LDTR: Transformer-based Lane Detection with Chain-anchor Representation

Zhongyu Yang[1], Chen Shen[2], Wei Shao[2], Tengfei Xing[2], Runbo Hu[2], Pengfei Xu[2], Hua Chai[2], and Ruini Xue[1](✉)

Abstract Despite recent advances in lane detection methods, scenarios with limited or no-visual-clue of lanes due to factors such as lighting conditions and occlusions remain a significant challenge. These scenarios are both common and crucial for automated driving. Moreover, current lane representations require complicated post-processing and fail to address special instances adequately. Inspired by the DETR architecture, we propose LDTR, a transformer-based model to overcome these problems. Lanes are modeled with Chain-anchor, regarding a lane as a whole from the beginning, which enables LDTR to handle special lanes inherently. To improve lane instance perception, LDTR incorporates a multi-referenced deformable attention module to distribute attention around the object. Additionally, LDTR devises two Line IoU algorithms to improve convergence efficiency and employs a Gaussian heatmap auxiliary branch to enhance model representation capability during training. By introducing Fréchet distance, parameterized F1-score and additional synthetic metrics are discussed to evaluate lane detection models more reasonably. Experimental results show that LDTR achieves SOTA performance on well-known datasets. Our code will be released soon.

Keywords Transformer, Lane detection, Chain-anchor.

1 Introduction

Autonomous driving is an important application of deep learning, in which the ability to perceive road elements is particularly crucial, especially for lane markings as one of the most essential components of road traffic signs. However, due to the complexity of road scenes and lane deformation from varying perspectives, accurate lane detection remains challenging, such as recalling lanes with little or no-visual-clue, and precise representation of special lanes.

Given good visibility and simple road conditions, traditional vision research [1, 2] performs very well, but it is not robust in complicated real-world scenarios. Recently, various DNN models [3–6] are trained on large-scale datasets and can infer lane positions via deep semantic features, delivering much better generalization and robustness compared to traditional approaches.

Some research leverages semantic segmentation [3, 5] to identify lanes by classifying pixels or picking up several keypoints. However, it is difficult to separate different instances from the lane foreground produced by semantic segmentation for lanes that are very close to each other. Instead, some other research turns to a top-down mechanism. LaneATT [9] first predicts a large number of candidates, then post-processes with Non-Maximum Suppression (NMS). Similarly, NMS can not tell apart adjacent lanes precisely which results in false deletions. CondLaneNet [6] and CANet [8] obtain lane instances by detecting keypoint responses on heatmaps, but due to the local perception characteristics of CNN, keypoints often respond weakly when the visual features are far away from them, which is prone to missed detections very likely. HoughLaneNet [10] leverages DHT-based feature aggregation to detect lanes with weak visual features, but is only applicable to straight lanes. To mitigate these challenges, emerging transformer-based research [11–15] uses the global attention mechanism to extract implicit semantic information. O2SFormer [14] proposes one-to-several label assignment to address label semantic conflicts. Chen [15] improves the convolutional kernel generation network of CondLaneNet [6] with transformer, thus the object query, after being processed by attention calculation, results in generated kernels pos-

1 University of Electronic Science and Technology of China, Chengdu, 611731, China. E-mail: Z. Yang, 202021080612@std.uestc.edu.cn; R. Xue, xueruini@uestc.edu.cn(✉).

2 Didi Chuxing, Beijing, 100081, China. E-mail: C. Shen, jayshenchen@didiglobal.com; W. Shao, wayneshaowei@didiglobal.com; T. Xing, xingtengfei@didiglobal.com; R. Hu, hurunbo@didiglobal.com; P. Xu, pengfeixu@didiglobal.com; H. Chai, chaihua@didiglobal.com.

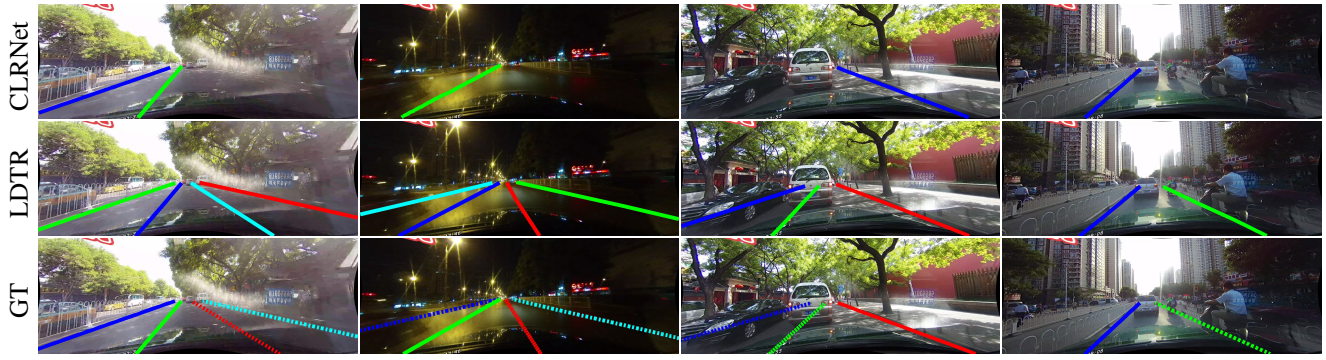


Fig. 1 Prediction results of the current SOTA method (CLRNet [7]) and LDTR for little or no-visual-clue lanes, like lens flare, weak lighting, occlusion, and hidden lines, in CULane dataset. CLRNet misses certain lanes, while LDTR can recall all instances.

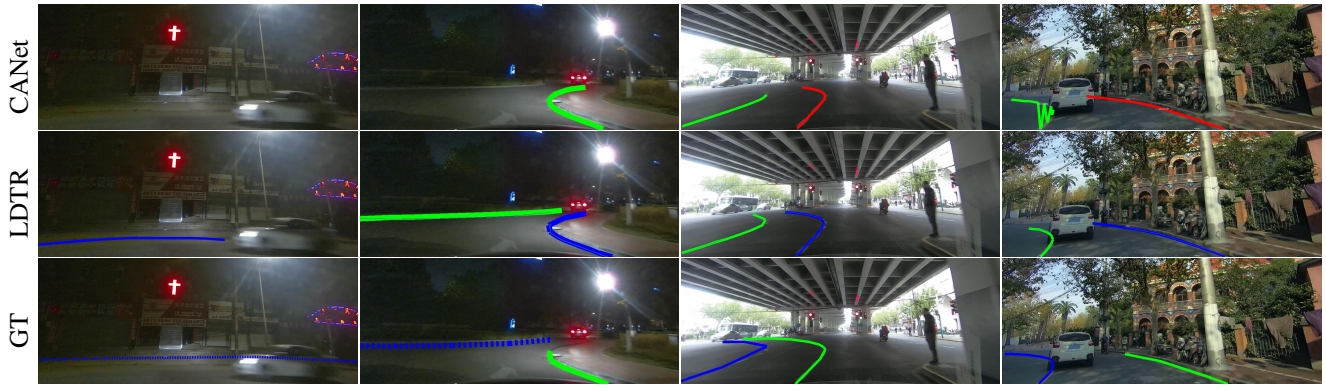


Fig. 2 Prediction results of current SOTA method (CANet [8]) and LDTR in CurveLanes dataset. Limited by its lane representation, CANet cannot describe lanes in special cases like T-junctions, roundabouts, waiting areas, and sharp turns, while LDTR can address all of them.

sessing more global information. However, these methods do not thoroughly utilize target locations to focus on attention. Therefore, a great amount of the attention calculation is wasted on the background which is irrelevant to the targets though, limiting the models' perception ability towards the lanes. Fig. 1 shows some cases they are hard to deal with, especially for those of little or no-visual-clue, which are critical for common downstream tasks, like lane keeping, and map-based lane information collection.

