WPT: World-to-Policy Transfer via Online World Model Distillation

Supplementary Material

6. More Experiment Details

6.1. World Model Structure

In this section, we provide a detailed description of the world model used in our WPT. As shown in Figs. 4 and 5, we utilize an occupancy-based world model [54] on the nuScenes dataset and an instance-based world model, which is similar to VAD [24] on the Bench2Drive dataset.

Occupancy-based World Model. The structure of the occupancy-based world model is illustrated in Fig. 4. The world decoder predicts the future BEV embedding based on the historical BEV features stored in the memory queue and the expected action conditions using an autoregressive approach. Specifically, the future BEV queries first establish contextual associations through a self-attention mechanism. Then, a temporal-cross attention layer extracts the corresponding features from multiple historical embeddings. Following this, an action cross-attention layer enables interaction between BEV queries and action conditions, injecting the action context into the prediction process. Finally, the feed-forward network generates the predicted BEV features, which are used for future occupancy predictions.

Instance-based World Model As illustrated in Fig. 5, our instance-based world model is based on a modified version of VAD [24]. In this model, multi-view images are first processed by a ResNet-based encoder to extract features. These features are then decoded into BEV space using a BEVFormer-based decoder. To predict the future world evolution, we model both the static road topology (*e.g.*, lane markings, lane centerlines, sidewalks, and other road structures) and dynamic agent motion. The future state is predicted by decoding the corresponding queries for both static elements and dynamic agents.

6.2. WPT-baseline Models

Occupancy-based Baseline. The architecture of the occupancy-based baseline planner is shown at the top of Fig. 6. The model extracts BEV features through the image backbone and BEV Encoder [32] from multi-view input images, which are then processed by a BEV decoder [32]. The initialized planning query interacts with these extracted BEV features, producing a refined plan query that predicts future trajectories.

Instance-based Baseline. The architecture of our instance-based baseline planner is shown in the bottom part of Fig. 6. Here, ego queries interact with map and agent queries via cross-attention, refining the ego query. These refined queries are then decoded by the plan head to generate future trajectory predictions.

6.3. Simulation Reward Design

In this section, we provide a detailed design of the Simulation Reward mechanism, which includes five distinct reward components: no collision (NC), drivable area compliance (DAC), ego progress (EP), time-to-collision (TTC), and comfort (Comf). These rewards assess the safety, efficiency, and comfort of the generated trajectories from an environment-centered perspective. Each reward is calculated based on the interaction of the candidate trajectory with the predicted world features, such as obstacles, drivable areas, and the ego's progress.

Below is the detailed description of how each reward is calculated and incorporated into the model.

1. NC. The NC reward evaluates whether the candidate trajectory intersects with obstacles in the environment. The trajectory points are compared against the predicted occupancy grid (instance occupancy) to identify collisions. If no collision occurs along the trajectory, the reward is maximized (i.e., a score of 1), and if a collision is detected, the reward is minimized (i.e., a score of 0). The final NC reward is computed as:

$$S_{\text{NC}}(\tau) = \begin{cases} 1 & \text{if no collision occurs,} \\ 0 & \text{if collision occurs.} \end{cases} \tag{17}$$

2. DAC. The DAC reward ensures that the trajectory stays within the predicted drivable areas. The penalty is applied whenever any point of the trajectory moves outside the permissible drivable zone, signifying a violation of traffic rules. The reward is calculated as follows: the trajectory receives a reward of 1 if all points lie within the drivable area, and 0 if any point of the trajectory lies outside the drivable area. The final DAC reward is given by:

$$S_{\text{DAC}}(\tau) = \begin{cases} 1 & \text{if } \tau \text{ is within the drivable area,} \\ 0 & \text{if } \tau \text{ is outside the drivable area.} \end{cases} \tag{18}$$

3. EP. The EP reward evaluates the ego's progress along the trajectory by measuring its forward movement in terms of longitudinal displacement. Positive progress is encouraged, while negative progress (backward movement) is penalized. To ensure that the trajectory remains safe, the progress reward is only activated if the NC and DAC conditions are satisfied. If either of these conditions is violated, the reward is set to zero. The reward is computed as follows:

$$S'_{\text{EP}}(\tau) = \begin{cases} \frac{C_{\text{EP}}(\tau)}{C_{\text{EP}}^{\max}} & \text{if } C_{\text{EP}}^{\max} > 5.0 \text{ and } C_{\text{EP}}(\tau) \geq 0, \\ 1 & \text{if } C_{\text{EP}}^{\max} \leq 5.0 \text{ and } C_{\text{EP}}(\tau) \geq 0, \\ 0 & \text{otherwise,} \end{cases} \tag{19}$$

