robotics, HKISI, CAS
[3]University of Chinese Academy of Sciences (UCAS)
[4]Tencent Maps, Tencent [5]Beijing University of Posts and Telecommunications
https://opensatmap.github.io

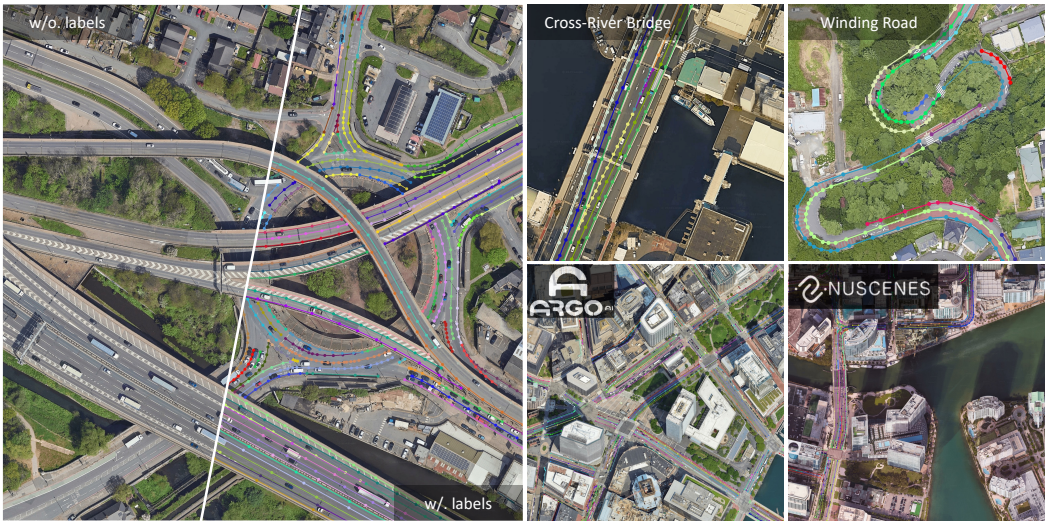


Figure 1: **Demonstrations of OpenSatMap dataset**. It contains high-resolution satellite images with fine-grained annotations, covering diverse geographic locations and popular driving datasets [8, 11].

Abstract

In this paper, we propose OpenSatMap, a fine-grained, high-resolution satellite dataset for large-scale map construction. Map construction is one of the foundations of the transportation industry, such as navigation and autonomous driving. Extracting road structures from satellite images is an efficient way to construct large-scale maps. However, existing satellite datasets provide only coarse semantic-level labels with a relatively low resolution (up to level 19), impeding the advancement of this field. In contrast, the proposed OpenSatMap (1) has fine-grained instance-level annotations; (2) consists of high-resolution images (level 20); (3) is currently the largest one of its kind; (4) collects data with high diversity. Moreover, OpenSatMap covers and aligns with the popular nuScenes dataset and Argoverse 2 dataset to potentially advance autonomous driving technologies. By publishing and maintaining the dataset, we provide a high-quality benchmark for satellite-based map construction and downstream tasks like autonomous driving.

[*]Equal contribution.
[†]Corresponding author.

38th Conference on Neural Information Processing Systems (NeurIPS 2024) Track on Datasets and Benchmarks.

1 Introduction

Large-scale, up-to-date, and fine-grained map construction is fundamental to many tasks such as traffic control and autonomous driving. Compared with on-road mapping, constructing maps from satellite images is a highly efficient way for this purpose because of the large geographic coverage.

Although it is promising, parsing fine-grained road structures from satellite images for map construction is a highly challenging task for the following reasons. *(i)* The resolution of satellite images is usually not high enough for fine-grained lane line detection. For example, the common level-19 satellite images have a resolution of 30 cm per pixel, which can barely make out a lane line that is 20 cm wide. *(ii)* The lane lines in modern cities possess highly complex topological structures and function-based semantics. It not only poses challenges for designing models with enough capacities but also makes it difficult to build fine-grained datasets.

Existing benchmarks [6, 14, 18, 30, 22] have made attempts to handle the challenging tasks. However, they each have their limitations, preventing them from effectively supporting large-scale and fine-grained map construction. Specifically, existing benchmarks have the following limitations.

- **Coarse annotations**. All of these benchmarks support only coarse semantic-level lane segmentation, which is far from enough to parse fine-grained lane markings with various functionalities and highly complicated topologies.
- **Low resolution**. The images in existing benchmarks often have relatively low resolutions. The highest resolution they have is 0.3 m per pixel, which is inadequate for accurate perception of lane lines.
- **Small scale**. Existing benchmarks have relatively small scales. The largest of them have less than 10k 1024×1024 images.
- **Unalignment with autonomous driving benchmarks**. Existing benchmarks solely focus on specific regions, limiting their further applications in autonomous driving, which also heavily relies on map construction.

Table 1 shows the basic information of them. Given the limitations of existing benchmarks and the challenges of this task, we construct **OpenSatMap**, a new dataset for map construction from satellite images, with the following unique features.

- **Fine-grained instance-level annotations**. Unlike the coarse semantic level annotations in previous datasets, we carefully label fine-grained instance-level lane lines according to fine-grained line attributes.
- **Higher resolution.** We collect level-20 satellite images with a resolution of 0.15 m per pixel, which is higher than the resolutions of all previous benchmarks. Considering the data accessibility and diversity, we additionally provide level-19 images with a resolution of 0.3 m per pixel.
- **Larger scale**. We collect and carefully annotate **38k** 1024×1024 images and around **445k** instances, which is around **5x larger** than the largest one of the previous datasets in terms of the number of images.
- **Higher diversity.** We collect data from 60 cities and 19 countries around the world. The collected data is highly diverse and covers various road types, geographic environments, special structures, and traffic regulations.
- **Alignment with autonomous driving benchmarks.** In order to advance the map construction in autonomous driving, OpenSatMap covers the regions of nuScenes [8] and Argoverse 2 [11] datasets, which are the most popular autonomous driving datasets. In addition, we manually align the GPS locations of our data with them for practical use.

Given these characteristics, we believe OpenSatMap can serve as a foundation for various applications, including city-scale map construction, lane detection, and autonomous driving.

2 Related Work

Satellite Imagery Datasets. Satellite imagery datasets can be used for several computer vision tasks such as instance segmentation [24, 46, 15], object detection [20, 50, 19, 33], semantic segmentation

Table 1: **Comparison against previous datasets.** OpenSatMap is the largest road extraction dataset with the highest resolution and the most detailed annotations. * denotes that we *standardize the size of images to 1024 × 1024* and calculate the number of images in all datasets.

Dataset	# of Images*	Resolution	GT Source	Labeling Level	Region
Massachusetts [39]	2513	1.00 *m/pixel*	OSM	Semantic	America
CasNet [14]	77	1.20 *m/pixel*	Manually	Semantic	-
DeepGlobe [18]	8570	0.50 *m/pixel*	QGIS	Semantic	3 Counties
SpaceNet [47]	4481	0.31 *m/pixel*	OSM	Semantic	4 Counties
Roadtracer [6]	4800	0.60 *m/pixel*	OSM	Semantic	6 Counties
Ottawa [37]	235	0.30 *m/pixel*	Manually	Semantic	Canada
CHN6-CUG [54]	4511	0.50 *m/pixel*	Manually	Semantic	China
OpenSatMap (Ours)	7224 31696	0.30 *m/pixel* 0.15 *m/pixel*	Manually	Instance, Vectorized	60 Cities, 19 Countries

[49, 10, 42, 37], scene classification [43, 31], and change detection [17, 40, 45, 12] *etc*.. These datasets include optical and hyper-spectral images with varying resolutions, covering a wide range of application fields such as urban [41, 10], agricultural [24, 15], transportation [37, 7, 6] and marine [3, 25] areas on a global scale. They provide a wealth of labeling information, such as building outlines, road networks, vegetation types, water distribution, *etc*., which greatly promotes the development of deep learning in the field of remote sensing images. In this paper, we are interested in the application of satellite images on road extraction and we will discuss the datasets for road construction using remote sensing imagery later.

