VisionTrap: Vision-Augmented Trajectory Prediction Guided by Textual Descriptionss Supplemental Material

Seokha Moon[1], Hyun Woo[1], Hongbeen Park[1], Haeji Jung[1], Reza Mahjourian[2], Hyung-gun Chi[3], Hyerin Lim[4], Sangpil Kim[1], and Jinkyu Kim[1]

[1] Korea University, Seoul 02841, Republic of Korea
[2] The University of Texas at Austin, Texas 78712, USA
[3] Perdue University, West Lafayette 95008, USA
[4] Hyundai Motor Company, Seongnam 13529, Republic of Korea

1 Details for Evaluation and Implementation

Dataset. Our proposed approach is developed and evaluated utilizing the widely employed nuScenes [2] dataset, which encompasses 1000 diverse scenes from Boston and Singapore. Annotations cover 10 classes for object detection, including car, truck, bus, trailer, construction vehicle, pedestrian, motorcycle, bicycle, barrier, and traffic cone. It also provides ego-centric surround-view images and HD map. In nuScenes, the model is trained with a 2-second history to predict a 6-second future trajectory. Unlike existing works [3, 4, 6, 10] that report about single-agent prediction performance, our research takes a different approach. Instead of utilizing only the dataset provided for the prediction task, we used the entire nuScenes dataset for training to conduct a multi-agent prediction approach that considers all agents in a scene simultaneously. Therefore, our nuScenes-Text dataset used for this study is created to cover all scenes in the nuScenes dataset. The Vision Language Model BLIP-2 [5] (VLM) used to generate this text is trained on the DRAMA [9] dataset, which provides an image of the driving environment, bounding box pointing to specific agent, and text representing this agent. To accurately use textual descriptions obtained from fine-tuned VLM, we refine the descriptions using GPT [1]. We also present metrics for all agents and metrics specifically for agents involved in the prediction task, offering a comprehensive evaluation.

Evaluation Metrics. Our model is evaluated using standard metrics for trajectory prediction, including minimum Average Displacement Error (ADE), minimum Final Displacement Error (FDE), and Miss Rate (MR). These metrics quantify the average and final displacement errors between the true trajectory and the best prediction sample. MR further denotes the percentage of scenarios where the distance between the endpoint of the true trajectory and the best prediction exceeds a 2m threshold.

* Corresponding author: J. Kim (jinkyukim@korea.ac.kr)

$$ADE = \frac{1}{T} \sum_{t=T_{curr}+1}^{T_{Fin}} \left\| \hat{Y}_{(k)}^{t} - Y^{t} \right\|_2 \tag{1}$$

$$FDE = \left\| \hat{Y}_{(k)}^{T_{Fin}} - Y^{T_{Fin}} \right\|_2 \tag{2}$$

Here, $\hat{Y}_{(k)}^{t}$ denotes the predicted position of the agent at timestep t in the (k)-th mode, and Y^{t} represents the ground truth position at timestep t. The (k) represents the mode with the smallest error when compared to the ground truth, while T indicates the number of timesteps to be predicted. Additionally, T_{Fin} represents the timestep at which the prediction concludes, while T_{curr} indicates the current timestep.

Implementation Details. We train the model for 48 epochs using AdamW optimizer [8] and four RTX 3090 Ti GPUs. The model has 32 batch sizes, 5×10^{-4} initial learning rates, 1×10^{-4} weight decay, and 0.1 dropout rates. To manage the learning rate, we adopt the cosine annealing scheduler [7]. For consistency, we set the number of offsets for deformable attention in the Scene-Agent Interaction Module, denoted as O, to 4. Additionally, augmentation techniques, including rotation within (-22.5, 22.5) degrees and excluding a random agents (10% of all agents in scene) from the loss calculation, are used to prevent overfitting and increase the generalization performance of the model.

