FORECASTEDIT: LEARNING TO EDIT WORLD FORECASTS FOR 4D OCCUPANCY PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Future-world models for autonomous driving typically improve prediction by learning to generate increasingly accurate future states from observations. We study a complementary problem: once a future world has already been predicted, can a model learn what should be edited rather than regenerate the entire forecast? We introduce **ForecastEdit**, which treats a frozen world-model prediction as an explicit reference and learns selective, evidence-grounded revisions. ForecastEdit combines evidence-guided boundary editing to retract unsupported occupancy with executable semantic editing that converts ontology-structured object dynamics into localized, utility-aware edits, followed by lightweight semantic commitment. On Occ3D-nuScenes, ForecastEdit achieves state-of-the-art semantic forecasting performance with 14.82 mIoU, improving the SparseWorld anchor by 1.62 points while also improving occupied IoU. The edited future representation further improves downstream planning, reducing average trajectory error from 0.27 to 0.23 m and collision rate from 0.29% to 0.16%. Controlled comparisons show that these gains cannot be explained by generic post-hoc correction alone. Our results suggest that stronger world models may require not only better forecasting, but also principled mechanisms for editing the futures they already predict.

1 INTRODUCTION

Anticipating how the surrounding world evolves is fundamental to autonomous driving, as planning depends not only on the current scene but also on future geometry, semantics, and motion (Hu et al., 2023; Li et al., 2022). Semantic occupancy provides a unified representation of scene structure and has increasingly become an interface between perception, forecasting, and planning (Wei et al., 2023; Zhang et al., 2023; Tian et al., 2023; Zheng et al., 2024). However, predicting an entire future world remains difficult, particularly over long horizons where geometric and semantic errors accumulate. This motivates a different question: *given an existing future forecast, can we improve it by learning what to edit rather than predicting the world again from scratch?*

Recent world models have substantially advanced occupancy forecasting through occupancy-centric representations (Zheng et al., 2024), future geometry modeling (Yang et al., 2024), spatiotemporal pre-training (Min et al., 2024), action-conditioned forecasting (Yang et al., 2025), and sparse dynamic representations (Dang et al., 2026). For example, SparseWorld already achieves 13.20 mIoU on Occ3D-nuScenes, showing that modern world models can recover substantial portions of future scene structure with high fidelity (Dang et al., 2026). However, as shown in Fig. 1(a), their remaining errors are therefore often local rather than global, including overextended occupancy, missed objects, and semantic misclassification (Kim et al., 2026). This makes full regeneration potentially redundant: when most of a forecast is already useful, the key challenge becomes identifying *where* it is wrong and *whether* a proposed correction should overwrite it. Unlike forecasting, which asks what the future should be, editing asks what should change in an already predicted future.

To address this problem, we introduce **ForecastEdit**, shown in Fig. 1(b), which treats the output of a frozen SOTA world model as an explicit editing reference. ForecastEdit first performs *evidence-guided boundary editing* (EGBE), using renderer provenance and local spatial support to retract unsupported occupancy while preserving well-supported regions. It then performs *executable semantic editing* (ESE), which reasons over object histories with ontology-structured states describing existence, kinematics, interaction, and visibility, refines their future trajectories, and translates the

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

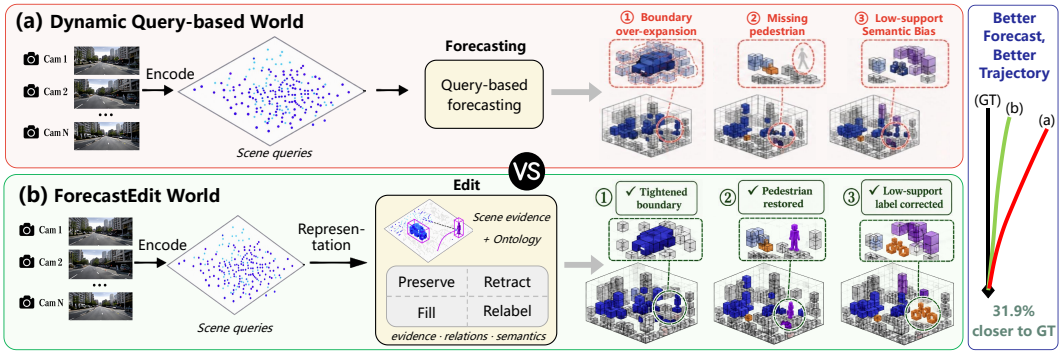

Figure 1: Overview of ForecastEdit. (a) Query-based world models may retain localized errors in future occupancy. (b) ForecastEdit treats the predicted future as an editable reference and selectively *Preserves*, *Retracts*, *Fills*, or *Relabels* states using scene evidence and ontology guidance, improving both prediction and planning.

resulting dynamics into localized voxel edits. Candidate edits are selectively committed relative to the original forecast, followed by a lightweight semantic correction step. Together, ForecastEdit learns *what to remove, what to write, and when to preserve the original forecast*, while jointly improving the downstream planning tasks.

We evaluate ForecastEdit on Occ3D-nuScenes (Tian et al., 2023) using SparseWorld (Dang et al., 2026) as the frozen forecasting anchor. ForecastEdit achieves a state-of-the-art 14.82 mIoU, improving SparseWorld from 13.20 by 1.62 points, while increasing occupied IoU from 22.03 to 22.84. The improved future forecast further benefits downstream planning, reducing average L2 error from 0.27 to 0.23 m and collision rate from 0.29% to 0.16%. Ablations show progressive gains to 14.17 with evidence-guided boundary editing, 14.53 with executable object editing, and 14.82 with semantic commitment. Moreover, ForecastEdit outperforms the strongest examined generic class-balanced correction (+1.28 vs. +1.62 mIoU) and improves 16 of 17 semantic classes. These results show that structured editing can consistently improve the world forecast and its downstream utility.

