Supplementary Material for Stay on Track: A Frenet Wrapper to Overcome Off-road Trajectories in Vehicle Motion Prediction

Marcel Hallgarten
Robert Bosch GmbH
Germany
marcel.hallgarten@de.bosch.com

Ismail Kisa
University of Tuebingen
Germany
ismail.kisa@student.uni-tuebingen.de

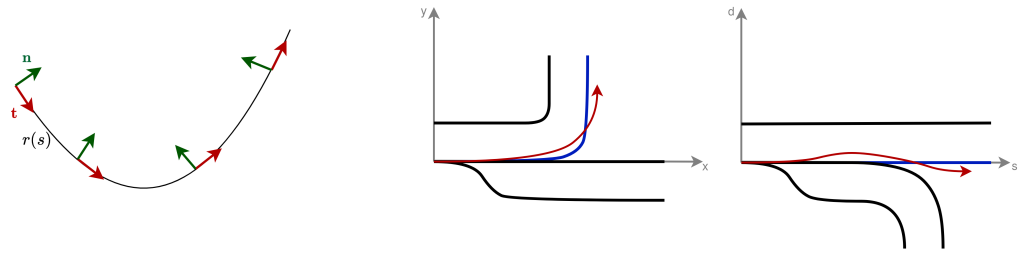
Martin Stoll
Robert Bosch GmbH
Germany
martin.stoll@de.bosch.com

Andreas Zell
University of Tuebingen
Germany
andreas.zell@uni-tuebingen.de

Abstract: In this **supplementary document**, we provide additional details about the Frenet frame wrapper approach. First, we give a brief overview of the relation between Cartesian representations and Frenet frame representations in traffic scenes. Next, we give a detailed description of our algorithm, used to transform the model inputs into the Frenet representation. We also discuss the computational overhead created by this transformation as well as relevant corner cases. Additionally, we report an upper bound for the estimation of the prior over relevant lanes. Finally, we provide additional qualitative examples to emphasise that the models trained with our Frenet frame wrapper generalise beyond simple lane following.

1 Frenet-Serret Frame

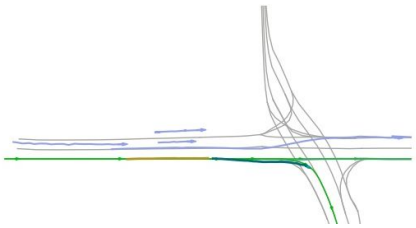
1.1 Overview



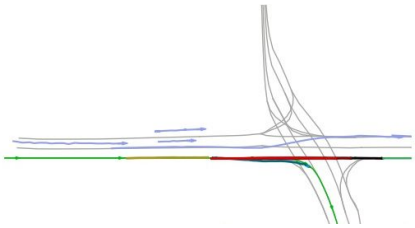
(a) Frenet System (b) Cartesian Representation vs. Frenet Representation

Figure 1: The definition of the Frenet coordinate system is depicted in Fig. 1a. In Fig. 1b a traffic scene is shown in Cartesian representation (i.e., x and y coordinates) and Frenet representation (i.e., s and d coordinates). It consists of one lane going straight, one branching to the right, and two making a left turn. The blue lane is chosen as the reference lane for the Frenet system. An exemplary trajectory is shown in red. It can be seen how the trajectory's representation in the Cartesian frame decomposes it into progress along the reference path s and lateral offset across it d.

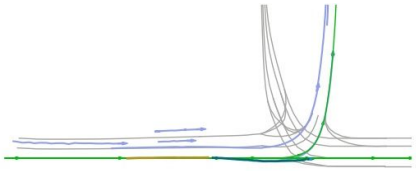
The Frenet-Serret coordinate frame [1, 2], commonly called Frenet frame, defines coordinates relative to a given curve $\mathbf{r}(s)$. In contrast, the Cartesian frame defines the position of a point with respect to an absolute origin and a set of pairwise orthogonal vectors. **Here we only consider the 2-dimensional case.** We denote the Cartesian coordinates of point i with (x_i, y_i) and the corre-



