JACoP: Joint Alignment for Compliant Multi-Agent Prediction

Supplementary Material

Section A details our sampling algorithm for generating scene predictions via prototype alignment. Section B defines the evaluation metrics used in our benchmark experiments. Section C provide an additional benchmark on the average ADE/FDE and KDE NLL metrics, highlighting JACoP's ability to produce predictions that are tightly concentrated around the ground-truth future and better align with agent intentions. Section D experiment on alternative agent-centric profiler baseline, confirming our proposed ACP's superiority in accuracy and collision metrics. Section E provides more qualitative analysis reflecting JACoP's ability to produce accurate, collision-free predictions in complex scenes. Section F provide additional benchmark on SDD dataset. Section G provides analysis on collision rate and number of agent per scene.

A. Sampling Algorithm

Algorithm 1 details our proposed sampling algorithm, which generates the scene prediction by aligning the prototype trajectories selected by our Agent-Centric Profiler (ACP). We employ Gibbs sampling, iteratively sampling a trajectory for each agent conditioned on the trajectories of all other agents from the previous sampling step. The process is initialized by sampling from the marginal distribution inferred for each agent by the ACP module. We perform a total of $B + K$ iterations across all agents and retain the samples from the final K steps as our set of scene predictions. This sampling procedure ensures that the samples converge to the distribution defined by the Markov Random Field (MRF) and the social filtering process, which is essential for achieving realistic predictions with no social collisions and high joint accuracy.

The sampling algorithm requires a time complexity of $O(N(B + K))$, where the overall complexity grows linearly with the number of agents (N). We can further parallelize the sampling process by initializing K independent processes, which allows the time complexity to be reduced to $O(BN)$.

B. Evaluation Metrics

Minimum ADE and FDE The commonly used metric $min_k ADE/FDE$ evaluates the average and final displacement error of the most accurate sample among the multi-modal prediction outputs. The ground truth future is denoted as $Y = \{Y_1, \cdots, Y_n\}$, representing a scene comprising multiple agents. Each agent's trajectory is further detailed as $Y_i = \{y_{i,1}, \cdots, y_{i,T_f}\}$. The K prediction samples are represented as $\hat{Y}^{(k)} = \{\hat{Y}_1^{(k)}, \cdots, \hat{Y}_n^{(k)}\}$, with each agent's sample expressed as $\hat{Y}_i^{(k)} = \{\hat{y}_{i,1}^{(k)}, \cdots, \hat{y}_{i,T_f}^{(k)}\}$, for each $k = 1, ..., K$. The metrics are defined as

$$min_k ADE(Y, \hat{Y}) = \frac{1}{TN} \sum_{i=1}^{N} \min_{k=1}^{K} \sum_{t=1}^{T} ||y_{t,i} - \hat{y}_{t,i}^{(k)}||_2^2,$$

$$min_k FDE(Y, \hat{Y}) = \frac{1}{N} \sum_{i=1}^{N} \min_{k=1}^{K} ||y_{T_f,i} - \hat{y}_{T_f,i}^{(k)}||_2^2. \quad (1)$$

JADE/JFDE The Joint Accuracy metrics, referred to as JADE/JFDE, were initially introduced in [6] with the objective of enhancing the widely used marginal $min_k ADE/FDE$ metrics. These joint metrics are designed to more accurately reflect a model's capability to predict the collective future trajectories of all agents present within a given scene. Unlike the conventional approach, which involves selecting each agent's most accurate prediction samples independently, JADE/JFDE calculates the average displacement error across all agents within a single prediction sample. The JADE and JFDE are defined as

$$JADE(Y, \hat{Y}) = \frac{1}{TN} \min_{k=1}^{K} \sum_{i=1}^{N} \sum_{t=1}^{T} ||y_{t,i} - \hat{y}_{t,i}^{(k)}||_2^2,$$

$$JFDE(Y, \hat{Y}) = \frac{1}{N} \min_{k=1}^{K} \sum_{i=1}^{N} ||y_{T_f,i} - \hat{y}_{T_f,i}^{(k)}||_2^2. \quad (2)$$

