

MachMap: End-to-End Vectorized Solution for Compact HD-Map Construction

Limeng Qiao^{1,*},[✉] Yongchao Zheng^{2,*},[†] Peng Zhang^{2,*},[†] Wenjie Ding^{1,*}
 Xi Qiu^{1,✉} Xing Wei² Chi Zhang¹
¹Mach Drive ²Xi'an Jiaotong University

{limeng.qiao, wenjie.ding, xi.qiu, chi.zhang}@mach-drive.com
 {zyc573823770, zp5070}@stu.xjtu.edu.cn weixing@mail.xjtu.edu.cn

Abstract

This report introduces the 1st place winning solution for the Autonomous Driving Challenge 2023 - Online HD-map Construction^b. By delving into the vectorization pipeline, we elaborate an effective architecture, termed as MachMap, which formulates the task of HD-map construction as the point detection paradigm in the bird-eye-view space with an end-to-end manner. Firstly, we introduce a novel map-compactness scheme into our framework, leading to reducing the number of vectorized points by 93% without any expression performance degradation. Build upon the above process, we then follow the general query-based paradigm and propose a strong baseline with integrating a powerful CNN-based backbone like InternImage, a temporal-based instance decoder and a well-designed point-mask coupling head. Additionally, an extra optional ensemble stage is utilized to refine model predictions for better performance. Our MachMap-tiny with IN-1K initialization achieves a mAP of 79.1 on the Argoverse2 benchmark and the further improved MachMap-huge reaches the best mAP of 83.5, outperforming all the other online HD-map construction approaches on the final leaderboard with a distinct performance margin (> 9.8 mAP at least).

1. Introduction

As one of the fundamental modules in the autonomous-driving, high-definition map (HD-map) provides centimeter level environment information for ego-vehicle navigation, including detailed geometric-topology relationships and semantic map categories, e.g. *ped-crossing*, *lane-divider* and *road-boundary*. Recently, with the development of deep neural network, online construction of local HD-map from onboard sensors (cameras) has gradually become a more advantageous and potential solution.

[†] Work done during an internship at Mach Drive

^{*} Equal Contribution Authors [✉] Corresponding Authors

^b <https://opendriveab.com/AD23Challenge.html#Track2>

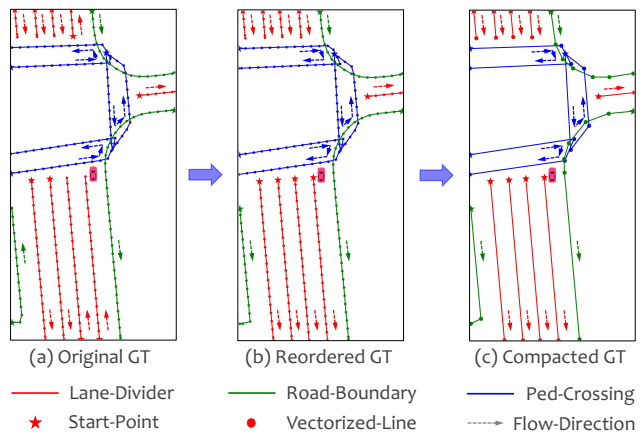


Figure 1. **The illustration of our HD-map processing principles.** (a) the original ground truth given in challenge. (b) reorder polylines to keep the *inter-element direction consistency*. (c) remove redundancy to keep the *intra-element sequence compactness*.

The Online HD-map Construction Track aims to dynamically construct local HD-map from onboard surrounding camera images. In this task, a local HD-map ground truth in Fig. 1 (a) is described by a set of map elements with three semantic categories and each element is designed to a polyline, which consists of a set of ordered points, to deal with complicated and even irregular road structures. Our method mainly focuses on three aspects to handle the competition, (1) **map modeling principles**. We propose the principles of *inter-element direction consistency* and *intra-element sequence compactness* to reduce the intrinsic redundancy of polyline-based map modeling. Concretely, without losing any expression performance, the flow directions of point sequences between different elements should be as consistent as possible, and the point sequences within the same map element should be reserved with as few points as possible. (2) **temporal-fusion instance decoder**. Based on the multi-cameras features from image backbone, we then employ a temporal-fusion based bird-eye-view (BEV) feature decoder for view-transformation and a bottom-up point-wise instance decoder to extract point descriptor.