Additionally, current lane representations are not suitable for cases in Fig. 2. Existing methods usually rely on manually crafted post-processing rules based on priori assumptions. Almost all methods [5, 6, 14–18] assume that lanes extend from the bottom of the view upwards, thus fail to detect those horizontal lanes illustrated in the first three columns of Fig. 2. Although CANet [8] uses an adaptive post-processing decoder to avoid such assumptions, a decoder depending on manual settings cannot cover cases like the last column of Fig. 2. It is necessary to represent these lanes properly with improved representations since they are common in daily driving.

To address the aforementioned problems, we propose a new top-down end-to-end lane detection network based on transformer, LDTR (**L**ane **D**etection **TR**ansformer). Specifically, we propose Chain-anchor to represent the shape of lanes and two new loss functions to supervise their overall trend and detailed descriptions. Moreover, to enhance LDTR's ability and efficiency during deep semantic information extracting, the multi-referenced deformable cross-attention algorithm is applied in the transformer decoder, along with the addition of auxiliary branches to extract more detailed target information. As the second row of Fig. 2 demonstrates LDTR's outstanding performance in various challenging scenarios, comprehensive experimental results indicate that LDTR achieves state-of-the-art performance on multiple datasets [3, 19].

The main contributions of this paper are as follows:

- A new lane representation method called "*Chain-anchor*" is proposed. Instead of describing lanes in discrete dots (pixels or keypoints), Chain-anchor considers multiple nodes as a whole, enabling it to avoid the rule-based post-processing and additional efforts on complicated lane shapes. Moreover, Chain-anchor requires fewer points to denote the important turning points of a lane, which is more efficient and precise.

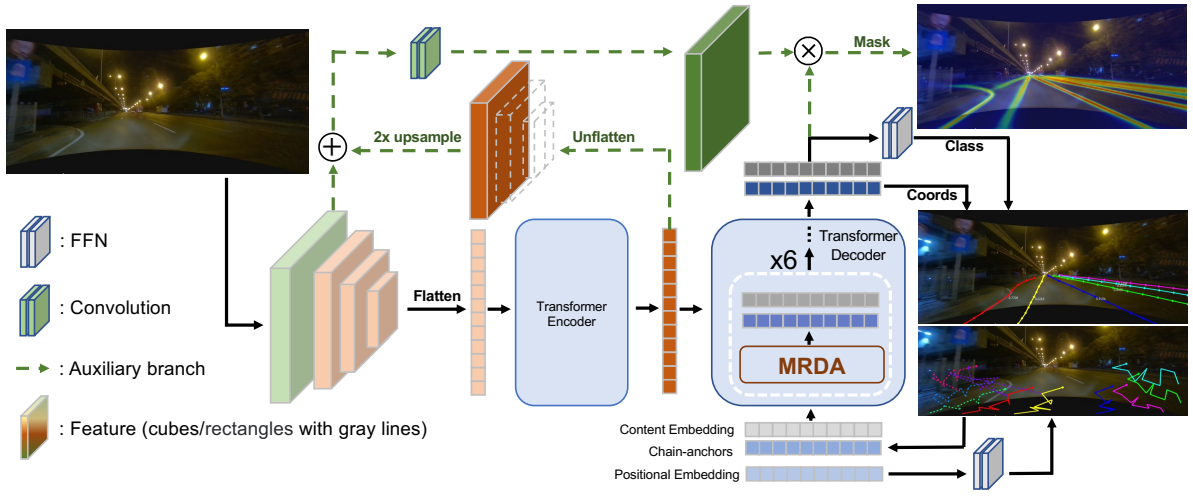


Fig. 3 LDTR follows the structural paradigm of DETR. After the 2D image features are extracted by the backbone, LDTR further extracts deep semantic information in the encoder through the self-attention mechanism. The input object queries to the decoder are composed of content embeddings and Chain-anchors. In the computation of each decoder layer, the object queries update themselves through MRDA and interact with image features, including the correction of Chain-anchors and differentiation of positive or negative objects. After 6 update iterations, the positive Chain-anchors could represent the lane shapes accurately. Additionally, LDTR introduces a Gaussian heatmap auxiliary branch to enhance the ability for the object query to perceive lane details.

- A multi-referenced deformable attention module (MRDA) is proposed to transmit the position prior information contained in Chain-anchor to the network, which evenly distributes attention around the targets. This, combined with the global semantic information extracted by the Encoder, enhances the model's perception ability toward the targets in case of little or no-visual-clue.
- Two Line IoU algorithms are devised, namely the "Point-to-Point" (P2P) IoU and "Dense-Sampling" (DS) IoU. They are applied in binary matching cost and loss during training, respectively. Compared to the traditional point-to-point L1 distance, the new algorithms introduce global optimization to improve training efficiency and inference performance.
- To facilitate downstream tasks, we evaluate LDTR with typical metrics, parameterized F1-score as well as synthetic metrics on public datasets. Experimental results demonstrate that LDTR performs better overall.

2 Related Work

We relate our work to both the existing lane detection approaches as well as the object detection methods in general.

2.1 Lane Detection

Deep learning-based lane detection algorithms can be boiled down into two main categories: *bottom-up* and *top-down*.

For bottom-up methods, they cluster or classify lane pixels or keypoints. Pixel-level segmentation [3–5], evolving from general visual semantic segmentation, first extracts foreground lane pixels using semantic segmentation and then clusters or classifies them using techniques like pixel embedding to differentiate between different lane instances. In contrast, keypoints detection-based methods [16, 17] can be regarded as sparse versions of segmentation models that replace dense pixel classification with discrete keypoints, which partially alleviates the problem of excessive focus on segmentation boundaries in pixel-level segmentation. However, bottom-up methods are usually unable to handle branching or merging lanes, and accurately distinguish adjacent boundaries of multiple closely located lanes. LDTR is designed to address all such challenging scenarios.

Top-down methods first obtain target instances and then refine the representation of the shape for each instance. Basically, there are three major categories based on the lane representation: parameterized curve fitting [11, 20], tilted anchor [7, 9, 14, 21], and row-wise classification [6, 10, 15, 18].

Curve fitting models complex lanes as simple polynomial curves, which could be very efficient due to the small number of descriptors that need to be predicted. However, the curves are hard to match sophisticated lanes in the real world, leading to poor precision and flexibility. Tilted anchor based methods obtain a large number of proposals by placing dense anchors, then filter out unqualified and overlapping instances through NMS in the post-processing stage. However, double solid lines and adjacent lanes are placed closely together under specific perspectives, which may be incorrectly deleted by

NMS during deduplication, leading to critical information loss. In contrast, row-wise classification improves efficiency based on the observation that lanes appear vertically often. Unfortunately, this also limits the expression for lanes that appear almost horizontally, like high-angle lanes [8] and U-turn lanes.

Different from the these three approaches, LDTR uses set prediction to distinguish lanes that are close in position, whose anchor ensures precise details while performing well in various complicated cases, such as U-turns, T-junctions, and roundabouts, commonly found in real-world situations and must be addressed.

2.2 Object Detection

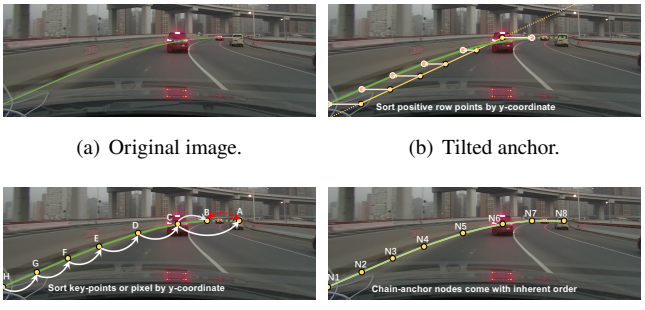
Object detection is closely related to lane detection, and many of its ideas and techniques can be leveraged directly. Those early CNN-based methods [22–26] mostly require rule-based post-processing operations, which can lead to poor model performance in some uncommon scenarios. DETR [27] proposed a new paradigm of end-to-end object detection, but suffers from problems like long training iterations and high computational cost. DETR inspired lots of following research. Deformable DETR [28] transformed the dense attention operation in the original cross-attention mechanism into a more efficient sparse attention mechanism by reference point sampling, reducing computational cost while improving model training convergence speed. DAB-DETR [29] explicitly modeled the object query as an anchor, using the position of the bounding box to guide attention focus near the target, which further optimizes the training iteration and improves model performance.