2 More Detail for nuScenes-Text Dataset

Prompt Engineering for LLM. We utilize the Large Language Model (LLM) GPT to refine textual descriptions obtained from VLM regarding issues stemming from the domain gap between datasets or completely missing parts, as well as inaccurate location information caused by the characteristics of surround view images (see Fig. 2). To enhance the quality of pseudo-text, we meticulously design prompt for LLM such as Fig. 1. The primary challenges in this improvement process involved i) removing inaccurate location information such as 'left,' 'right lane,' or 'ego lane,' caused by the characteristics of surround view images (see Fig. 2) and ii) refining parts that are incorrectly predicted or completely missing due to domain gaps between datasets. Given that the nuScene dataset includes not only a front view but also a surround view, including back view, i) is crucial to avoid confusion in the model caused by these location details. Additionally, for ii), we explicitly integrate task details such as maneuvering and agent types to eliminate hallucinations and generate clear information. Finally, we include examples of both effective ('good') and ineffective ('bad') outputs to optimize the capabilities of the LLM. **Maneuvering extraction Algorithm.** To integrate information about the intention of each agent into our generated text dataset, we utilize the maneuvering attribute. We classify the maneuvering of the agent based on the actual future trajectory. Maneuvering is defined by comparing the

Your Role: You are a writer tasked with generating descriptive captions about objects without including their location information.

Inputs Explained:
1. Caption: Describes an object but might contain location info which we don't want.
2. GT class: The actual type of the object.
3. Maneuvering: Predicted future movement of the object.

Your Task:
- DO NOT include location information like 'left side', 'right lane', 'away from the ego car', 'in the ego lane' etc.
- If the object described in the Caption is different from the GT class, craft your caption using only the GT class and Maneuver.
- If the action described in the Caption does not align with the Maneuver, adjust the description to fit the provided Maneuver.
- Explicitly mention the object's expected maneuver using the provided "Maneuver" input.

Example Input 1:
Caption: a white suv driving in the left lane, away from the intersection, in the rain
GT class: vehicle.car
Maneuvering: straight

Bad Output: A white SUV is driving in the left lane, away from the intersection, in the rain, and is expected to continue straight.
Good Output: A white SUV is driving in the rain, and is expected to continue straight.

Example Input 2:
Caption: a white suv parked on the left side of the road
GT class: vehicle.car
Maneuvering: stationary

Bad Output: A white SUV is parked on the left side of the road.
Good Output: A white SUV is parked, and is anticipated remain stationary.

Now, based on the above guidelines and examples, create a caption for the following:

Fig. 1: Example of prompt given to the LLM, specifically designed to generate accurate descriptions. Inputs for this prompt, highlighted in red for emphasis, include the caption obtained from VLM, the object's class (GT class), and maneuvering information.

initial position and orientation with the final position and orientation. The generated maneuvering information is provided to the LLM to offer insights into the agent's intention. Therefore, the refined text, including information on the agent's characteristic points, current movement, and future intention, may be utilized, thereby contributing to enhancing the performance of the model. The maneuvering extraction algorithm can be observed in Fig. 3.