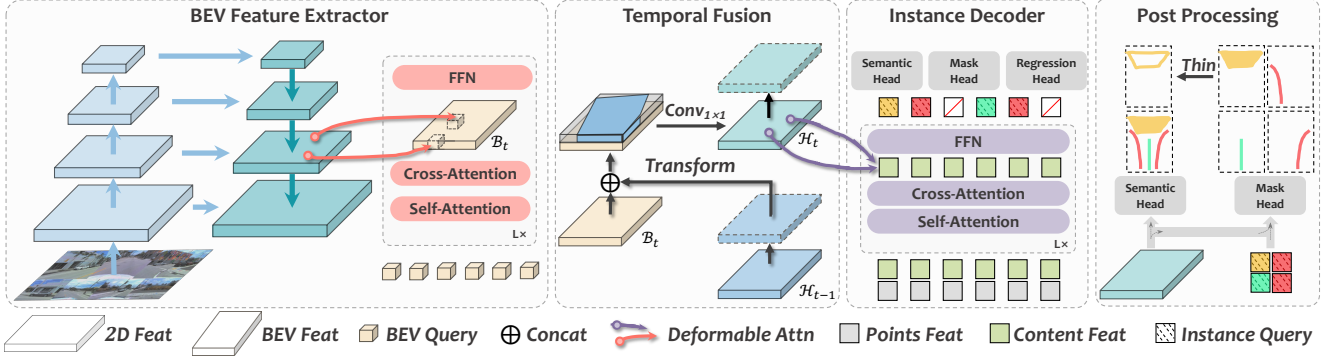


Figure 2. **The architecture of our proposed *MachMap*.** Given surrounding images, we generate 2D features from each of views through image backbone and neck. Then the deformable attention are used to aggregate the 3D feature among different views and average it along z-axis. The temporal fusion module fuses the new BEV feature with the one of hidden state of BEV feature, based on which the hidden state is updated. Finally, we conduct instance decoder which utilize instance-level deformable attention to refine content and points features and format the final results. It is worth noting that the results of *ped-crossing* and *lane-divider* are thinned from the mask.

(3) **point-mask coupling head.** Considering that different map elements have distinct shape priors, *e.g.* *lane-divider* is usually polyline and *ped-crossing* is convex polygon, we equip each semantic map category with both segmentation and detection heads under the MaskDINO [6] framework, which greatly improves the flexibility and scalability of our model. Furthermore, the above multi-task training strategy also accelerates the model convergence performance.

Inspired by the above motivations, we propose an end-to-end vectorized *HD-map* construction architecture, named as *MachMap*. The entire framework is illustrated in Fig.2 and all technical details are presented in the next section.

2. Method

This section introduces the details of our winning method. We first present the map compaction pipeline, which significantly reduces the difficulty of model training and makes the inference results more compact and efficient. Next the design scheme of each module is presented, and some task-specific improvements are integrated into some off-the-shelf methods. Lastly, we introduce our novel ensemble ideas, which can further enhance our approach.

2.1. Map Compaction Pipeline

Different from rasterized scheme, vectorized *HD-maps* in the given annotations explicitly express the spatial relation between map elements and instance information in their respective categories. Following the newly proposed map modeling principles, we compact the original evenly sampled map representation in two steps, namely orientation rearrangement and redundancy removal.

