# RMP: 3rd Place Solution for 2024 Waymo Open Dataset Challenge - Motion Prediction

Jiawei Sun[1], Jiahui Li[1], Tingchen Liu[1], Chengran Yuan[1], Shuo Sun[1],
Yuhang Han[1], Anthony Wong[2], Keng Peng Tee[2], Marcelo H. Ang Jr.[1]
[1]National University of Singapore, [2]Moovita Pte Ltd

***Abstract*— In this report, we introduce our method, Robust Motion Predictor (RMP), for the Waymo Open Dataset Challenge 2024, Motion Prediction. Accurately predicting the future trajectories of surrounding traffic participants is crucial for autonomous vehicles. Current prediction methods depend heavily on complete historical trajectory data, and their performance deteriorates rapidly when certain timestamps are missing. However, various challenging situations, such as occlusion, sensor failures, and adverse weather conditions, can result in incomplete historical trajectories. Therefore, we propose a simple recovery module designed to restore incomplete historical trajectories, which is also plug-and-play. The overall network structure is modified based on MTR [1] framework. Our method, the RMP_ensemble, ranked third in SoftmAP with a score of 0.4726 and first in Overlap Rate with a score of 0.1257. Our end-to-end version, RMP_e2e, ranked second in both ADE (0.5529 meters) and FDE (1.0932 meters) on the Waymo test leaderboard.**

## I. INTRODUCTION

Prediction accuracy is crucial for autonomous vehicle systems. Current learning-based motion predictors [1]–[9] are typically trained and evaluated using selected target agents with nearly complete historical trajectories. For instance, in the Waymo prediction leaderboard [10], [11], all methods must predict the trajectories of 2 to 8 agents per scenario, and the proportion of valid past timestamps must be greater than 97% of all past timesteps (see TABLE I). However, this ideal condition may not always be feasible in practice due to various challenging situations, such as occlusion, sensor failures, and adverse weather conditions, which can result in incomplete historical trajectories. In some extreme cases, there may be only a single frame of historical trajectory available. For example, in situations like a 'ghost pedestrian' suddenly darting out from behind an obstacle, the vehicle's sensors might only capture one frame of the pedestrian's movement, making accurate prediction highly challenging. To tackle this situation, we propose our RMP network structure, which can realize accurate predictions even when receiving only one timestep historical data. The model is built based on MTR [1], which is a powerful query-based motion predictor.

[1]Jiawei Sun, Tingchen Liu, Chengran Yuan, Shuo Sun, Yuhang Han, Jiahui Li and Marcelo H. Ang Jr. are with the Department of Mechanical Engineering, National University of Singapore, Singapore 119077 (e-mail: {sunjiawei, e1010862, chengran.yuan, shuo.sun, yuhang han }@u.nus.edu; ddawangjiahui@gmail.com; mpeangh@nus.edu.sg).

[2]Keng Peng Tee, Anthony Wong are with Moovita Pte Ltd, Singapore, 599489 (e-mail: anthonywong, kptee@moovita.com).

TABLE I: Data for valid past timesteps, all past timesteps and valid ratio for selected agents to predict in Waymo motion dataset.

| | Training | Validation | Testing |
|---|---|---|---|
| **Valid Past Timesteps** | 23367425 | 2,060,239 | 2,100,823 |
| **All Past Timesteps** | 23956394 | 2,113,892 | 2,156,550 |
| **Valid Ratio** | 0.9754 | 0.9746 | 0.9742 |

## II. METHOD

### A. Input Representation

Unlike the MTR framework, which does not consider traffic light information, our approach only uses current traffic light information as waypoints on map polylines, discarding historical traffic lights data. Additionally, we incorporate historical relative movement between target agents and map polylines as an additional input context. By employing an agent-centric strategy, the input for each target agent can be represented as follows:

$\mathcal{A} \in \mathbb{R}^{N_a \times T_p \times F_1}$, where $N_a$ represents the number of agents in the scenarios, $T_p$ is the number of past timesteps, and $F_1$ includes features such as position, heading, velocity, acceleration, agent type, agent size, valid sign, and one-hot embeddings for past timesteps.

$\mathcal{M} \in \mathbb{R}^{N_l \times N_p \times F_2}$, where $N_l$ denotes the number of map polylines, $N_p$ represents the number of waypoints per polyline, and $F_2$ includes position, direction, and waypoint type.