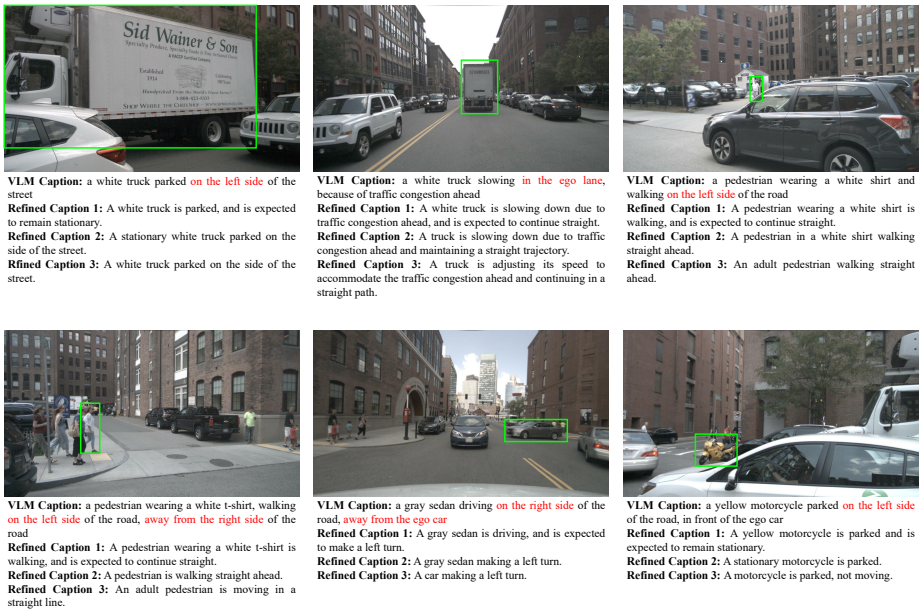

Fig. 2: Example of captions for objects in the nuScenes [2] dataset are provided in ego-centric surround-view images from a single scene. These captions describe each agent within the images, and each agent is accompanied by three versions of text.

```
Algorithm 1: Maneuver Classification Algorithm
Data: traj, traj_mask, category
Result: Classify agent's maneuver as "stationary", "lane change", "straight", "left turn", "right turn", "left U-turn", or "right U-turn"

begin
    // Initialize thresholds without specific values
    Initialize thresholds: displacement, lateral_displacement, and heading thresholds for both vehicle and pedestrian

    // Rotate future path to align with the positive direction of the x-axis based on current heading
    Rotate agent_positions based on origin_heading_vector

    // Calculate heading_delta and final_displacement
    Calculate heading_delta as the change in heading from the start to the last valid index in the trajectory
    Calculate final_displacement as the Euclidean norm of the position difference (xy_delta) from the start to the last valid index

    // Define xy_delta as the displacement vector from the starting to the final valid position in the rotated agent_positions
    Define xy_delta

    if category is a vehicle then
        if final_displacement < vehicle_displacement_for_stationary then
            return "stationary"
        end if
        if abs(xy_delta[1]) < vehicle_lateral_displacement_for_straight and abs(heading_delta) < heading_delta_for_straight then
            if abs(xy_delta[1]) > lane_change_threshold then
                return "lane change"
            else
                return "straight"
            end if
        end if
        if heading_delta < -heading_delta_for_straight then
            if xy_delta[0] < vehicle_longitudinal_displacement_for_uturn then
                return "right U-turn"
            else
                return "right turn"
            end if
        end if
        if xy_delta[0] < vehicle_longitudinal_displacement_for_uturn then
            return "left U-turn"
        else
            return "left turn"
        end if
    else if category is a pedestrian then
        if final_displacement < pedestrian_displacement_for_stationary then
            return "stationary"
        end if
        if abs(xy_delta[1]) < pedestrian_lateral_displacement_for_straight and abs(heading_delta) < heading_delta_for_straight then
            return "straight"
        end if
        if heading_delta < -heading_delta_for_straight then
            if xy_delta[0] < pedestrian_logitudinal_displacement_for_uturn then
                return "right U-turn"
            else
                return "right turn"
            end if
        end if
        if xy_delta[0] < pedestrian_logitudinal_displacement_for_uturn then
            return "left U-turn"
        else
            return "left turn"
        end if
    end if
end
```

Fig. 3: Maneuver Classification Algorithm. This algorithm shows the process for classifying the maneuvers of agents based on their future path and heading vector.

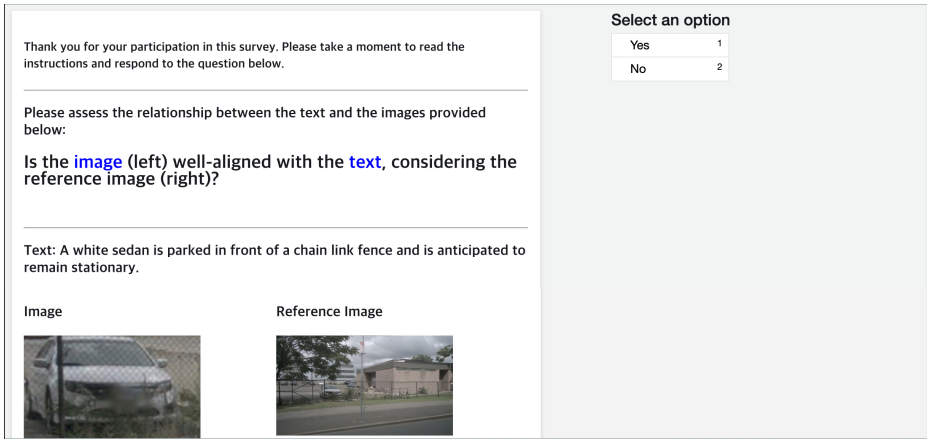

Fig. 4: Example of the Mechanical Turk evaluation interface used for assessing the alignment between generated text descriptions and corresponding images in the nuscenes-text dataset.