Our contributions are threefold: (1) We formulate future-world editing as a complementary problem to forecasting, using existing predictions as explicit revision references. (2) We propose ForecastEdit, combining evidence-guided boundary editing with ontology-informed trajectory and semantic editing for structured forecast refinement. (3) ForecastEdit achieves state-of-the-art 14.82 mIoU on Occ3D-nuScenes, improving the SparseWorld anchor by 1.62 points and further benefiting downstream trajectory planning.

2 Related Work

Occupancy Forecasting and Driving World Models. Forecasting future scene evolution is central to autonomous driving, as planning benefits from explicitly modeling how surrounding occupancy and interactions evolve beyond the current observation (Huang et al., 2026). Semantic occupancy offers a unified representation for this purpose, and recent driving world models have explored occupancy-centric generation, future geometry modeling, spatiotemporal pre-training, action-conditioned prediction, and sparse dynamic representations (Zheng et al., 2024; Yang et al., 2024; Min et al., 2024; Yang et al., 2025; Dang et al., 2026). These advances substantially improve how future worlds are generated from observations and actions. However, as forecasts become increasingly accurate, their remaining errors are often localized within an otherwise useful prediction, making complete regeneration potentially redundant and disruptive to already-correct states. ForecastEdit instead treats the predicted future as an explicit reference and learns where and how it should be selectively edited.

Object-centric Forecasting and Prediction Refinement. Object-centric forecasting models future behavior by explicitly representing agent histories, interactions, and multimodal motion, providing structured cues beyond dense scene prediction (Gao et al., 2020; Yuan et al., 2021; Shi et al., 2022; Ngiam et al., 2022). Related work on occupancy refinement further improves coarse predictions through additional geometric or semantic correction (Shi et al., 2025). These approaches show

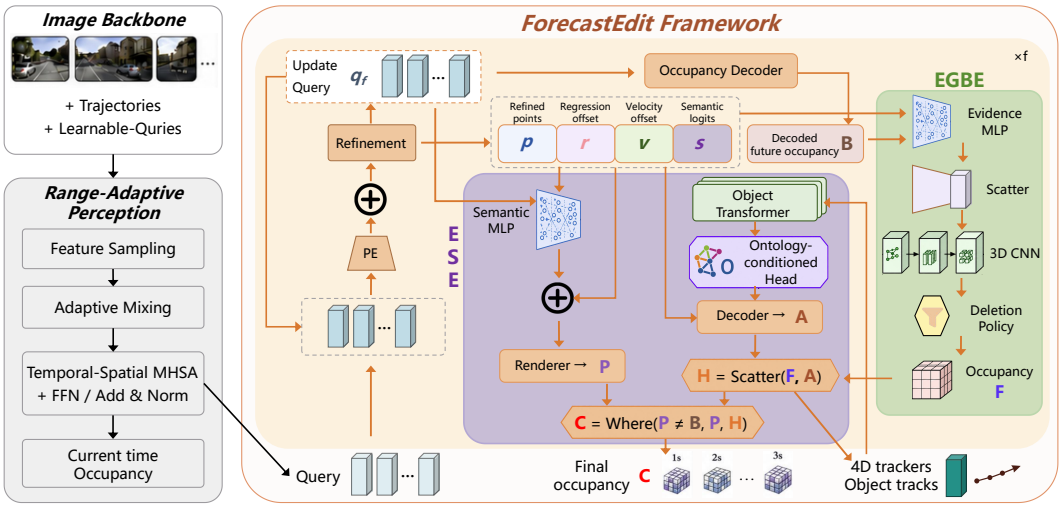

Figure 2: Framework of ForecastEdit. Starting from a frozen reference forecast B, EGBE retracts unsupported occupancy to obtain F. ESE converts object tracks and ontology-structured dynamics into sparse edits A, yielding H, while semantic commitment produces a complementary candidate P. Deterministic composition returns the final edited forecast C.

that both structured object reasoning and post-hoc correction can improve future scene prediction. However, they typically either predict future object states directly or refine outputs without explicitly treating existing future occupancy as a state whose correct regions should be preserved. ForecastEdit combines these perspectives by using object-level semantic states to construct localized revisions and learning when an existing world forecast should be preserved, retracted, or overwritten.

3 Forecast Editing as a Learning Problem

Given historical observations X, a frozen world forecaster f_θ produces a future semantic occupancy forecast

$$B = f_\theta(X), \tag{1}$$

where $B_t(v)$ denotes the predicted semantic state at voxel v and future horizon t. Rather than regenerating the future world, we treat B as an explicit *reference forecast* and learn which states should be preserved or revised.

Forecast Editing. Given auxiliary evidence Z, an editor G_ϕ produces localized candidate edits $\mathcal{A}$ and applies committed revisions to the reference:

$$\mathcal{A} = G_\phi(B, Z), \qquad C = \mathcal{E}(B, \mathcal{A}). \tag{2}$$

Editing is reference-relative: states without a committed revision retain their predictions in B.

Learning to Edit. A candidate edit should be judged by whether it improves the state it replaces relative to preserving the reference, rather than by its plausibility alone. Let $\mathcal{U}(\cdot, Y)$ denote task utility with respect to ground truth Y. For a candidate edit a, we define its relative editing utility as

$$\Delta\mathcal{U}(a; B, Y) = \mathcal{U}(\mathcal{E}(B, \{a\}), Y) - \mathcal{U}(B, Y). \tag{3}$$

Forecast editing therefore learns both *where* the reference should change and *what supported alternative* should replace it. Eq. (3) provides the general principle rather than a unified training objective, which ForecastEdit instantiates through module-specific editing objectives.

