HAL open science

HOME: Heatmap Output for future Motion Estimation

Thomas Gilles, Stefano Sabatini, Dzmitry Tsishkou, Bogdan Stanciulescu, Fabien Moutarde

To cite this version:

Thomas Gilles, Stefano Sabatini, Dzmitry Tsishkou, Bogdan Stanciulescu, Fabien Moutarde. HOME: Heatmap Output for future Motion Estimation. 2021 IEEE International Intelligent Transportation Systems Conference (ITSC'2021), Sep 2021, Indianapolis, United States. hal-03356794

HAL Id: hal-03356794

https://minesparis-psl.hal.science/hal-03356794v1

Submitted on 28 Sep 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

arXiv:2105.10968v2 [cs.CV] 2 Jun 2021

HOME: Heatmap Output for future Motion Estimation

Thomas Gilles[1,2], Stefano Sabatini[1], Dzmitry Tsishkou[1], Bogdan Stanciulescu[2], Fabien Moutarde[2]

***Abstract*— In this paper, we propose HOME, a framework tackling the motion forecasting problem with an image output representing the probability distribution of the agent's future location. This method allows for a simple architecture with classic convolution networks coupled with attention mechanism for agent interactions, and outputs an unconstrained 2D top-view representation of the agent's possible future. Based on this output, we design two methods to sample a finite set of agent's future locations. These methods allow us to control the optimization trade-off between miss rate and final displacement error for multiple modalities without having to retrain any part of the model. We apply our method to the Argoverse Motion Forecasting Benchmark and achieve 1st place on the online leaderboard.**

I. INTRODUCTION

Forecasting the future motion of surrounding actors is an essential part of the autonomous driving pipeline, necessary for safe planning and useful for simulation of realistic behaviors. In order to capture the complexity of a driving scenario, the prediction model needs to take into account the local map, the past trajectory of the predicted agent and the interactions with other actors. Its output needs to be multimodal to cover the different choices a driver could make, between going straight or turning, slowing down or overtaking. Each modality proposed should represent a possible trajectory that an agent could take in the immediate future.

The challenge in motion prediction resides not in having the absolute closest trajectory to the ground truth, but rather in avoiding big failures where a possibility has not been considered, and the future is totally missed by all modalities. An accident will rarely happen because most predictions are offset by half a meter, but rather because of one single case where a lack of coverage led to a miss of more than a few meters.

A classic way to obtain k modalities is to design a model that outputs a fixed number of k future trajectories [6, 20, 21, 14], as a regression problem. This approach has however significant drawbacks, as training predictions all together leads to mode collapse. The common solution to this problem is to only train the closest prediction to the ground truth, but this diminishes the training data allocated to each predicted modality as only one is learning at each sample.

Later methods adapt the model to the multi-modal problem by conditioning the prediction to specific inputs such as lanes [11] or targets [34]. Finally, recent methods use the topological lane graph itself to generate trajectory for each node [32]. However each of these model constrains its prediction space to a restricted representation, that may be limited to represent the actual diversity of possible futures. For example, if the predicted modalities are constrained to the High Definition map graph, it becomes very hard to predict agent breaking traffic rules or slowing down to park at the side of the road.

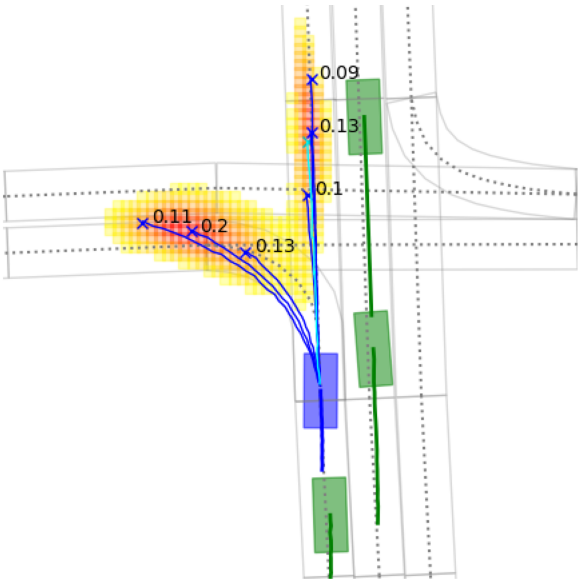


Fig. 1: Summary of our approach. The yellow/red heatmap is our predicted probability distribution and the blue points are the sampled final point predictions.

In this paper, following the same principle as recent state of the art method, which is that a future trajectory can be almost fully defined by its final point [34, 32], we reformulate the prediction problem in three steps. We first represent the possible futures distribution by a 2D probability heatmap that gives an unconstrained approximation of the probability of the agent position. This heatmap is represented as a squared image and it naturally accommodates for multimodal predictions where each pixel represent a possible future position of the target agent. It also enables to fully describe the future uncertainty in a probability distribution, without having to choose its modes or means. In a second step, we sample from the heatmap a finite number of possible future locations with the possibility to choose which metric we want to optimize without retraining the model. Finally, we build the full trajectories based on the past history and conditioned on the sampled final points.