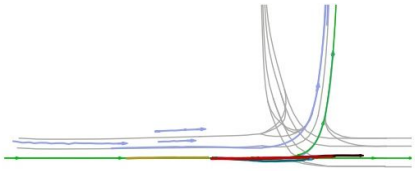
(a) Frenet representation of $C^{(1)}$



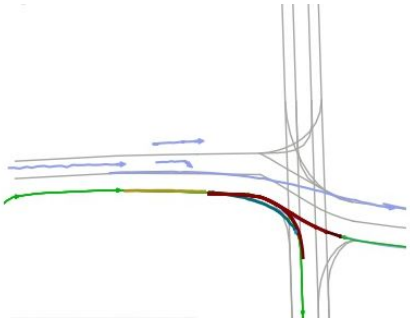
(b) Predictions conditioned $C^{(1)}$



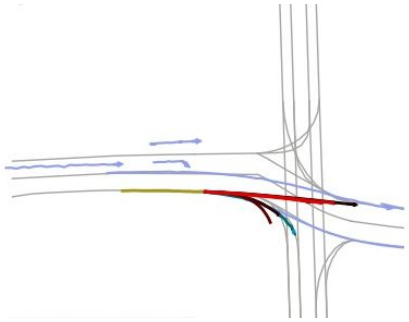
(c) Frenet representation of $C^{(2)}$



(d) Predictions conditioned on $C^{(2)}$



(e) Aggregated predictions of LaneGCN-SD



(f) Aggregated predictions of LaneGCN [3]

Figure 2: Transformation

sponding Frenet coordinates with (s_i, d_i). s measures the arc length along the curve, d the lateral displacement to the curve (in direction of the normal vector $\mathbf{n}(s)$). Note that $\mathbf{n}(s)$ depends on the position on the curve $\mathbf{r}(s)$.

The Cartesian position of a point can be reconstructed

$$(x_i, y_i) = \mathbf{r}(s_i) + d_i \cdot \mathbf{n}(s_i)\,. \tag{1}$$

An illustration is given in Fig. 1a.

The Frenet frame is a natural choice for trajectory forecasting and planning. Commonly, the centreline of a lane is chosen as the reference path. Consequently, this representation is invariant to the exact road topology. For instance, a trajectory that perfectly follows the reference path, i.e. the centreline, will have $d = 0$. Moreover, as s describes the longitudinal progress, $\frac{ds}{dt}$ gives the velocity along the reference path, i.e. the rear axle velocity of the vehicle. Fig. 1b shows how the representation of a traffic scene differs between the Cartesian and the Frenet representation.

1.2 Application in the Frenet Wrapper

See Fig. 2 for a practical example of how the Frenet representation is leveraged in the prediction task. Commonly, a Cartesian representation is used to describe the inputs of a prediction model. Fig. 2f visualises the scene representation in the Cartesian frame. It can be seen that the LaneGCN [3] model is not able to predict trajectories that accurately follow the centrelines in the scene. In particular, the set of predicted trajectories includes off-road trajectories and trajectories that lead into oncoming traffic. In contrast, Fig. 2a and 2c show the Frenet representation of the scene based on the two centrelines $C^{(1)}$ and $C^{(2)}$ that the target vehicle could follow. The respective predictions, conditioned on the respective lane, can be seen in Fig. 2b and 2d. This intuitively shows that fol-

lowing a respective lane in this representation is a much easier task than in the Cartesian frame. After transforming the predicted trajectories back into the Cartesian frame, the output represented in Fig. 2e is obtained. Compared to the output of the Cartesian model (Fig. 2f), it clearly leverages the map context more effectively.

2 Transformation Algorithm

2.1 General Transformation Algorithm

Notation We consider the discrete case where a centreline is given by a sequence of N points $\{x_i^c, y_i^c\}_{i=1}^N$, where x and y denote the Cartesian coordinates of the points. The points are ordered along the centreline. We can represent them as $c \in \mathbb{R}^{N\times 2}$. x_i and y_i are given in a local Cartesian coordinate frame centred around the target vehicle.

In the following, we describe our algorithm to transform a point $P = (x, y)$ given in the same Cartesian frame into a Frenet frame defined by the centreline c, i.e.

$$(s, d) = \mathcal{F}^{(c)}(x, y)\,. \tag{2}$$

As we will discuss afterwards (cf. Sec. 2.2), we can vectorise this algorithm to transform a set of M points at once efficiently.