4 ForecastEdit

4.1 Overview

ForecastEdit instantiates the above formulation on top of a frozen query-based world model, as illustrated in Fig. 2. The frozen forecaster provides both the reference occupancy forecast B and intermediate states used as editing evidence. ForecastEdit then performs selective revision through two mechanisms: *evidence-guided boundary editing* (EGBE) and *executable semantic editing* (ESE).

EGBE traces renderer provenance and local spatial support to retract unsupported occupied states, producing a boundary-edited forecast F. ESE then introduces supported revisions through two complementary paths: object-grounded executable editing and semantic commitment. The object path reasons over predicted object histories with ontology-structured states and multimodal dynamics, converts future object support into localized voxel edits, and applies the selected edits to F to obtain H. In parallel, semantic commitment derives a complementary semantic candidate P from frozen forecasting states. The final edited forecast is composed relative to the original reference:

$$C_t(v) = \begin{cases} P_t(v), & P_t(v) \neq B_t(v), \\ H_t(v), & \text{otherwise.} \end{cases} \tag{4}$$

Thus, semantic commitment overrides the object-edited forecast only when it changes the reference decision; otherwise, the structured edit in H is preserved. This deterministic composition defines the hard inference-time decision flow of ForecastEdit while leaving the underlying world model frozen.

4.2 Evidence-Guided Boundary Editing (EGBE)

Forecast editing first requires deciding which occupied states in the reference forecast should be retracted. A semantic label alone does not reveal whether an occupied voxel is supported by consistent scene evidence or arises from sparse contributions, local propagation, or uncertain boundaries (Wang et al., 2024). EGBE therefore traces each occupied prediction back to its frozen renderer provenance and evaluates its local structural support. Its action space is deliberately restricted to *preserve* or *retract-to-*`free`, leaving foreground insertion and relabeling to subsequent semantic editing.

Provenance-Aware Boundary Evidence. Let $\mathcal{V}_B = \{v \mid B(v) \neq \texttt{free}\}$ denote the occupied candidates in the reference forecast. For each $v \in \mathcal{V}_B$, we retrieve its contributing refine points $\mathcal{S}_v = \{p_i\}_{i=1}^{N_v}$ from the frozen renderer, including direct contributors and class-consistent supporting points. Each contributor is encoded from its semantic response s_i and relative geometry δ_{iv} as $e_i = \phi(s_i, \delta_{iv})$. In parallel, $q_v = \psi(g_v, \mathrm{Emb}(B(v)))$ encodes the predicted class and local structural statistics g_v, including contributor support and neighborhood consistency. Since contributors vary in number and order, we aggregate them with candidate-conditioned attention:

$$\alpha_i = \mathrm{softmax}_{i \in \mathcal{S}_v}\left(\frac{(W_q q_v)^\top (W_k e_i)}{\sqrt{d}}\right), \qquad h_v^{\mathrm{prov}} = \left[q_v;\ \sum_{i \in \mathcal{S}_v} \alpha_i e_i;\ \max_{i \in \mathcal{S}_v} e_i\right]. \tag{5}$$

Here, ϕ and ψ are lightweight encoders, W_q and W_k are learnable projections, and d is the attention dimension. The attention term captures candidate-specific contributor support, while max pooling retains strong local evidence.

Renderer provenance alone is insufficient to determine whether an occupancy boundary is spatially reliable (Zhang et al., 2023; Wei et al., 2023). We therefore scatter the provenance representations back to the voxel grid and contextualize them with a lightweight local 3D network:

$$h_v^{\mathrm{loc}} = \mathcal{G}_{3\mathrm{D}}\left(\mathrm{Scatter}\left(\{h_u^{\mathrm{prov}}\}\right)\right)_v. \tag{6}$$

Provenance captures *why* an occupied state exists, while local context evaluates whether that evidence forms coherent 3D structure. Detailed provenance features, local architecture, and source-side calibration are provided in Appendix A.2.

Utility-Aware Retraction. Given the contextualized evidence h_v^{loc}, EGBE predicts a retraction score $d_v = w_d^\top h_v^{\mathrm{loc}}$ with target $y_v^{\mathrm{ret}} = \mathbf{1}[Y(v) = \texttt{free}]$. A uniform binary objective treats all preserve/retract decisions equally, although their effects on semantic IoU vary across classes. We therefore derive class-conditional weights from the marginal utility of modifying the reference forecast. Let $c = B(v)$ and (I_c, U_c) denote the source-side intersection and union of class c. Retracting a false-positive voxel yields the marginal gain

$$\Delta_c^+ = \frac{I_c}{U_c - 1} - \frac{I_c}{U_c} = \frac{I_c}{U_c(U_c - 1)}, \qquad \Delta_c^- = \frac{I_c - 1}{U_c} - \frac{I_c}{U_c} = -\frac{1}{U_c}, \tag{7}$$

where Δ_c^- denotes the corresponding utility change from incorrectly retracting a true-positive voxel. After class-wise normalization, Δ_c^+ and $-\Delta_c^-$ provide positive weights for beneficial retraction and costly removal, respectively. The resulting objective combines utility-weighted binary classification with pairwise ranking:

$$\mathcal{L}_{\mathrm{EGBE}} = \mathcal{L}_{\mathrm{uBCE}} + \lambda_{\mathrm{rank}} \mathcal{L}_{\mathrm{rank}}. \tag{8}$$

The ranking term further encourages beneficial retractions to receive higher scores than states that should be preserved.