Our contributions are summarized as follow:

[1]IoV team, Paris Research Center, Huawei Technologies France
[2]MINES ParisTech, PSL University, Center for robotics
Contact: thomas.gilles@mines-paristech.fr

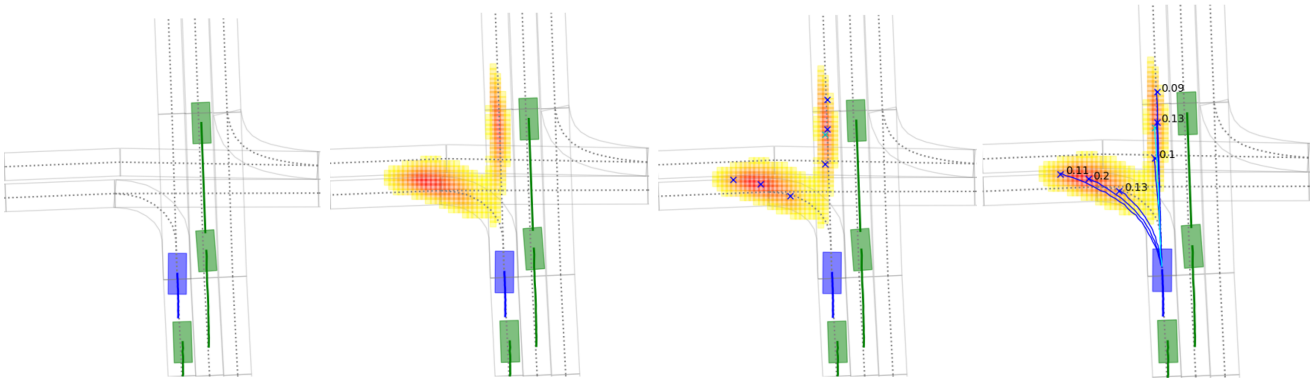


Fig. 2: HOME pipeline. a) Context map, target agent (blue) and neighbor (green) trajectories are given as input to the network. b) Heatmap output of the network. c) Sampled final points. d) Trajectories are built for each final point

- We present a simple model architecture made of a convolutional neural network (CNN), a recurrent neural network (RNN) and an Attention module, with a heatmap output allowing for easy and efficient training.
- We design two sampling algorithms from this heatmap output, optimizing MR_k or minFDE_k respectively.
- We highlight a trade-off between both metrics, and show that our sampling algorithm allows us to control this trade-off with a simple parameter.

II. RELATED WORK

Deep learning has brought great progress to the motion forecasting results [22]. A classic CNN architecture can be applied to a rasterized map to predict 2D coordinates [6].

In order to model interactions better between driving agents, attention has been introduced in multiple methods. The approach of [20] encodes separately agents and centerlines with 1D CNN and LSTM and then applies multi-head attention from actors to other actors and lines. MHA-JAM [21] concatenates agent features to a CNN-encoded map at their specific coordinates, and then applies attention on this joint representation. The work of [17] also uses attention between agents for interactions, and parallely applies an attention head on encoded lane to obtain lane probabilities and generate a modality for each given lane. mmTransformer [16] applies a general Transformer [30] architecture to fuse history, map and interactions.

Another family of methods use a pool of anchor trajectories, predefined [4] or model-based [23, 28], and rank them with a learned model. This allows to avoid any mode collapse and assert realistic trajectories, but removes the ability to tune the trajectories accurately to the current situation.

Multimodality can also be obtained using generative approaches that model the actual future probability distribution [12, 19, 29, 24, 25]. However, generative models require multiple independent sampling at inference time without any optimization of coverage or average distance.

More recently, methods have started to leverage the graph obtained from HD-map in order to better represent lane connectivity. VectorNet [9] encodes both map features and agent trajectories as polylines then merge them with a global interaction graph. LaneGCN [14] treats actor past and the lane graph separately, and then fuse them with a series of attention layers between lane and actors.

Other methods then use the graph to structure their multimodal outputs. TNT [34] builds from the VectorNet backbone and combines it with multiple target proposals sampled from the lanes in order to diversify the prediction points. GoalNet [33] also identifies possible goals and applied a prediction head for each on a localized raster in order to base the modalities on reachable lanes. WIMP [11] matches possible polylines to the past trajectory and uses them as conditional input to their model. LaneRCNN [32] adds actor features from the start to sampled nodes on the lanes, and then predicts a future point for each node along a probability.

Grid-based outputs have already been used in pedestrian behavior prediction such as [13, 7, 18, 10, 26]. Their model architecture, training and sampling strategies however differ greatly from ours. The work of [27] produces a future grid occupancy output prediction for each vehicle class in order to plan from it, but it is not instance-based and doesn't allow for individual vehicle prediction.

III. METHOD

We describe our general pipeline in Fig. 2. Our method takes as input a rasterized image of the agent environment, and outputs a probability distribution heatmap representing where the agent could be at a fixed time horizon T in the future. A finite set of possible locations are then extracted from the heatmap to ensure appropriate coverage. Future locations are sampled to minimize either rate of misses or final displacement errors. Finally for each sampled future location, a trajectory representing the motion of agent from the initial state to the future location is computed.