Algorithm Our algorithm first generates a lookup table that stores the longitudinal progress p_i of each point i in c. Then, the point C_l in c, which is closest to P is calculated. The longitudinal coordinate s is then the progress of C_l, i.e., p_l. The lateral offset can be calculated as $|d| = \|P - C_l\|_2$. This makes use of the property that if C_l is the closest point to P, then the vector $P - C_l$ is perpendicular to the reference lane c in C_l. We define the sign of d to be positive if P is located on the left of c, i.e.,

$$sign(d) = sign((C_l - C_{l-1}) \times (P - C_l))\,. \tag{3}$$

Hence, the transformed coordinates are

$$(s, d) = \mathcal{F}^{(c)}(x, y) = (p_j,\ sign(d) \cdot \|P - C_l\|_2)\,. \tag{4}$$

Lookup table for longitudinal progress along path In the first step, we calculate a lookup table that stores the progress p_i of each point i. We start by calculating the progress increments between the points on the reference path

$$\Delta p_i = \|(x_i^c - x_{i-1}^c, y_i^c - y_{i-1}^c)\|_2 \qquad \forall\, i = 2..M\,. \tag{5}$$

The progress of each point along the reference lane is obtained by integration of Δp_i:

$$\hat{p}_i = \sum_{j=2}^{i} \Delta p_j \qquad \forall\, i = 2..M \tag{6}$$

The progress of the first point p_1 is zero by definition. Both operations can be computed efficiently as they are parallelisable vectorised operations.

In order to clearly define a Frenet frame based on the reference centreline c, we need to define an origin on the centreline c. We chose to select the point which is closest to the target vehicle's current position as the origin. As the local Cartesian coordinate frame is centred around the vehicle, its current position is $(0, 0)$. Hence, we can infer the index o of the closest point as

$$o = \underset{i}{\operatorname{argmin}} \|c\|_2 \tag{7}$$

where $\|.\|_2$ denotes an elementwise Euclidean distance. This is as well a vectorised operation that can be computed efficiently. To shift the origin to o, we calculate the progress

$$p_i = \hat{p}_i - p_o\,. \tag{8}$$

Matching the closest point Our goal is to find the index l of the closest point in c to P. Therefore, we first calculate the distance between all points in c and P,

$$|d_i| = \|x_i^c - x, x_i^c - y\|_2 \,. \tag{9}$$

The closest point can be obtained by $l = \text{argmin}_i \, |d_i|$.

Output. The final transformation of P is hence given as

$$\mathcal{F}^{(c)}(x, y) = (p_l, sign(d_l) \cdot |d_l|) \tag{10}$$

2.2 Vectorisation

The algorithm can be vectorised to efficiently transform a set of M points $P_j = (x_j, y_j)$, $j = 1..M$ in parallel, i.e.,

$$\{s_j, d_j\}_{j=1}^M = \mathcal{F}^{(c)}(\{x_j, y_j\}_{j=1}^M) \,. \tag{11}$$

As the centreline is identical for all points, Eqs. 5 to 8 remain unchanged. In order to calculate the closest point $C_{l(j)}$ in c for every point P_j in parallel, we first calculate the pairwise distance between all points in c and P_j

$$D_{ij} = \|x_i^c - x_j, x_i^c - y_j\|_2 \,. \tag{12}$$

This is the most computationally intensive step of our transformation. Still, there are relatively efficient implementations for this [4]. Then, $l(j) = \text{argmin}_i \, D_{ij}$ denotes the index of the point in c that is closest to P_j. Finally, we get the transformed coordinates

$$(s_j, d_j) = \mathcal{F}^{(c)}(x_j, y_j) = \left(P_{l(j)}, sign(d_j) \cdot |D_{l(j),j}|\right) \,, \tag{13}$$

where

$$D_{l(j),j} = \|P_j - C_{l(j)}\|_2 \tag{14}$$

is the distance of P_j to the reference line.

Computational Burden in Multi-Agent Settings. In a multi-agent setting, the prediction model has to be inferred for each actor in the scene. This makes additional operations, such as the coordinate transform, expensive. However, in our case, the number of operations does not increase linearly with the number of agents. For instance, if several agents are located on the same centreline, the transformed scene representation can be reused. Only the origin has to be shifted, which is a computationally cheap operation (cf. Eq. 8). Thus, the number of operations depends on the number of lanes that are used as reference paths for the transformation. Hence, computational complexity does not scale linearly with the number of agents and is instead bounded by the number of relevant centrelines in the scene.