More Details about Dataset Statistics. We further explore the details of the dataset we have created. The dataset contains 15,369,058 words, leading to a total of 17,134,981 tokens. This significant amount of text reflects the dataset's comprehensive scope, encompassing a variety of subjects and scenarios relevant to autonomous vehicles. With an average of 13.08 words and 14.58 tokens per text, the dataset showcases a wide-ranging vocabulary and linguistic diversity. Additionally, An example of the Mturk evaluation interface we used can be seen in Fig. 4. The results from the human evaluation conducted via Mechanical Turk further demonstrate how well the captions included in our dataset describe the corresponding objects, indicating their substantial validity.

More Details about nuScenes-Text Dataset. In this section, we provide additional examples of our created nuScenes-Text dataset. Fig. 2 represents textual descriptions obtained from surround-view images. Each agent has three distinct versions of textual descriptions and shows this descriptions of each agent in the bounding box. The description generated through VLM in the top center image (CAM_FRONT) includes location information based on the perspective of the ego vehicle (highlighted in red). However, this may differ from the perspective of other vehicles and pedestrians. Additionally, the location data highlighted in red in the top right image (CAM_FRONT_RIGHT) indicates the position of a person located on the left side of the image, but from the perspective of the autonomous vehicle, it may inaccurately depict the location (from the perspective of the autonomous vehicle, the person is positioned to the right). Such inaccuracies in image-based location data have the potential to compromise the trajectory prediction functionality of the model. This issue is addressed by removing incorrect information through LLM, and the improvements are clearly evident in the refined captions. Through this, we demonstrate the capability to

Fig. 5: Textual descriptions of unique scenarios in out dataset.

generate accurate textual descriptions for all objects visible in surround-view images.

Fig. 5 provides additional examples of unique situations that can be captured by camera images. Surprisingly, the textual description describes scenarios of rainy conditions and can also describe situations where camera data is compromised, such as low-light conditions. In addition, the text description shows that it can also capture situation information and details well, such as pedestrians holding umbrellas, unloading from trucks, people riding cycle, a driver getting out of a vehicle and pedestrians sitting on concrete blocks. Please refer to the images and captions together.

Table 1: Results for various types: **A** uses only observed trajectory data, **B** without and **C** with our Visual Semantic Encoder and Text-driven Guidance Module. The data is used from the nuScenes [2] whole set.

Model	Type: All			Type: Vehicles			Type: Pedestrians		
	$ADE_{10}\downarrow$	$FDE_{10}\downarrow$	$MR_{10}\downarrow$	$ADE_{10}\downarrow$	$FDE_{10}\downarrow$	$MR_{10}\downarrow$	$ADE_{10}\downarrow$	$FDE_{10}\downarrow$	$MR_{10}\downarrow$
A	0.425	0.641	0.081	0.453	0.702	0.102	0.341	0.471	0.019
B	0.407	0.601	0.075	0.431	0.649	0.095	0.339	0.463	0.017
C	**0.368**	**0.535**	**0.051**	**0.386**	**0.573**	**0.064**	**0.319**	**0.430**	**0.016**

3 Further Results

Additional Quantitative Results. In Table 1, results for various agent types in the nuScenes whole dataset are presented. VisionTrap conducts predictions for both vehicles and pedestrians, showcasing information for both types. Model A employs only observed trajectories, Model B incorporates map data in addition to trajectory information, and Model C represents the results of VisionTrap. The outcomes demonstrate that the Visual Semantic Encoder and Text-driven Guidance Module contribute to improved performance across all agents.

Additional Qualitative Examples. We present additional qualitative examples obtained from various scenes. The examples are selected from the nuScenes dataset. Results from Fig.6 to Fig.12 illustrate without and with our Visual Semantic Encoder and Text-driven Guidance Module. Refer to the respective captions for explanations about the figures.