The aim of motion estimation is to predict the future positions of the target agent a for T timesteps $\{(x_a^t, y_a^t)$ for t in $[\![1, T]\!]\}$. The model is given the past H timesteps $\{(x_a^t, y_a^t)$ for t in $[\![-H, 0]\!]\}$ for the target agent a and the N neighbor agents a'. Supplementary context informations are available in the shape of a graph High

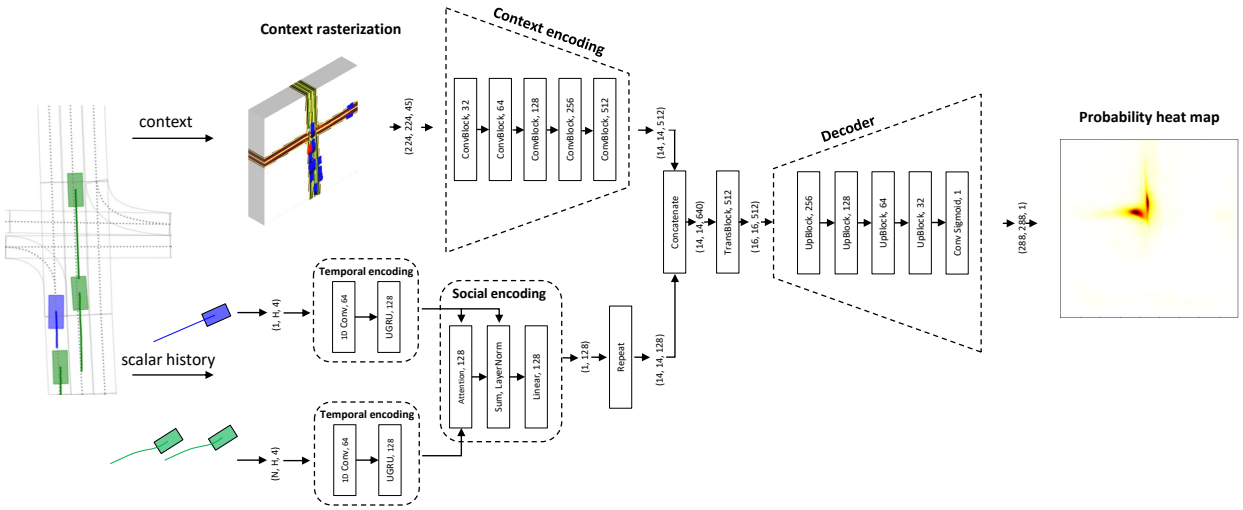


Fig. 3: Example of input and output data for our model with brief description of architecture

Definition Map (HD map). We will focus in this paper on the prediction of the final points (x_a^T, y_a^T), and then regress the whole trajectory conditioned on the end point.

A. Encoding history and local context information

1) Map and past trajectory encoding: The local context is available as a High Definition Map centered on the target agent. We rasterize the HD-Map in 5 semantic channels: drivable area, lane boundaries and directed center-lines with their headings encoded using HSV on 3 channels. We also add the target agent trajectory as a moving rectangle on 20 history channels and the other agents history on 20 more channels. The final input is a (224, 224, 45) image with a 0.5 x 0.5 m² resolution per pixel. This image is processed by a classic CNN model alternating convolutional layers and max-pooling for downscaling to obtain a (14, 14, 512) encoding E_{raster} as illustrated in the top-left part of Fig. 3.

The scalar history of the agents is also taken as input to the model as a list of 2D coordinates. Missing timesteps are padded with zeros and a binary mask indicating if padding was applied is concatenated to the trajectory, as well as the timestamps for each step, so to obtain a $(H, 4)$ input for each agent. Each agent trajectory goes through a 1D convolutional layer followed by a UGRU[8] recurrent layer. The weights are shared for all agents except the target agent.

2) Inter-agent attention for interaction: Similar to [20, 21, 17], we use attention [30] to model agent interaction. A query vector is generated for the target agent, while key and value vectors are created for the other actors. The normalized dot product of query and keys creates an attention map from the target agent to the other agent, then used to pool their value features into a context vector. The context vector is then added to the target vehicle feature vector through a residual connection followed by LayerNormalization [2]. The obtained trajectory encoding $E_{trajectory}$ is then repeated to match the context encoding E_{raster} dimensions. The final encoding $E_{context}$ is the result of the concatenation of both encodings E_{raster} and $E_{trajectory}$.

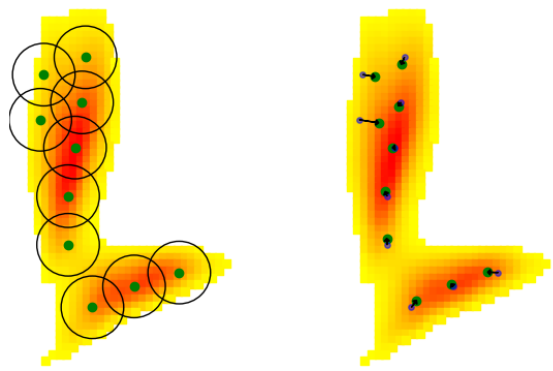
3) Increased output size for longer range: Due to high speed, some cars may go through a greater range in the time horizon T that is covered by the input range of 56m. However, simply increasing input size would greatly add to the computational burden while not necessarily bringing useful information. We therefore want to increase the output size while retaining the spatial correspondences through the layers. In order to do so, we apply Tranpose Convolutions with stride 1 and kernel size 3. Since 1 input pixel is connected to a grid of 3x3 output pixels, the edge pixels generate a new border of pixels around them, increasing the encoding size by 1 in each direction. We apply 2 of these layers, resulting in a (18, 18, 512) augmented encoding so that once upscaled the decoded image output will be of size (288, 288), corresponding to a 72m range.