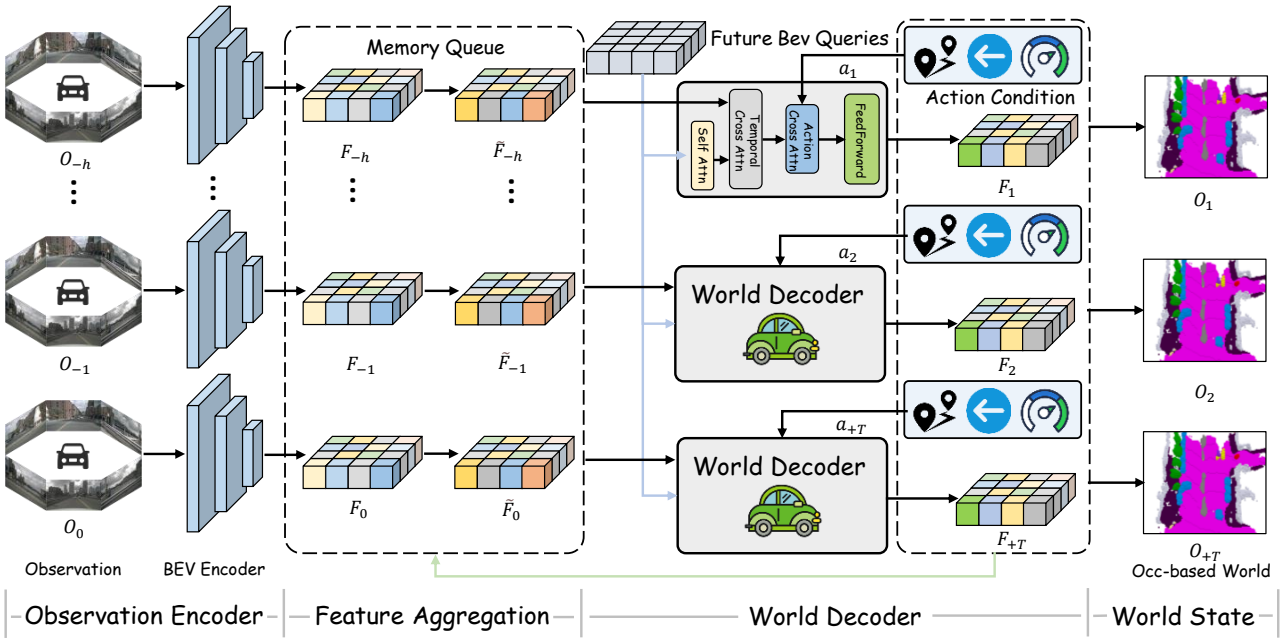

Figure 4. **Detailed structure of the occupancy-based world model,** which predicts the future world states through an autoregressive manner. The model utilizes an observation encoder to process multi-view images, a feature aggregation module to capture temporal consistency, and a world decoder to predict the future BEV embedding based on historical and current world features. This approach allows the model to predict future occupancy states.