2.3 Handling Corner Cases

What if the property is violated, that the shortest distance is orthogonal on the closest point? The problem is depicted in Fig. 3. It demonstrates how point p_2 cannot be accurately transformed, as the distance d_2 to the closest point is not orthogonal to the curve. The problem is a jump in curvature. Green shows all points that can be transformed without error. In the second part, the same centreline is sampled at a higher resolution. It can be seen how the area of points where the assumption breaks shrinks. We make use of this and increase the resolution of the reference centreline c before applying the transformation algorithm. Thus, we fit a spline to the sequence of points and supersample it. However, the increased number of points N in the centreline increases the computational burden, especially when calculating the distances as described in Eq. 12. We found that supersampling the resolution by a factor of 100 effectively balances computational overhead and transformation accuracy. In our experiments, this results in spacing between adjacent points of 0.05m.

No unique closest point. In cases where a point p_j cannot be assigned to a unique closest point $c_{l(j)}$, the transformation is ambiguous. This is demonstrated in Fig. 4, where the distance of p_1 is at the centre of a circular curve. Hence, its distance to multiple points on the reference lane is d_1. To resolve the ambiguity, we pick the point which is closest to the target vehicle, i.e., the one with the smallest absolute progress $|p|$. Note that this choice does not prevent accurate reconstruction.

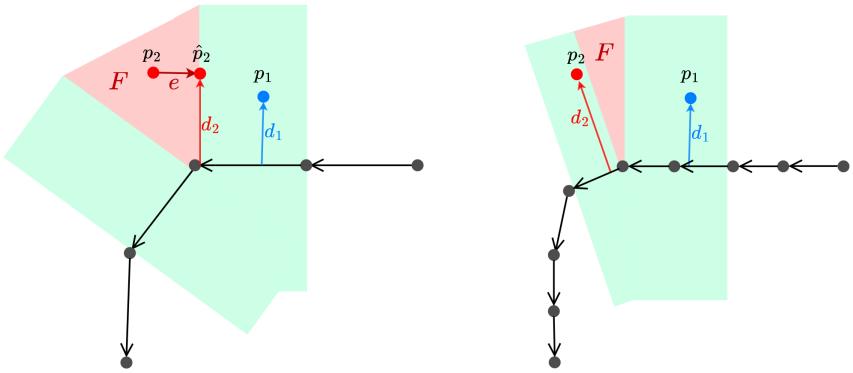


Figure 3: Transformation errors

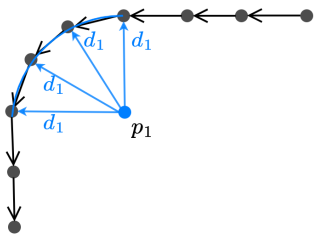


Figure 4: Corner case, the closest point on the reference path is not unique.

3 Further Ablations

In Tab. 1 we report additional ablations that shed light on the reduced performance of the Frenet models compared to the Cartesian models.

3.1 Privliedged Prior

Models denoted with an asterisk * only predict conditioned on the ground truth future centreline, i.e. they have perfect centreline scoring, and results serve as an upper bound. We find that this significantly reduces the gap between the models with $\hat{K} = N \cdot 6$ and the ones with $\hat{K} = 6$. Moreover, Multipath++SD* even outperforms the Cartesian Multipath++ [5] model. This emphasizes that an accurate estimation of the prior over the lanes significantly improves prediction performance on the original scenes. Hence, we conclude that the reduced performance of the $\hat{K} = 6$ models without the perfect lane scoring is mostly caused by the naive assumption of a uniform prior over the lanes.