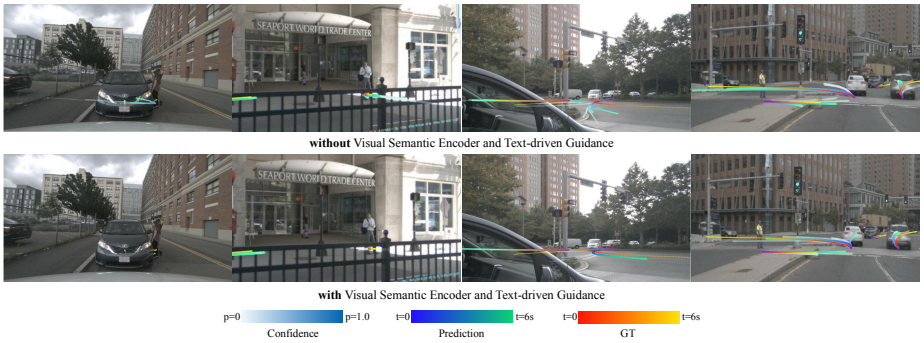

Fig. 6: Examples where visual semantic information is used to improve the performance of trajectory prediction

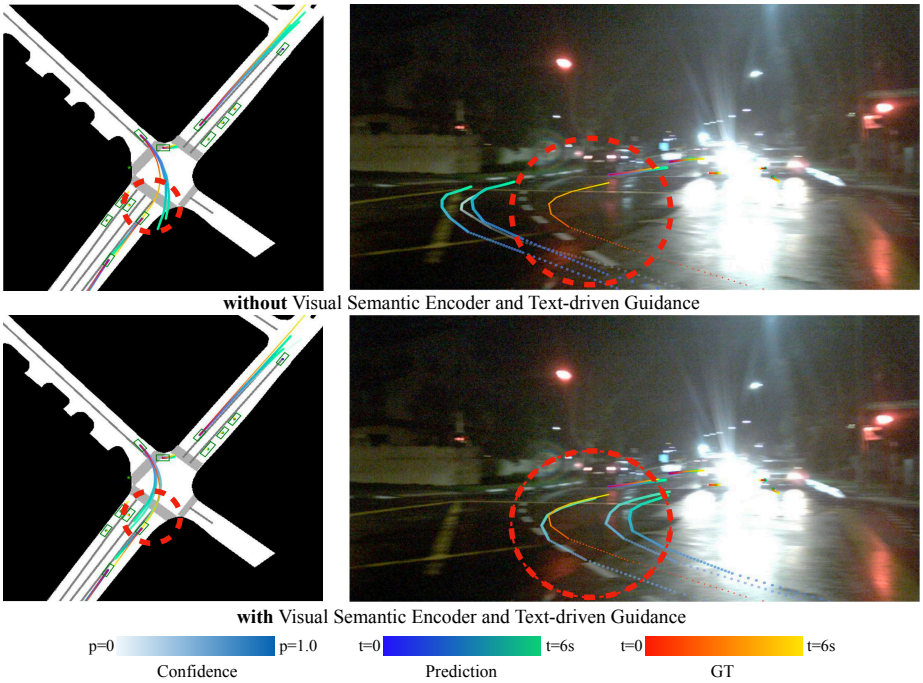

Fig. 7: Align trajectory to lane: Despite the use of nighttime images, it effectively aids in course adjustment when the vehicle makes a right turn.