(1) **inter-element direction consistency.** The directions of elements in original map annotations are in a state of chaos, such as *lane-dividers* of moving forward from front-to-back or back-to-front as shown in Fig.1 (a). We noticed that the inconsistency of directions can negatively affect the training

of the model. To reduce the discreteness of map organization, we follow a certain strategy to make the orientation of map elements as orderly as possible, and guarantee that this process does not lose any details of the map. Specifically, under the principle of conforming to the observation order of human eyes, a simple and intuitive strategy is to reorganize all polylines according to the rules *from-front-to-back* and *from-left-to-right* in bird-eye-view space.

(2) **intra-element sequence compactness.** Vectorized maps with evenly-distributed points have redundant semantic information, while compacted-points representation is sparse, which is more suitable for expression and storage of maps. To this end, we extract keypoints for all elements to supervise model training. Concretely, we adopt Douglas-Peucker algorithm [11] and Visvalingam algorithm [12] to condense a polyline composed of line segments to a similar polyline with fewer points. For these methods, points are removed in order of least to most importance, with importance related to the distance and triangular area respectively.

2.2. MachMap Architecture

We follow the general query-based design paradigm [5, 10], as illustrated in the Fig.2, where the overall structure can be roughly divided into three parts: *BEV feature extractor*, *temporal-fusion instance decoder*, and *point-mask coupling head*. Afterwards, we introduce each module sequentially according to the flow of information.

Backbone. Giving a list of 2D images $\mathcal{I} \in \mathcal{R}^{N \times 3 \times H \times W}$, extracting unified textures representation within images is a top-priority task. With regards to this, we utilize a shared InternImage [13] as strong backbone to extract image features, which employs deformable convolutions [1] as its core operator and has been meticulously designed. During the downsampling process, a series of feature maps in varying scales are generated and then aggregated by the Bi-directional Feature Pyramid Network, *i.e.* BiFPN [16].

Multi-view Encoder. Since the map vectors we ultimately need to predict lay in 3D space, it is necessary to elevate surrounding features from camera-view to 3D ego-view space. Rather than direct transformation to 3D-view, we predefine a set of reference points and arrange them in a *BEV* raster. After that, we employ the camera intrinsics and extrinsics to project them onto several images and then aggregate the surrounding features. By averaging on the z -axis, we obtain the final bird-eye-view features $\mathcal{B} \in \mathcal{R}^{H_B \times W_B \times C}$.

Temporal Fusion Module. The provided dataset is collected and organized chronologically, with precise poses for each sample. This makes it possible to align current features with previous ones by poses, resulting in a larger real-world perception range beyond the current position. We follow the long-term fusion strategy proposed in VideoBEV [3], which affines the previous hidden state \mathcal{H}_{t-1} of *BEV* feature into the current one \mathcal{B}_{t-1} using vehicle ego pose. The latter is concatenated with the current *BEV* feature \mathcal{B}_t in the channel dimension and fused by a 1×1 convolutional layer as,

$$\mathcal{B}_{t-1} = \text{Affine}_{pre \rightarrow cur}(\mathcal{H}_{t-1}) \quad (1)$$

$$\mathcal{H}_t = \text{Conv}_{1 \times 1}(\mathcal{B}_{t-1} \oplus \mathcal{B}_t) \quad (2)$$

where \oplus denotes the concatenation operator. The fused features are cached as the next hidden state and used as input for subsequent instance decoder. In practice, since the timestamp offset between adjacent frame is too small, we group the timestamps at specific intervals to expand the performance gain brought by this temporal-fusion module.

Instance Decoder. To benefit from multi-task loss, we opt for the MaskDINO [6] framework, which conducts object detection and segmentation tasks simultaneously. Each query consists of content and position vectors, with the former is utilized to generate instance masks, while the latter undergoes iterative updates to yield normalized coordinates directly. Yet, due to the hierarchical relationship between map elements and their corresponding points sets, we adopt the query design paradigm in MapTR [7] for better adaptation to map element modeling. This implies that the query is point-wise, and a set of which can be aggregated to form a single instance and obtain its corresponding instance mask.