The global attention mechanism in the DETR paradigm equip the model with powerful semantic awareness capability, which helps to improve the model performance in cases of little or no-visual clues. Based on the structures and optimization techniques of DETR-family, this paper proposes an end-to-end lane detection model, LDTR.

3 Methods

3.1 Network Architecture

As a transformer-based model, LDTR is inspired by DETR [27] architecture as shown in Fig. 3. The black solid line indicates the lane prediction process. Firstly, LDTR takes a front view as the network input and extracts feature at different levels through a backbone network composed of multiple layers of CNN. The high-level features are reduced to one dimension and input into the transformer encoder for further interaction and output. Secondly, the transformer decoder takes a small fixed number of learnable content embeddings and Chain-anchors (Section 3.2) as input object queries, computes MRDA (Section 3.3) with the output of the encoder and outputs the modified content embeddings and anchors. Finally, the modified content embeddings are passed to the shared parameter feed-forward network (FFN), which predicts the presence or absence of targets, and the Chain-anchor could accurately describe target positions.



(a) Original image. (b) Tilted anchor.

(c) Keypoints/Segmentation. (d) Chain-anchor.

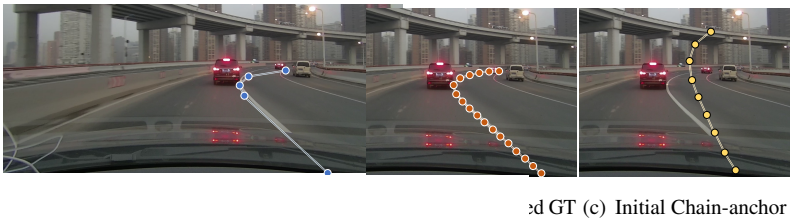
Fig. 4 Various lane representations. It is hard for current methods to represent the horizontal part of lanes, but easy for Chain-anchor.

3.2 Chain-anchor

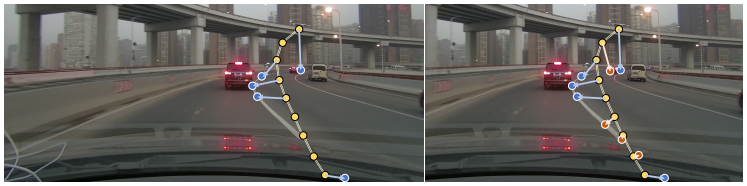
There are various lane representations available, such as tilted anchor lines in Fig. 4(b) [9], row-wise classification [6, 18] predicting a set of points and sorting them by y-coordinates [16, 17], and keypoints with adaptive decoders [8] in Fig. 4(c). Most of them share the same assumption that lanes extend vertically in the view. Therefore, models with such assumption usually have promising performance metrics for such lanes, but fail to recall the horizontal ones. Particularly, tilted anchor does not support curved lanes, while keypoints-based anchor can not sort the keypoints properly in case of storngly curved lanes in which y-coordinates are messed up.

To address these chanlleges, LDTR describes a lane as a whole with "Chain-anchor", $\text{Lane}_{ca} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$, where N is the number of nodes in Chain-anchor, x_i, y_i are the normalized relative positions with values in the range of [0, 1] as shown in Fig. 5.

Regarding supervision, LDTR uses two types of ground truths: an ordered set of manually annotated nodes, Lane_m (Fig. 5(a)), and a densely sampled set of nodes, Lane_s (Fig. 5(b)), obtained by uniformly sampling along the Lane_m. To predict the lane, Lane_{pr} (Fig. 5(c), the initial Chain-anchor), LDTR first matches it with Lane_m using the Hungarian algorithm [30]. Nodes in Lane_{ca} that have not been successfully matched are then matched with Lane_s using the



:d GT (c) Initial Chain-anchor



(d) Chain-anchor w/ mannual GT (e) Chain-anchor w/ uniform sampled GT

Fig. 5 The regression supervision approach of Chain-anchor enables it to efficiently utilize a small number of nodes to accurately describe curves, similar to how humans recognize lanes in essense.

same algorithm, ensuring that each node in Lane_{pr} is matched with one corresponding ground truth, forming the final Chain-anchor. Finally, LDTR employs L1 distance to supervise the horizontal and vertical coordinates of each predicted node on the Chain-anchor, and the loss is presented in Equation (1).

$$L_{reg} = -\frac{1}{N} \sum_{(x,y)\in \text{Lane}} |\hat{P}_{xy} - P_{xy}| \tag{1}$$

where $\hat{P}_{xy}$ and P_{xy} denote the predicted and the ground truth nodes, respectively.

Different from the uniform sampling in Point Set [31], LDTR samples $M (M \gg N)$ nodes on the annotated lane and performs Hungarian matching between these nodes and predictions. It matches each predicted node with the closest ground truth, giving the nodes higher degrees of freedom. This allows Chain-anchor to learn the implicit human preferences in the annotations during training and distribute the nodes at higher information density turning points. Besides, thanks to the low prior assumption setting, Chain-anchor can describe lanes of any shape and requires no longer prior conditions.

In addition, Chain-anchor can also provide fine-grained position information for the network. The cross-attention module needs to gather features from the entire feature map, thus it is necessary to provide appropriate position priors for each query to focus attention on the locality surrounding the targets. LDTR explicitly models the query position as an anchor, which is a similar approach to DAB-DETR [29]. Chain-anchor can effectively help the network aggregate features from nearby regions of different parts of a target using multi-referenced deformable cross-attention modules.

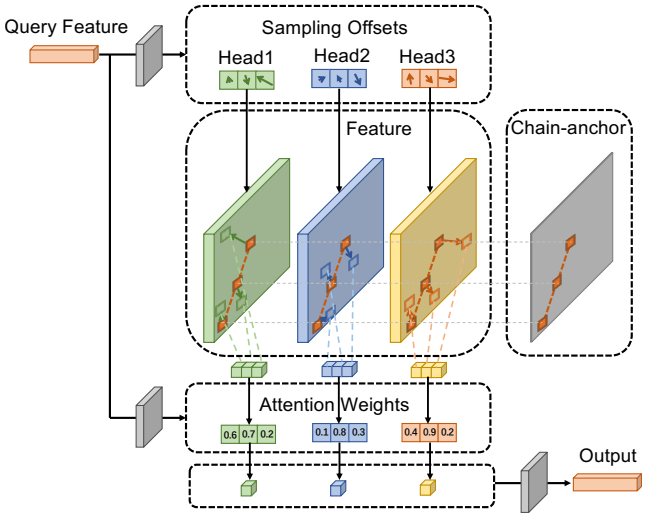


Fig. 6 The multi-referenced deformable cross-attention module utilizes positional information from Chain-anchor to guide the attention distribution.

3.3 Multi-Referenced Deformable Cross-Attention

As most adjacent features contain similar appearance information, traditional cross-attention modules bring a lot of additional computation, most of which is useless computation on backgrounds. The improved deformable attention module [32] samples partial information based on a learnable offset field. However, due to the lack of explicit supervision to guide the sampling along the object contour, the deformable attention module could still lead to wasted computation when sampling around the center of elongated lanes, and can not balance the sampling of endpoints far away from the center point. Therefore, LDTR uses points (anchors) distributed along the lanes as reference points and samples only part of the information around each point, as shown in Fig. 6. By assigning only a small number of keys along the lanes for each query, the convergence speed and computational efficiency can be improved significantly.