Satellite Imagery Datasets for Road Extraction. Early datasets mainly focus on semantic labeling for the satellite images and contribute significantly to semantic-level road extraction. Massachusetts [39] contains 1171 images of size 1500 × 1500 with a resolution of 1 m/pixel. DeepGlobe [18] collects images from Thailand, Indonesia, and India and scrapes labels from QGIS [5]. It contains 8570 images with image size 1024 × 1024. Roadtracer [6] uses OpenStreetMap [4] to label 300 images from big cities with 0.6 m/pixel. SpaceNet [47] contains 2780 images collected from Las Vegas, Paris, Shanghai, and Khartoum. There are some datasets manually labeled. Ottawa [37] collects several typical urban areas with 0.30 m/pixel spatial resolution and annotates the road surface, road edges and road centerlines. CHN6-CUG [54] collects 6 cities in China with 0.5 m/pixel. and CasNet [14] is composed of 224 images with a spatial resolution of 1.2 m per pixel.

However, existing satellite road extraction datasets tend to be relatively small [37, 14, 29, 53], or labeled by OpenStreetMap [47, 39, 6] and QGIS [18, 52] platforms, which are limited by the meta information in the database. Besides, *semantic* labeling results in a wealth of under-utilized information. With the booming development of high-resolution remote sensing imagery, academics urgently need a higher resolution, more richly labeled dataset. Our work fills this gap by providing a high quality, higher resolution, and bigger dataset with *instance-level* vectorized annotations. The readers may refer to Table 1 for a detailed comparison against previous datasets.

3 OpenSatMap Dataset

In this section, we introduce our pipeline to collect (Sec. 3.1), annotate (Sec. 3.2) high-resolution satellite images and demonstrate the statistics of OpenSatMap in Sec. 3.3.

3.1 Data Collection

We collect images from Google Maps [1] using public Maps Static API, which is publicly available and has been processed by Google to remove sensitive information. Considering the limitations mentioned in Sec. 1, we collect the data OpenSatMap with the following guidelines and factors.

High resolution. To fulfill the goal of higher resolution. We collect level-20 satellite images, which have a resolution of 0.15 cm/pixel. In this way, our dataset has a higher resolution than all the existing datasets as Table 1 shows. Google Maps does not have real level-20 images in some regions. Therefore, during the collection, we carefully verify the resolution for each image to avoid the

(a) Landmark correspondences between a satellite image and a nuScenes image.

(b) Overlaying the driving trajectories from the nuScenes dataset onto OpenSatMap (Boston Seaport).

Figure 2: **Alignment with driving benchmark**.

"pseudo level-20 images" which are actually resized from level-19 images. Considering the practical requirements of users, we also additionally collect level-19 images. To be precise, **OpenSatMap19** stands for the level-19 part, and **OpenSatMap20** stands for the level-20 part.

Geographic Diversity The roads in different geographic locations exhibit significant differences in the surrounding natural environment, construction regulations, spatial distribution, road conditions, and appearance. To ensure such diversities, we sample the geographic locations considering *(i)* famous and typical cities around the world; *(ii)* terrain with different appearances such as plains, mountains, and deserts; *(iii)* regions with different road densities such as urban and suburban. In this way, we ensure the overall diversity of our dataset by controlling the geographic diversity. In the following, we take more factors into account to further improve the data quality and diversity.

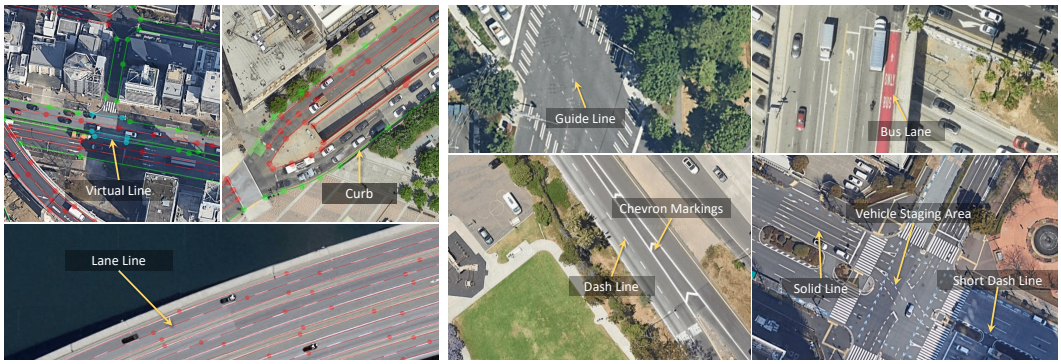
Special Road Structures Unlike the common straight roads, some roads have special and complicated structures such as flyovers, roundabouts, and winding roads. The lane instances of these regions demonstrate different shapes and distributions from the straight lanes. To ensure the models trained on our data could well handle these structures, we manually search and collect some regions containing them. Figure 1 shows examples of special road structures.

Traffic Feature Here we use the term traffic feature to represent left-hand/right-hand traffic regulations and vehicle density. The left-hand/right-hand regulations determine the direction of lanes. The vehicle density has an impact on the difficulty of lane detection since vehicles may occlude the lanes. During data collection, we deliberately consider the regions with different driving regulations and balance the samples with different vehicle densities.

Alignment with Driving Benchmark To facilitate the map construction task in the field of autonomous driving, we make the collecting regions fully cover the driving region in famous driving benchmark nuScenes [8] and Argoverse 2 [11]. Since Argoverse 2 only provides the relative pose without GPS, we manually recognize a couple of landmarks of each city in Argoverse 2 and find the GPS coordinates of these landmarks. Then the GPS coordinates of all samples in Argoverse 2 can be derived by the relative poses. Figure 2 showcases the alignment.

3.2 Annotation

After the data collection, we employ experienced annotators to carefully annotate the data at a fine-grained instance level. The annotations are performed by a professional remote sensing imagery labeling team of approximately 50 annotators for labeling and 7 for quality checking. The total cost of labeling is roughly \$72,000, *i.e.*, \$19 per image. Note that, for labeling, the image size of OpenSatMap20 we collect is 4096×4096 and 2048×2048 for OpenSatMap19.



(a) An example of three categories. (b) Examples of attributes.

Figure 3: **Examples of categories and attributes**.

Line representation. To represent the lane lines with highly variable lengths and curvatures, we adopt vectorized polylines as the representation. Each line contains a wealth of category and attribute information. Meanwhile, the order of the points in the polylines defines the line direction.

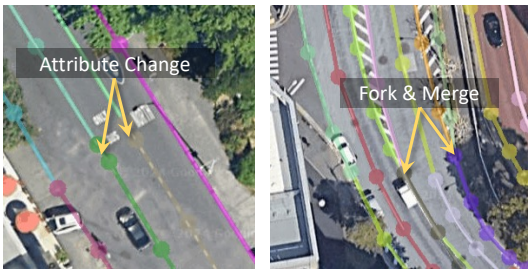
Line categories. We first categorize all lines into three categories: **curb**, **lane line**, and **virtual line**. A curb is the boundary of a road. Lane lines are those visible lines forming the lanes. A virtual line means that there is no lane line or curb here, but logically there should be a boundary to form a full lane. For example, a fork in the road breaks a curb into two segments, then we use a virtual line to connect the two segments. Figure 3a shows examples of these three categories.

Where should it be labeled? Generally speaking, we label all three categories above. However, there is usually occlusion in the satellite images. If a line is partially occluded by buildings, trees, and vehicles but its complete shape can be inferred with high confidence, we annotate it. Fully occluded lines are not labeled. A special case is that overpasses usually occlude the bottom roads, where the occluded part is not labeled. We show these cases in the supplementary materials.

Fine-grained line attributes. Lines are assigned with a couple of attributes based on their appearance and functionalities. Specifically, there are 8 attributes as follows:

- The colors of lines.
- The line type such as solid line, thick solid line, dashed line, and short dashed line.
- The number of lines such as single lines and double lines.
- Lines with special functionalities such as bus lanes, guide lines, lane-borrowing areas.
- Whether it is bi-directional.
- Whether it is the outermost boundary of the pavement.
- The level of occlusion.
- The level of clearness.