$\mathcal{R} \in \mathbb{R}^{N_l \times T_p \times F_3}$, where $F_3$ indicates the relative position and orientation between the target agent and the center of each road polyline over the past timesteps.

This approach ensures a comprehensive representation of the current traffic conditions and the interactions between agents and their environment.

### B. Network Encoder

**Feature aggregation.** Temporal information, including agent historical states $\mathcal{A}$ and relative movement $\mathcal{R}$, is processed using a Multi-Scale LSTM (MSL) model. Initially, the data is concurrently passed through a 1D CNN module with kernel sizes of 1, 3, and 5. It then progresses through a two-layer LSTM network, where the output at the final timestep is captured, concatenated across feature dimensions, and passed through an additional MLP layer to generate the final feature token (see Fig. 1). For the road polylines data $\mathcal{M}$, a simple PointNet-like network is employed to extract the spatial features of each polyline.

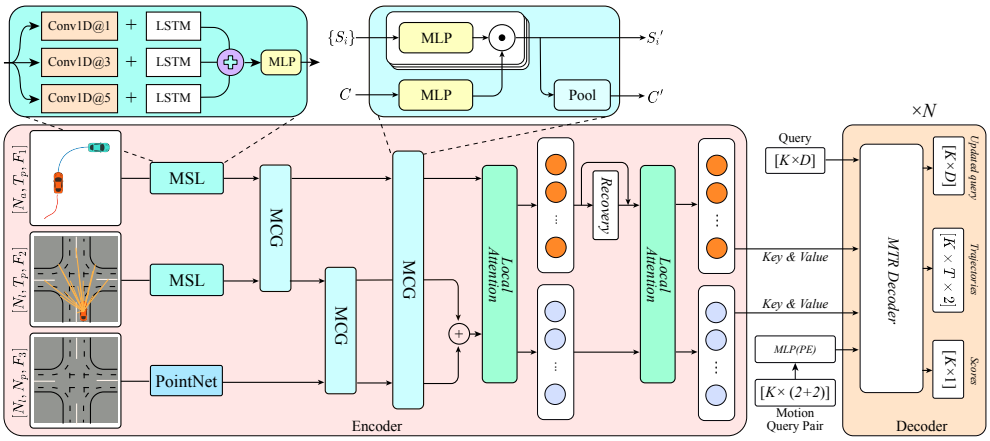


Fig. 1: Network structure of our proposed method.

$$\begin{aligned} \mathcal{A}_1 &= \text{MSL}(\mathcal{A}), & \mathcal{A}_1 &\in \mathbb{R}^{N_a \times D_1}, \\ \mathcal{R}_1 &= \text{MSL}(\mathcal{R}), & \mathcal{R}_1 &\in \mathbb{R}^{N_l \times D_1}, \\ \mathcal{M}_1 &= \phi(\text{MLP}(\mathcal{M})), & \mathcal{M}_1 &\in \mathbb{R}^{N_l \times D_1}, \end{aligned} \tag{1}$$

where $\phi(\cdot)$ denotes the max-pooling operation.

**Feature Fusion and Recovery Module.** In this process, we integrate encodings from various input modalities using Multi-Context Gating (MCG) as proposed in [5]. We utilize a cascading method where, at each stage, two distinct modalities are chosen from a set of three to be input into the MCG module. The output from one MCG module is then fed into the subsequent MCG module in the sequence, as illustrated in Fig. 1.

$$\begin{aligned} (\mathcal{A}_2, \mathcal{R}_2) &= \text{MCG}(\mathcal{A}_1, \mathcal{R}_1), \\ (\mathcal{M}_2, \mathcal{R}_3) &= \text{MCG}(\mathcal{M}_1, \mathcal{R}_2), \\ (\mathcal{A}_3, \mathcal{M}_3) &= \text{MCG}(\mathcal{A}_2, \mathcal{M}_2). \end{aligned} \tag{2}$$

Then, we use $\mathcal{A}_3$ as the final agent tokens $\mathcal{A}_{agent} \in \mathbb{R}^{N_a \times D_1}$ and $\mathcal{M}_3 + \mathcal{R}_3$ as the final map tokens $\mathcal{M}_{map} \in \mathbb{R}^{N_p \times D_1}$.