3.4 Line IoU

L1 distance loss (Equation (1)) can independently optimize the position of each node on the chain, but lacking the overall error calculation of the target nodes means slow convergence. IoU loss is a widely adopted overall loss function in object detection/segmentation, computed by bounding box overlap or pixel-level intersection. For thin and long objects such as lanes, the bounding box method suffers from large errors while the latter does not support gradient backpropagation with the current lane representation.

Hence, CLRNet [7] suggested Line IoU, an approximate IoU algorithm for lanes. However, it assumes lanes are all

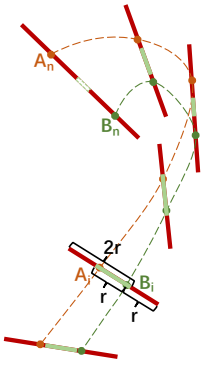


Fig. 7 Point-to-point (P2P) Line IoU.

vertical and only calculates the horizontal errors between two lines, thus the IoU results for a pair of lines vary a lot when they appear at different angles. In addition, the existing Line IoU algorithm is bound to a row-wise classification head and can only describe lanes that extend vertically in the view, which cannot be used to optimize Chain-anchor that can describe lane lines of any shape. To address such limitations, we propose two Line IoU algorithms inspired by the Chain-anchor: "*point-to-point*" and "*dense-sampling*" Line IoU.

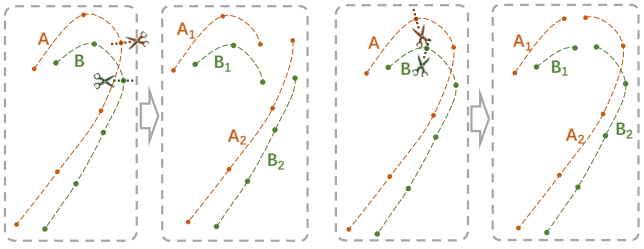
3.4.1 *Point-to-Point Line IoU*

To calculate the IoU of two lines A and B, the Point-to-Point (P2P) algorithm picks the same number (N) of keypoints in them uniformly and pairs the i-th points together (A_i, B_i). Given a fixed length r, draw two line segments along the direction of (A_i, B_i) with length $2r$, taking A_i and B_i as midpoints, respectively (thick solid red lines in Fig. 7). Let $L_{union}^{A_iB_i}$ be the distance of the far endpoints (the union) of the two segments, $L_{inter}^{A_iB_i}$ the distance of the near endpoints (the intersection, green lines in Fig. 7) and $||A_iB_i||_2$ the length of (A_i, B_i). If the two segments overlap, $L_{inter}^{A_iB_i}$ is $2r - ||A_iB_i||_2$ (solid green lines), while they are separated from each other (like (A_n, B_n) in Fig. 7), this expression is still used instead of 0, describing how far away they are in negative values (thick white line with the dashed green border between A_n and B_n). This is helpful for gradients and optimization. Then, LIoU_{P2P} is defined as Equation ((2)).

$$\text{LIoU}_{\text{P2P}}(A, B) = \frac{\sum_{i=1}^{N} L_{inter}^{A_i\ B_i}}{\sum_{i=1}^{N} L_{union}^{A_i\ B_i}} = \frac{\sum_{i=1}^{N}(2r - ||A_i\ B_i||_2)}{\sum_{i=1}^{N}(2r + ||A_i\ B_i||_2)} \tag{2}$$

Fig. 7 illustrates how LIoU_{P2P} is computed. LIoU_{P2P} ranges from -1 to 1. When the two lines overlap completely, it is 1, while when they are infinitely apart, it converges to -1.

P2P Line IoU describes the trend similarity between lanes and is not affected by the lane orientation. Thanks to its stability in the optimization process, LDTR leverages P2P Line IoU to compute the matching error as a supplement to the L1 distance and classification costs.



(a) Vertical splitting. (b) Horizontal splitting.

Fig. 8 Split U-turn lines.

3.4.2 *Dense-sampling Line IoU*

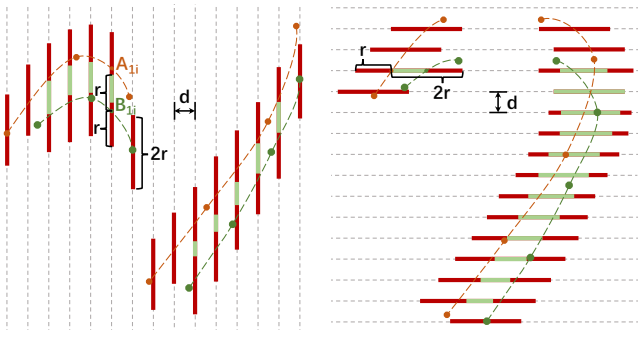
DS Line IoU consists of two steps:

Step 1: Line splitting. DS Line IoU samples keypoints in both x and y directions. To be compatible with curved lanes and backtracking lanes, it first splits the lines in the sampling direction into multiple segments of one-way sub-lines. As shown in Fig. 8, after the lines A and B are split, they respectively consist of sub-lines $\{A_1, A_2\}$ and $\{B_1, B_2\}$, each of which contains multiple sample points.

Step 2: Segment-wise calculation. For each unidirectional sub-line, the algorithm sets reference lines for every distance d in the sampling direction. If a reference line intersects both sub-lines, the two intersection points are paired together like P2P, in which case DS shares the definitions of L_{inter} and L_{union} with P2P in Equation (3).

$$\begin{aligned} L_{inter}^{A_{ij}B_{ij}} &= 2r - ||A_{ij}B_{ij}||_2 \\ L_{union2}^{A_{ij}B_{ij}} &= 2r + ||A_{ij}B_{ij}||_2 \end{aligned} \tag{3}$$

where i is the segment index, j the reference point index, and `X` in `unionX` means the number of intersection points.



(a) Vertical Sampling (b) Horizontal Sampling

Fig. 9 Dense-Sampling (DS) Line IoU.

Otherwise, if a reference line intersects only one sub-line, L_{inter} is set to 0, and L_{union1} is in Equation ((4)). Fig. 9

illustrates how the reference lines intersect the segments and cases of different intersection points.

$$L^A_{union1} = L^B_{union1} = 2r \tag{4}$$

Then, DS Line IoU is defined as the ratio of accumulated intersection distances to the union ones as P2P in Equation (5).

$$\text{LIoU}_{\text{DS}}(A, B) = \frac{\sum_{i=1}^{O}\sum_{j=1}^{N_i} L^{A_{ij}B_{ij}}_{inter}}{\sum_{i=1}^{O}\sum_{j=1}^{N_i} L^{A_{ij}B_{ij}}_{union2} + \sum_{i=1}^{U_A} L^{A}_{union1} + \sum_{i=1}^{U_B} L^{B}_{union1}} \tag{5}$$

where O is the number of `union2` segments, N_i the number of sampling points in the i-th `union2` segment, U_A and U_B the numbers of `union1` segments on line A and U_B, respectively.

Changing sampling interval d can make a flexible balance between precision and speed. LDTR sets d to 8 pixels by default. Though DS consumes more computation than P2P, it is straightforward to parallelize the algorithm in GPU. This will not be discussed due to space limits.

LDTR applies DS Line IoU to the overall loss of the target in Equation (6), which can record subtle differences between the predictions and GT. Since the entire chain of nodes is viewed as a whole, the prediction of the horizontal and vertical coordinates are optimized in x and y directions independently.

$$L_{iou} = 1 - \text{LIoU}_{\text{DS}}(P_{chain}, \hat{P}_{chain}) \tag{6}$$

where P_{chain} and $\hat{P}_{chain}$ are the ground truth and predicted chains, respectively.