Restricted Boundary Editing. At inference, class-specific thresholds τ_c are calibrated exclusively on held-out source data. For each candidate $v \in \mathcal{V}_B$, we define $m_v = \mathbf{1}[d_v > \tau_{B(v)}]$ and obtain

$$F(v) = \begin{cases} \texttt{free}, & m_v = 1, \\ B(v), & \text{otherwise.} \end{cases} \tag{9}$$

All states outside $\mathcal{V}_B$ remain unchanged. EGBE therefore only preserves or retracts existing occupancy, leaving foreground insertion and relabeling to executable semantic editing.

4.3 Executable Semantic Editing (ESE)

EGBE retracts unsupported occupancy but cannot recover missing future states or correct remaining semantic errors. ESE addresses this complementary problem by converting structured object dynamics into sparse, localized edits over the boundary-edited forecast F. Rather than densely reclassifying the voxel grid, ESE predicts how objects evolve and translates their future support into executable voxel-level revisions.

Ontology-Structured Object Dynamics. Structured interaction modeling provides useful relational inductive biases for driving (Zhang et al., 2025; Sima et al., 2024; Cheng et al., 2025). We adapt this structure to future-state editing rather than action reasoning. Given camera-only predicted object histories, we encode each object's temporal evolution and interactions with surrounding agents into a relational representation h_i^{rel}. We further predict four ontology-defined states—existence, kinematics, interaction, and visibility—with distributions p_i^{exist}, p_i^{kin}, p_i^{int}, and p_i^{vis}. Rather than serving only as auxiliary predictions, these states are projected back into the object representation:

$$z_i = h_i^{\text{rel}} + W_o \left[p_i^{\text{exist}}; p_i^{\text{kin}}; p_i^{\text{int}}; p_i^{\text{vis}} \right]. \tag{10}$$

The resulting representation directly conditions a multimodal decoder that predicts future object geometry, existence, and local occupancy support. Thus, structured semantics influence where an object may persist or move, providing spatial support for executable edits. Detailed object inputs, ontology definitions, and multimodal dynamics are provided in Appendix A.3.

From Future Support to Localized Edits. We project each object's multimodal future geometry and occupancy support into the common voxel space. For each motion mode and horizon, predicted center, size, and orientation are converted into spatial support using an oriented-box renderer, together with history, query, and temporal evidence. Rather than overwriting the forecast with an entire predicted object, we decompose this support into localized candidates $a = (t, v, c)$, where t is the future horizon, v the voxel location, and c the proposed semantic state. Collecting candidates from all object futures gives $\widetilde{\mathcal{A}} = \bigcup_i \text{Voxelize}(\mathcal{D}_{\text{future}}(z_i))$, while $\mathcal{A}_{t,v} \subseteq \widetilde{\mathcal{A}}$ denotes candidates addressing voxel–horizon pair (t, v). These candidates specify *what could be written*; whether they should overwrite the reference is determined by utility-aware selection.

Utility-Aware Edit Selection. Object support alone does not imply that an edit should replace the current forecast. For each candidate $a = (t, v, c)$, we aggregate object support, local geometry, forecast evidence, and temporal consistency into an action representation h_a and predict three editing consequences: *repair*, *damage*, and *remain-wrong*. We then convert these consequences into class-conditional utility and select against an explicit no-op baseline:

$$\begin{aligned} p_o(a) &= \text{softmax}(g_{\text{sel}}(h_a))_o, \qquad o \in \{\text{repair}, \text{damage}, \text{remain-wrong}\}, \\ u(a) &= \sum_o w_o^c p_o(a), \qquad u(\varnothing) = 0, \\ a_{t,v}^* &= \arg \max_{a \in \mathcal{A}_{t,v} \cup \{\varnothing\}} u(a). \end{aligned} \tag{11}$$

Here, w_o^c encodes the class-conditional consequence of each outcome, while $\varnothing$ explicitly preserves the current state. The selected edit is applied to F, yielding $H_t(v) = c^*$ if $a_{t,v}^* = (t, v, c^*)$ and $H_t(v) = F_t(v)$ otherwise. ESE therefore commits an alternative only when its estimated utility exceeds preservation, turning structured object dynamics into sparse executable forecast edits.

4.4 Semantic Commitment and Edit Composition

Object-grounded edits capture structured future changes, but semantic errors may remain beyond object-derived support. We therefore introduce a lightweight semantic commitment branch that revises frozen semantic states while preserving the original geometry and motion.

Semantic Commitment. Let ξ_s collect the frozen query/refine representation and its associated semantic, geometric, and temporal features, and let s denote the frozen semantic logits. A lightweight residual head predicts

$$\Delta s = r_\psi(\mathrm{stopgrad}(\xi_s)), \qquad s' = \mathrm{stopgrad}(s) + \Delta s. \tag{12}$$

Utilizing the unchanged geometry and motion states, the frozen renderer converts s' into a semantic candidate $P = \mathcal{R}(s')$. Thus, semantic commitment revises class decisions without perturbing the underlying spatial forecast.

Reference-Aware Edit Composition. The object path produces H, while semantic commitment produces P. We assign final write ownership relative to the original reference B:

$$C_t(v) = \begin{cases} P_t(v), & P_t(v) \neq B_t(v), \\ H_t(v), & \text{otherwise.} \end{cases} \tag{13}$$

Semantic commitment therefore overrides H only when it changes the reference decision; otherwise, the object-grounded edit is retained.