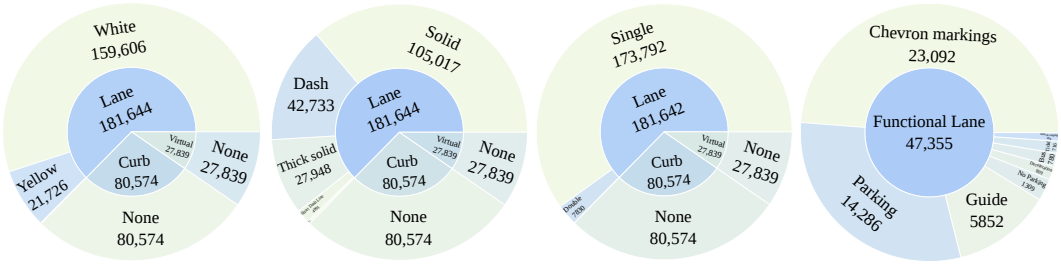
Figure 3b shows some examples of different attributes.



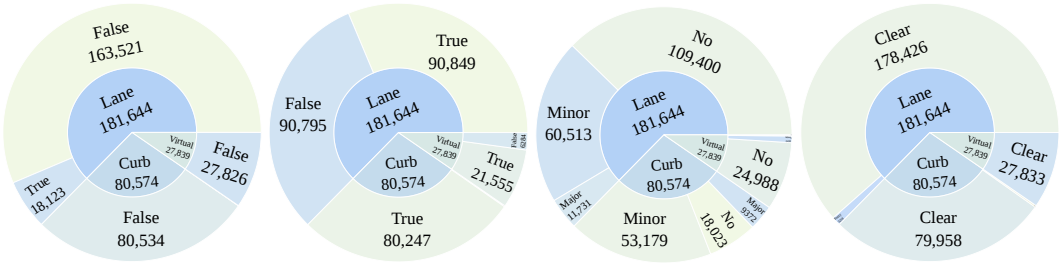
(a) Attribute change. (b) Lines fork and merge.

Figure 4: **Definition of instances** (*zoom in* for best viewing). In (a), a change of line type results in two instances sharing the same point. In (b), a line should be divided into three instances when it is forked or merged.

Instances definition. After defining the line attributes, we further define the instance using the following guidelines. *(i)* Different instances have different attributes. For example, if a solid line becomes a dashed line, we cut the line into two instances. Figure 4a shows an example of such attribute change. *(ii)* When lines fork or merge (*e.g.*, "Y"-shape point), we break the lines into multiple instances as Figure 4b shows.



(a) Color distribution. (b) Line type distribution. (c) # lines distribution. (d) Function distribution.



(e) Bidirection distribution. (f) Boundary distribution. (g) Occlusion distribution. (h) Clearness distribution.

Figure 6: Distribution of line attributes in OpenSatMap20.

Image-level tags. Furthermore, we provide OpenSatMap20 with 4 additional image-level tags to describe general information, including image clearness, overall vehicle density, urban/suburban/rural, and the existence of special road structures. We present the details in the supplementary materials.

3.3 Statistics of OpenSatMap

In this section, we summarize the statistics of the established OpenSatMap. There are 1806 2048×2048 images in OpenSatMap19 and 1981 4096×4096 images in OpenSatMap20. We randomly divided them into training set, validation set and testing set in the ratio of 6:2:2. Figure 5 shows the number of instances in each image in OpenSatMap19 and OpenSatMap20. The number of instances of most images is under 300. Figure 6 illustrates the distributions of attributes. Most attributes have a non-uniform distribution, which reflects the real condition of the roads. Figure 7 shows the tag distribution in OpenSatMap20.

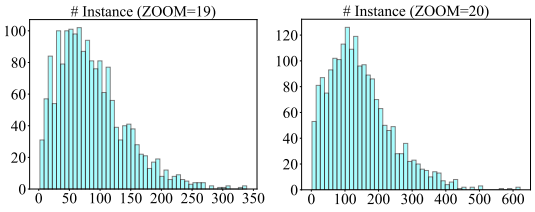


Figure 5: Number of instances in each image in OpenSatMap19 (left) and OpenSatMap20 (right).

4 Instance-level Line Detection

4.1 Formulations and Baseline Method

Given an input image $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$, where $H \times W$ indicates the input resolution, we aim to detect each line instance in the input image. For each instance, we use polylines as the *vectorized* representation and pixel-wise instance-level masks as the *rasterized* representation. A polyline is defined as a set of points $\mathbf{p} \in \mathbb{R}^{N \times 2}$, where N means the number of sampled points. $\mathbf{p}[:, 0]$ and $\mathbf{p}[:, 1]$ represent the x and y coordinates, respectively. One can rasterize polylines to pixel-level masks by simply connecting adjacent points with lines with a pre-defined line width.

The instance-level line detection task needs to convert an RGB image $\mathbf{I}$ into a set of polylines $\{\mathbf{p}_i\}_{i=1}^{M}$, where M is the number of instances. We develop a simple baseline without whistle and bells illustrated in Figure 8. We decompose this task into 3 steps: (1) semantic segmentation, (2) instance detection, and (3) instance vectorization. (2) and (3) are post-processing techniques. Detailed formulations for each step are provided as follows and implementation details are provided in the *Supplementary materials*.

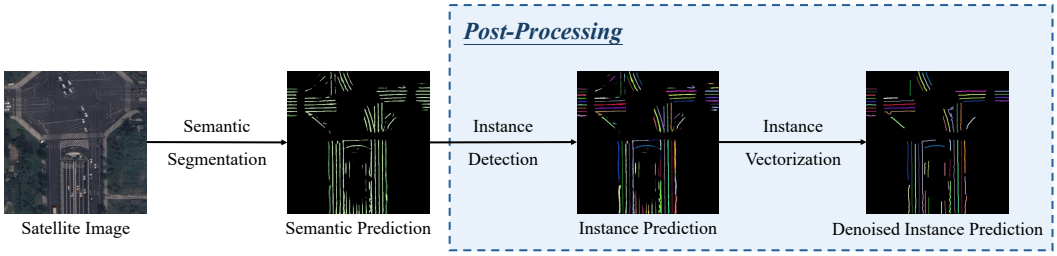


Figure 8: Illustration of our baseline method.

Semantic segmentation $f : \mathbb{R}^{H\times W\times 3} \rightarrow \{0, 1, \cdots, C-1\}^{H\times W}$ aims to classify each pixel into a unique semantic category, where C is the number of categories. We denote $\mathbf{y} = f(\mathbf{I})$ as the segmentation result. We adopt SegNeXt [27] as the segmentation network since it is quite efficient for high-resolution images.

Instance detection $g : \{0, 1\}^{H\times W} \rightarrow \{0, 1, \cdots, M\}^{H\times W}$ converts the binary segmentation mask of each category into a pixel-level mask, where each instance shares the same value within this mask. Technically, we first extract the binary mask $\mathbf{y}_c \in \{0, 1\}^{H\times W}$ of each category c and then apply the watershed algorithm [2] to obtain instance-level masks.

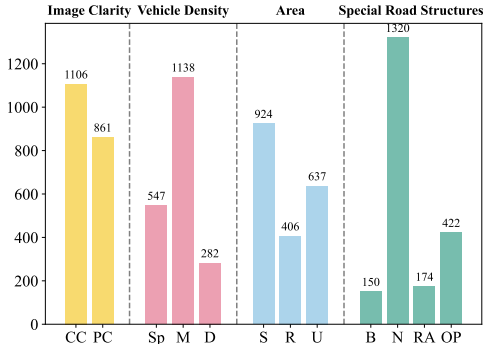


Figure 7: **Image-level tag distribution in OpenSatMap20.** CC = complete clear, PC = partially clear, Sp = sparse, M = moderate, D = dense, S = suburban, R = rural, U = urban, B = bridge, N = None, RA = roundabout, OP = overpass.