We assume that local interactions are crucial and sufficient to reconstruct incomplete historical trajectories. Therefore, we use a single layer of local attention [1] to ensure that each agent token attends to its K nearest neighbor tokens (including both agent and map tokens). Following this, a simple MLP layer is employed to recover the complete past trajectories for all agents. Next, a PointNet-like layer aggregates these recovered trajectories into agent tokens. A residual connection is added between the input and output of the recovery module to ensure stable training.

$$\begin{aligned} \mathcal{A}_{agent} &= \mathcal{A}_{agent} + \text{Recovery}(\mathcal{A}_{agent}), \\ \text{Recovery}(\mathcal{A}_{agent}) &= \text{MLP}(\mathcal{A}_{Past}), \\ \mathcal{A}_{Past} &= \text{MLP}(\mathcal{A}_{agent}), \end{aligned} \tag{3}$$

Where $\mathcal{A}_{Past} \in \mathbb{R}^{N_a \times (T_p \times 4)}$ and 4 denotes the position and velocity for the recovered past trajectories. This recovery module aims to reconstruct the incomplete historical trajectories for each agent and integrate this enriched historical information into the agent tokens. After the recovery module, both agent tokens and map tokens will go through another four layers of local attention for further feature fusion. The $i_{th}$ transformer encoder layer can be formulated as:

$$\begin{aligned} Q^i = \text{MHA}(&Q^{i-1} + \text{PE}(Q^{i-1}), \\ &\mathcal{K}(Q^{i-1}) + \text{PE}(\mathcal{K}(Q^{i-1})), \mathcal{K}(Q^{i-1})), \end{aligned} \tag{4}$$

where $\text{MHA}(\cdot,\cdot,\cdot)$ represents the multi-head attention function, $Q^0 = [\mathcal{M}_{map}, \mathcal{A}_{agent}] \in \mathbb{R}^{(N_a+N_m)\times D_1}$, and $\mathcal{K}(\cdot)$ denotes the $K$-nearest neighbours (KNN) algorithm, which is used to identify the $K$ nearest tokens relative to each query. The term $\text{PE}(\cdot)$ refers to the sinusoidal positional encoding assigned to input tokens, incorporating the most recent position of each agent and the central point of each map polyline. The final output of the local attention layer will be snet into the MTR decoder, $[\mathcal{M}_{map}, \mathcal{A}_{agent}] = Q^{Final}$.

### C. Network Decoder

The decoder part is identical to the MTR [1] decoder, except for how the output trajectories are used to calculate the loss. Between each decoder layer's output and loss calculation, we apply evolving and distinct anchors tricks referred in [2]. We use 6 decoder layers, perform evolving anchors at the second and fourth layers, and select distinct anchors at each layer.

### D. Loss Function

We put forward a combined loss function which is consist of two components: Original MTR [1] loss and recovery loss. The total loss function can be defined as follows:

$$\mathcal{L}_{Total} = \mathcal{L}_{MTR} + \mathcal{L}_{Recovery} \tag{5}$$

For the MTR loss, We follow the loss function of MTR [1], using a decoder loss $\mathcal{L}_{Decoder}$ and a dense future prediction loss $\mathcal{L}_{Df}$.

The recovery loss $\mathcal{L}_{Recovery}$ aims to optimize the recovery module to resume the incomplete past trajectories and it is simply the $\mathcal{L}_1$ loss of recovered $\mathcal{A}_{Past}$ and ground truth past trajectories $\mathcal{A}_{PastGT}$.

TABLE II: Performance on the test leaderboard of the motion prediction track of the Waymo Open Dataset Challenge. Our approach is termed as RMP, i.e., Robust Motion Predictor. Soft mAP is the official ranking metric, while miss rate is the secondary ranking metric. The first place is denoted by **bold**, the second place by underline, and the third place by $*$asterisk.