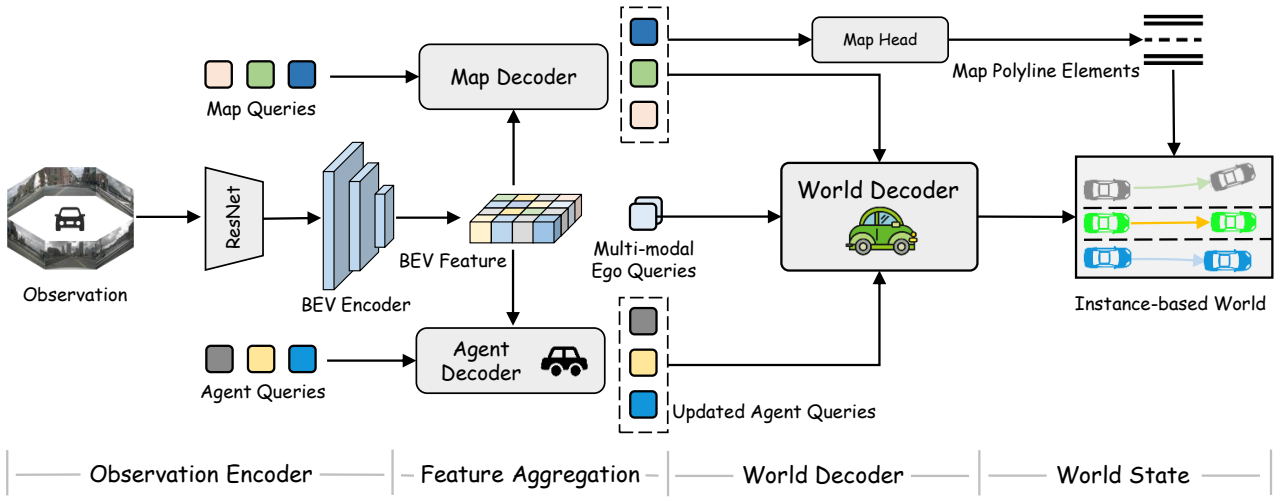

Figure 5. **Detailed structure of the instance-based world model,** which predicts the future map elements and agent motion. The model uses a ResNet encoder to process multi-view images, a BEV encoder to transform the features into BEV space, and two decoders: the map decoder for static road elements and the world decoder for dynamic agents.

$$S_{\mathrm{EP}}(\tau) = S'_{\mathrm{EP}}(\tau)\, S_{\mathrm{NC}}(\tau)\, S_{\mathrm{DAC}}(\tau), \tag{20}$$

where $C_{\mathrm{EP}}(\tau)$ represents the longitudinal displacement along the trajectory τ. $C_{\mathrm{EP}}^{\max}$ is the maximum longitudinal displacement of the candidate trajectory within the batch.

4. TTC. The TTC reward evaluates the ego vehicle's distance to potential obstacles in its future trajectory. To ensure safety, the trajectory is extended forward by a fixed distance $d_{\mathrm{fix}} = 10$m, and obstacles within the drivable region are checked. If no collision risk is detected within this extended range, the TTC reward is assigned a value of 1, indicating a safe trajectory. If a collision risk is detected, the reward is set to 0, indicating an unsafe trajectory. The