3.5 Gaussian Heatmap Auxiliary Branch

Multi-task training has been widely used to enhance the generalization ability of single-task models. Therefore, LDTR introduces the Gaussian heatmap branch [8] as an auxiliary training branch. The structure and workflow of the branch are similar to that of Mask-DINO [33], as shown by the green dotted line in Fig. 3.

The Gaussian heatmap auxiliary training branch only performs forward and backward propagation during the training process, and the gradient is backpropagated to update the network weights. During the inference stage, this branch is discarded for efficiency. Networks with auxiliary branches have a larger parameter size, making it easier to learn how to fit the relationship between input images and ground truth from the initial state, effectively improving the convergence speed and stability of the optimization direction of the model. LDTR uses the same L_{mask} and L_{offset} as CANet [8] to supervise the training of the auxiliary branch.

3.6 Total Loss

In addition, LDTR adopts Focal loss to supervise the classification head like DETR, which is used to determine whether each query corresponds to a target in the input. The classification loss is shown in Equation (7).

$$L_{cls} = \frac{-1}{N_Q}\sum_{q\in Q}\begin{cases}(1-\hat{P}_q)^{\gamma}\log\hat{P}_q & P_q = 1\\ (1-P_q)^{\lambda}\hat{P}_q^{\gamma}\log(1-\hat{P}_q) & \text{otherwise}\end{cases} \tag{7}$$

where Q represents the set of all queries, P_q and $\hat{P}_q$ denote the prediction and binary match ground truth for Query q, respectively, and N_Q is the query number. Then, the total loss function in LDTR is Equation (8).

$$L_{total} = aL_{cls} + bL_{reg} + cL_{iou} + L_{mask} + L_{offset} \tag{8}$$

The hyperparameters a, b, and c are set to 1, 5, and 1, respectively.

4 Experiments

4.1 Datasets

To evaluate LDTR, extensive experiments are conducted on two widely used datasets for lane detection, CULane [3] and CurveLanes [19]. As a comprehensive dataset, CULane contains images from urban street views, rural roads, and highways in diverse conditions, like glare and occlusion. Many lanes have little or no-visual-clue, which requires a deep understanding of the overall scene by the models as demonstrated in Fig. 1. Though lanes in CurveLanes are more obvious than those in CULane, CurveLanes has more topologically complicated lanes, such as forks, convergences, sharp turns, and T-junctions as Fig. 2, which are not well addressed before.

4.2 Performance Indicators

4.2.1 Parameterized F1-score

F1-score is the default evaluation indicator for lane detection models. Predictions and ground truth are expanded into fixed-width masks and the IoU between masks greater than the threshold is regarded as TP. The default IoU threshold of 0.5 originated from general object detections, but is too high for lane detection. Because lanes are thin, long objects, slight jitter in predicted points may lead to huge IoU variation. IoU is so sensitive that lots of predictions may be dropped even though they are qualified for downstream tasks. The situation would get worse for cases in Fig. 10.

On the one hand, many lanes are lacking obvious visual features to provide sufficient information for models to accurately locate. Some predicted lanes would run almost parallelly along the ground truth with a tiny distance,

(a) Predictions are determined as FP for lanes lacking visual clues.

(b) Incorrect predictions with high IoU.

Fig. 10 Abnormal predictions with typical IoU threshold. Thin solid lines are ground truth, and thick dotted ones are predicted lanes. "Dist" is the Fréchet distance between the prediction and its ground truth.

like the predictions with IoU of 0.34, 0.35, 0.28, and 0.21 in Fig. 10(a). These predictions should be recalled but are dropped because of the small IoU. Hence, it is reasonable to decrease the threshold of IoU to recall predictions with inaccurate but acceptable locations. Concerning Fig. 10(a), the threshold could be adjusted from 0.5 to 0.2.

On the other hand, certain high IoU predictions should not be accepted because IoU only focuses on pixel overlapping but lacks measurement of lane trend as Fig. 10(b) illustrates. Specifically, two situations cause such misjudgments: incomplete prediction and incorrect trend prediction, both of which are harmful to real-world downstream tasks. Simply decreasing the IoU threshold may exacerbate the occurrence of such misjudgments, so Fréchet distance is introduced to filter out predictions that deviate significantly from the real trend. The original Fréchet distance expects the lines to be roughly the same length and calculates the maximum shortest distance bidirectionally. However, concerning lane detection, the predicted lanes may be longer than the ground truth. Though the additional predicted segment implies the lane trend, it is difficult to justify and should be weighted less.

Thus, we modify Fréchet distance to calculate unidirectionally from ground truth to predictions only, to make sure every point in ground truth counts but not vice versa. It follows then predicted lanes are considered matched only if they satisfy a hybrid constraint: IoU$\geqslant \alpha$ and Fréchet distance$\leqslant \beta$. That is, F1-score will depend on two parameters, IoU threshold (α) and Fréchet distance threshold (β), denoted as F1(α, β). The classic F1-score configuration is actually F1(0.5, $+\infty$), while in the experiments we set β to be 4% of the image width, i.e., F1(0.2, 60) for CULane and F1(0.2, 10) for CurveLanes. For simplicity, F1 means F1(0.5, $+\infty$) in the following unless otherwise specified. Not only lane detection but also other object detections could benefit from the idea of parameterized F1-score.

The video material in the supplementary materials compares the prediction results of the current SOTA model CLRNet and LDTR on CULane under two different evaluation indicators. The video visually demonstrates that LDTR has better instance recall in situations where visual cues are lacking. However, these additional recalls compared to CLRNet are often regarded as false positives (red lines).

4.2.2 *MIoU and MDis*

In Fig. 10, all predictions of IoU below 0.5 will be dropped by F1 because they are regarded as FP, though they are qualified for downstream tasks and should be accepted instead. Consequently, if a model predicts more such lanes, its precision will get worse and worse. This contradiction stems from the definition of precision, and extra insightful indicators are required to investigate lane detection models together.

Based on such observation, we suggest turning to "*MIoU*" (Mean IoU) and "*MDis*" (Mean Fréchet Distance) as in Equation (9) to compare position precision and trend similarity of predicted lanes under similar recall rates. Both metrics describe how close the predictions relate to ground truth overall, which makes more sense than precision with respect to lane detection. In Equation (9), N_{TP} is the number of TP predictions, and U_{TP} is the set of all TP instances.

$$\text{MIoU} = \frac{\sum_{i \in U_{TP}} IoU_i}{N_{TP}}, \quad \text{MDis} = \frac{\sum_{i \in U_{TP}} Dis_i}{N_{TP}} \tag{9}$$

Table 1 Performance of SOTA models and LDTR on CULane.

Model	F1	Precision	Recall	MIoU	MDis	F1(0.2, 60)
CANet	79.86	**88.03**	73.08	**82.61**	15.83	81.85
CLRNet	**80.47**	87.13	74.77	82.13	22.62	82.11
LDTR	78.16	81.53	**75.06**	82.23	**12.89**	**84.18**

4.2.3 *Necessity Verification of New Indicators*

To verify the effectiveness of the synthetic indicators, experimental results of current SOTA models and LDTR in terms of all metrics are presented in Table 1. Some valuable insights could be revealed from the results. Although CLRNet reaches the best F1 and precision, LDTR has a better recall rate. Additionally, with superior MIoU and MDis, LDTR predicts lane positions more accurately. CANet has the highest precision and MIoU due to its adaptive decoder's great description

ability for common lanes and a bit of degradation in recall rate. Therefore, the highest F1(0.2, 60) of LDTR indicates that some FP predictions otherwise are correctly recalled, while the smallest MDis guarantees these recalls are safe.

Table 2 LDTR performance by adopting the two IoU algorithms in different cost and loss combinations.