Agent-to-Agent Collision The agent-to-agent collision rate measures the proportion of predicted trajectories that intersect with another agent's path in the same scene prediction. Two agents are considered to collide if their positions come within $r = 0.2$ meters at any future time step. We first define an indicator function for collision as

$$\mathbb{1}_{col}(\hat{Y}_i^{(k)}, \hat{Y}_j^{(k)}) = 1, \quad \text{if } \exists\, t, \text{such that } ||\hat{y}_{t,i}^{(k)} - \hat{y}_{t,j}^{(k)}||_2^2 < r. \quad (3)$$

Then we compute the agent-to-agent collision rate as

$$\text{A2A}_{CR} = \frac{1}{NK} \sum_{i=1}^{N} \sum_{k=1}^{K} \mathbb{1}_{col}(\hat{Y}_i^{(k)}, \hat{Y}_j^{(k)}), \quad \text{for any } j, i \neq j. \quad (4)$$

Environmental Collision Rate The environmental collision rate evaluates the percentage of predicted trajectories overlap with non-navigable areas or environmental obstacles. We first define an indicator function for environmental

Table 1. Average ADE/FDE and KDE-based NLL Benchmark. The best performance is boldfaced and the 2nd place is marked as blue.

Model	ETH	HOTEL	UNIV	ZARA1	ZARA2	ETH	HOTEL	UNIV	ZARA1	ZARA2
Avg. # Agents	2.6	3.5	25.7	3.7	6.3	2.6	3.5	25.7	3.7	6.3
Model	Average ADE/FDE					KDE NLL				
AgentFormer	1.99/4.39	0.87/2.01	1.01/2.26	0.72/1.63	0.74/1.71	2.70	1.60	1.88	1.52	1.53
GP-Graph	1.21/2.51	0.71/1.53	**0.91/1.31**	0.68/1.50	0.57/1.27	2.31	1.51	1.80	1.52	1.42
EqMotion	1.42/2.99	0.64/1.32	2.30/5.39	0.82/1.83	0.68/1.54	2.23	1.37	**1.78**	1.60	1.47
SingularTrajectory	1.47/2.77	0.89/1.88	1.12/2.36	0.92/1.96	1.02/2.23	**2.04**	1.58	1.93	1.67	1.77
JACoP	**1.03/1.97**	**0.50/1.05**	0.82/1.70	**0.59/1.26**	**0.46/0.99**	2.54	**1.28**	2.07	**1.44**	**1.35**

violations given the prediction $\hat{Y}_i^{(k)}$ and navigability map M,

$$\mathbb{1}_{env}(\hat{Y}_i^{(k)}, M) = 1, \text{if } \exists t \text{ such that } M(\hat{y}_{t,i}) \neq 1. \quad (5)$$

Then we compute the proportion of predictions with environmental violation via

$$\text{ENV}_{CR} = \frac{1}{NK} \sum_{i=1}^{N} \sum_{k=1}^{K} \mathbb{1}_{env}(\hat{Y}_i^{(k)}, M). \quad (6)$$

Average ADE/FDE Both the minimum k Average/Final Displacement Error ($min_k ADE/FDE$) and the Joint Average/Final Displacement Error (JADE/JFDE) primarily evaluate prediction accuracy based on the best marginal or joint sample. However, in a real-world deployment setting, distinguishing the most accurate output from a set of model predictions is impossible. Therefore, it is crucial that the entire set of predicted futures does not deviate excessively from the true trajectory. To better evaluate the model's overall capability in producing realistic future predictions, we propose the Average ADE/FDE as an additional accuracy measure.

$$avgADE(Y, \hat{Y}) = \frac{1}{TNK} \sum_{i=1}^{N} \sum_{k=1}^{K} \sum_{t=1}^{T} ||y_{t,i} - \hat{y}_{t,i}^{(k)}||_2^2,$$

$$avgFDE(Y, \hat{Y}) = \frac{1}{NK} \sum_{i=1}^{N} \sum_{k=1}^{K} ||y_{T_f,i} - \hat{y}_{T_f,i}^{(k)}||_2^2. \quad (7)$$

The key difference between $min_k ADE/FDE$ and $avgADE/FDE$ lies in their aggregation strategy: instead of selecting the best sample (i.e., the minimum error) from the K outputs, $avgADE/FDE$ averages the displacement error across all K model outputs. A smaller value for these average metrics indicates that the model's entire distribution of predictions is tightly concentrated around the ground-truth future.