4.5 Training and Inference

Training. The base world model remains frozen throughout training. EGBE is first optimized on source data using the utility-aware retraction objective in Eq. (8), after which its boundary model and calibrated execution policy are fixed. The ESE object path is then trained to jointly model ontology-conditioned multimodal dynamics and edit utility, including the selection of sparse voxel-level revisions from predicted future support. Finally, semantic commitment updates only the lightweight residual head on detached forecasting states using source-derived class-conditional supervision and scene-group reweighting. The editing modules are therefore optimized stage-wise against the same reference forecast, without impairing the predictive ability of the underlying world model. Full training and calibration details are provided in Appendix A.4.

Inference. Given an input sequence, the frozen world model produces the reference forecast B and its intermediate states. EGBE applies its frozen retraction policy and class-specific calibration to remove unsupported occupancy, yielding F. ESE then constructs candidate revisions from predicted object futures, resolves competing edits at each voxel–horizon location, and applies the selected sparse edits to obtain H. In parallel, semantic commitment generates the candidate forecast P from frozen semantic states using the learned residual head and original renderer. The final forecast C follows the deterministic composition rule in Eq. (13). All editing parameters and calibration rules remain fixed at inference, requiring no target annotations, online source-statistic updates, or parameter adaptation.

5 Experiments

5.1 Experimental Setup

Implementation. ForecastEdit is built on the frozen SparseWorld forecasting anchor, whose perception, renderer, and planning components remain unchanged. EGBE, the ESE object path, and semantic commitment are trained stage-wise against the same reference forecasts. EGBE learns provenance-aware retraction from renderer evidence, while the ESE object path uses camera-only predicted tracks to model ontology-conditioned future dynamics and select sparse voxel edits. Semantic commitment updates only a lightweight residual head on detached forecasting states. During inference, all editing policies and calibration parameters are fixed, requiring neither target annotations nor online source-statistic estimation. All ablations and correction baselines share the same forecasting anchor, renderer, data split, and evaluation protocol. Full implementation, training, calibration, and inference details are provided in Appendix A.

Table 1: Future occupancy forecasting on Occ3D-nuScenes. We report semantic mIoU and occupied IoU at 1, 2, and 3 s prediction horizons. **Bold** and underlined denote the best and second-best results.

Method	Aux. Sup.	mIoU ↑				IoU ↑			
		1s	2s	3s	Avg.	1s	2s	3s	Avg.
OccWorld-T	Semantic LiDAR	4.68	3.36	2.63	3.56	9.32	8.23	7.47	8.34
OccWorld-D	3D Occ	11.55	8.66	6.98	8.66	18.90	16.26	14.43	16.53
OccLLaMA-F	3D Occ	10.34	8.66	6.98	8.66	**25.81**	**23.19**	19.97	**22.99**
PreWorld	3D Occ	11.69	8.72	6.77	9.06	23.01	20.79	18.84	20.88
PreWorld + Pre-training	2D & 3D Occ	12.27	9.24	7.15	9.55	23.62	21.76	19.63	21.62
SparseWorld	3D & 4D Occ	14.93	13.15	11.51	13.20	22.96	22.10	21.05	22.03
ForecastEdit (Ours)	3D & 4D Occ	**16.70**	**14.75**	**13.02**	**14.82**	23.89	22.98	**21.64**	22.84

Table 2: Motion planning on Occ3D-nuScenes. We report trajectory L2 error and collision rate over 1–3 s horizons. † denotes the use of ego state during training and inference. **Bold** and underlined denote the best and second-best results.

Method	Aux. Sup.	L2 (m) ↓				Collision Rate (%) ↓			
		1s	2s	3s	Avg.	1s	2s	3s	Avg.
UniAD	Map & Box & Occ	0.48	0.96	1.65	1.03	**0.05**	0.17	0.71	0.31
OccNet	Map & Box & 3D Occ	1.29	2.13	2.99	2.14	0.21	0.59	1.37	0.72
OccWorld-D	3D Occ	0.52	1.27	2.41	1.40	0.12	0.40	2.08	0.87
OccLLaMA-F†	3D Occ	0.38	1.07	2.15	1.20	0.06	0.39	1.65	0.70
PreWorld	2D & 3D Occ	0.41	1.16	2.32	1.30	0.50	0.88	2.42	1.27
PreWorld†	2D & 3D Occ	0.22	0.30	0.40	0.31	0.21	0.66	0.71	0.53
SparseWorld	3D & 4D Occ	0.49	0.94	1.47	0.97	0.18	0.78	1.88	0.95
SparseWorld†	3D & 4D Occ	0.19	0.25	0.36	0.27	0.11	0.29	0.46	0.29
ForecastEdit (Ours)	**3D & 4D Occ**	0.46	0.76	1.14	0.78	0.22	0.56	1.12	0.64
ForecastEdit (Ours)†	**3D & 4D Occ**	0.19	**0.23**	**0.27**	**0.23**	0.14	**0.16**	**0.19**	**0.16**

Dataset and Metrics. We evaluate ForecastEdit on Occ3D-nuScenes (Tian et al., 2023) following the SparseWorld (Dang et al., 2026) protocol. It provides 700 training scenes and 150 validation scenes, with future occupancy evaluated at 1, 2, and 3 s using semantic mIoU and occupied IoU. For motion planning, we evaluate 4,819 valid windows using trajectory L2 error and collision rate.

Baselines. We compare with representative occupancy world models, including OccWorld (Zheng et al., 2024), OccLLaMA (Wei et al., 2024), PreWorld (Li et al., 2025), and SparseWorld (Dang et al., 2026). For planning, we additionally compare with UniAD (Hu et al., 2023) and OccNet (Tong et al., 2023). SparseWorld serves as the frozen forecasting anchor for all ForecastEdit variants and controlled comparisons.