Instance vectorization $h : \{0, 1\}^{H\times W} \rightarrow \mathbb{R}^{N\times 2}$ aims to build a *vectorized* representation of each instance, which contains (1) instance denoising, and (2) point sampling. Instance denoising follows a simple sample-then-reconstruct pipeline. Specifically, for each instance i that belongs to category c, we first extract its binary mask $\mathbf{y}_c^i \in \{0, 1\}^{H\times W}$. We subsequently sample N points and obtain their coordinates $\mathbf{p}_c^i \in \mathbb{R}^{N\times 2}$. Then, we simply connect adjacent points with lines and remove instances with fewer than 100 pixels, obtaining the denoised instance map $\hat{\mathbf{y}}_c^i = h(\mathbf{y}_c^i)$ for each instance i that belongs to category c. Note that $\hat{\mathbf{y}}_c^i$ is the final *rasterized* representation of a line instance. As for point sampling, we sample N points from $\hat{\mathbf{y}}_c^i$, resulting in $\hat{\mathbf{p}}_c^i = \phi(\hat{\mathbf{y}}_c^i)$ to build the *vectorized* representation.

4.2 Evaluation

Semantic-level evaluation. We adopt the mean intersection over union (mIoU) [21] over different categories as the metric to evaluate the performance at the semantic level.

Instance-level evaluation. We adopt the average precision (AP) as the metric to evaluate the performance at the instance level. Under instance segmentation settings, a common practice is to use the Mask AP (AP^{M}) by leveraging a mask IoU threshold to identify whether an instance is a true positive sample or not [13, 48, 36, 9, 28]. However, line markings typically exhibit *narrow* features and segmentation outputs frequently lack precision in edge definition. Consequently, relying exclusively on Mask IoU thresholds may be excessively strict. To this end, we incorporate Chamfer distance proposed by [34] as an alternative. Chamfer distance between two sets of points $\mathbf{p} \in \mathbb{R}^{M\times 2}$ and $\mathbf{q} \in \mathbb{R}^{K\times 2}$ is defined as:

$$D_{\text{Chamfer}}(\mathbf{p}, \mathbf{q}) = \frac{1}{M}\sum_{i=1}^{M} \min_{j=1,2,\ldots,K} d(p_i, q_j), \tag{1}$$

where $d(\cdot, \cdot)$ is the Euclidean distance between two points. Then, we get Chamfer AP (AP_D^{C}) by choosing the distance threshold D.

Table 2: **Evaluation on OpenSatMap validation set.** The line width is set to 6 pixels in OpenSatMap20 and 3 pixels in OpenSatMap19 by default. AP^{M} means that the mask IoU is used when determining true positives, while AP^{C}_{D} means Chamfer AP with a threshold of D meters. AP_{x} denotes that the threshold is set to x. $\text{AP}_{x:y}$ indicates the *averaged* values, varying the threshold from x to y.

Dataset	$\text{AP}^{\text{C}}_{0.9}$	$\text{AP}^{\text{C}}_{1.5}$	$\text{AP}^{\text{C}}_{3.0}$	$\text{AP}^{\text{C}}_{4.5}$	$\text{AP}^{\text{M}}_{50:95}$	$\text{AP}^{\text{M}}_{50}$	$\text{AP}^{\text{M}}_{75}$	mIoU
OpenSatMap19	$16.04_{\pm 0.35}$	$22.68_{\pm 0.35}$	$26.88_{\pm 0.52}$	$29.18_{\pm 0.22}$	$3.66_{\pm 0.15}$	$10.66_{\pm 0.44}$	$1.45_{\pm 0.12}$	$28.71_{\pm 0.38}$
OpenSatMap20	$20.30_{\pm 0.21}$	$25.93_{\pm 0.35}$	$29.50_{\pm 0.40}$	$31.38_{\pm 0.43}$	$6.98_{\pm 0.21}$	$16.05_{\pm 0.32}$	$5.26_{\pm 0.13}$	$33.69_{\pm 0.45}$

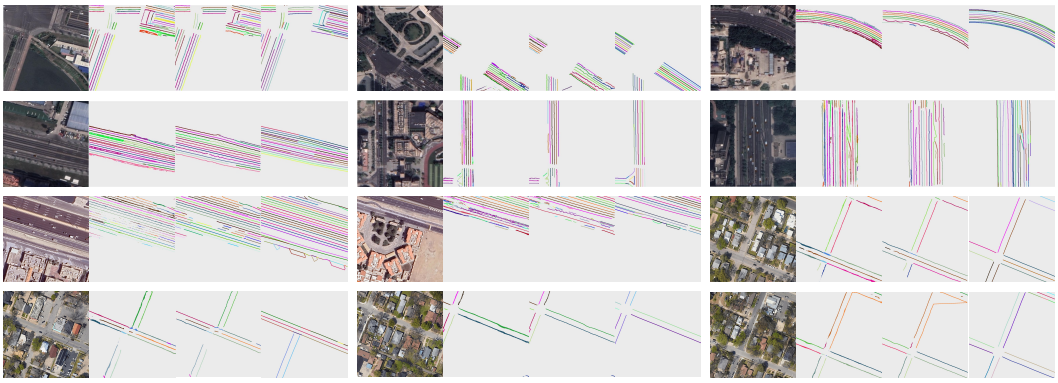


Figure 9: **Qualitative results** on OpenSatMap19 (the first two rows) and OpenSatMap20 (the last two rows) test split. For each scene, we put **(a)** the input image, **(b)** the instance prediction, **(c)** the *denoised* instance prediction, and **(d)** the annotation, **from left to right**, respectively.

4.3 Main results

Empirical results in Table 2 indicate that *instance-level* line detection is a much more challenging task compared with semantic-level segmentation, since values of AP^{C} and AP^{M} are much lower than that of mIoU.

Fundamentally, this issue arises because instances are associated with fine-grained attributes while semantic segmentation only involves several categories. Consequently, identifying true positive instance predictions becomes a notably difficult task.

Qualitative results. We provide qualitative results in Figure 9. Our method manages to detect line instances. Although the visual results of our model are almost satisfactorily presented, they are accompanied by disappointingly low quantitative scores of AP^{C} and AP^{M}. Usually, our method fails to detect accurate line instances, when the segmentation performs poorly, such as failing to predict separate masks when attributes change, and adhered segmentation masks of two separate lines. This disparity indicates that this benchmark is of substantial difficulty, necessitating deeper exploration of an effective end-to-end method.

5 Potential Application in Autonomous Driving

To facilitate the map construction task in autonomous driving, we make OpenSatMap deliberately cover the regions of nuScenes [8] and Argoverse 2 [11] datasets. Recently, a line of work has proved the effectiveness and success of using additional map information to boost the performance of HDMap construction for autonomous driving. MapEX [44] uses historically stored map information to optimize the construction of current local high-precision maps. NMP [51] uses a neural map prior to boost the performance of VectorMapNet [38].

Some more closed work uses satellite image information to enhance map construction. SatforHDMap [23] leverages satellite images as an additional input modality for MapTR [35]. P-MapNet [32] extracts weakly aligned SDMap from OpenStreetMap [4], and encode it as an additional feature for MapTR, achieving better performance. Our OpenSatMap has the potential to advance this field. We take SatforHDMap as an example. We use intersection-over-union (IoU) as the metric following

semantic map learning [34]. Let $\mathcal{D}_1, \mathcal{D}_2 \in \mathbb{R}^{H \times W \times D}$ be dense predictions of shapes, H and W are the height and width of the grid, D is the number of categories. IoU can be expressed as follows:

$$\text{IoU}\left(\mathcal{D}_1, \mathcal{D}_2\right) = \frac{|\mathcal{D}_1 \cap \mathcal{D}_2|}{|\mathcal{D}_1 \cup \mathcal{D}_2|}, \tag{2}$$

where $|\cdot|$ is the size of the set. Table 3 shows the results using SatforHDMap [23] with OpenSatMap. The use of corresponding satellite imagery significantly enhanced the performance of online map construction, demonstrating the potential superiority of this approach.

Table 3: Evaluation of semantic map segmentation on nuScenes validation set.

Method	Divider	Crossing	Boundary	All
HDMapNet [34]	40.6	18.7	39.5	32.9
SatforHDMap [23]	**50.2**	**53.2**	**49.4**	**50.9**

6 Availability and Maintenance

OpenSatMap is collected from Google Maps, which is publicly available. According to the regulations of Google, the use of this dataset is restricted to **non-commercial research**. Our project page is https://opensatmap.github.io, which contains the full dataset with annotations and code repository. The repository contains the full-stack tools to use this dataset, including code for collecting images from Google Maps, documents, baseline models, and evaluation tools. We encourage the community to further explore more tracks and effective methods using OpenSatMap. This dataset will be maintained over the long term and continually iterated upon.