Output Head. Using only coordinates from point regression has some drawbacks. Firstly, there is a keypoint mismatching issue, where a well predicted instance may occur a mismatched point which belongs to other instance, as a result, a single bad apple spoils the whole bunch. Secondly, for *ped-crossing*, there exists a strong geometric prior, which is difficult to depict through vectors. However, masks not only can effectively constrain the geometry shape of instances, but it also impose a significant penalty on mismatched points during training. Empirically, we obtain *ped-crossing* and *lane-divider* through post-processing of instance masks, while point regression is employed only for *road-boundary*. As the common practice, we adopt cross-

entropy and dice loss [9] for masks and L1 loss for point regression. In addition, we also add semantic loss to the *BEV* features as auxiliary supervision, and our final loss as,

$$\mathcal{L} = \lambda_{cls} \mathcal{L}_{cls} + \lambda_{pts} \mathcal{L}_{pts} + \lambda_{mask} \mathcal{L}_{mask} + \lambda_{sem} \mathcal{L}_{sem} \quad (3)$$

where λ_* is the balance weight for different losses.

2.3. Ensemble Strategy

The predicted map vectors of our model are represented in normalized coordinates, which are then rescaled to the actual range $60 \times 30m$ in the ego coordinate system during the post-processing stage. Yet the actual visible content from images greatly exceeds this range, which often leads to ambiguities in the existence of certain elements at the border position of exact map region that may be ignored by a single model. Accordingly, the use of ensemble techniques can mitigate prediction variability and curb overfitting by summarizing multiple models together.

By utilizing chamfer distance as a metric for measuring the similarity between instances, we present the ensemble algorithm in the Algorithm 1. Given a base set and a list of proposals, which are derived from multiple other predictions and sorted by confidence in descending order, we can compare each proposal with the base set one by one. If their similarity is low, we can consider them as missed true positives and add them to the base set. In addition to multi-model ensemble, we also conduct multi-frame ensemble. Despite the utilization of temporal fusion module, some instances are still absent, which were accurately recalled in previous frames. This inspires us to compensate some erratic predictions by ensemble with predictions from previous frames. It’s worth noting that the integration of multi-frame and multi-model can share the same algorithm, with only modifying the source of candidate proposal list.

Algorithm 1 MachMap Ensemble Algorithm

Input: Base-list B , Proposal-list P and score-list S , CD-threshold T
Output: Added proposal list and score A , AS

```

1:  $P, S \leftarrow \text{SortProposalByScore}(P, S)$ 
2:  $A \leftarrow [], AS \leftarrow []$ 
3: while  $P.length \neq 0$  do
4:    $Flag \leftarrow False$ 
5:    $Head \leftarrow P.pop, HeadScore \leftarrow S.pop$ 
6:   for  $Base$  in  $B$  do
7:      $Sim \leftarrow \text{ChamferDistance}(Head, Base)$ 
8:     if  $Sim < T$  then
9:        $Flag \leftarrow True$ 
10:      break
11:    end if
12:  end for
13:  if  $Flag$  then
14:     $B.append(Head)$ 
15:     $A.append(Head)$ 
16:     $AS.append(HeadScore \cdot \sigma)$   $\triangleright \sigma$  is a score decay factor
17:  end if
18: end while

```

Category	# images	# instances	# points (raw)	# points (compacted)	AP _{0.2m}	AP _{0.3m}	AP _{0.4m}	AP _{0.5m}
<i>ped-crossing</i>	19523	55686	3593548	252219 (↓ 93.0%)	98.33	99.46	99.92	100.00
<i>lane-divider</i>	26222	133186	7426425	335534 (↓ 95.5%)	99.91	99.98	99.99	100.00
<i>road-boundary</i>	27283	84384	7018193	469533 (↓ 93.3%)	97.38	99.70	99.92	100.00

Table 1. **The effectiveness and correctness verification of map compaction principles.** All statistical numbers are collected on both training and validation sets. Note that # means 'the number of' and the blue color means the proportion of point reduction. AP_τ indicates that the average precision between before and after the compaction, where a prediction as true-positive only if the distance is less than τ.