5.2 Main Results

Future Occupancy Forecasting. As shown in Table 1, ForecastEdit achieves state-of-the-art semantic forecasting performance, ranking first in mIoU at every horizon and reaching 14.82 on average. Compared with SparseWorld, it improves average mIoU by 1.62 points, with consistent gains of 1.77, 1.60, and 1.51 points at 1, 2, and 3 s, respectively, showing that the benefit of editing persists as forecasting uncertainty increases. ForecastEdit also improves average occupied IoU from 22.03 to 22.84 and achieves the best 3 s IoU of 21.64. While OccLLaMA-F remains slightly higher in average occupied IoU, ForecastEdit delivers substantially stronger semantic accuracy together with competitive occupancy quality, indicating that structured editing improves semantic discrimination without sacrificing long-horizon scene structure.

Downstream Motion Planning. As shown in Table 2, ForecastEdit consistently improves downstream planning over the SparseWorld anchor under both settings. With ego state, ForecastEdit

Table 3: Progressive ablation of ForecastEdit on Occ3D-nuScenes. Components are removed in reverse editing order, and Δ reports the change in average performance.

Configuration	mIoU ↑				Δ Avg. mIoU	IoU ↑				Δ Avg. IoU
	1s	2s	3s	Avg.		1s	2s	3s	Avg.	
ForecastEdit	16.70	14.75	13.02	14.82	–	23.89	22.98	21.64	22.84	–
w/o Semantic Commitment	16.45	14.47	12.68	14.53	-0.29	23.90	22.98	21.64	22.84	0.00
w/o ESE	16.05	14.12	12.33	14.17	-0.36	23.89	22.98	21.63	22.83	-0.01
w/o ESE & EGBE	14.93	13.15	11.51	13.20	-0.97	22.96	22.10	21.05	22.03	-0.80

Table 4: Controlled comparison with generic correction mechanisms. All variants build on the same SparseWorld anchor and EGBE output; ΔmIoU is measured relative to the anchor.

Method	ΔmIoU (pp)
Anchor	+0.00
Anchor + EGBE	+0.97
EGBE + Generic Corrector: Point Sup.	+1.18
EGBE + Generic Corrector: Voxel CE	+1.20
EGBE + Generic Corrector: Class-weighted Voxel CE	+1.28
ForecastEdit (EGBE + ESE)	**+1.62**

achieves the best average L2 error of 0.23 m and collision rate of 0.16%, reducing them from 0.27 m and 0.29% for SparseWorld, respectively. The gains are particularly pronounced at longer horizons: at 3 s, L2 error decreases from 0.36 to 0.27 m, while collision rate drops from 0.46% to 0.19%. Without ego state, ForecastEdit similarly improves average L2 from 0.97 to 0.78 m and collision rate from 0.95% to 0.64%. These results indicate that the edited future representation provides more useful long-horizon information for downstream trajectory planning.

5.3 Ablation and Controlled Comparisons

Progressive Editing Ablation. As shown in Table 3, each editing stage contributes complementary gains. Removing semantic commitment reduces average mIoU from 14.82 to 14.53, while further removing ESE decreases it to 14.17. Removing EGBE finally recovers the SparseWorld anchor at 13.20. Notably, ESE substantially improves semantic mIoU while leaving occupied IoU nearly unchanged, whereas removing EGBE causes a pronounced IoU drop from 22.83 to 22.03. This suggests that EGBE primarily repairs occupancy boundaries, while ESE contributes complementary object- and semantic-level corrections.

Comparison with Generic Correction. We further examine whether ForecastEdit's gains can be explained by generic post-hoc correction. As shown in Table 4, conventional correction improves the SparseWorld anchor by 1.18–1.28 mIoU points, confirming that post-hoc refinement itself is beneficial. ForecastEdit achieves a larger 1.62-point gain, exceeding the strongest class-weighted alternative by 0.34 points. This gap suggests that the improvement is not explained by generic correction capacity or class balancing alone, but benefits from the structured executable editing introduced by ESE. Additional analyses are provided in Appendix B.

5.4 Understanding Forecast Editing

Complementary Roles Across Semantic Classes. As shown in Fig. 3, EGBE and ESE exhibit complementary class-wise behaviors. EGBE contributes most strongly to spatially extended and boundary-sensitive classes, including driveable surface, sidewalk, terrain, and car, consistent with its role in retracting unsupported occupancy. In contrast, ESE provides larger gains for small and dynamic traffic participants, such as bicycle, motorcycle, pedestrian, and traffic cone, where structured object dynamics and semantic editing are particularly important. Together, the two mechanisms improve 16 of 17 classes and yield a 1.62-point average gain. The only slight drop occurs for the

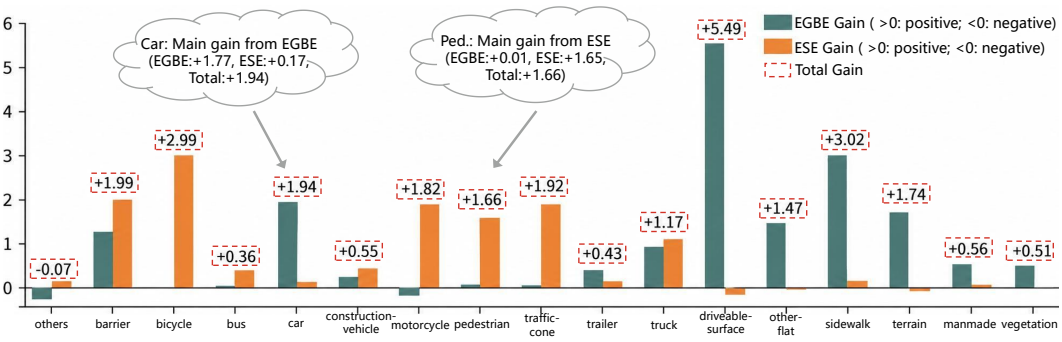

Figure 3: Class-wise editing gains on Occ3D-nuScenes. Bars report IoU improvements over SparseWorld from EGBE, ESE, and their combined ForecastEdit result; Avg. denotes the 17-class mean.