KDE-based NLL To further examine the likelihood of each method producing the ground-truth (GT) future among all their prediction outputs, we benchmark the models using the Kernel Density Estimation (KDE)-based Negative Log-Likelihood (NLL) metric, as proposed in [3, 5]. The KDE-NLL metric quantifies the probability density of the GT future trajectory under a distribution inferred from the K predicted samples produced by the models. A smaller NLL value indicates a tighter concentration of the model's predicted distribution around the GT future, reflecting a more accurate and confident overall prediction set. Here we use Silverman's rule of thumb to determine the optimal bandwidth for the KDE estimation.

C. Average ADE/FDE and KDE NLL Analysis

Previous accuracy metrics primarily emphasize the model's best sample; this design was originally introduced to compensate for the inherent stochastic nature of human trajectory. While this metric effectively reflects the upper bound of a model's predictive capability, its optimization can inadvertently lead to models learning shortcuts. Specifically, a model may produce highly diverse or scattered predictions hoping that at least one sample happens to align closely with the ground-truth future. Our previous benchmark results have shown that state-of-the-art (SOTA) models, despite their good accuracy upper bound, frequently produce highly unlikely predictions that violate social and environmental constraints. Consequently, these best-case accuracy measures become less convincing as a true indicator of model quality.

To address this limitation, we introduce two additional benchmark experiments to provide a more complete assessment of the overall model capability for producing accurate predictions. Specifically, we measure the Average ADE/FDE and KDE-NLL of the model outputs, as presented in Table 6. Our JACoP model achieves a notable improvement in Average ADE/FDE performance compared with state-of-the-art (SOTA) methods. This suggests that the model's output distribution, on average, more closely aligns with the ground-truth (GT) agent intentions.

To further support this finding, we evaluate the KDE-NLL metric, which measures the likelihood of the GT future trajectory under the sampling distribution of the model

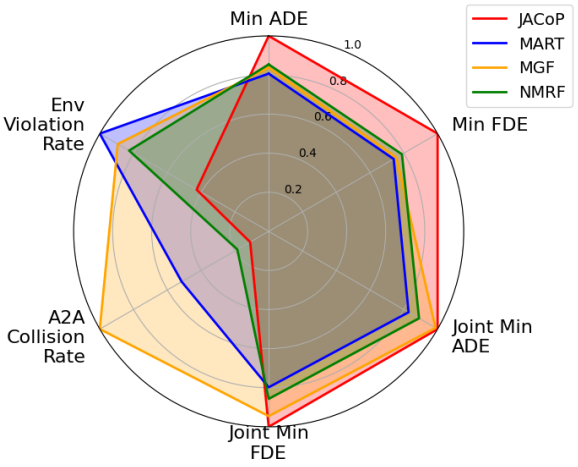


Figure 1. Radar Chart for normalized SDD evaluation result

Table 2. Accuracy and Collision Benchmark on SDD dataset. $min_k ADE$ and JADE measured in pixel space. A2A collision threshold equals 5 pixels (roughly 0.2m). Best result in **bold face** and second best marked by blue color

Model	$min_k ADE / min_k ADE$	JADE/JFDE	A2A Collision	A2E Collision
MART	**7.42/11.81**	**9.48/16.77**	0.010	0.055
MGF	7.71/12.07	11.30/19.88	0.019	0.058
NMRF	7.86/12.59	10.19/18.02	0.004	0.066
Ours	9.20/15.97	11.45/21.02	**0.002**	**0.028**

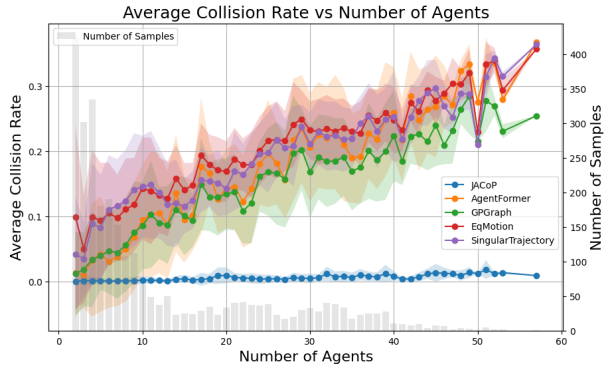


Figure 2. A2A collision rate versus number of agents on the ETH-UCY dataset, showing our JACoP model is clearly better at avoiding collisions, especially in crowded scenes.

outputs. Our model demonstrates strong performance relative to SOTA methods, particularly on the Hotel, Zara1, and Zara2 splits, where JACoP sets a new performance benchmark.