7 Limitations and Potential Social Impact

OpenSatMap has the following limitations. *(i)* OpenSatMap is collected from Google Maps, which is not updated in real time, and certain areas may be outdated and not reflect the current conditions. Besides, the timing of captured remote sensing images may not be aligned with nuScenes and Argoverse dataset. Although we noticed this potential misalignment on road structures and asked the annotators to check the consistency of road structures between our data and the data in the driving datasets during annotating process, there are still very few inconsistencies. *(ii)* High-resolution (level-20) images in certain areas are not available or may be covered by the cloud, limiting the diversity of collection locations to some extent. *(iii)* Although the annotators are quite professional and we conduct strict sanity checks, the subjectivity of annotators may inevitably lead to slightly inconsistent annotation results since this project employs around 60 annotators. In addition, the inconsistent annotation can also stem from the different training and expertise levels of the annotators. A potential social impact is that it might cause a safety risk for driving if the model is directly trained on our dataset, which contains outdated data as discussed above.

8 Conclusion

In summary, we introduce OpenSatMap, a large-scale, high-resolution, geographically diverse satellite dataset with detailed annotations and alignment with driving benchmarks. To validate the utility of OpenSatMap, we construct the instance-level line detection track and provide a baseline for it. Moreover, we use OpenSatMap to improve the performance of online map construction, which shows the great potential of autonomous driving. We believe that our carefully annotated high-quality OpenSatMap can serve as the foundation for various applications, including lane detection, city-scale map construction, and autonomous driving.

Acknowledgments and Disclosure of Funding

This work was supported in part by the National Key R&D Program of China (No. 2022ZD0116500), the National Natural Science Foundation of China (No. U21B2042, No. 62320106010), and in part by the 2035 Innovation Program of CAS, and the InnoHK program, and in part by the Tencent Maps Collaborative Research Project.

References

[1] Google Maps. https://maps.google.com.

[2] Image segmentation with watershed algorithm. https://docs.opencv.org/3.4/d3/db4/tutorial_py_watershed.html.

[3] Noaa fisheries steller sea lion population count | kaggle. https://www.kaggle.com/c/noaa-fisheries-steller-sea-lion-population-count.

[4] OpenStreetMap. https://www.openstreetmap.org.

[5] QGIS. https://qgis.org/en/site/.

[6] Favyen Bastani, Songtao He, Sofiane Abbar, Mohammad Alizadeh, Hari Balakrishnan, Sanjay Chawla, Sam Madden, and David DeWitt. Roadtracer: Automatic extraction of road networks from aerial images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4720–4728, 2018.

[7] Martin Büchner, Jannik Zürn, Ion-George Todoran, Abhinav Valada, and Wolfram Burgard. Learning and aggregating lane graphs for urban automated driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13415–13424, 2023.

[8] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020.

[9] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020.

[10] Javiera Castillo-Navarro, Bertrand Le Saux, Alexandre Boulch, Nicolas Audebert, and Sébastien Lefèvre. Semi-supervised semantic segmentation in earth observation: The minifrance suite, dataset analysis and multi-task network study. *Machine Learning*, 111(9):3125–3160, 2022.

[11] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, et al. Argoverse: 3d tracking and forecasting with rich maps. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8748–8757, 2019.

[12] Hao Chen and Zhenwei Shi. A spatial-temporal attention-based method and a new dataset for remote sensing image change detection. *Remote Sensing*, 12(10), 2020.

[13] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1290–1299, 2022.

[14] Guangliang Cheng, Ying Wang, Shibiao Xu, Hongzhen Wang, Shiming Xiang, and Chunhong Pan. Automatic road detection and centerline extraction via cascaded end-to-end convolutional neural network. *IEEE Transactions on Geoscience and Remote Sensing*, 55(6):3322–3337, 2017.

[15] Mang Tik Chiu, Xingqian Xu, Yunchao Wei, Zilong Huang, Alexander G Schwing, Robert Brunner, Hrant Khachatrian, Hovnatan Karapetyan, Ivan Dozier, Greg Rose, et al. Agriculture-vision: A large aerial image database for agricultural pattern analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2828–2838, 2020.

[16] MMSegmentation Contributors. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. https://github.com/open-mmlab/mmsegmentation, 2020.

[17] Rodrigo Caye Daudt, Bertr Le Saux, Alexandre Boulch, and Yann Gousseau. Urban change detection for multispectral earth observation using convolutional neural networks. In *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*, pages 2115–2118, 2018.

[18] Ilke Demir, Krzysztof Koperski, David Lindenbaum, Guan Pang, Jing Huang, Saikat Basu, Forest Hughes, Devis Tuia, and Ramesh Raskar. Deepglobe 2018: A challenge to parse the earth through satellite images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 172–181, 2018.

[19] Jian Ding, Nan Xue, Yang Long, Gui-Song Xia, and Qikai Lu. Learning roi transformer for detecting oriented objects in aerial images. *arXiv preprint arXiv:1812.00155*, 2018.

[20] Jian Ding, Nan Xue, Gui-Song Xia, Xiang Bai, Wen Yang, Michael Yang, Serge Belongie, Jiebo Luo, Mihai Datcu, Marcello Pelillo, and Liangpei Zhang. Object detection in aerial images: A large-scale benchmark and challenges. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2021.

[21] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010.

[22] Lipeng Gao, Yiqing Zhou, Jiangtao Tian, and Wenjing Cai. Ddctnet: A deformable and dynamic cross transformer network for road extraction from high resolution remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing*, 2024.

[23] Wenjie Gao, Jiawei Fu, Yanqing Shen, Haodong Jing, Shitao Chen, and Nanning Zheng. Complementing onboard sensors with satellite map: A new perspective for hd map construction, 2024.

[24] Vivien Sainte Fare Garnot and Loic Landrieu. Panoptic segmentation of satellite image time series with convolutional temporal attention networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4872–4881, 2021.

[25] Jan Gasienica-Jozkowy, Mateusz Knapik, and Bogusław Cyganek. An ensemble deep learning method with optimized weights for drone-based water rescue and surveillance. *Integrated Computer-Aided Engineering*, 28(3):221–235, 2021.

[26] Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé Iii, and Kate Crawford. Datasheets for datasets. *Communications of the ACM*, 64(12):86–92, 2021.

[27] Meng-Hao Guo, Cheng-Ze Lu, Qibin Hou, Zhengning Liu, Ming-Ming Cheng, and Shi-Min Hu. Segnext: Rethinking convolutional attention design for semantic segmentation. *Advances in Neural Information Processing Systems*, 35:1140–1156, 2022.

[28] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[29] Songtao He and Hari Balakrishnan. Lane-level street map extraction from aerial imagery. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2080–2089, 2022.

[30] Songtao He, Favyen Bastani, Satvat Jagwani, Mohammad Alizadeh, Hari Balakrishnan, Sanjay Chawla, Mohamed M Elshrif, Samuel Madden, and Mohammad Amin Sadeghi. Sat2graph: Road graph extraction through graph-tensor encoding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIV 16*, pages 51–67. Springer, 2020.

[31] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

[32] Zhou Jiang, Zhenxin Zhu, Pengfei Li, Huan-ang Gao, Tianyuan Yuan, Yongliang Shi, Hang Zhao, and Hao Zhao. P-mapnet: Far-seeing map generator enhanced by both sdmap and hdmap priors. *arXiv preprint arXiv:2403.10521*, 2024.

[33] Darius Lam, Richard Kuzma, Kevin McGee, Samuel Dooley, Michael Laielli, Matthew Klaric, Yaroslav Bulatov, and Brendan McCord. xview: Objects in context in overhead imagery. *arXiv preprint arXiv:1802.07856*, 2018.

[34] Qi Li, Yue Wang, Yilun Wang, and Hang Zhao. Hdmapnet: An online hd map construction and evaluation framework. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 4628–4634. IEEE, 2022.

[35] Bencheng Liao, Shaoyu Chen, Xinggang Wang, Tianheng Cheng, Qian Zhang, Wenyu Liu, and Chang Huang. Maptr: Structured modeling and learning for online vectorized hd map construction. *arXiv preprint arXiv:2208.14437*, 2022.