B. Heatmap output

The final part of the model is a convolutional decoder alternating transpose convolutions for upscaling and classic convolutions, topped with a sigmoid activation. We output an image $\hat{Y}$ with similar resolution as the raster input (0.5 x 0.5 m² / pixel). The output target is an image Y with a Gaussian centered around the ground truth position. This image is trained with a pixel-wise focal loss inspired from [35], averaged over the total P pixels p of the heatmap:

$$L = -\frac{1}{P}\sum_p (Y_p - \hat{Y}_p)^2 f(Y_p, \hat{Y}_p)$$
$$\text{with } f(Y_p, \hat{Y}_p) = \begin{cases} \log(\hat{Y}_p) & \text{if } Y_p\text{=1} \\ (1 - Y_p)^4 \log(1 - \hat{Y}_p) & \text{else} \end{cases} \quad (1)$$

where the non-null pixels around the Gaussian center serve as penalty-reducing coefficients, and the square factor of



(a) MR sampling (b) FDE sampling

Fig. 4: Illustration of sampling methods

error allows the gradient to focus on poorly-predicted pixels. We use a standard deviation of 4 pixels for the Gaussian.

C. Modality sampling

Our aim is here to sample the probability heatmap in order to optimize the performance metric of our choice. In most datasets such as Argoverse [5] and NuScenes [3], two main metrics are used for the final predicted point: MissRate (MR) and Final Displacement Error (FDE). MissRate corresponds to the percentage of prediction being farther than a certain threshhold to the ground truth, and FDE is simply the mean of l_2 distance between the prediction and the ground truth. When the output is multimodal, with k predictions, minimum Final Displacement Error minFDE$_k$ and Miss Rate over the k predictions MR$_k$ are used.

1) Optimizing Miss Rate: We design a sampling method in order to optimize the Miss Rate between the predicted modalities and the ground truth. A case is defined as missed if the ground truth is further than 2m from the prediction. For a given area A, the probability of the ground truth Y being in this area is equal to the integral of the probability distribution p under this ground truth.

$$P(Y \in A) = \int_{x \in A} p(x)dx \tag{2}$$

Therefore, for k predictions, given a 2D probability distribution, the sampling minimizing the expected MR is the one maximizing the integral of the future probability distribution under the area defined as 2m radius circles around the k predictions:

$$E(\mathbb{1}_{\min_k \|c_k - Y\| > 2}) = 1 - \sum_k \int_{\|c_k - x\| < 2} p(x)dx \tag{3}$$

We therefore process in a greedy way as described in Algo. 1, and iteratively select the location with the highest integrated probability value in its 2m circle. Once we obtain such a point, we set to zero the heatmap values under the defined circle and move on to selecting the next point with the same method.

Algorithm 1: MR Sampling Algorithm

input: Probability map p(x)
K number of predictions
R threshhold for Miss Rate
for k = *1..K* **do**
  Find c_k maximizing $\int_{\|c_k - x\| < R} p(x)dx$
  Set $p(x) = 0$ for all x such that $\|c_k - x\| < R$
end

The result is illustrated in Fig. 4a. We see that each sampled point can be surrounded by a circle of radius 2m that barely overlaps with other circles. Each point is sampled almost equidistant to the others, as setting the probability under previous points to zero sets a very strict limit to the minimum distance between points.

For implementation, we process the covered area for each point using a convolution layer with kernel weights fixed so to approximate a 2m circle. In practice, we don't actually use a radius of 2 meters, but a 1.8 meters one as we found out it to yield better performance. We also upscale the heatmap to 0.25 x 0.25 m_2 per pixel with bilinear interpolation to have a more refined prediction location.

2) Optimizing Final Displacement Error: We inspire ourselves from KMeans to optimize minFDEk. The image output can be represented as a discrete probability distribution (x_i, p_i) where x_i represents the pixel centers and p_i the associated probability value. Optimizing the Final Displacement Error over k predictions means finding k centroids c_k that minimize the following quantity:

$$minimize_c \sum_i p_i \|c - x_i\| \tag{4}$$

To do so we design our sampling algorithm for FDE optimization detailed in Algo. 2.

We replace the classic weighted average $\sum_i p_i x_i$ for each centroid c_k by $\sum_i \frac{p_i}{d_i^k} x_i$ where d_i^k is the distance between point x_i and centroid c_k to be more robust to outliers and take into account the optimisation of l_2 norm instead of its square.

In essence, we update each prediction as a weighted average of its local neighborhood in a radius of 3m. The coefficient $\frac{m_i}{d_i^k}$, with m_i the distance between point x_i and its closest centroid allows for flexible partition boundaries compared to KMeans (where we would use $\mathbb{1}_{d_i^k <= m_i}$ instead): when x_i is in the partition of prediction k, its value is 1, while when it's outside it decreases, so as to be 0 when at the exact position of another prediction k', where it could never be improved by a displacement of k.