| Method | Soft mAP ↑ | mAP ↑ | minADE ↓ | minFDE ↓ | Miss rate ↓ | Overlap Rate ↓ |
|---|---|---|---|---|---|---|
| MTR v3 [4] | **0.4967** | **0.4859** | 0.5554* | 1.1062* | **0.1098** | 0.1279 |
| ModeSeq | 0.4737 | 0.4665 | 0.5680 | 1.1766 | 0.1204 | 0.1275 |
| Betop | 0.4698 | 0.4587* | 0.5716 | 1.1668 | 0.1183 | 0.1272 |
| BehaveOcc | 0.4678 | 0.4566 | 0.5723 | 1.1668 | 0.1176 | 0.1278 |
| QMTR | 0.4649 | 0.4445 | 0.5702 | 1.1627 | 0.1177 | 0.1269 |
| EDA [2] | 0.4596 | 0.4487 | 0.5718 | 1.1702 | 0.1169 | 0.1266* |
| ControlMTR [3] | 0.4572 | 0.4414 | 0.5897 | 1.1916 | 0.1282 | 0.1259 |
| LLM-Augmented-MTR v4 | 0.4423 | 0.4270 | 0.5987 | 1.2084 | 0.1316 | 0.1274 |
| MTR [1] | 0.4403 | 0.4249 | 0.5964 | 1.2039 | 0.1312 | 0.1274 |
| Traj_pred | 0.4320 | 0.4218 | 0.6030 | 0.1409 | 1.9136 | 0.1283 |
| FMAT | 0.3438 | 0.3000 | **0.5362** | **1.0788** | 0.1364 | 0.1342 |
| LSTM | 0.1931 | 0.1863 | 1.0065 | 2.3553 | 0.3750 | 0.1898 |
| RMP | 0.4572 | 0.4423 | 0.5695 | 1.1658 | 0.1160* | **0.1257** |
| RMP e2e | 0.3828 | 0.3440 | 0.5529 | 1.0932 | 0.1354 | 0.1295 |
| RMP Ensemble | 0.4726* | 0.4553 | 0.5596 | 1.1272 | 0.1113 | **0.1257** |

TABLE III: Detailed performance comparison of RMP variants across four categories: Vehicle, Pedestrian, Cyclist, and Average. The variants include RMP, RMP e2e, and RMP Ensemble.

| Our Model | Category | Soft mAP ↑ | mAP ↑ | minADE ↓ | minFDE ↓ | Miss rate ↓ | Overlap Rate ↓ |
|---|---|---|---|---|---|---|---|
| RMP | Vehicle | 0.5013 | 0.4776 | 0.6627 | 1.3425 | 0.1103 | 0.0397 |
| | Pedestrian | 0.4758 | 0.4670 | 0.3524 | 0.7333 | 0.0725 | 0.2651 |
| | Cyclist | 0.3946 | 0.3822 | 0.6936 | 1.4217 | 0.1652 | 0.0724 |
| | **Avg** | 0.4572 | 0.4423 | 0.5695 | 1.1658 | 0.1160 | 0.1257 |
| RMP (Ensemble) | Vehicle | 0.4922 | 0.4686 | 0.6731 | 1.3433 | 0.1143 | 0.0399 |
| | Pedestrian | 0.4923 | 0.4812 | 0.3358 | 0.6935 | 0.0627 | 0.2652 |
| | Cyclist | 0.4334 | 0.4160 | 0.6698 | 1.3448 | 0.1567 | 0.0721 |
| | **Avg** | 0.4726 | 0.4553 | 0.5596 | 1.1272 | 0.1113 | 0.1257 |
| RMP (e2e) | Vehicle | 0.4279 | 0.3824 | 0.6763 | 1.3105 | 0.1353 | 0.0416 |
| | Pedestrian | 0.3846 | 0.3420 | 0.3239 | 0.6584 | 0.0767 | 0.2699 |
| | Cyclist | 0.3359 | 0.3074 | 0.6586 | 1.3108 | 0.1943 | 0.0771 |
| | **Avg** | 0.3828 | 0.3440 | 0.5529 | 1.0932 | 0.1354 | 0.1295 |

TABLE IV: Results of randomly masking various percentages of historical trajectories for all agents per scenario on the Waymo validation dataset.