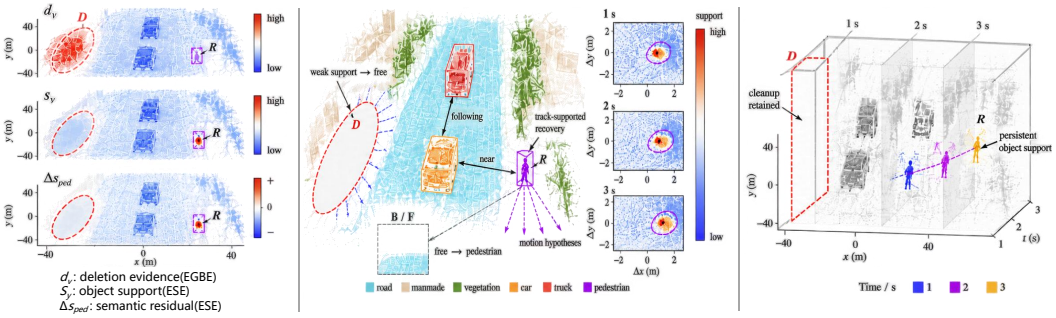

Figure 4: Evidence grounding of ForecastEdit. (a) Scene-aligned responses from EGBE deletion evidence d_v, ESE object support S_y, and semantic residual Δs. (b) Object-level semantic reasoning underlying executable edits. (c) Spatiotemporal support across future horizons.

heterogeneous *others* category, whose broad semantics are less aligned with the structured object categories modeled by ESE. Overall, ForecastEdit improves both occupancy structure and object-level semantic fidelity rather than favoring a single error type.

Evidence Grounding of Forecast Edits. As shown in Fig. 4, ForecastEdit grounds different revisions in complementary evidence sources. EGBE deletion evidence d_v concentrates on unsupported occupancy boundaries, while ESE object support S_y follows the spatial extent of predicted future objects. The semantic residual Δs further targets class-specific errors that remain after object-grounded editing. The decoded-scene and temporal views show that these responses are tied to structured object states and evolve consistently across future horizons. These observations suggest ForecastEdit performs evidence-specific, spatially grounded revisions rather than uniform post-hoc correction.

6 Conclusions

We study future-world prediction from a complementary perspective: rather than forecasting the entire future, an existing forecast can serve as a reference for deciding what should change and what should be preserved. We introduce **ForecastEdit**, which combines evidence-guided boundary editing with executable semantic editing to produce selectively committed revisions. On Occ3D-nuScenes, ForecastEdit achieves 14.82 mIoU, improving the SparseWorld anchor by 1.62 points while improving occupied IoU and downstream motion planning. Ablations and comparisons show that these gains arise from complementary boundary, object, and semantic editing rather than generic correction alone. Our results suggest that stronger world models may require not only better forecasting, but also principled mechanisms for editing the futures they already predict.

AI USE STATEMENT

Generative AI tools, including ChatGPT and AI coding assistants such as Codex, were used during the preparation of this work. They assisted with language editing, improving clarity and organization, and the implementation, debugging, and refinement of selected code components. All AI-assisted code was reviewed, tested, and integrated by the authors. The research questions, methodological design, experimental protocol, analysis of results, and conclusions were developed and verified by the authors. Generative AI was not used to generate experimental data or reported results. The authors take full responsibility for the final manuscript, implementation, and reported findings.

REPRODUCIBILITY STATEMENT

We provide the formulation, model components, training procedure, and evaluation protocol required to reproduce ForecastEdit. The main paper describes the complete forecast-editing pipeline and its interaction with the frozen SparseWorld forecasting anchor. The appendix provides the implementation environment, feature definitions, architectural configurations, training objectives, calibration procedures, hyperparameters, and controlled-baseline settings for EGBE and ESE. Unless otherwise stated, all ForecastEdit variants and controlled comparisons use the same frozen forecasting anchor, renderer, data splits, and evaluation protocol, enabling matched comparisons across editing mechanisms. The source code and configuration files will be publicly released upon acceptance to facilitate reproducibility and future research.

REFERENCES

Tongtong Cheng, Rongzhen Li, Yixin Xiong, Tao Zhang, Jing Wang, and Kai Liu. MCAM: multimodal causal analysis model for ego-vehicle-level driving video understanding. In *ICCV*, pp. 5479–5489. IEEE, 2025.

Chenxu Dang, Haiyan Liu, Jason Bao, Pei An, Xinyue Tang, An Pan, Jie Ma, Bingchuan Sun, and Yan Wang. Sparseworld: A flexible, adaptive, and efficient 4d occupancy world model powered by sparse and dynamic queries. In *AAAI*, pp. 3497–3505. AAAI Press, 2026.

Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. Vectornet: Encoding HD maps and agent dynamics from vectorized representation. In *CVPR*, pp. 11522–11530. Computer Vision Foundation / IEEE, 2020.

Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, Lewei Lu, Xiaosong Jia, Qiang Liu, Jifeng Dai, Yu Qiao, and Hongyang Li. Planning-oriented autonomous driving. In *CVPR*, pp. 17853–17862. IEEE, 2023.