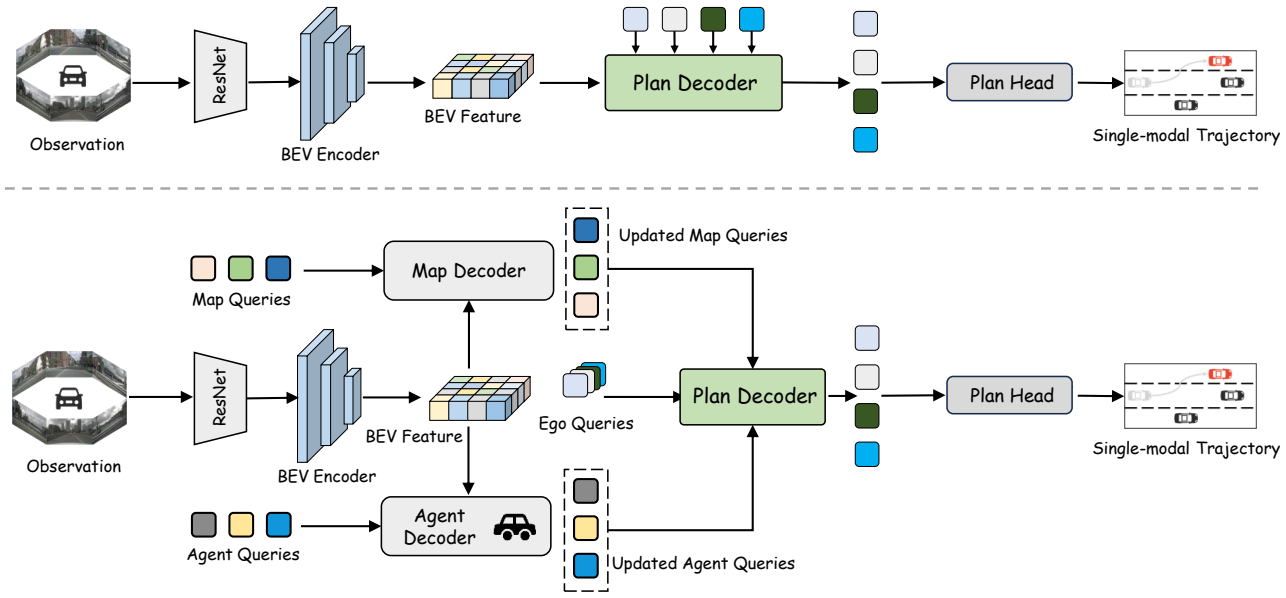

Figure 6. **Illustration of our Occ-based and instance-based baseline models.** The top part shows the occupancy-based baseline model, while the bottom part illustrates the instance-based baseline model. Both approaches utilize a BEV decoder, but differ in how planning queries interact with features.

final TTC reward is given by:

$$S'_{\text{TTC}}(\tau) = \begin{cases} 1 & \text{if no collision risk is detected,} \\ 0 & \text{if collision risk is detected,} \end{cases} \tag{21}$$

$$S_{\text{TTC}}(\tau) = S'_{\text{TTC}}(\tau)\, S_{\text{DAC}}(\tau)\,. \tag{22}$$

5. Comf. The Comf. reward evaluates the smoothness and comfort of the trajectory by penalizing abrupt longitudinal/lateral accelerations and jerk. Given a candidate trajectory τ, we compute the longitudinal acceleration a_{lon}, lateral acceleration a_{lat}, jerk magnitude $\|\mathbf{j}\|$, and longitudinal jerk j_{lon} based on the ego-vehicle's kinematic profile. The comfort reward is assigned only when all motion quantities fall within acceptable thresholds:

$$\begin{aligned} S_{\text{Comf.}}(\tau) = \mathbf{1}\left[a_{\text{lon}} \in [a_{\min}, a_{\max}]\right] \\ \cdot \mathbf{1}\left[|a_{\text{lat}}| \le a_{\text{lat}}^{\max}\right] \\ \cdot \mathbf{1}\left[\|\mathbf{j}\| \le j^{\max}\right] \\ \cdot \mathbf{1}\left[|j_{\text{lon}}| \le j_{\text{lon}}^{\max}\right], \end{aligned} \tag{23}$$

where $\mathbf{1}[\cdot]$ is an indicator function. The thresholds are based on NAVISIM comfort standards:

- $a_{\min} = -4.05\,\text{m/s}^2$: minimum longitudinal deceleration, $a_{\max} = 2.40\,\text{m/s}^2$: maximum longitudinal acceleration,
- $a_{\text{lat}}^{\max} = 4.89\,\text{m/s}^2$: maximum lateral acceleration,
- $j^{\max} = 8.37\,\text{m/s}^3$: maximum jerk magnitude,
- $j_{\text{lon}}^{\max} = 4.13\,\text{m/s}^3$: maximum longitudinal jerk.

Here, a_{lon} and a_{lat} represent longitudinal and lateral accelerations, respectively, while $\mathbf{j}$ is the jerk vector, and j_{lon} is its longitudinal component. A trajectory is considered comfortable when all constraints are satisfied.

7. More Closed-loop Experiments

7.1. Multi-Ability Results

Tab. 7 further compares the success rates of different methods across five challenging driving scenarios. A scenario is considered successful only when the ego vehicle reaches the target destination without any collisions or infractions.

Both WPT-Teacher and WPT-Student achieve strong improvements over the Baseline across all abilities. In particular, WPT-Teacher obtains the highest success rates in *Merging* (47.50%), *Overtaking* (73.33%), *Emergency Brake* (65.00%), and *Traffic Sign* (53.16%), leading to the best overall ability score of **57.80%**. Meanwhile, the lightweight WPT-Student also surpasses the Baseline notably (49.25% vs. 39.36%), demonstrating that our distillation strategy preserves strong multi-ability performance. These results highlight the effectiveness of WPT in handling complex, interactive driving scenarios with superior generalization capability.