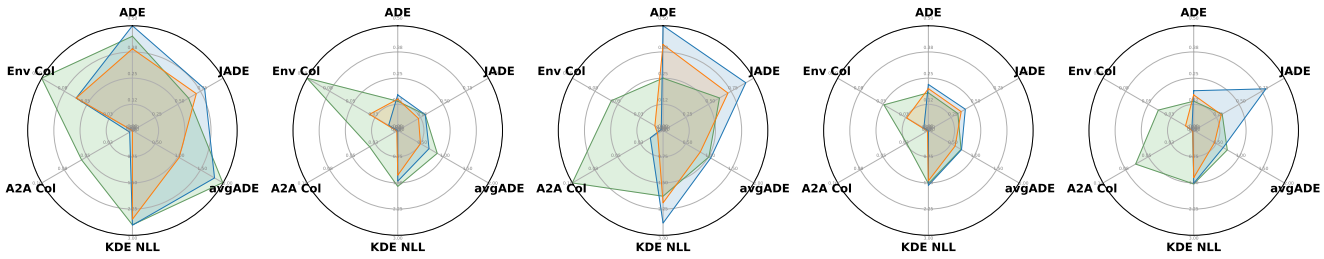
Taken together, these results—combined with JACoP's ability to generate trajectories with minimal scene violations—support the conclusion that our JACoP model functions as a reliable trajectory predictor capable of generating multiple future possibilities that better reconcile both GT agent intentions and environmental and social contexts.

D. Alternative Profiler Choice – AgentFormer

Our ablation study in Section 4.3 shows that the proposed environmental and social collision filter and the Gibbs sampling module are vital for generating collision-free scene-level predictions. Given this result, we recognize that both the filtering technique and our Joint Prediction Aligner module (Figure 2, right) are generally applicable to other Profiler modules, provided the module is probabilistic or offers a scoring mechanism for trajectory proposals. However, we argue that our light-weight anchor-based Profiler(ACP), which performs matching in the future trajectory's feature space, offers a sufficient and efficient solution. To further verify this hypothesis on the sufficiency of our approach, we experimented with an alternative Profiler choice, using the predictions and probability estimates from AgentFormer [7] as inputs to our Joint Prediction Aligner. We benchmark this alternative baseline against the original AgentFormer and our proposed JACoP method.

Table 3, Table 4 and Table 5 show the performance on the accuracy and feasibility benchmarks, while Figure 3 summarizes the numerical results on the radar charts for better head-to-head comparison. The results concerning environmental and A2A collisions demonstrate that the combination of AgentFormer with our proposed aligner and filtering technique (shown in blue polygon), significantly reduces the collision rate compared to the original AgentFormer (shown in green polygon). This robustly indicates the utility and generalizability of the JACoP Aligner module.

However, the alternative AgentFormer Profiler fails to outperform our anchor-based Agent-Centric Profiler (ACP) (shown in orange) across the three main accuracy metrics, KDE NLL and the A2A collision rate. Our utilization of environmental and social contexts as part of the query embeddings helps the model select more accurate and suitable prototypes for subsequent alignment and scene prediction generation. While the AgentFormer output shows good performance in the best-case scenario (indicated by the $min_k ADE/FDE$ metrics), its inherent lack of environmental awareness renders a majority of its predictions unusable for alignment after the environmental filtering process. The remaining filtered predictions, despite achieving good environmental compliance, lack the necessary accuracy for the alignment process to produce a high-quality final output, which also leads to more unavoidable social collisions. The superior performance of JACoP in the average ADE/FDE and KDE NLL further demonstrate that our proposed ACP module helps the model achieve good average accuracy across all selected samples and generate trajectories that have a tight concentration around the GT future.