We initialize the centroids with the results of the Miss Rate optimization algorithm and use the number of iterations L as a parameter to tune the trade-off between Miss Rate and FDE: when L is zero, Miss Rate is optimized while

Algorithm 2: FDE Sampling Algorithm

input: Set of points x_i with probability weight p_i
L number of iterations to run the algorithm
Initialization of K centroids c_k

for l = *1..L* **do**

Compute d_i^k the matrix of distance of point x_i to each centroid c_k

Compute m_i the distance of point x_i to the closest centroid c_k

for k = *1..K* **do**

Compute new centroid coordinates :

$$c_k = \frac{1}{N} \sum_i \mathbb{1}_{d_i^k <= 3} \frac{p_i}{d_i^k} \frac{m_i}{d_i^k} x_i$$

with $N = \sum_i \mathbb{1}_{d_i^k <= 3} \frac{p_i}{d_i^k} \frac{m_i}{d_i^k}$

end

end

when L increases MR is sacrificed to get better FDE. The output of the algorithm is illustrated in Fig. 4b, where is it can be observed that centroids are brought closer together, sacrificing total coverage but getting closer to areas with high probabilities to reduce the expected distance. Results of this trade-off are illustrated further in Sec. IV-C.2, where we show in Fig. 6 that every iteration of Algo. 2 diminishes minFDE$_6$ and increases MR$_6$.

D. Full trajectory generation

We use a separate model to generate full trajectories connecting the initial agent position to all sampled locations. This model applies a fully-connected layer to encode the target agent history into a vector of 32 features, which is then concatenated with the (x, y) coordinates of the target future location. Another fully-connected layer is then applied to obtain a 64 feature vector, which is then transformed through a last fully-connected layer to a set of locations representing the intermediate position of the agent in the time frame $[\![1, T]\!]$. The probability of a trajectory is the integral of the probability heatmap under the circle of radius 2m around the end point of the trajectory.

IV. EXPERIMENTS

A. Experimental settings

1) Dataset: We use the Argoverse Motion Forecasting Dataset [5]. It is a car trajectory prediction benchmark with 205942 training samples, 39472 validation samples and 78143 test samples. Each sample contains the position of all agents in the scene in the past 2s as well as the local map, and the labels are the 3s future positions of one target agent in the scene.

2) Metrics: We report the previously defined metrics MR$_k$ and minFDE$_k$ for k=1,6, completed by the minimum Average Displacement Error minADE$_k$ which is the average l_2 error over all successive trajectory points. We also report the metrics p-minFDE$_6$ and p-minADE$_6$ for the test set, where $-\log(p)$ is added to the metric, p being the probability assigned to the best (closest to ground-truth) predicted trajectory. These later metrics allow to measure the quality of the probability distribution assigned to the predictions.

3) Implementation details: We train all models for 16 epochs with batch size 32, using Adam optimizer initialized with a learning rate of 0.001. Each sample frame is centered on the target agent and aligned with its heading. We divide learning rate by half at epochs 3, 6, 9 and 13. We augment the training data by dropping each raster channel with a probability of 0.1 and rotating the frame by a uniform random angle in $[-\pi/4, \pi/4]$ in 50% of the samples. All convolution layers are CoordConv [15] with a kernel of size 3x3 (3 for 1D Convs) and are followed by BatchNormalization and ReLU activation.

B. Comparison with State-of-the-art

We show in Tab. I our results compared to other methods on the Argoverse motion forecasting test set. The benchmark is ranked by MR$_6$, where we rank first and significantly improve on previous results, demonstrating that having the heatmap output enables the best coverage with respect to the prior art. We also outperform other methods on both

TABLE I: Results on Argoverse Motion Forecasting Leaderboard [1] (test set)

	K=1			K=6				
	minADE	minFDE	MR	minADE	minFDE	p-minADE	p-minFDE	MR
WIMP [11]	1.82	4.03	62.9	0.90	1.42	2.69	3.21	16.7
LaneGCN [14]	1.71	3.78	59.1	0.87	1.36	2.66	3.16	16.3
Alibaba-ADLab	1.97	4.35	63.4	0.92	1.48	2.64	3.23	15.9
TPCN [31]	**1.64**	**3.64**	58.6	**0.85**	**1.35**	2.61	3.11	15.9
HIKVISION-ADLab-HZ	1.94	3.90	58.2	1.21	1.83	3.00	3.62	13.8
TNT [34]	1.78	3.91	59.7	0.94	1.54	2.73	3.33	13.3
Jean [20]	1.74	4.24	68.6	1.00	1.42	2.79	3.21	13.1
TMP [16]	1.70	3.78	58.4	0.87	1.37	2.66	3.16	13.0
LaneRCNN [32]	1.69	3.69	**56.9**	0.90	1.45	2.70	3.24	12.3
SenseTime_AP	1.70	3.76	58.3	0.87	1.36	2.66	3.16	12.0
poly (3rd)	1.70	3.82	58.8	0.87	1.47	2.67	3.28	12.0
PRIME (2nd) [28]	1.91	3.82	58.7	1.22	1.56	2.71	3.04	11.5
Ours-HOME (FDE L=4)	1.72	3.73	58.4	0.92	1.36	2.64	3.08	11.3
Ours-HOME (MR) (1st)	1.73	3.73	58.4	0.94	1.45	**2.52**	**3.03**	**10.2**

TABLE II: Ablation study on output representation (Argoverse validation set)

Bottleneck	Output	K=1		K=6	
		minFDE	MR	minFDE	MR
Scalar	Regression	3.81	61.7	1.26	13.0
Scalar	Heatmap	3.07	51.9	1.30	8.0
Image	Heatmap	3.02	50.7	1.28	6.8

p-minFDE$_6$ and p-minADE$_6$, demonstrating superior modelling of the probability distribution between predictions. Another interesting observation is that methods performing very well on minFDE$_6$ such as LaneGCN [14] and TPCN [31] have a worse MR$_6$ as drawback. PRIME [28] has the closest MR$_6$ to ours but a much higher minFDE$_6$ in comparison. We show the results of both our sampling optimized for MR and minFDE with the same trained model. Our FDE sampling with $L = 4$ sacrifices 1.1 points of MR$_6$ for 9 cm of minFDE$_6$, which gets us second best on minFDE$_6$ while still being good enough for 1st position on the leaderboard.