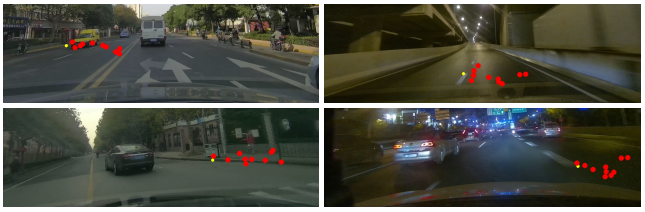
Cost	Loss	F1	Precision	Recall	MIoU	MDis	F1(0.2, 10)
N/A	N/A	86.24	88.20	84.36	79.86	2.88	88.60
P2P	P2P	87.15	90.92	83.69	81.25	2.77	88.51
DS	DS	87.74	90.28	**85.33**	81.19	2.86	89.00
P2P	DS	**87.96**	**91.19**	84.95	**81.31**	**2.85**	**89.01**

4.3 Evaluation of Line IoU Algorithms

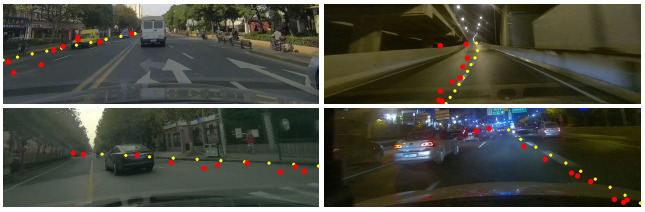
Line IoU can be used as a binary matching cost to improve the stability of matching, or as a loss function to optimize model training. Table 2 presents LDTR performance using the two proposed IoU algorithms in different combinations. Both algorithms adopted as either cost or loss function outperform the baseline, while P2P is more suitable for cost and DS for loss. This is because P2P is insensitive to shape jitter, so using P2P as the cost for bipartite matching can make the training more stable, while DS can capture the subtle differences and is suitable for loss function intrinsically to optimize prediction details.

4.4 Ablation Study

Table 3 presents the ablation experimental results of different components of LDTR on CurveLanes. Different from other lane description methods that directly perform local computation on the image feature map, the transformer encoder-decoder query-based structure adopted in this paper does not have a strong mapping relationship between the query representing each lane instance and the positions in the image. If replacing chain-anchor with other methods, the entire decoder needs to be removed, making it impossible to control variables effectively. Therefore, we have to keep chain-anchor as the baseline. The baseline (1st row) is the DETR-based model with Chain-anchor, which utilizes the



(a) Original deformable attention (ODA).



(b) Multi-referenced deformable attention (MRDA).

Fig. 11 Distribution of reference points (yellow) and sample points (red) in cross-attention modules. In MRDA, the attention tends to be distributed along the line, while in ODA, the attention is often concentrated near the central point.

Table 3 Ablation experiments of different components in LDTR on CurveLanes.

No.	Model	F1	Precision	Recall	MIoU	MDis	F1(0.2, 10)
1	Chain-anchor	85.57	87.41	83.80	78.86	3.01	88.44
2	+MRDA	$86.24_{+0.67}$	$88.20_{+0.79}$	$84.36_{+0.56}$	$79.86_{+1.00}$	$2.88_{-0.13}$	$88.60_{+0.16}$
3	+Line-IoU	$87.96_{+1.72}$	$91.19_{+2.99}$	$84.95_{+0.59}$	$81.31_{+1.45}$	$2.85_{-0.03}$	$89.01_{+0.41}$
4	+Auxiliary	$\mathbf{88.44}_{+0.48}$	$\mathbf{91.55}_{+0.36}$	$\mathbf{85.53}_{+0.58}$	$\mathbf{81.39}_{+0.08}$	$\mathbf{2.69}_{-0.16}$	$\mathbf{89.66}_{+0.65}$

Table 4 Performance comparison on CULane. The first two groups (CNN-based and Transformer-based) are measured in F1, while the last one (SOTA, marked with "*") is in F1(0.2, 60).

	Method	Total	Normal	Crowded	Dazzle	Shadow	No line	Arrow	Curve	Cross	Night
CNN-based (F1)	SCNN [3]	71.60	90.60	69.70	58.50	66.90	43.40	84.10	64.40	1990	66.10
	CurveLane-L [19]	74.80	90.70	72.30	67.70	70.10	49.40	85.80	68.40	1746	68.90
	LaneATT-L [9]	77.02	91.74	76.16	69.47	76.31	50.46	86.29	64.05	1264	70.81
	CondLaneNet-L [6]	79.48	93.47	77.44	70.93	80.91	54.13	90.16	75.21	1201	74.80
	GANet-L [17]	79.63	93.67	78.66	71.82	78.32	53.38	89.86	**77.37**	1352	73.85
	CANet-L [8]	79.86	93.60	78.74	70.07	79.35	52.88	90.18	76.69	1196	74.91
	CLRNet(DLA34) [7]	**80.47**	**93.73**	**79.59**	**75.30**	**82.51**	**54.58**	**90.62**	74.13	**1155**	**75.37**
Transformer-based (F1)	PriorLane [12]	76.27	92.36	73.86	68.26	78.13	49.60	88.59	**73.94**	2688	70.26
	LaneFormer [13]	77.06	91.77	75.41	70.17	75.75	48.73	87.65	66.33	**19**	71.04
	LDTR	**78.16**	**93.22**	**75.91**	**72.57**	**79.53**	**53.02**	**88.70**	70.41	1352	**73.66**
SOTA (F1(0.2, 60))	CANet-L [8] *	81.85	95.45	80.57	77.43	77.99	55.45	91.68	**71.22**	1196	77.10
	CLRNet(DLA34) [7] *	82.11	94.54	80.85	80.96	81.14	58.66	91.85	58.33	**1155**	78.80
	LDTR *	**84.18**	**96.12**	**83.27**	**81.49**	**87.39**	**62.93**	**92.06**	62.77	1352	**81.52**

classification loss (L_{cls} as Equation (8)) and regression loss (L_{reg} as Equation (1)).

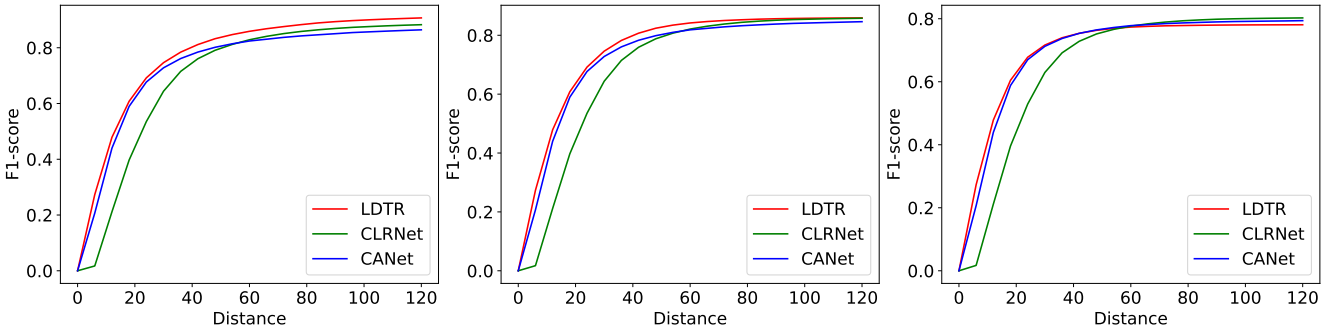
In the 2nd row, the original deformable attention is replaced with MRDA, whose attention positions are shown in Fig. 11. Because of the more precise location prior, MRDA can focus on more comprehensive detailed features, which boosts all indicators as the results.

Particularly, Line IoU adopted (3rd row) in the loss and matching error calculation optimizes each lane instance as a whole, thus significantly improving the accuracy and MIoU. In the last row, F1(0.2, 60) and MDis are improved remarkably, indicating the auxiliary branch can enhance the model to extract global semantic information.