[36] 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 *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[37] Yahui Liu, Jian Yao, Xiaohu Lu, Menghan Xia, Xingbo Wang, and Yuan Liu. Roadnet: Learning to comprehensively analyze road networks in complex urban scenes from high-resolution remotely sensed images. *IEEE Transactions on Geoscience and Remote Sensing*, 57(4):2043–2056, 2018.

[38] Yicheng Liu, Tianyuan Yuan, Yue Wang, Yilun Wang, and Hang Zhao. Vectormapnet: End-to-end vectorized hd map learning. In *International conference on machine learning*. PMLR, 2023.

[39] Volodymyr Mnih. *Machine Learning for Aerial Image Labeling*. PhD thesis, University of Toronto, 2013.

[40] Sorour Mohajerani and Parvaneh Saeedi. Cloud and cloud shadow segmentation for remote sensing imagery via filtered jaccard loss function and parametric augmentation. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 14:4254–4266, 2021.

[41] Maria Papadomanolaki, Maria Vakalopoulou, and Konstantinos Karantzalos. A deep multitask learning framework coupling semantic segmentation and fully convolutional lstm networks for urban change detection. *IEEE Transactions on Geoscience and Remote Sensing*, 59(9):7651–7668, 2021.

[42] Maryam Rahnemoonfar, Tashnim Chowdhury, Argho Sarkar, Debvrat Varshney, Masoud Yari, and Robin Roberson Murphy. Floodnet: A high resolution aerial imagery dataset for post flood scene understanding. *IEEE Access*, 9:89644–89654, 2021.

[43] Gencer Sumbul, Marcela Charfuelan, Begüm Demir, and Volker Markl. Bigearthnet: A large-scale benchmark archive for remote sensing image understanding. In *IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium*, pages 5901–5904. IEEE, 2019.

[44] Rémy Sun, Li Yang, Diane Lingrand, and Frédéric Precioso. Mind the map! accounting for existing map information when estimating online hdmaps from sensor data. *arXiv preprint arXiv:2311.10517*, 2023.

[45] Shiqi Tian, Ailong Ma, Zhuo Zheng, and Yanfei Zhong. Hi-ucd: A large-scale dataset for urban semantic change detection in remote sensing imagery. *arXiv preprint arXiv:2011.03247*, 2020.

[46] Adam Van Etten, Daniel Hogan, Jesus Martinez Manso, Jacob Shermeyer, Nicholas Weir, and Ryan Lewis. The multi-temporal urban development spacenet dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6398–6407, 2021.

[47] Adam Van Etten, Dave Lindenbaum, and Todd M Bacastow. Spacenet: A remote sensing dataset and challenge series. *arXiv preprint arXiv:1807.01232*, 2018.

[48] Haochen Wang, Junsong Fan, Yuxi Wang, Kaiyou Song, Tong Wang, and ZHAO-XIANG ZHANG. Droppos: Pre-training vision transformers by reconstructing dropped positions. *Advances in Neural Information Processing Systems*, 36, 2023.

[49] Junjue Wang, Zhuo Zheng, Ailong Ma, Xiaoyan Lu, and Yanfei Zhong. Loveda: A remote sensing land-cover dataset for domain adaptive semantic segmentation. *arXiv preprint arXiv:2110.08733*, 2021.

[50] Gui-Song Xia, Xiang Bai, Jian Ding, Zhen Zhu, Serge Belongie, Jiebo Luo, Mihai Datcu, Marcello Pelillo, and Liangpei Zhang. Dota: A large-scale dataset for object detection in aerial images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[51] Xuan Xiong, Yicheng Liu, Tianyuan Yuan, Yue Wang, Yilun Wang, and Hang Zhao. Neural map prior for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17535–17544, 2023.

[52] Jiawei Yao, Xiaochao Pan, Tong Wu, and Xiaofeng Zhang. Building lane-level maps from aerial images. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3890–3894. IEEE, 2024.

[53] Andi Zang, Runsheng Xu, Zichen Li, and David Doria. Lane boundary geometry extraction from satellite imagery. *arXiv preprint arXiv:2002.02362*, 2020.

[54] Qiqi Zhu, Yanan Zhang, Lizeng Wang, Yanfei Zhong, Qingfeng Guan, Xiaoyan Lu, Liangpei Zhang, and Deren Li. A global context-aware and batch-independent network for road extraction from vhr satellite imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 175:353–365, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See in Sec. 7.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See in Sec. 7
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A] We are not including these.
   (b) Did you include complete proofs of all theoretical results? [N/A] We are not including these.
3. If you ran experiments (e.g. for benchmarks)...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See our project page.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See the supplemental material.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See in Table 2
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See the supplemental material.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] See the project page.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] See Sec. 6.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] Sec. 3.1 and Sec. 6.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [Yes] See Sec. 3.2.
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A] This work does not use human subjects.
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [Yes] See Sec. 3.2.

In this Supplementary Material, we:

- Provide a detailed description of the dataset following guides on dataset publication [26].
- Provide the details and results on instance-level line detection and satellite-enhanced online map construction for autonomous driving.

A OpenSatMap Description

A.1 Datasheets

A.1.1 Motivation

1. **For what purpose was the dataset created?** Was there a specific task in mind? Was there a specific gap that needed to be filled? Please provide a description.
   OpenSatMap is created to parse fine-grained road structures from satellite images and can serve as a foundation for various applications, including city-scale map construction, lane line detection, and autonomous driving, *etc.*. Existing benchmarks [37, 6, 18, 14] have made attempts to handle this task. However, their limitations including coarse annotations, low resolution, small scale and unalignment with autonomous driving benchmarks prevent them from effectively supporting large-scale and fine-grained map construction.
2. **Who created the dataset and on behalf of which entity?**
   The dataset was developed by a consortium of ML researchers from CASIA, HKISI and employees from Tencent Maps listed in the author list.
3. **Who funded the creation of the dataset?**
   This work was supported in part by the National Key R&D Program of China (No. 2022ZD0116500), the National Natural Science Foundation of China (No. U21B2042, No. 62320106010), and in part by the 2035 Innovation Program of CAS, and the InnoHK program, and in part by the Tencent Maps Collaborative Research Project.

A.1.2 Composition

1. **What do the instances that comprise the dataset represent?**
   Our dataset contains satellite images and annotations. For each image, we annotate lane lines with three categories and eight attributes.
2. **How many instances are there in total (of each type, if appropriate)?**
   There are 1806 images in OpenSatMap19 with image size 2048 $\times$ 2048 and 1981 images in OpenSatMap20 with image size 4096 $\times$ 4096. For each image, we use polylines to annotate lane lines and 8 attributes (color, line type, number of lines, line function, bi-direction, boundary, occlusion and clearness) for each line. There are 446,645 lines in total.
3. **Does the dataset contain all possible instances or is it a sample (not necessarily random) of instances from a larger set?**
   Yes, it contains all possible instances.
4. **Is there a label or target associated with each instance?**
   Yes.
5. **Is any information missing from individual instances?**
   Yes. In OpenSatMap19, we can not provide the GPS information because of the regulation in China. However, this does not affect the use of our dataset.
6. **Are relationships between individual instances made explicit?**
   Yes. Our dataset is well-organized and the relationships between instances are explicit.
7. **Are there recommended data splits (e.g., training, development/validation, testing)?**
   Yes. We have already done this for users when the dataset releases.
8. **Are there any errors, sources of noise, or redundancies in the dataset?**
   Yes. Noise comes from some poor-quality images from Google Maps.
9. **Is the dataset self-contained, or does it link to or otherwise rely on external resources (e.g., websites, tweets, other datasets)?**
   The dataset is self-contained.

10. **Does the dataset contain data that might be considered confidential?**
    No.
11. **Does the dataset contain data that, if viewed directly, might be offensive, insulting, threatening, or might otherwise cause anxiety?**
    No.