(a) ETH (b) Hotel (c) Univ. (d) Zara1 (e) Zara2

Figure 3. Radar plot for all metrics among the five ETH-UCY dataset splits comparing between AgentFormer+JACoP Aligner and JACoP. Green: AgentFormer; Blue: AgentFormer Profiler + JACoP Aligner; Orange: JACoP. A smaller overall area of the polygon aligns with better performance across all metrics.

Table 3. Collision Rate of our JACoP vs. AgentFormer + JACoP Aligner. The best performance is boldfaced and the 2nd place is marked as blue.

Model	ETH	HOTEL	UNIV	ZARA1	ZARA2	ETH	HOTEL	UNIV	ZARA1	ZARA2
Avg. # Agents	2.6	3.5	25.7	3.7	6.3	2.6	3.5	25.7	3.7	6.3
Model	Environmental Collision					A2A Collision				
AgentFormer	0.32	0.147	0.057	0.049	0.039	0.056	0.031	0.205	0.024	0.064
AgentFormer + JACoP Aligner	**0.008**	**0.010**	**0.000**	**0.005**	**0.002**	0.003	0.002	0.014	0.00	0.002
JACoP	0.062	0.031	0.009	0.024	0.009	**0.00**	**0.001**	**0.006**	**0.001**	**0.001**

E. Additional Qualitative Evaluations

We offer additional visualizations for qualitative analysis. Figure 4 presents examples of densely packed scenes, where our JACoP model accurately predicts without social or environmental collisions. For instance, row 2 depicts a crowd navigating dense areas. Compared to other SOTA models, JACoP avoids agent collisions. It also provides accurate scene predictions, evident in the JADE performance for rows 1, 3, and 4.

Figure 5 illustrates individual predictions that demand a deep understanding of environmental constraints. Row 1 features an ETH split example where the model predicts a slowdown as the agent nears the scene's edge. Our model, incorporating environmental context into the query embedding and filtering, generates trajectories that adhere to navigable areas and explores alternatives while respecting constraints. The SingularTrajectory model often fails due to post-prediction corrections, leading to infeasible predictions. Other models, ignoring environmental contexts, tend to overshoot or overlap non-navigable areas. In two other instances, our JACoP models also achieve environmentally compliant predictions, especially in row 2, where our predictions align closely with the ground truth future.

F. Additional Benchmark – SDD

We added an SDD experiment comparing three SOTA models: MART [4], MGF[1], and NMRF[2], summarized in the radar plot in Figure 1. Our model attains the best social and environmental collision rates—our main focus—with only slightly lower accuracy. We will include full numerical results in the final version.

G. Analysis on Collision Rate vs Number of Scene Agent

We further showcase this robustness via the A2A collision rate (Fig. 2) across scenes of varying density, where our method decisively outperforms prior SOTA in generating feasible, collision-free predictions.

Table 4. Prediction Accuracy Comparison between our JACoP vs. AgentFormer + JACoP Aligner. The best performance is boldfaced and the 2nd place is marked as blue.

Model Avg. # Agents	ETH 2.6	HOTEL 3.5	UNIV 25.7	ZARA1 3.7	ZARA2 6.3	ETH 2.6	HOTEL 3.5	UNIV 25.7	ZARA1 3.7	ZARA2 6.3
Model	$min_k ADE/FDE$					JADE/JFDE				
AgentFormer	**0.45/0.79**	**0.14/0.22**	**0.25/0.45**	**0.18/0.30**	**0.14/0.23**	**0.619/1.136**	0.303/0.603	**0.622/1.311**	**0.325/0.660**	0.314/0.663
AgentFormer + JACoP Aligner	0.52/0.94	0.17/0.29	0.62/1.29	0.22/0.43	0.19/0.37	0.796/1.572	0.303/0.614	0.912/1.942	0.407/0.869	0.372/0.807
JACoP	0.59/1.06	0.15/0.25	0.41/0.82	0.20/0.39	0.17/0.32	0.704/1.226	**0.229/0.420**	0.715/1.472	0.361/0.724	**0.304/0.623**

Table 5. Average ADE/FDE and KDE-based NLL Benchmark Comparison between our JACoP vs. AgentFormer + JACoP Aligner. The best performance is boldfaced and the 2nd place is marked as blue.