Model	**prior/mode selection**	$\hat{K}$	**minADE [m]**	**minFDE [m]**	**ORP [%]**	**MR1 [%]**	**MIED [m]**
Multipath++ [5]	-	6	0.772	1.399	1.7	56.7	2.908
Multipath++SD*	privileged	6	0.808	1.390	1.2	59.3	3.098
Multipath++SD-GS	greedy-sampling	6	0.837	1.470	1.2	59.8	3.216
Multipath++SD-LS	lane-scoring	6	0.915	1.688	1.5	59.8	3.019
Multipath++SD-KM	clustering	6	0.833	1.470	1.3	65.3	3.518
MultiPath++SD-UP	uniform	6	0.936	1.742	1.6	61.7	2.942
MultiPath++SD-UP	uniform / all	$N \cdot 6$	0.767	1.274	1.4	61.7	3.473
LaneGCN [3]	-	6	0.645	1.076	0.7	48.7	3.800
LaneGCN-SD*	privileged	6	0.716	1.173	1.1	53.6	4.347
LaneGCN-SD-GS	greedy-sampling	6	0.737	1.233	1.1	54.1	4.261
LaneGCN-SD-LS	lane-scoring	6	0.760	1.292	1.0	54.1	4.016
LaneGCN-SD-KM	clustering	6	0.759	1.315	1.4	65.1	4.703
LaneGCN-SD-UP	uniform	6	0.777	1.335	1.3	55.4	4.091
LaneGCN-SD-UP	uniform / all	$N \cdot 6$	0.704	1.138	1.3	55.4	4.678

Table 1: **Results on original scenes**

3.2 Uniform Prior

In contrast to the privileged prior models, models denoted with UP assume a uniform prior over the centrelines $C^{(i)}$, i.e. $p(C^{(i)}) = \mathcal{U}$. Then, we pick the $\hat{K}$ trajectories with the highest probabilities $p(t_i)$, which are obtained by marginalising the predicted conditioned probabilities $p(t_i|C^{(i)})$. We use the same uniform prior assumption in order to evaluate the miss-rate when selecting a variable number of trajectories $\hat{K} = N \cdot 6$.

3.3 Lane Scoring Model

We employ a simple feedforward model that predicts a score for each centreline in the scene. A subsequent softmax layer across all centrelines in the scene yields the probability for each lane. The model inputs comprise the target vehicle's past trajectory as well as a sequence of poses along the centreline, given in Cartesian coordinates w.r.t. the current pose. Both are independently processed by a single linear layer and then concatenated to form a feature vector. Afterwards, an MLP with two hidden layers, each comprising 512 neurons, regresses the score for the given centerline. The lane scoring model is trained for ten epochs at a learning rate of $1e-4$ with a batch size of 128. We use the cross-entropy loss as training objective. We denote the models that are evaluated with the lane scoring prior with the suffix LS.

3.4 Clustering

As an alternative to estimating the prior, we employ a K-means clustering of the $N \cdot K$ trajectories to obtain a set of K output trajectories [6]. The estimated probability of each trajectory is obtained from the inverse rank of the cluster. We denote this ablation with the suffix KM.

3.5 Greedy Sampling

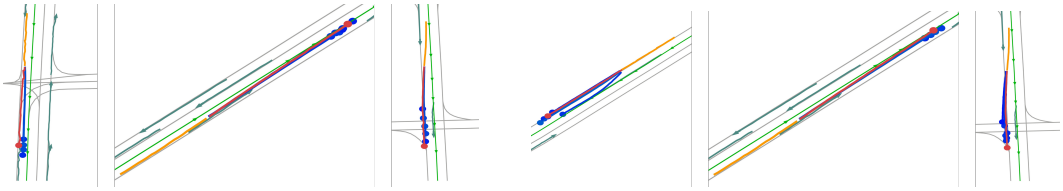
Similar to [7], the set of K output trajectories can be selected greedily. This extends the idea of the lane scoring model with a non-maximum suppression. First, the lane scoring model is inferred to obtain the estimated probability of each trajectory. Next, a greedy algorithm selects the most probable trajectory and removes all trajectories from the candidate set whose endpoint is within a radius of 1.0m of the selected one. This is repeated until $K = 6$ trajectories are selected. We denote this model with the suffix GS.

3.6 Results

We find that selecting a set of $\hat{K} = 6$ trajectories with a uniform prior results ($-UP$) in the highest displacement errors. As the ablation that leverages a privileged prior (*) performs substantially better, we conclude that the impaired performance results from a suboptimal selection of the trajectories. Consequently, leveraging the lane-scoring model (LS) to estimate the prior over the lanes results in decreased displacement errors. Moreover, the miss-rate closely approaches the level of privileged prior ablation. Combining this with non-maxima-suppression, as it's done by our greedy sampling method (GS), results in the lowest displacement errors among the SD models while maintaining the low miss rate. Besides, the diversity (MIED) is higher than for the lane-scoring models, which select the trajectories with the highest estimated probability.