| Method | Missing timestamps | Soft mAP ↑ | mAP ↑ | minADE ↓ | minFDE ↓ | Miss rate ↓ | Overlap Rate ↓ |
|---|---|---|---|---|---|---|---|
| MTR | 40% | 0.4264 | 0.4093 | 0.624 | 1.2446 | 0.1403 | 0.1289 |
| | 50% | 0.4192 | 0.4023 | 0.6354 | 1.2594 | 0.1431 | 0.1297 |
| | 60% | 0.4117 | 0.3954 | 0.6497 | 1.2771 | 0.1458 | 0.1306 |
| | 70% | 0.4019 | 0.3858 | 0.673 | 1.3073 | 0.1517 | 0.1309 |
| | 80% | 0.3784 | 0.3633 | 0.7087 | 1.3534 | 0.1603 | 0.1327 |
| | 90% | 0.3425 | 0.329 | 0.7749 | 1.451 | 0.1802 | 0.1361 |
| | only current timestamps | 0.2801 | 0.2692 | 0.9171 | 1.6648 | 0.2207 | 0.146 |
| Ours | 40% | 0.4498 | 0.4317 | 0.5774 | 1.1874 | 0.1221 | 0.1273 |
| | 50% | 0.4492 | 0.4312 | 0.5787 | 1.1894 | 0.1226 | 0.1274 |
| | 60% | 0.449 | 0.4311 | 0.5821 | 1.195 | 0.1226 | 0.127 |
| | 70% | 0.4454 | 0.4275 | 0.5845 | 1.1979 | 0.1235 | 0.1267 |
| | 80% | 0.4392 | 0.4214 | 0.5919 | 1.2081 | 0.1264 | 0.1275 |
| | 90% | 0.4287 | 0.4118 | 0.6063 | 1.2317 | 0.1302 | 0.1275 |
| | only current timestamps | 0.4083 | 0.3925 | 0.6309 | 1.273 | 0.1378 | 0.1308 |

## III. EXPERIMENTS

### A. Experimental Setup

*1) Dataset and Metrics:* Our model is trained using the Waymo Open Motion Dataset (WOMD), which is divided into two sets: 486,995 scenes for training and 44,097 scenes for validation. The performance of our model is evaluated using several metrics, including the minimum average displacement error (minADE), minimum final displacement error (minFDE), miss rate, overlap rate, and soft mean average precision (Soft mAP). Among these, Soft mAP is a crucial metric for assessing the model's performance.

*2) Network Details:* The output channel size for the 1D CNN is 64. The hidden dimension for the encoder is 256, and we utilize a 2-layer LSTM, 2 layers of MCG for each modality fusion part, one layer of local attention before the recovery module, and 4 layers of local attention after. The decoder part is identical to that in MTR [1]. Our RMP model outputs 64 different trajectories, and we use Non-maximum Suppression(NMS) to select the final 6 trajectories. In contrast, our RMP_e2e model directly generates 6 trajectories.

*3) Training Details:* We use the AdamW optimizer to train our model in an end-to-end manner, with an initial

learning rate set to 0.0001. Beginning at epoch 20, the learning rate is halved every two epochs. We train the model for 30 epochs and then fine-tune it for an additional 5 epochs, maintaining a learning rate of 6.25e-6. The best model is trained on a single Nvidia 4090 GPU with a batch size of 6. Other models are trained on two Nvidia 3090 GPUs with a batch size of 12. No data augmentation is used.

*4) Model Ensemble Details:* Given $N$ well-trained models, each model generates 64 different trajectories, resulting in a total of $64N$ multimodal future trajectories for each target agent. Each trajectory is accompanied by a confidence score predicted by its respective model.

Initially, we apply a softmax operation on the scores of the $64N$ trajectories. Subsequently, we employ non-maximum suppression (NMS) to select the top 6 future trajectories based on their endpoints. The distance threshold $\sigma$ for NMS is scaled according to the length $L$ of the trajectory with the highest confidence among the $64N$ predictions, as follows:

$$\sigma = \min\left(3.5, \max\left(2.5, \frac{L-10}{50-10} \times 1.5 + 2.5\right)\right). \quad (6)$$

Our best model, `RMP_Ensemble`, is generated by an ensemble of four different models:

1) RMP model.
2) RMP model without recovery module.
3) RMP model using Binary Cross Entropy (BCE) for classification loss.
4) RMP model without using evolving and distinct anchors.