C. Ablation studies

We discuss the importance of our difference contributions, starting by comparing our output representation to the traditional scalar coordinates output, then decomposing our model architecture and sampling strategies. All metrics are reported on the Argoverse validation set. If not specified otherwise, MR sampling is used.

1) Heatmap output: We show the effect of output representation in Tab. II by using the same encoding backbone and replacing the image decoder with a global pooling followed by a regression head of 6 coordinate modalities. We train the regression output with a winner-takes-all l_1 regression loss similar to [21, 14, 11, 31, 6] and a classification loss where target is obtained through a softmax on distances between predictions and ground-truth, as in [34, 28]. Since the global pooling leads to loss of spatial information from the image, for fair comparison we also include a model with "scalar bottleneck" where pooling is also applied on the image encoding and is then reshaped to form an image on which is applied the heatmap decoder. We observe that heatmap outputs yields much better Miss Rate, and that having a scalar pooling bottleneck diminishes performance as it creates information loss, but not significantly. Interestingly, the regression output reaches better minFDE$_6$ when compared to the MR-optimized sampled image output models, but is still worse than FDE-optimized model, as this scalar coordinates output doesn't leave room for any post-processing optimization.

We also show the effect of adding more modalities to a regression output in Fig 5 : even if the MR$_k$ improves for the total number of modalities as k increases, the performance for a fixed k such as 1 or 6 worsens. [11] and [33] notice a similar trend, obtaining much better results for lower k metrics when training less modalities. Furthermore, for a regression output model a new training is required each time to accommodate the maximum number of modalities, whereas with heatmap output any number of modalities can

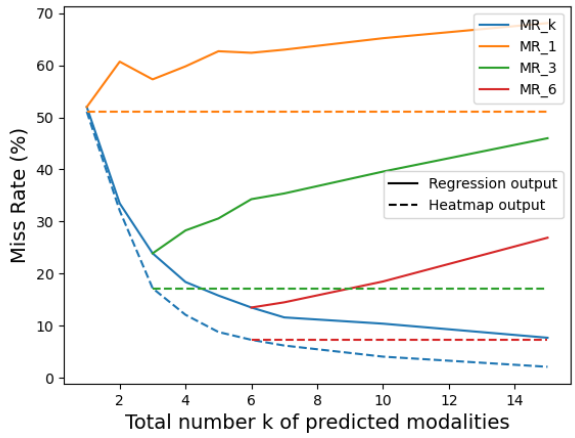


Fig. 5: Effect of maximum number k of modalities trained on metrics of lower fixed modality numbers. Full lines are results of regression output model. Dashed lines are result of our heatmap output model. We show the Miss Rate for total number of predicted modalities k (blue) and fixed number of modalities 1 (orange), 3 (green) and 6 (red).

be obtained at will with the same training, and the lower k numbers are not impacted by the total number of modalities extracted, as showed by the dashed horizontal lines displayed for MR$_1$, MR$_3$ and MR$_6$. Finally, our model heatmap output scales better with the number of k modalities, converging to a 0% MR faster that the regression output model.

2) Trajectory sampling: We show in Fig. 6 the results of our trade-off between MR$_6$ and FDE$_6$ on the Argoverse test set thanks to the parameter L of Algo. 2. We also include points for the other top 10 methods of the leaderboard for comparison. Our method reaches best possible MR$_6$, and

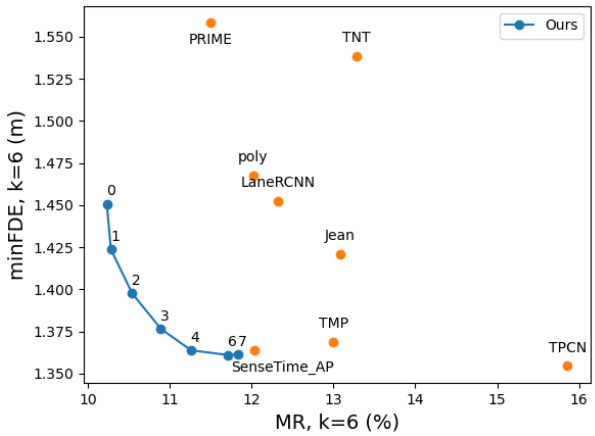


Fig. 6: FDE$_6$ - MR$_6$ trade-off. Lower-left is better. Points of the curve (blue) are obtained increasing number of iteration L of Algorithm 2 from 0 to 7. Points for other top-10 leaderboard methods are also included (orange).