4.5 Performance on Datasets

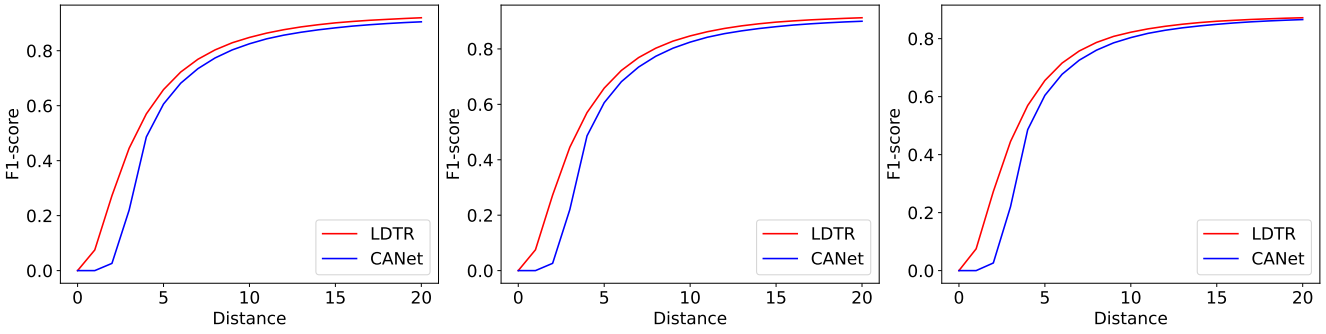
4.5.1 Results on CULane

As mentioned in Section 4.1, many lanes are having little or no-visual-clue in CULane, which is suitable to distinguish the recall capabilities of different models. Table 4 presents the comprehensive experimental results. Models are clustered according to their basic techniques into CNN-based and Transformer-based groups, and SOTA items are marked as bold in each group. LDTR is almost ahead of other Transformer-based models in all subsets, implying the effectiveness of LDTR's network structure design. However, it lags behind CLRNet because CNN-based models often predict fewer false positives, which results in higher precision and F1. As discussed in Section 4.2, F1 and precision are not friendly to downstream tasks concerning lane detection, so an additional group measured in F1(0.2, 60) is provided in the last of the table. LDTR outperforms both CANet and CLRNet by 2.33 and 2.07 overall, respectively, especially in scenes with fewer visual clues such as Crowded, Dazzle, and Shadow. It is worth noting that LDTR's F1 (0.2, 60) score has significantly improved by 6.02 percentage points compared to the overall F1 score. This is because, for the no-visual-clue scenarios that exist in CULane, LDTR's predictions are recalled more frequently under a more reasonable TP standard, while CLRNet and CANet have weaker recall performance in this scenario, resulting in poorer performance than LDTR in F1 (0.2, 60).



(a) IoU threshold=0 (b) IoU threshold=0.2 (c) IoU threshold=0.5

Fig. 12 F1-scores of different models vs. Fréchet distance thresholds on CULane.



(a) IoU threshold=0 (b) IoU threshold=0.2 (c) IoU threshold=0.5

Fig. 13 F1-scores of different models vs. Fréchet distance thresholds on CurveLanes. As CLRNet did not provide metrics and trained weights on CurveLanes, it is not included.

Table 5 Performance comparison on CurveLanes. The first group models are measured in F1, while the second group is in F1(0.2, 10) (marked with "*").

Models	F1	Precision	Recall	FPS
SCNN	65.02	76.13	56.74	7.5
ENet-SAD	50.31	63.60	41.60	75
PointLaneNet	78.47	86.33	72.91	71
CurveLane-L	82.29	91.11	75.03	-
CondLaneNet-L	86.10	88.98	83.41	48
CANet-L	87.87	**91.69**	84.36	36.6
LDTR	**88.44**	91.55	**85.53**	25.2
CANet-L*	88.48	92.33	84.95	36.6
LDTR *	**89.66**	**92.82**	**86.72**	25.2

4.5.2 *Results on CurveLanes*

Compared to CULane, the CurveLanes dataset covers a wider range of scenes and has more complicated lane shapes, which is more qualified to evaluate the ability of lane shape modeling. Table 5 shows the inference results in detail. LDTR performs best in terms of either F1 or F1(0.2, 10), especially for recall rate, which is emphasized by LDTR.

4.5.3 *Average Performance*

All previous experiments are executed with fixed hyper-parameters α and β. To evaluate the capability of LDTR with different hyperparameter configurations, we borrow COCO [34] object detection dataset performance indicators AP and AR and define AF1 (average F1-score) similarly. By conducting extensive experiments with lots of α, β combinations, Table 6 shows that LDTR surpasses other networks on average, implying the effectiveness of the architecture design independent of specific parameter settings.

Table 6 Average performance comparison on CULane and CurveLanes. As CLRNet did not provide metrics and trained weights on CurveLanes, it is not included.

Datasets	Models	AF1	AP	AR
CULane	CANet-L	62.97	**69.85**	57.33
	CLRNet	58.99	63.87	54.81
	LDTR	**63.65**	66.39	**61.12**
CurveLanes	CANet-L	58.55	61.10	56.22
	LDTR	**62.76**	**65.06**	**60.62**

To thoroughly understand how the Fréchet distance threshold is determined, Fig. 12 and Fig. 13 present how F1-score varies along the Fréchet distance thresholds with different IoU settings in the two datasets, respectively. LDTR generally performs better than the other two models, especially, when the IoU threshold is small. As the IoU threshold increases, no-visual-clue lanes are more likely to be dropped, thus the

(a) CLRNet, F1(0.5, $+\infty$) (b) LDTR, F1(0.5, $+\infty$) (c) LDTR, F1(0.2, 60) (d) GT

Fig. 14 Prediction ability comparison between CLRNet and LDTR for "Congested Scenarios" in CULane. The green and red lines represent the true positive (TP) and false positive (FP) of the model's prediction results under a specific evaluation indicator, respectively, while the ground truth (GT) is represented by blue lines.

advantage of LDTR gradually diminishes. These figures indicate that it is very flexible to choose different Fréchet distance thresholds for LDTR to exceed in F1-score.

5 Visual Comparison

For a more intuitive comparison, we have selected and included several frames in Fig. 14. These scenes include different lighting conditions, occlusions and most of them have no-visual-clue for lanes. By comparing the ground truth, LDTR can nearly predict all the instances while CLRNet misses certain ones sometimes. However, those predictions recalled by LDTR but missed by CLRNet are often mistakenly identified as FP in the default evaluation indicator (F1(0.5, $+\infty$)), leading to underestimated model performance. The use of the F1(0.2, 60) evaluation indicator can partially alleviate such misjudgement and provide a more objective and comprehensive evaluation of model performance.

6 Conclusion and Future work

There are still fundamental challenges in lane detection to be addressed: predict lanes with little or no-visual-clue and describe lanes of any shape. Aiming at these goals, this paper proposes LDTR, a transformer-based network, to leverage the global perception ability of transformer to improve the instance recall capability. The Chain-anchor representation enables LDTR to model lanes flexibly and precisely. To speed up convergence and reduce computation, a multi-referenced deformable cross-attention module is proposed to work together with the Chain-anchor. In addition, two Line IoU algorithms are designed to facilitate the cost and loss functions, which further enhances LDTR's representation capability along with the auxiliary branch. Considering the differences between lanes and typical objects, F1-score is extended to accept Fréchet distance as an additional parameter besides reducing the IoU threshold. Meanwhile, several synthetic metrics are devised to evaluate LDTR along with those classic ones. The experimental results show that LDTR achieves SOTA on two well-known datasets.

In the future, we plan to first improve the inference speed of LDTR (it is the slowest in Table 5 now). In addition, the temporal information implied in the video can provide valuable insights when there are no-visual-clue for lanes in the following frames. Existing methods, such as RVLD [21], have demonstrated this. We will explore the potential to share the semantic information stored in the updated queries in the decoder between frames in real-time videos.