Model Avg. # Agents	ETH 2.6	HOTEL 3.5	UNIV 25.7	ZARA1 3.7	ZARA2 6.3	ETH 2.6	HOTEL 3.5	UNIV 25.7	ZARA1 3.7	ZARA2 6.3
Model	Average ADE/FDE					KDE NLL				
AgentFormer	1.99/4.39	0.87/2.01	1.01/2.26	0.72/1.63	0.74/1.71	2.70	1.60	**1.88**	1.52	1.53
AgentFormer + JACoP Aligner	1.81/3.92	0.69/1.56	1.05/2.26	0.73/1.65	0.65/1.48	2.71	1.45	2.65	1.57	1.51
JACoP	**1.03/1.97**	**0.50/1.05**	**0.82/1.70**	**0.59/1.26**	**0.46/0.99**	**2.54**	**1.28**	2.07	**1.44**	**1.35**

References

[1] Jiahe Chen, Jinkun Cao, Dahua Lin, Kris Kitani, and Jiangmiao Pang. Mgf: Mixed gaussian flow for diverse trajectory prediction. *Advances in Neural Information Processing Systems*, 37:57539–57563, 2024. 4

[2] Zilin Fang, David Hsu, Gim Hee Lee, and Gim Hee Lee. Neuralized markov random field for interaction-aware stochastic human trajectory prediction. In *ICLR*, 2025. 4

[3] Boris Ivanovic and Marco Pavone. The trajectron: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2375–2384, 2019. 2

[4] Seongju Lee, Junseok Lee, Yeonguk Yu, Taeri Kim, and Kyoobin Lee. Mart: Multiscale relational transformer networks for multi-agent trajectory prediction. In *European Conference on Computer Vision*, pages 89–107. Springer, 2024. 4

[5] Luca Anthony Thiede and Pratik Prabhanjan Brahma. Analyzing the variety loss in the context of probabilistic trajectory prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9954–9963, 2019. 2

[6] Erica Weng, Hana Hoshino, Deva Ramanan, and Kris Kitani. Joint metrics matter: A better standard for trajectory forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20315–20326, 2023. 1

[7] Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris M Kitani. Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9813–9823, 2021. 3