A.1.3 Collection Process

1. **How was the data associated with each instance acquired?** Was the data directly observable (e.g., raw text, movie ratings), reported by subjects (e.g., survey responses), or indirectly inferred/derived from other data (e.g., part-of-speech tags, model-based guesses for age or language)? If the data was reported by subjects or indirectly inferred/derived from other data, was the data validated/verified?
   The data is acquired from Google Maps using static public API. The collection process is in accordance with Google Maps terms. Please refer to https://maps.google.com/help/terms_maps/ for more details.
2. **What mechanisms or procedures were used to collect the data (e.g., hardware apparatuses or sensors, manual human curation, software programs, software APIs)?**
   We use one NVIDIA A30 to run a program with the Maps Static API in Google Maps. We will provide the source code in our GitHub repository. The collection process is in accordance with Google Maps terms. Please refer to https://maps.google.com/help/terms_maps/ for more details.
3. **Who was involved in the data collection process (e.g., students, crowdworkers, contractors) and how were they compensated?**
   The authors in the author list collect the data and a professional remote sensing imagery labeling team of approximately 50 annotators label the data. The total cost of labeling is about $72,000. For each crowdworker, the hourly rate is $10.
4. **Over what timeframe was the data collected?**
   The data was collected over 7 weeks, during the Spring of 2024 (March 15th, 2024 through May 10th, 2024).
5. **Were any ethical review processes conducted (e.g., by an institutional review board)?**
   Yes. We obtain the publicly available images from Google Maps. Google has conducted the ethical review processes for the data.

A.1.4 Preprocessing/Labeling

1. **Was any preprocessing/cleaning/labeling of the data done?**
   Yes. We manually cleaned the images with poor quality (blurred, distorted, spliced, *etc*. and without roads.).
2. **Was the "raw" data saved in addition to the preprocessed/cleaned/labeled data (e.g., to support unanticipated future uses)?**
   Yes. We will provide it along with the release of the dataset.
3. **Is the software that was used to preprocess/clean/label the data available?**
   The annotators use software to help the annotation process, and the software is developed themselves.

A.1.5 Uses

1. **Has the dataset been used for any tasks already?**
   No, this dataset has not been used for any tasks yet.
2. **What (other) tasks could the dataset be used for?**
   It could be used to conduct instance-level line detection, satellite-enhanced online map construction for autonomous Driving, super-resolution image reconstruction, *etc.*.
3. **Are there tasks for which the dataset should not be used?**
   No.
4. **Is there anything about the composition of the dataset or the way it was collected and preprocessed/cleaned/labeled that might impact future uses?**

Yes. We only annotated the lane lines in the dataset, leaving a large amount of information unlabeled (*e.g.*, buildings, cars *etc.*), which limits the further uses of this dataset. We have already discussed this in Sec. 7.

A.1.6 Distribution

1. **Will the dataset be distributed to third parties outside of the entity (e.g., company, institution, organization) on behalf of which the dataset was created?**
   Yes, the dataset is open to the public.
2. **How will the dataset be distributed (e.g., tarball on website, API, GitHub)?**
   We intend to distribute this dataset through Hugging Face and tarball on website. The code for baselines will be released on GitHub.
3. **When will the dataset be distributed?**
   It will be distributed in fall 2024.
4. **Will the dataset be distributed under a copyright or other intellectual property (IP) license, and/or under applicable terms of use (ToU)?**
   The dataset will be licensed under a Creative Commons CC-BY-NC-SA 4.0 license. Meanwhile, the use of the images from Google Maps must respect the "Google Maps" terms of use (https://about.google/brand-resource-center/products-and-services/geo-guidelines/, https://maps.google.com/help/terms_maps/).
5. **Have any third parties imposed IP-based or other restrictions on the data associated with the instances?**
   No.
6. **Do any export controls or other regulatory restrictions apply to the dataset or to individual instances?**
   No.

A.1.7 Maintenance

1. **Who will be supporting/hosting/maintaining the dataset?**
   All the authors will be supporting/hosting/maintaining the dataset in the long term.
2. **How can the owner/curator/manager of the dataset be contacted (e.g., email address)?**
   Through our project page https://opensatmap.github.io/ which contains our email addresses or GitHub issues.
3. **Is there an erratum?**
   Not yet.
4. **Will the dataset be updated?**
   Yes, the datasets will be updated whenever necessary to ensure accuracy, and announcements will be made accordingly. All the updates can be found in our project page https://opensatmap.github.io/.
5. **If the dataset relates to people, are there applicable limits on the retention of the data associated with the instances?**
   Not applicable.
6. **Will older versions of the dataset continue to be supported/hosted/maintained?**
   Yes, older versions of the dataset will continue to be supported/hosted/maintained.
7. **If others want to extend/augment/build on/contribute to the dataset, is there a mechanism for them to do so?**
   Yes. They can contact us from our project page or post a GitHub issue.

A.2 Dataset Annotation Documentation

In this subsection, we provide a detailed description of our instance-level and image-level annotation rules, which helps the dataset consumers to better understand and use our dataset.

A.2.1 Overview

OpenSatMap is a high-resolution, geographically diverse, large-scale, satellite image dataset with fine-grained instance-level annotations. Besides, in order to advance the map construction in autonomous driving, OpenSatMap covers the regions of nuScenes [8] and Argoverse 2 [11] datasets.

A.2.2 Instance-level Annotation Rules

We use vectorized polylines to represent a line instance. We first categorize all lines into three categories: **curb, lane line,** and **virtual line**. A curb is the boundary of a road. Lane lines are those visible lines forming the lanes. A virtual line means no lane line or curb here, but logically there should be a boundary to form a full lane.

Each line is assigned a couple of attributes based on its appearance and functionalities. Specifically, there are 8 attributes as follows:

- Color: White, Yellow, Others, None.
- Line type: Solid line, Thick solid line, Dashed line, Short dashed line, Others, None.
- Number of lines: Single line, Double line, Others, None.
- Function: Chevron markings, No parking, Deceleration lane, Bus lane, Others, Tidal lane, Parking spaces, Vehicle staging area, Guidance line, Lane-borrowing area.
- Bidireciton: True, False.
- Boundary: True, False.
- Occlusion: No occlusion, Minor occlusion, Major occlusion.
- Clearness: Clear, Fuzzy.

Note that there is no man-made visible line on curbs and virtual lines, so we annotate their colors, line types, numbers of lines, and functions as None. In terms of line colors, line types, numbers of lines, functions, boundaries, and clearness, they are apparent and easy to distinguish.

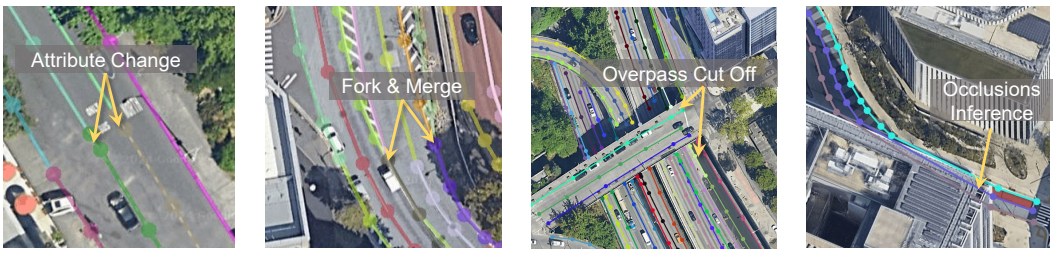
In the following, we give more details about bidirection and occlusion. For bidirection, the order of the points in the polylines defines the line direction. We can easily determine the direction of the lines by traffic flow, arrows, *etc*. However, it is hard to tell the directions of some lines, *e.g.*, the center line of a road, which is bidirectional. Therefore, we design bidirection attribute to handle this condition. For occlusion, we use the ratio of the lines to be blocked by trees, cars, and buildings. When the ratio is greater than 50%, we give it major occlusion. When the ratio is smaller than 50%, we label minor occlusion. When the line is fully visible, we label no occlusion.

After defining the line attributes, we give the definition of each instance using the following rules. Although we have already talked about the definition of the instance in Sec. 3.2, here is a more detailed discussion for better understanding. The detailed rules of each tag are shown below. *(i)* Different instances have different attributes. For example, if a solid line becomes a dashed line, we cut the line into two instances. Please refer to Figure 5a for an example. *(ii)* When a line fork or merge (*e.g.*, "Y"-shape point), we break it into three instances. Please refer to Figure 5b for an example.