Electronic Supplementary Material

A demo video is available in the online version of this article, which shows how LDTR and CLRNet (the current SOTA model) perform on CULane validation set.

The frame is divided into four parts. The upper and lower background areas of the frame represent the prediction results of LDTR and CLRNet respectively, where green and red lines indicate TP (true positive) and FP (false positive) respectively measured in the default evaluation indicator. The picture-in-picture area in the upper part of the frame also displays the prediction results of LDTR, but in F1(0.2, 60) evaluation indicator. The picture-in-picture area in the lower part of the frame displays the visualization results of GT (ground truth).

Declarations

Availability of data and materials

The datasets analyzed during the current study are available in the following repositories: https://github.com/SoulmateB/CurveLanes and https://xingangpan.github.io/projects/CULane.html.

Competing interests

The authors have no competing interests to declare that are relevant to the content of this article.

Funding

This paper is supported by National Natural Science Foundation of China (No. U23A6007).

Author contributions

Zhongyu Yang, Chen Shen and Wei Shao conceived of the presented idea, Zhongyu Yang proposed and implemented the prototype system. Zhongyu Yang, Chen Shen, Wei Shao and Tengfei Xing designed and performed the experiments and analysed the data. Runbo Hu, Pengfei Xu, Hua Chai and Ruini Xue supervised the project. Ruini Xue contributed to the interpretation of the results and took the lead in writing the manuscript. All authors discussed the results and contributed to the final manuscript.

Acknowledgements

We would like to express our sincere gratitude to Xingxu Yao and Yueming Zhang for their valuable assistance in analyzing the experimental results. We would also like to thank Ge Zhang for her valuable insights and assistance in refining the wording of the paper.

References

[1] Kim Z. Robust lane detection and tracking in challenging scenarios. *IEEE Transactions on intelligent transportation systems*, 2008, 9(1): 16–26.

[2] Borkar A, Hayes M, Smith MT. Robust lane detection and tracking with ransac and kalman filter. In *2009 16th IEEE International Conference on Image Processing (ICIP)*, 2009, 3261–3264, doi:10.1109/ICIP.2009.5413980.

[3] Pan X, Shi J, Luo P, Wang X, Tang X. Spatial as deep: Spatial cnn for traffic scene understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[4] Neven D, De Brabandere B, Georgoulis S, Proesmans M, Van Gool L. Towards end-to-end lane detection: an instance segmentation approach. In *2018 IEEE intelligent vehicles symposium (IV)*, 2018, 286–291, doi:10.1109/ivs.2018.8500547.

[5] Abualsaud H, Liu S, Lu DB, Situ K, Rangesh A, Trivedi MM. Laneaf: Robust multi-lane detection with affinity fields. *IEEE Robotics and Automation Letters*, 2021, 6(4): 7477–7484.

[6] Liu L, Chen X, Zhu S, Tan P. Condlanenet: a top-to-down lane detection framework based on conditional convolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, 3773–3782.

[7] Zheng T, Huang Y, Liu Y, Tang W, Yang Z, Cai D, He X. Clrnet: Cross layer refinement network for lane detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, 898–907.

[8] Yang Z, Shen C, Shao W, Xing T, Hu R, Xu P, Chai H, Xue R. CANet: Curved Guide Line Network with Adaptive Decoder for Lane Detection, 2023.

[9] Tabelini L, Berriel R, Paixao TM, Badue C, De Souza AF, Oliveira-Santos T. Keep your eyes on the lane: Real-time attention-guided lane detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, 294–302.

[10] Zhang JQ, Duan HB, Chen JL, Shamir A, Wang M. HoughLaneNet: Lane detection with deep hough transform and dynamic convolution. *Computers & Graphics*, 2023, 116: 82–92.

[11] Liu R, Yuan Z, Liu T, Xiong Z. End-to-end lane shape prediction with transformers. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2021, 3694–3702.

[12] Qiu Q, Gao H, Hua W, Huang G, He X. PriorLane: A Prior Knowledge Enhanced Lane Detection Approach Based on Transformer. *arXiv preprint arXiv:2209.06994*, 2022.

[13] Han J, Deng X, Cai X, Yang Z, Xu H, Xu C, Liang X. Laneformer: Object-aware Row-Column Transformers for Lane Detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 2022, 799–807.

[14] Zhou K, Zhou R. End to End Lane detection with One-to-Several Transformer. *arXiv preprint arXiv:2305.00675*, 2023.

[15] Chen Z, Liu Y, Gong M, Du B, Qian G, Smith-Miles K. Generating Dynamic Kernels via Transformers for Lane Detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, 6835–6844.

[16] Qu Z, Jin H, Zhou Y, Yang Z, Zhang W. Focus on local: Detecting lane marker from bottom up via key point. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, 14122–14130.

[17] Wang J, Ma Y, Huang S, Hui T, Wang F, Qian C, Zhang T. A Keypoint-based Global Association Network for Lane Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, 1392–1401.

[18] Qin Z, Wang H, Li X. Ultra fast structure-aware deep lane detection. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIV 16*, 2020, 276–291, doi:10.1007/978-3-030-58586-0_17.

[19] Xu H, Wang S, Cai X, Zhang W, Liang X, Li Z. Curvelane-nas: Unifying lane-sensitive architecture search and adaptive point blending. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV 16*, 2020, 689–704, doi:10.1007/978-3-030-58555-6_41.

[20] Feng Z, Guo S, Tan X, Xu K, Wang M, Ma L. Rethinking Efficient Lane Detection via Curve Modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, 17062–17070.

[21] Jin D, Kim D, Kim CS. Recursive Video Lane Detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, 8473–8482.

[22] Ren S, He K, Girshick R, Sun J. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 2015, 28.

[23] Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, 779–788.

[24] Redmon J, Farhadi A. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[25] Duan K, Bai S, Xie L, Qi H, Huang Q, Tian Q. Centernet: Keypoint triplets for object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, 6569–6578.

[26] Tian Z, Shen C, Chen H, He T. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, 9627–9636.

[27] Carion N, Massa F, Synnaeve G, Usunier N, Kirillov A, Zagoruyko S. End-to-end object detection with transformers. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, 2020, 213–229, doi:10.1007/978-3-030-58452-8_13.

[28] Zhu X, Su W, Lu L, Li B, Wang X, Dai J. Deformable detr:

Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020.

[29] Liu S, Li F, Zhang H, Yang X, Qi X, Su H, Zhu J, Zhang L. Dab-detr: Dynamic anchor boxes are better queries for detr. *arXiv preprint arXiv:2201.12329*, 2022.

[30] Kuhn HW. The Hungarian method for the assignment problem. *Naval research logistics quarterly*, 1955, 2(1-2): 83–97.

[31] Liao B, Chen S, Wang X, Cheng T, Zhang Q, Liu W, Huang C. Maptr: Structured modeling and learning for online vectorized hd map construction. *arXiv preprint arXiv:2208.14437*, 2022.

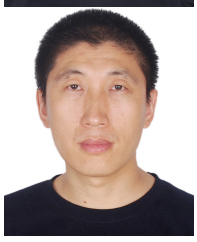
[32] Zhu X, Su W, Lu L, Li B, Wang X, Dai J. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020.

[33] Li F, Zhang H, Liu S, Zhang L, Ni LM, Shum HY, et al.. Mask dino: Towards a unified transformer-based framework for object detection and segmentation. *arXiv preprint arXiv:2206.02777*, 2022.

[34] Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, 2014, 740–755, doi: 10.1007/978-3-319-10602-1_48.

Author biography

Zhongyu Yang received his M.S degree from University of Electronic Science and Technology of China, Chengdu, in 2023. He is now an Algorithm engineer in Didi Chuxing.



Ruini Xue received Ph.D degree from Tsinghua University in 2009, and is currently an associated professor with School of Computer Science and Engineering in UESTC. His research interests include distributed storage systems, graph computing and AI.