ID	Data	Backbone	PreTrain	# Epochs	w/o <i>Opt.</i>	AP _{crossing}	AP _{divider}	AP _{boundary}	mAP
1	<i>train</i>	<i>tiny</i>	ADE20K	6	✗	61.01	65.87	65.70	64.19
2	<i>train</i>	<i>tiny</i>	ADE20K	72	✗	76.75	73.51	74.68	74.98
□	<i>train+val</i>	<i>tiny</i>	ADE20K	72	✗	78.34	74.74	76.02	76.37
3	<i>train+val</i>	<i>tiny</i>	<i>from</i> - □	↔ 6	✓	84.82	79.66	80.63	81.70
♡	<i>train+val</i>	<i>huge</i>	ADE20K	36	✗	81.45	75.34	77.14	77.98
4	<i>train+val</i>	<i>huge</i>	<i>from</i> - ♡	↔ 12	✓	86.66	81.54	82.29	83.50
△	<i>train+val</i>	<i>tiny</i>	IN-1K	72	✗	76.46	72.32	75.91	74.90
5	<i>train+val</i>	<i>tiny</i>	<i>from</i> - △	↔ 6	✓	82.01	76.23	79.10	79.11

Table 2. **The performance of different MachMap milestone models** under thresholds of [0.5, 1.0, 1.5]m. we employ InternImage [13] as backbone and '*tiny/huge*' means its scale. The '*from* - □/♡/△' means loading corresponding checkpoint and ↔ is more epochs finetuning. Note the weights of ADE20K/IN-1K are public. The term *Opt.* means our improving techniques, e.g. *ema, ida, temporal* and *ensemble*.

3. Experiments

3.1. Existing Benchmarks

The Argoverse2 [14] contains 700, 150 and 150 video clips in the training, validation, and testing sets respectively. Each sequence has 6-DOF map-aligned pose and seven ring views with the image resolution of 2048 × 1550 or 1550 × 2048 pixels. The given data from challenge is a subset of Argoverse2. We utilize all frames from the challenge training set to verify the effect of different ablations but finally all frames from training and validation sets are used to reach better performance. We focus on three categories, i.e. *lane-divider, ped-crossing* and *road-boundary*.

3.2. Implementation Details

Training Setup. We adopt common data augmentation, e.g. random scaling, cropping, and flipping. At the same time, an *IDA* [4] matrix is updated to record view transformation to maintain spatial consistency. Then the final input shape is fixed at 896 × 768, as this aspect ratio is close to the front view, i.e. 2048 × 1550, which contains the most abundant visual map information. For *BEV* features, the default spatial shape of *BEV* queries is 64 × 32, which corresponds to the perception ranges in lidar coordinate system are [-30, 30]m for the *Y*-axis and [-15, 15]m for the *X*-axis. Note all map masks are interpolated to 400 × 200 to ensure that distinct elements can be easily distinguished without occupying too much memory. As for the hyperparameters of loss function, we set $\lambda_{cls}, \lambda_{pts}, \lambda_{mask}, \lambda_{sem}$ to 2, 20, 1, and 3 respectively.

Training Strategy. We train our model with a total batch of 8 on 8 GPUs. The AdamW [8] optimizer is employed with

Rank	Team	AP _{crossing}	AP _{divider}	AP _{boundary}	mAP
1	Mach (ours)	86.66	81.54	82.29	83.50
2	MapNeXt	68.94	76.66	75.34	73.65
3	SCR	70.37	75.08	74.73	73.39
4	LTS	72.67	73.20	71.80	72.56
5	USTC-VGG	69.05	73.24	70.76	71.02

Table 3. Top 5 entries on the test leaderboard of challenge.

a weight decay of 5×10^{-2} and a learning rate of 3×10^{-4} . Our training process consists of two stages: base training and fine-tuning. Firstly, we initialize the InternImage [13] with public pretrained weights [2, 15] and then train our model for 60 epochs without any tricks except a multi-step schedule with milestone [0.7, 0.9] and $\gamma = \frac{1}{5}$. Afterward, we apply all proposed improving techniques to fine-tune the model for extra epochs with a learning rate of 1×10^{-4} .