TABLE III: Ablation study on trajectory sampling (Argoverse validation set)

Bottleneck	K=1		K=6	
	minFDE	MR	minFDE	MR
Pixel ranking with NMS	3.07	51.0	1.21	10.7
KMeans	3.06	51.6	1.23	9.3
Ours (MR)	3.02	50.7	1.28	**6.8**
Ours (FDE L=6)	**3.01**	**50.5**	**1.16**	7.4

allows to improve FDE_6 to second-best while still being first in MR_6 (fourth curve point obtained with $L = 4$)

We highlight our sampling results in Tab III and compare them to other possible sampling strategies: we try ranking pixels by probability and select them in decreasing order while removing overlapping pixels that are closer than a 1.8m radius following a classic Non-Maximum Suppression method. We also try KMeans as is used in [18].

D. Qualitative results

We show supplementary qualitative results in Fig. 7. We highlight examples of straight line, overtaking, curve road, going outside the map and intersections. Our model heatmap output makes use and usually follows the prior from the context map, but it is also able to divert from it based on interactions, realistic observations and hints of divergence from history.

V. CONCLUSION

We have presented HOME, a novel representation for multimodal trajectory prediction. It is based on predicting the future final point position on a 2D top-view grid, decoding then this final point into a full trajectory. This heatmap output represents the complete future probability distribution and its uncertainties, from which we design two prediction sampling methods. Sampling directly from the heatmap distribution enables a more optimized coverage, achieving state-of-the-art performance on the Argoverse Motion Forecasting benchmark.

ACKNOWLEDGMENT

We would like to thank Thomas Wang and Camille Truong-Allié for useful comments on the paper, as well as Arthur Moreau and Joseph Gesnouin for insightful discussions.

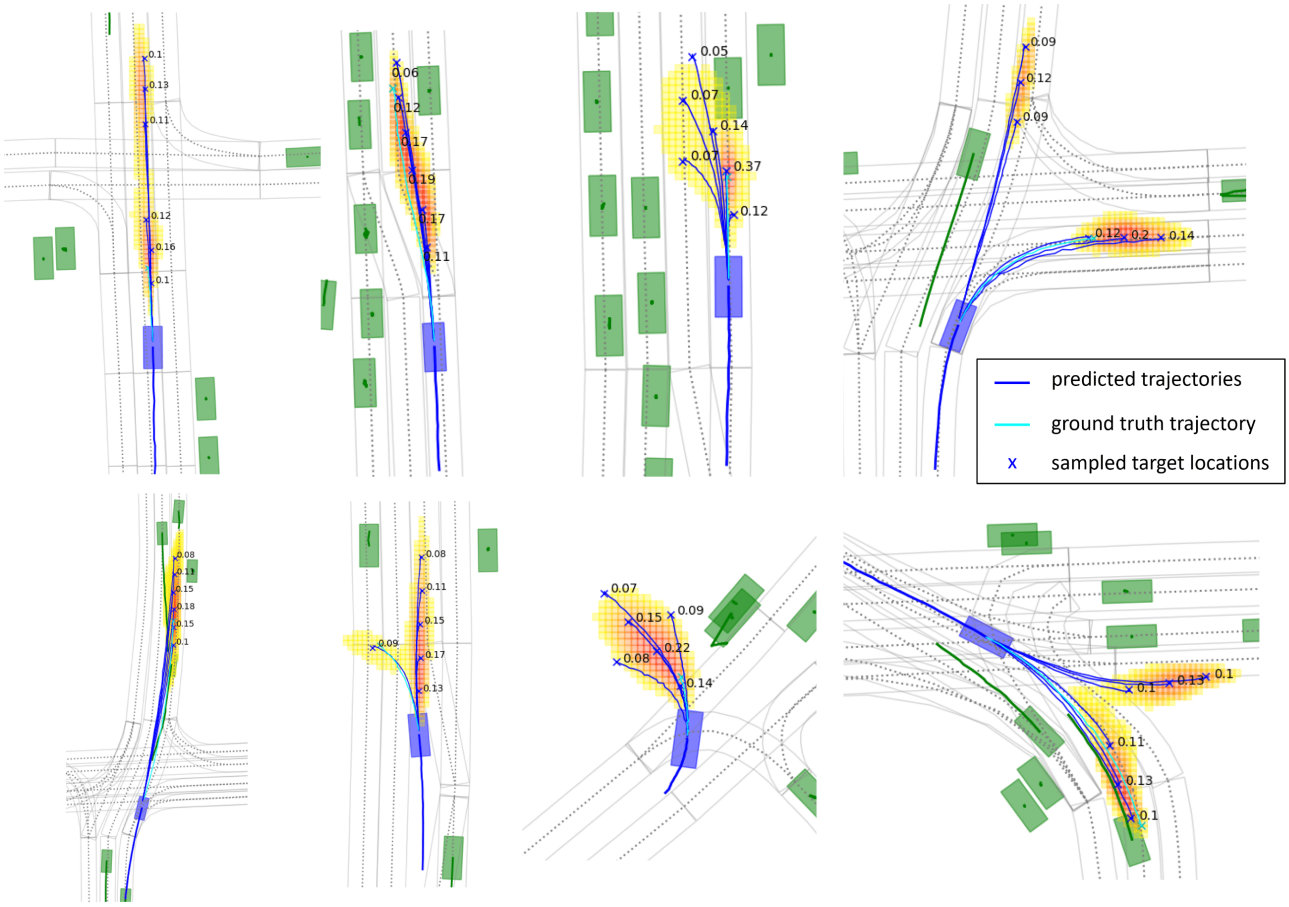


Fig. 7: Qualitative examples. The yellow/red heatmap is our predicted probability distribution and the blue points are the sampled final point predictions. The ground truth trajectory is shown in green.