7.2. Computation and Latency

We analyze compute and inference cost for the teacher (WPT-Teacher) and the distilled compact student (WPT-Student) in Tables 8 and 9. Here, Baseline-T is the teacher without reward model, and Baseline is the student trained without reward model and distillation.

Training Process. Relative to Baseline-T, WPT-Teacher halves training to 12 epochs and reduces GPU time to 248h (**–46.6%**) while improving planning to 0.61m / 0.11% (Ta-

Table 7. **Multi-ability performance of E2E methods.** * denotes expert feature distillation.

Method	Ability (%) ↑					
	Merging	Overtaking	Emergency Brake	Give Way	Traffic Sign	Mean
AD-MLP [56]	0.00	0.00	0.00	0.00	4.35	0.87
UniAD-Tiny [19]	8.89	9.33	20.00	20.00	15.43	14.73
UniAD-Base [19]	14.10	17.78	21.67	10.00	14.21	15.55
VAD [24]	8.11	24.44	18.64	20.00	19.15	18.07
DriveTransformer [23]	17.57	35.00	48.36	40.00	52.10	38.60
DiffAD [43]	30.00	35.55	46.66	40.00	46.32	38.79
Baseline	26.26	40.00	40.00	50.00	40.53	39.36
WPT-Student (Ours)	30.00	55.56	63.33	50.00	47.37	49.25
WPT-Teacher (Ours)	**47.50**	**73.33**	**65.00**	**50.00**	**53.16**	**57.80**
TCP* [48]	16.18	20.00	20.00	10.00	6.99	14.63
TCP-Ctrl* [48]	10.29	4.44	10.00	10.00	6.45	8.23
TCP-Traj* [48]	8.89	24.29	51.67	40.00	46.8	34.22
ThinkTwice* [21]	27.38	18.42	35.82	50.00	54.23	37.17
DriveAdapter* [20]	28.82	26.38	48.76	50.00	**56.43**	42.08

Table 8. **Comparison of training process.**

Method	**Training Settings**		**Planning**	
	Epoch	GPU Hours (h)	Avg. L2 (m) ↓	Avg. Col. (%) ↓
Baseline-T	24	464	0.72	0.71
WPT-Teacher	12	248	0.61	0.11
Baseline	24	488	0.88	1.06
WPT-Student	12	168	0.66	0.24

Table 9. **Comparison of planning inference time.**

Method	**Rwd Model**	**Planning**		**Latency**
		Avg. L2 (m)↓	Avg. Col. (%)↓	(ms)↓
Baseline-T	-	0.72	0.71	286
WPT-Teacher	-	0.62	0.14	286
WPT-Teacher	✓	0.61	0.11	312
Baseline	-	0.88	1.06	64
WPT-Student	-	0.66	0.24	64

ble 8). For the lightweight policy, distillation cuts training from 488 h (Baseline) to 168h (**–65.6%**) and improves planning from 0.88m / 1.06% to 0.66 m / 0.24%. Thus, WPT-Teacher accelerates convergence and lowers compute for both teacher and student, with especially large relative savings for the compact model.

Inference Latency. On the teacher, enabling the reward model online adds a small overhead (286 → 312ms, but yields the best safety (0.11% collisions). Turning the reward model off restores baseline latency (286ms) while still outperforming Baseline-T (0.62m / 0.14% vs. 0.72m / 0.71%). The distilled student maintains the **64ms** latency of Baseline yet delivers markedly better plans (0.66m / 0.24% vs. 0.88m / 1.06%), indicating that world-aware distillation transfers most of the benefit without any test-time world-model calls.

7.3. Visualizations

We present visualizations of some challenging closed-loop simulation scenarios from the Bench2Drive benchmark, as shown in Fig. 7. Additionally, more closed-loop visualizations in video format can be found in the supplementary materials.

(a) Obstacle avoidance when other vehicles merge into the lane.

(b) Change lanes when encountering stationary vehicles.

(c) Slow down and give way when encountering a vehicle that suddenly starts to change lanes in the night.

(d) Slow down and give way when encountering pedestrians who suddenly appear in the rain.

(e) Slow down and give way when encountering pedestrians who suddenly appear in the night.

(f) Change lanes in the fog.

Figure 7. Visualization of representative closed-loop scenarios from the Bench2Drive benchmark.