### B. Performance

*1) Leaderboard:* As shown in tableII, the RMP Ensemble is our best submission, ranking 3*rd* in the Waymo Open Dataset Challenge 2024, Motion Prediction Track, based on soft mAP, and 1*st* in terms of overlap rate.

*2) Detailed performance:* We conducted experiments to test the performance of three submissions of our RMP method: RMP, RMP e2e, and RMP Ensemble. Each submission was evaluated across four categories: Vehicle, Pedestrian, Cyclist, and Average. The results are summarized in TableIII. It can be infered that RMP excels in predicting Vehicle behavior while RMP Ensemble outperforms the other two submissions in the remaining three categories: Pedestrian, Cyclist, and Average. This analysis highlights the strengths of each submission and provides valuable insights for further optimization.

### C. Experiments result

*1) Missing timestamps research:* We conducted an experiment by randomly masking historical timestamps in the validation set, varying the mask ratio from 40% to 100%, to assess the robustness of both the MTR and our RMP method. The summarized results in TableIV clearly indicate that our method, RMP, consistently surpasses MTR across all performance metrics when historical frames are missing. Notably, when all historical timestamps are masked (i.e., only current timestamps are available), our RMP method excels with a Miss Rate (MR) of 0.1378, compared to MTR's MR of 0.2207. Furthermore, as the mask ratio increases, MTR experiences a more rapid decline in performance compared to our method. These findings underscore the robustness and superiority of our RMP approach, especially when dealing with missing historical data.

## IV. CONCLUSIONS

In this technical report, we introduce a Robust Motion Predictor (RMP) designed for multimodal motion prediction, particularly effective when historical trajectories are incomplete. Our proposed method can achieve high prediction accuracy even with only a single timestep of historical trajectory information.

## REFERENCES


[1] S. Shi, L. Jiang, D. Dai, and B. Schiele, "Motion transformer with global intention localization and local movement refinement," 2023.

[2] L. Lin, X. Lin, T. Lin, L. Huang, R. Xiong, and Y. Wang, "Eda: Evolving and distinct anchors for multimodal motion prediction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, pp. 3432–3440, 2024.

[3] J. Sun, C. Yuan, S. Sun, S. Wang, Y. Han, S. Ma, Z. Huang, A. Wong, K. P. Tee, and M. H. A. Jr, "Controlmtr: Control-guided motion transformer with scene-compliant intention points for feasible motion prediction," 2024.

[4] S. Shi, L. Jiang, D. Dai, and B. Schiele, "Mtr++: Multi-agent motion prediction with symmetric scene modeling and guided intention querying," 2024.

[5] B. Varadarajan, A. Hefny, A. Srivastava, K. S. Refaat, N. Nayakanti, A. Cornman, K. Chen, B. Douillard, C. P. Lam, D. Anguelov, and B. Sapp, "Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction," 2021.

[6] N. Nayakanti, R. Al-Rfou, A. Zhou, K. Goel, K. S. Refaat, and B. Sapp, "Wayformer: Motion forecasting via simple efficient attention networks," 2022.

[7] C. M. Jiang, A. Cornman, C. Park, B. Sapp, Y. Zhou, and D. Anguelov, "Motiondiffuser: Controllable multi-agent motion prediction using diffusion," 2023.

[8] Z. Zhou, L. Ye, J. Wang, K. Wu, and K. Lu, "Hivt: Hierarchical vector transformer for multi-agent motion prediction," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8813–8823, 2022.

[9] Z. Zhou, J. Wang, Y.-H. Li, and Y.-K. Huang, "Query-centric trajectory prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[10] S. Ettinger, S. Cheng, B. Caine, C. Liu, H. Zhao, S. Pradhan, Y. Chai, B. Sapp, C. Qi, Y. Zhou, Z. Yang, A. Chouard, P. Sun, J. Ngiam, V. Vasudevan, A. McCauley, J. Shlens, and D. Anguelov, "Large scale interactive motion forecasting for autonomous driving : The waymo open motion dataset," 2021.

[11] K. Chen, R. Ge, H. Qiu, R. AI-Rfou, C. R. Qi, X. Zhou, Z. Yang, S. Ettinger, P. Sun, Z. Leng, M. Baniodeh, I. Bogun, W. Wang, M. Tan, and D. Anguelov, "Womd-lidar: Raw sensor dataset benchmark for motion forecasting," 2024.