In addition to these generalized rules above, there are some rules for specific situations.

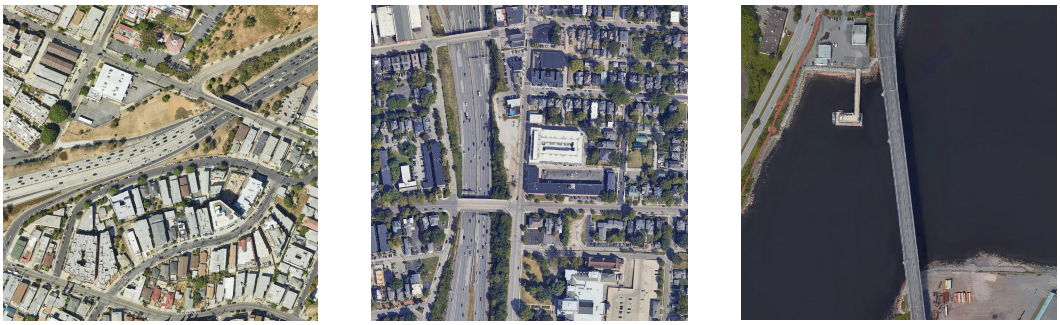
(i) Occlusion inference. If a line is partially occluded by buildings, trees, and vehicles but its complete shape can be inferred with high confidence, we will annotate it. Fully occluded lines are not labeled. However, the overpass is a special case and we will discuss it later.

(ii) Overpass. When there is a multi-level interchange, the upper-level overpass will obscure the lane lines on the lower level. Although in this case, the underlying lane lines can be inferred, the portion covered by the upper level of the interchange is *not labeled* to avoid conflicts with the upper ones. Besides, the lower overpass breaks off at this point for two instances.



(a) Attribute change. (b) Lines fork and merge. (c) Overpass cut off. (d) Occlusions inference.

Figure 10: **Definition of instances** (*zoom in* for best viewing). In (a), a change of line type results in two instances sharing the same point. In (b), a line should be divided into three instances when it is forked or merged. (c) shows an example of multi-level interchange. (d) shows the annotation inferences caused by the occlusion of the buildings.



(a) Dense. (b) Moderate. (c) Sparse.

Figure 11: Vehicle density examples.

A.2.3 Image-level Annotation Rules

We provide OpenSatMap20 with 4 additional image-level tags including image clarity, vehicle density, settlement type and special road structure to describe general information of each image. The following are detailed rules for each tag.

Image clarity. This attribute is used to describe the clarity of the image.

- Completely clear: Clear view of lane lines and surroundings.
- Partially clear: Some lane lines are clear, some are blurry or have ghosting, *etc.* Less than 30% of the lane lines in the image are blurred.
- Not clear: More than 30% of the roads are unclear. For images with this tag, we will not annotate them and remove them from OpenSatMap.

Vehicle density. We have three levels to describe the vehicle density of an image. Figure 11 shows an example of each case.

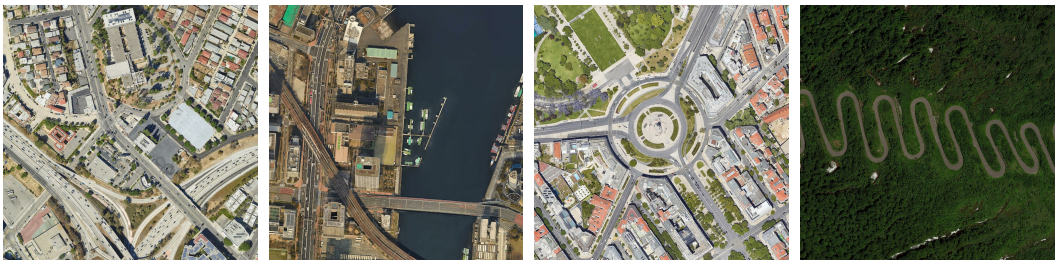
- Dense: The distance between cars is less than 5 vehicle lengths.
- Moderate: There are some cars on the road and the distance between cars is greater than 5 vehicle lengths.
- Sparse: Only less than 20% of major roads have cars and the rest have no cars.

Settlement type. Settlement type is used to describe the surroundings of the road. It includes suburban, rural and urban. Figure 12 shows an example of each case.

- Urban: The surroundings are all high rises and office buildings.
- Suburban: A form of urban settlement with medium population density, it can be seen as the middle ground between city and country life.

(a) Urban. (b) Suburban. (c) Rural.

Figure 12: Settlement type examples.



(a) Flyover. (b) Cross-river bridge. (c) Roundabout (d) Winding road.

Figure 13: Special road structure examples.

- Rural: A human settlement with a low population density. The number of buildings is the least of the three.

Special road structure. This tag describes the special road structures of an image and multiple special structures can exist in one image. We define four special road structures including roundabout, flyover (overpass), winding road and cross-river bridge. Figure 13 shows an example of each case.

B Details of Baseline Methods

B.1 Instance-level Line Detection

Implementation details. We use MMSegmentation [16] to conduct the semantic segmentation. For OpenSatMap19, we cut each image into 512 $\times$ 512 and for OpenSatMap20, we cut them into 1024 $\times$ 1024. After cutting process, we resize each image into 2048 $\times$ 2048, *i.e.*, we use "pseudo level-21 images" for segmentation. In the "pseudo level-21 images", we set the line width as 6 pixel. We follow the setting of SegNeXt [27] tiny and use cross entroy loss and dice loss as loss function. All the experiments are conducted on 8 RTX 3090 GPUs. Table 4 shows the hyperparameters we use.

In our baseline method, we use line categories (lane line, curb, background, virtual line) and line types (solid line, thick solid line, dashed line, short dashed line and others) to conduct semantic segmentation and the number of categories is eight. We get this number according to our definition of instances, *i.e.*, when one attribute changes, we should break the line into two instances. With this definition, we further find only line types and line categories may change between two connected line segments, *e.g.*, a solid line turns into a dashed line or a virtual line breaks the curb. So we only consider these line categories and line types in our semantic segmentation model. Other attributes do not influence the definition of instances and are not used in this benchmark.

B.2 Satellite-enhanced Online Map Construction for Autonomous Driving

In this subsection, we demonstrate that our dataset could help the online map construction for autonomous driving.

Table 4: Hyperparameters in instance-level line detection track.

Config	Value
optimizer	AdamW
learning rate	8e-5
weight decay	0.01
momentum	β_1, β_2 = 0.9, 0.999
batch size	8
learning rate schedule	cosine decay
warmup iterations	400
training iterations	40000
augmentation	RandomFlip
drop path	0.1
class weight in CE loss	1, 20, 30, 40

Task definition. We follow the setting of semantic map learning, which was first proposed by HDMapNet [34]. Inputs are camera images $\mathcal{I}$ of an autonomous driving vehicle and outputs are vectorized map elements $\mathcal{M}$ around the vehicle. SatforHDMap [23] adds the corresponding satellite images of each sample as additional input $\mathcal{I}_{\mathcal{S}} \in \mathbb{R}^{H \times W \times 3}$ and improves the performance, where $H \times W$ indicates the input resolution. It is a hyperparameter and decided by the region of a satellite map tile (*e.g.*, $60m \times 30m$). Based on SatforHDMap, we add the mask of the satellite images $\mathcal{I}_{\mathcal{MS}} \in \mathbb{R}^{H \times W}$ as an additional input.

Implementation details. We use the model from SatforHDMap [23] and add an additional branch to process the mask input. All the experiments only use the carema images and additional satellite images as input. We set the satellite image sampling area as $60m \times 30m$, and the corresponding satellite images tile size is shown in Table 5. Table 6 shows the hyperparameters for this track.

Table 5: Resolution of satellite map tiles in nuScenes.

Region	Resolution
Boston Seaport	545 × 273
Singapore One North	403 × 202
Singapore Queenstown	403 × 202
Singapore Holland Village	404 × 202

Table 6: Hyperparameters in satellite-enhanced online map construction track.

Config	Parameters
optimizer	Adam
learning rate	0.0016
weight decay	1e-7
momentum	β_1, β_2 = 0.9, 0.999
batch size	32
training epochs	30