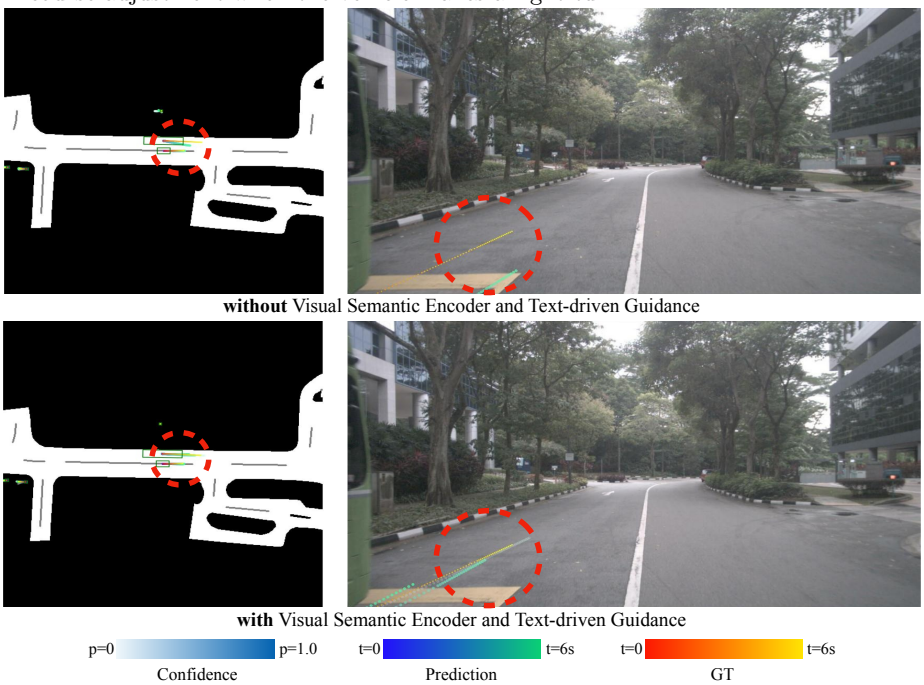

Fig. 8: Align trajectory to lane: The trajectory is adjusted to align with the lane when the parked bus starts moving.

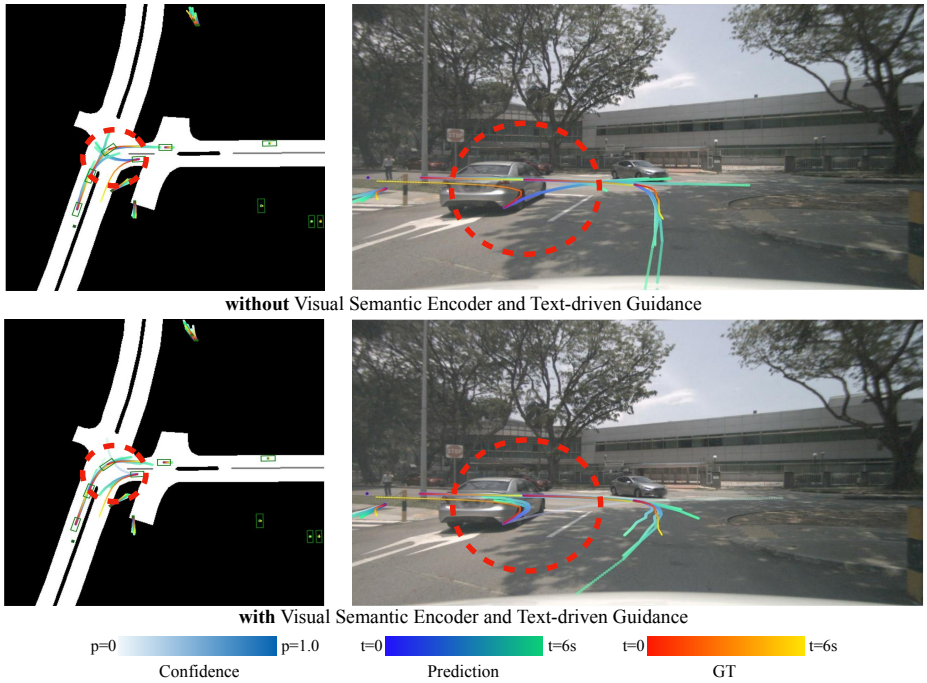

Fig. 9: Prevent collision: Vision data enables an understanding of the detailed situations of agents, enhancing interactions among them based on this understanding.