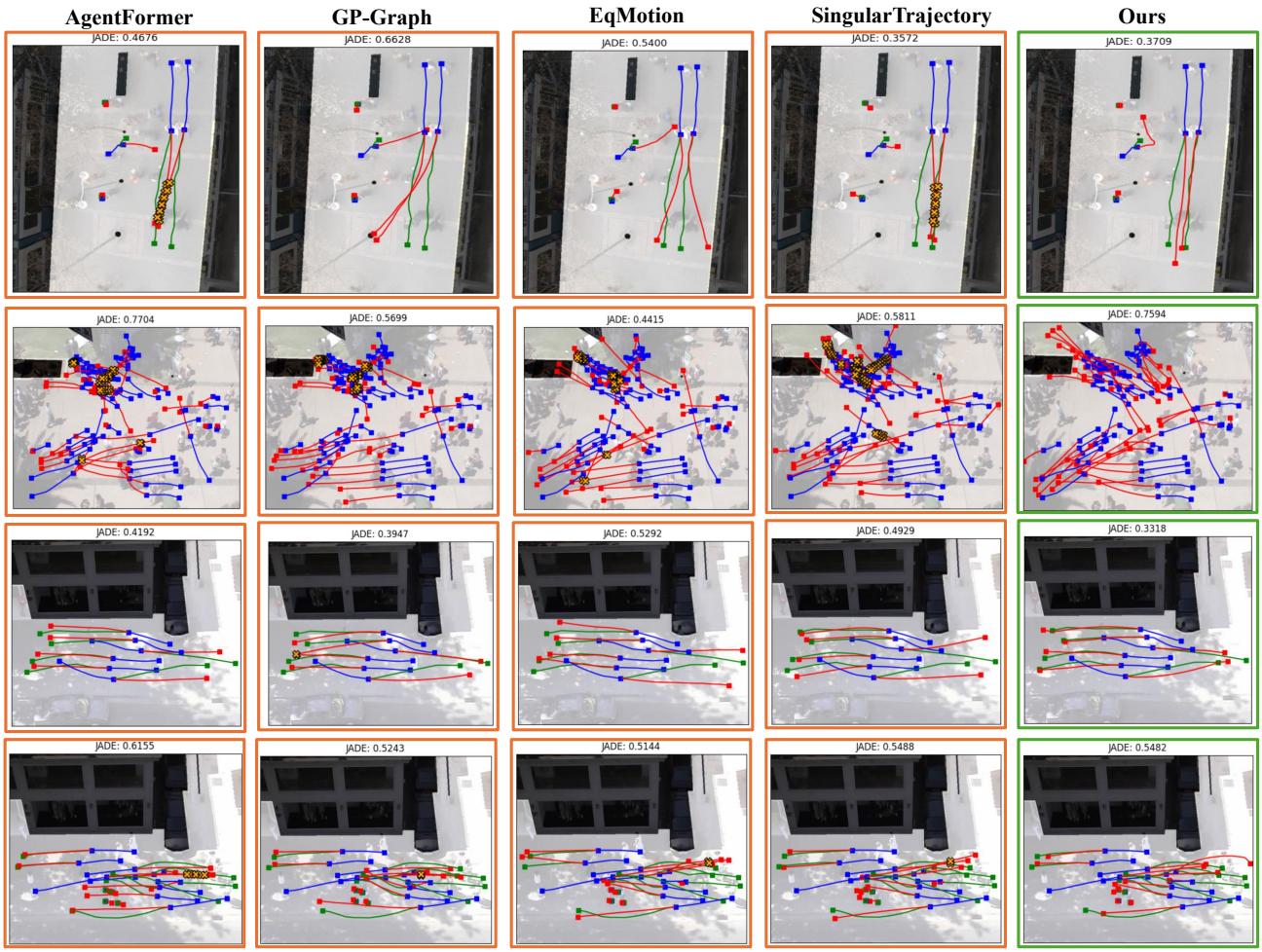


Figure 4. Scene prediction with best JADE performance from Hotel (Row 1), Univ (Row 2), Zara1 (Row 3) and Zara2 (Row 4) Blue line:Historical Trajectory, Green line: Ground Truth Future Trajectory, Red line: Predictions.

Algorithm 1 Gibbs Sampling for Scene-Level Trajectory Prediction

Require: Number of burn-in steps B, number of samples K, and a set of agents $\mathcal{I}$.
Ensure: A set of scene predictions $\mathcal{Y}$.

$Y^{(0)} \sim P_{\text{unary}}$ ▷ Initialize from marginal samples
$\mathcal{Y} :=$ Empty List
for $\tau = 1$ to $B + K$ **do**
  for each agent $i \in \mathcal{I}$ **do**
    $Y_i^{(\tau)} \sim P(Y_i | Y_{\setminus i}^{(\tau-1)})$ ▷ Sample agent i's trajectory conditioned on all others
  end for
  if $\tau > B$ **then**
    Append $Y^{(\tau)}$ to $\mathcal{Y}$ ▷ Save the scene prediction after burn-in
  end if
end for
return $\mathcal{Y}$