REFERENCES

[1] *Argoverse motion forecasting competition*. https://eval.ai/web/challenges/challenge-page/454/leaderboard/1279. Accessed: 2021-03-12.

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. "Layer normalization". In: *arXiv:1607.06450*. 2016.

[3] Holger Caesar et al. "nuScenes: A multimodal dataset for autonomous driving". In: *CVPR*. 2020.

[4] Yuning Chai et al. "MultiPath: Multiple Probabilistic Anchor Trajectory Hypotheses for Behavior Prediction". In: *CoRL*. 2020.

[5] Ming-Fang Chang et al. "Argoverse: 3d tracking and forecasting with rich maps". In: *CVPR*. 2019.

[6] Henggang Cui et al. "Multimodal trajectory predictions for autonomous driving using deep convolutional networks". In: *ICRA*. 2019.

[7] Nachiket Deo and Mohan M Trivedi. "Trajectory forecasts in unknown environments conditioned on grid-based plans". In: *arXiv:2001.00735* (2020).

[8] Ahmet Erdem. *6th Place Solution: Very Custom GRU*. www.kaggle.com/c/riiid-test-answer-prediction/discussion/209581.

[9] Jiyang Gao et al. "Vectornet: Encoding hd maps and agent dynamics from vectorized representation". In: *CVPR*. 2020.

[10] Ajay Jain et al. "Discrete residual flow for probabilistic pedestrian behavior prediction". In: *ECCV*. 2020.

[11] Siddhesh Khandelwal et al. "What-If Motion Prediction for Autonomous Driving". In: *arXiv:2008.10587* (2020).

[12] Namhoon Lee et al. "Desire: Distant future prediction in dynamic scenes with interacting agents". In: *CVPR*. 2017.

[13] Junwei Liang et al. "The garden of forking paths: Towards multi-future trajectory prediction". In: *CVPR*. 2020.

[14] Ming Liang et al. "Learning lane graph representations for motion forecasting". In: *ECCV*. 2020.

[15] Rosanne Liu et al. "An intriguing failing of convolutional neural networks and the CoordConv solution". In: *NeurIPS*. 2018.

[16] Yicheng Liu et al. "Multimodal Motion Prediction with Stacked Transformers". In: *arXiv preprint arXiv:2103.11624* (2021).

[17] Chenxu Luo et al. "Probabilistic Multi-modal Trajectory Prediction with Lane Attention for Autonomous Vehicles". In: *arXiv:2007.02574* (2020).

[18] Karttikeya Mangalam et al. "From Goals, Waypoints & Paths To Long Term Human Trajectory Forecasting". In: *arXiv:2012.01526* (2020).

[19] Karttikeya Mangalam et al. "It is not the journey but the destination: Endpoint conditioned trajectory prediction". In: *ECCV*. 2020.

[20] Jean Mercat et al. "Multi-head attention for multi-modal joint vehicle motion forecasting". In: *ICRA*. 2020.

[21] Kaouther Messaoud et al. "Multi-Head Attention with Joint Agent-Map Representation for Trajectory Prediction in Autonomous Driving". In: *arXiv:2005.02545* (2020).

[22] Sajjad Mozaffari et al. "Deep learning-based vehicle behavior prediction for autonomous driving applications: A review". In: *IEEE Transactions on Intelligent Transportation Systems* (2020).

[23] Tung Phan-Minh et al. "Covernet: Multimodal behavior prediction using trajectory sets". In: *CVPR*. 2020.

[24] Nicholas Rhinehart, Kris M Kitani, and Paul Vernaza. "R2p2: A reparameterized pushforward policy for diverse, precise generative path forecasting". In: *ECCV*. 2018.

[25] Nicholas Rhinehart et al. "Precog: Prediction conditioned on goals in visual multi-agent settings". In: *CVPR*. 2019.

[26] Daniela Ridel et al. "Scene compliant trajectory forecast with agent-centric spatio-temporal grids". In: *IEEE Robotics and Automation Letters* (2020).

[27] Abbas Sadat et al. "Perceive, predict, and plan: Safe motion planning through interpretable semantic representations". In: *ECCV*. 2020.

[28] Haoran Song et al. "Learning to Predict Vehicle Trajectories with Model-based Planning". In: *arXiv:2103.04027* (2021).

[29] Yichuan Charlie Tang and Ruslan Salakhutdinov. "Multiple Futures Prediction". In: *NeurIPS*. 2019.

[30] A Vaswani et al. "Attention is all you need". In: *NIPS*. 2017.

[31] Maosheng Ye, Tongyi Cao, and Qifeng Chen. "TPCN: Temporal Point Cloud Networks for Motion Forecasting". In: *arXiv:2103.03067* (2021).

[32] Wenyuan Zeng et al. "LaneRCNN: Distributed Representations for Graph-Centric Motion Forecasting". In: *arXiv:2101.06653* (2021).

[33] Lingyao Zhang et al. "Map-Adaptive Goal-Based Trajectory Prediction". In: *CoRL*. 2020.

[34] Hang Zhao et al. "TNT: Target-driven trajectory prediction". In: *CoRL* (2020).

[35] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. "Objects as points". In: *arXiv:1904.07850* (2019).