3.3. Experimental Results

Table 1. Our statistical results show the compacted map can reduce more than 93% points without expression performance losing under the threshold of 0.5m, even it can still maintain more than 97% performance under stricter 0.2m.

Table 2. Comparing the results in row-1&2, training more epochs brings a performance gain of more than 10 points, which shows that accelerating the convergence speed is still a vital future work. Compared with row-3&4, 5&6, 7&8, using the proposed improving techniques can always bring more than 5 points of increase. Moreover, even starting with IN-1K as pretrained weights, our model still achieves 79.1. **Table 3.** We succeed the championship with a performance advantage of 9.85 mAP over the second place, demonstrating the effectiveness of our proposed *MachMap* method.

References

- [1] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017. 2
- [2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 4
- [3] Chunrui Han, Jianjian Sun, Zheng Ge, Jinrong Yang, Runpei Dong, Hongyu Zhou, Weixin Mao, Yuang Peng, and Xiangyu Zhang. Exploring recurrent long-term temporal fusion for multi-view 3d perception. *arXiv*, 2023. 3
- [4] Junjie Huang, Guan Huang, Zheng Zhu, Ye Yun, and Dalong Du. Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. *arXiv preprint arXiv:2112.11790*, 2021. 4
- [5] Feng Li, Hao Zhang, Shilong Liu, Jian Guo, Lionel M Ni, and Lei Zhang. Dn-detr: Accelerate detr training by introducing query denoising. In *CVPR*, 2022. 2
- [6] Feng Li, Hao Zhang, Huaizhe Xu, Shilong Liu, Lei Zhang, Lionel M Ni, and Heung-Yeung Shum. Mask dino: Towards a unified transformer-based framework for object detection and segmentation. In *CVPR*, 2023. 2, 3
- [7] 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. 3
- [8] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. 2019. 4
- [9] F. Milletari, N. Navab, and S. A. Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. *IEEE*, 2016. 3
- [10] Limeng Qiao, Wenjie Ding, Xi Qiu, and Chi Zhang. End-to-end vectorized hd-map construction with piecewise bezier curve. In *CVPR*, 2023. 2
- [11] Mahes Visvalingam and J Duncan Whyatt. The douglas-peucker algorithm for line simplification: re-evaluation through visualization. In *Computer Graphics Forum*, volume 9, pages 213–225. Wiley Online Library, 1990. 2
- [12] Mahes Visvalingam and Peter J Williamson. Simplification and generalization of large scale data for roads: a comparison of two filtering algorithms. *Cartography and Geographic Information Systems*, 22(4):264–275, 1995. 2
- [13] Wenhai Wang, Jifeng Dai, Zhe Chen, Zhenhang Huang, Zhiqi Li, Xizhou Zhu, Xiaowei Hu, Tong Lu, Lewei Lu, Hongsheng Li, et al. Internimage: Exploring large-scale vision foundation models with deformable convolutions. In *CVPR*, 2023. 2, 4
- [14] B. Wilson, W. Qi, T. Agarwal, J. Lambert, J. Singh, S. Khandelwal, B. Pan, R. Kumar, A. Hartnett, and J. K. Pontes. Argoverse 2: Next generation datasets for self-driving perception and forecasting. In *Neural Information Processing Systems*, 2021. 4
- [15] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 633–641, 2017. 4
- [16] Lei Zhu, Zijun Deng, Xiaowei Hu, Chi-Wing Fu, Xuemiao Xu, Jing Qin, and Pheng-Ann Heng. Bidirectional feature pyramid network with recurrent attention residual modules for shadow detection. In *ECCV*, 2018. 2