4 Additional Qualitative Results

Our proposed Frenet frame wrapper induces a bias to follow the lane, but the resulting models are still able to deviate significantly from the lane centreline. Fig. 5 shows that both LaneGCN-SD and Multipath++SD are able to predict lane changes. These can be induced by conditioning the model on an adjacent lane. We also observe that a lane change that has already been initiated is completed, even if the model is conditioned on the centreline that the target vehicle is about to leave.



(a) LaneGCN-SD (b) Multipath++SD

Figure 5: The two left images in both subfigures shows how the models predict a lane change onto an adjacent lane. In the rightmost image of each subfigure, the target vehicle already started to change to the neighbouring lanes. Both models are able to forecast a feasible completion for that lane change, even if they are conditioned on the centreline that the target vehicle is about to leave.

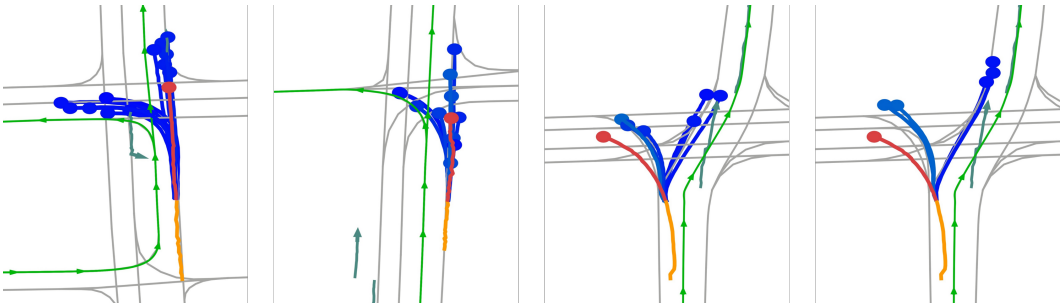


Figure 6: The two left columns show how the Multipath++SD model (leftmost) and the LaneGCN-SD model (second from left) predict an illegal left turn if they are conditioned on the neighbouring centreline. The two right columns show how the models are able to anticipate a left turn because the vehicle started turning to the left, even though they are conditioned on a centreline going straight. It can be seen how this significant deviation from the centreline results in an offroad prediction in the case of the LaneGCN-SD model (rightmost).

Moreover, Fig. 6 demonstrates non-compliant behaviour. Again, these predictions can be induced by conditioning the model on a neighbouring centreline. If the target vehicle already started a turning manoeuvre, the models are clearly able to anticipate the full turn irrespective of the chosen reference centreline.

We conclude that the Frenet frame wrapper does not prevent the model from learning deviations from trivial lane following.

References

[1] F. Frenet. Sur les courbes à double courbure. *Journal de mathématiques pures et appliquées*, 17:437–447, 1852.

[2] J.-A. Serret. Sur quelques formules relatives à la théorie des courbes à double courbure. *Journal de mathématiques pures et appliquées*, 16:193–207, 1851.

[3] M. Liang, B. Yang, R. Hu, Y. Chen, R. Liao, S. Feng, and R. Urtasun. Learning lane graph representations for motion forecasting. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 541–556. Springer, 2020.

[4] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0

Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi:10.1038/s41592-019-0686-2.

[5] B. Varadarajan, A. Hefny, A. Srivastava, K. S. Refaat, N. Nayakanti, A. Cornman, K. Chen, B. Douillard, C. P. Lam, D. Anguelov, et al. Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 7814–7821. IEEE, 2022.

[6] N. Deo, E. Wolff, and O. Beijbom. Multimodal trajectory prediction conditioned on lane-graph traversals. In *Conference on Robot Learning*, pages 203–212. PMLR, 2022.

[7] T. Gilles, S. Sabatini, D. Tsishkou, B. Stanciulescu, and F. Moutarde. Home: Heatmap output for future motion estimation. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 500–507. IEEE, 2021.