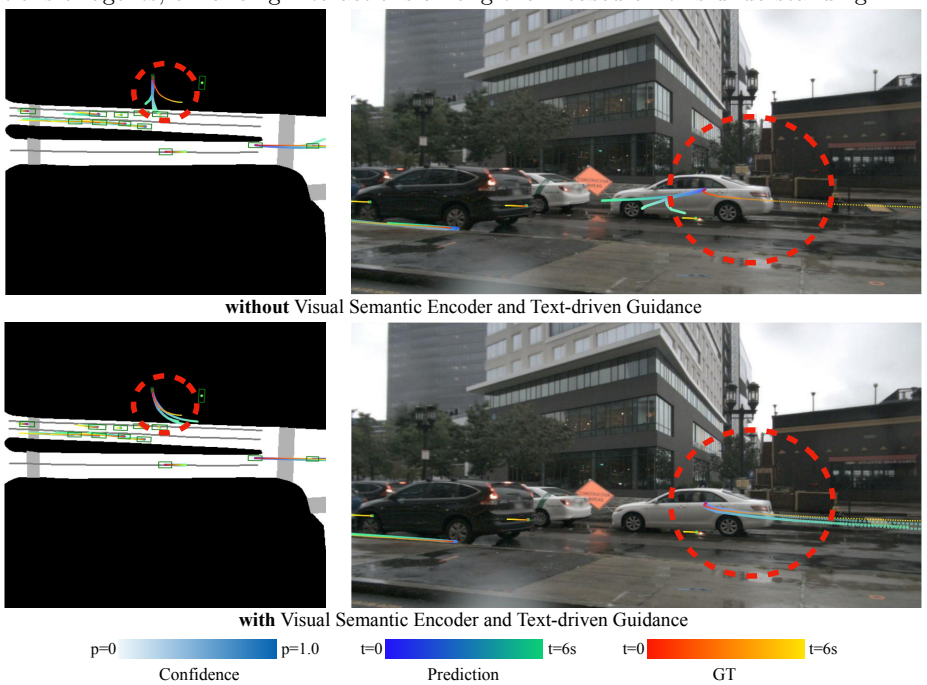

Fig. 10: Prevent collision: The pedestrian's trajectory is adjusted to ensure there is no collision with the car and align with walking on the sidewalk.

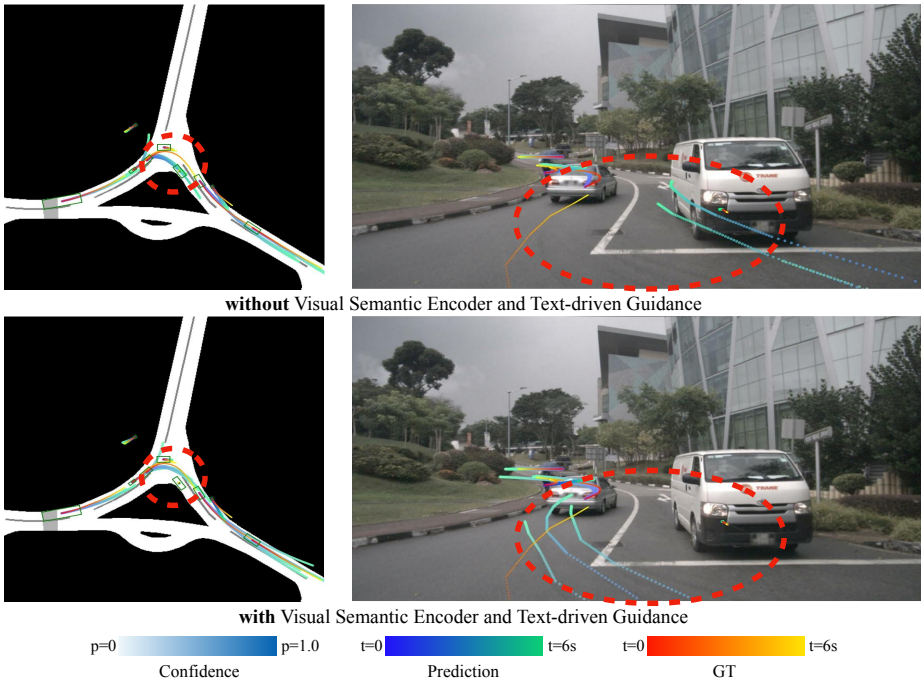

Fig. 11: The direct utilization of vision information: Vision information can determine the direction of the lane and the heading of the agents.