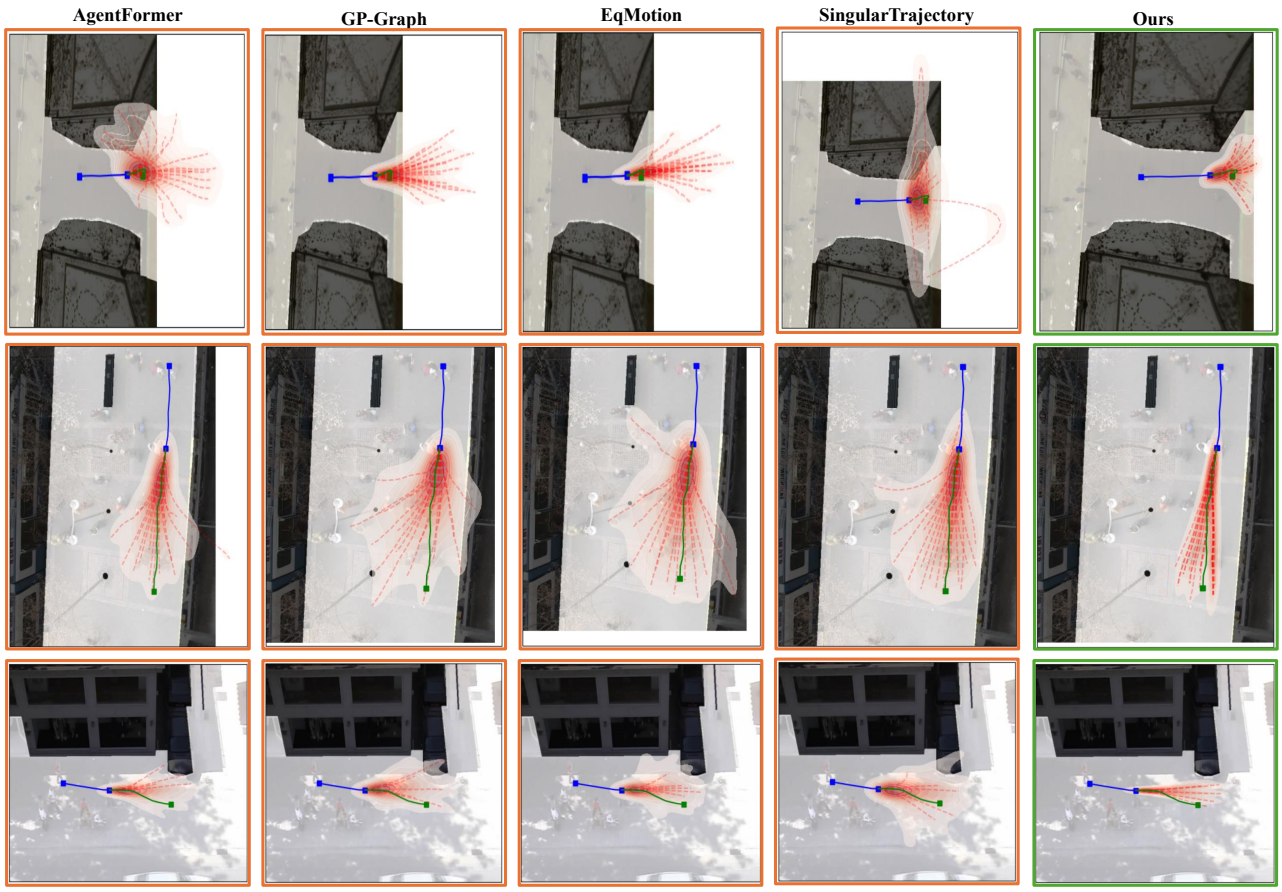


Figure 5. Individual prediction from ETH (Row 1), Hotel (Row 2) and Zara1 (Row 3)
Blue line:Historical Trajectory, Green line: Ground Truth Future Trajectory, Red Dashed line: Predictions, Red Shade: Sample Distribution.

Table 6. Average ADE/FDE and KDE-based NLL Benchmark. The best performance is boldfaced and the 2nd place is marked as blue.

Model Avg. # Agents	ETH 2.6	HOTEL 3.5	UNIV 25.7	ZARA1 3.7	ZARA2 6.3	ETH 2.6	HOTEL 3.5	UNIV 25.7	ZARA1 3.7	ZARA2 6.3
Model	Average ADE/FDE					KDE NLL				
AgentFormer	1.99/4.39	0.87/2.01	1.01/2.26	0.72/1.63	0.74/1.71	2.71	1.35	1.72	1.30	1.29
AgentFormer + JACoP Aligner	1.81/3.92	0.69/1.56	1.05/2.26	0.73/1.65	0.65/1.48	2.81	1.19	2.93	1.38	1.29
GP-Graph	1.21/2.51	0.71/1.53	**0.91/1.31**	0.68/1.50	0.57/1.27	2.30	1.28	1.65	1.32	1.19
EqMotion	1.42/2.99	0.64/1.32	2.30/5.39	0.82/1.83	0.68/1.54	2.15	1.10	**1.60**	1.40	1.22
SingularTrajectory	1.47/2.77	0.89/1.88	1.12/2.36	0.92/1.96	1.02/2.23	**1.91**	1.34	1.77	1.46	1.57
JACoP	**1.03/1.97**	**0.50/1.05**	0.82/1.70	**0.59/1.26**	**0.46/0.99**	2.76	**1.01**	2.10	**1.24**	**1.12**