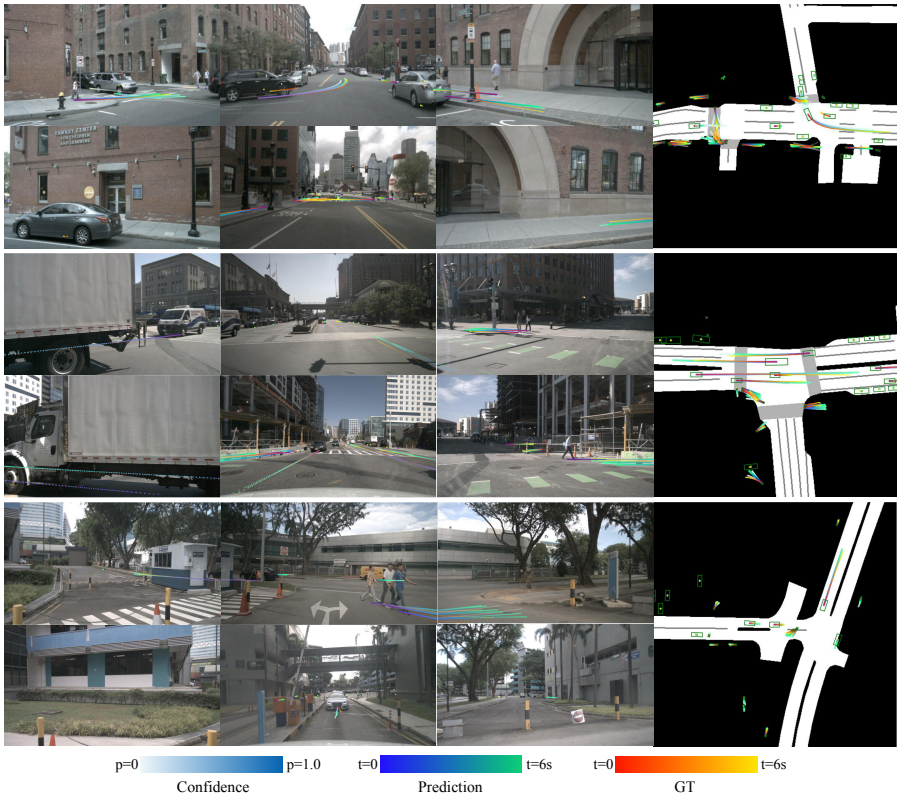

Fig. 12: Visualization results for trajectory prediction by our model for all objects (vehicles, pedestrians) in ego-centric surround view images.

References

1. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. Advances in neural information processing systems **33**, 1877–1901 (2020)
2. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuscenes: A multimodal dataset for autonomous driving. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11621–11631 (2020)
3. Deo, N., Wolff, E., Beijbom, O.: Multimodal trajectory prediction conditioned on lane-graph traversals. In: Conference on Robot Learning. pp. 203–212. PMLR (2022)
4. Gilles, T., Sabatini, S., Tsishkou, D., Stanciulescu, B., Moutarde, F.: Thomas: Trajectory heatmap output with learned multi-agent sampling. arXiv preprint arXiv:2110.06607 (2021)
5. Li, J., Li, D., Savarese, S., Hoi, S.: Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. arXiv preprint arXiv:2301.12597 (2023)
6. Liu, M., Cheng, H., Chen, L., Broszio, H., Li, J., Zhao, R., Sester, M., Yang, M.Y.: Laformer: Trajectory prediction for autonomous driving with lane-aware scene constraints. arXiv preprint arXiv:2302.13933 (2023)
7. Loshchilov, I., Hutter, F.: Sgdr: Stochastic gradient descent with warm restarts. arXiv preprint arXiv:1608.03983 (2016)
8. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
9. Malla, S., Choi, C., Dwivedi, I., Choi, J.H., Li, J.: Drama: Joint risk localization and captioning in driving. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 1043–1052 (2023)
10. Park, D., Ryu, H., Yang, Y., Cho, J., Kim, J., Yoon, K.J.: Leveraging future relationship reasoning for vehicle trajectory prediction. arXiv preprint arXiv:2305.14715 (2023)