Zhao Huang, Bowen Zhang, Zhongzhu Li, and Di Lin. Occdriver: Future occupancy guided dual-branch trajectory planner in autonomous driving. In *ICLR*, pp. 141940–141962, 2026.

Wonjune Kim, In-Jae Lee, Sihwan Hwang, Sanmin Kim, and Dongsuk Kum. Class-distribution guided active learning for 3d occupancy prediction in autonomous driving. *IEEE Robotics Autom. Lett.*, 11(6):6999–7006, 2026.

Xiang Li, Pengfei Li, Yupeng Zheng, Wei Sun, Yan Wang, and Yilun Chen. Semi-supervised vision-centric 3d occupancy world model for autonomous driving. In *ICLR*, 2025.

Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng Dai. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *ECCV (9)*, volume 13669 of *Lecture Notes in Computer Science*, pp. 1–18. Springer, 2022.

Chen Min, Dawei Zhao, Liang Xiao, Jian Zhao, Xinli Xu, Zheng Zhu, Lei Jin, Jianshu Li, Yulan Guo, Junliang Xing, Liping Jing, Yiming Nie, and Bin Dai. Driveworld: 4d pre-trained scene understanding via world models for autonomous driving. In *CVPR*, pp. 15522–15533. IEEE, 2024.

Jiquan Ngiam, Vijay Vasudevan, Benjamin Caine, Zhengdong Zhang, Hao-Tien Lewis Chiang, Jeffrey Ling, Rebecca Roelofs, Alex Bewley, Chenxi Liu, Ashish Venugopal, David J. Weiss, Ben Sapp, Zhifeng Chen, and Jonathon Shlens. Scene transformer: A unified architecture for predicting future trajectories of multiple agents. In *ICLR*. OpenReview.net, 2022.

Hao Shi, Song Wang, Jiaming Zhang, Xiaoting Yin, Guangming Wang, Jianke Zhu, Kailun Yang, and Kaiwei Wang. Offboard occupancy refinement with hybrid propagation for autonomous driving. *IEEE Trans. Intell. Transp. Syst.*, 26(10):16001–16016, 2025.

Shaoshuai Shi, Li Jiang, Dengxin Dai, and Bernt Schiele. Motion transformer with global intention localization and local movement refinement. In *NeurIPS*, 2022.

Chonghao Sima, Katrin Renz, Kashyap Chitta, Li Chen, Hanxue Zhang, Chengen Xie, Jens Beißwenger, Ping Luo, Andreas Geiger, and Hongyang Li. Drivelm: Driving with graph visual question answering. In *ECCV (52)*, volume 15110 of *Lecture Notes in Computer Science*, pp. 256–274. Springer, 2024.

Xiaoyu Tian, Tao Jiang, Longfei Yun, Yucheng Mao, Huitong Yang, Yue Wang, Yilun Wang, and Hang Zhao. Occ3d: A large-scale 3d occupancy prediction benchmark for autonomous driving. In *NeurIPS*, 2023.

Wenwen Tong, Chonghao Sima, Tai Wang, Li Chen, Silei Wu, Hanming Deng, Yi Gu, Lewei Lu, Ping Luo, Dahua Lin, and Hongyang Li. Scene as occupancy. In *ICCV*, pp. 8372–8381, 2023.

Song Wang, Jiawei Yu, Wentong Li, Wenyu Liu, Xiaolu Liu, Junbo Chen, and Jianke Zhu. Not all voxels are equal: Hardness-aware semantic scene completion with self-distillation. In *CVPR*, pp. 14792–14801. IEEE, 2024.

Julong Wei, Shanshuai Yuan, Pengfei Li, Qingda Hu, Zhongxue Gan, and Wenchao Ding. Occllama: An occupancy-language-action generative world model for autonomous driving. *CoRR*, abs/2409.03272, 2024.

Yi Wei, Linqing Zhao, Wenzhao Zheng, Zheng Zhu, Jie Zhou, and Jiwen Lu. Surroundocc: Multi-camera 3d occupancy prediction for autonomous driving. In *ICCV*, pp. 21672–21683. IEEE, 2023.

Yu Yang, Jianbiao Mei, Yukai Ma, Siliang Du, Wenqing Chen, Yijie Qian, Yuxiang Feng, and Yong Liu. Driving in the occupancy world: Vision-centric 4d occupancy forecasting and planning via world models for autonomous driving. In *AAAI*, pp. 9327–9335. AAAI Press, 2025.

Zetong Yang, Li Chen, Yanan Sun, and Hongyang Li. Visual point cloud forecasting enables scalable autonomous driving. In *CVPR*, pp. 14673–14684. IEEE, 2024.

Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris Kitani. Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. In *ICCV*, pp. 9793–9803. IEEE, 2021.

Yunpeng Zhang, Zheng Zhu, and Dalong Du. Occformer: Dual-path transformer for vision-based 3d semantic occupancy prediction. In *ICCV*, pp. 9399–9409. IEEE, 2023.

Yunpeng Zhang, Deheng Qian, Ding Li, Yifeng Pan, Yong Chen, Zhenbao Liang, Zhiyao Zhang, Yingzong Liu, Jianhui Mei, Maolei Fu, Yun Ye, Zhujin Liang, Yi Shan, and Dalong Du. Graphad: Interaction scene graph for end-to-end autonomous driving. In *IJCAI*, pp. 2422–2430. ijcai.org, 2025.

Wenzhao Zheng, Weiliang Chen, Yuanhui Huang, Borui Zhang, Yueqi Duan, and Jiwen Lu. Occworld: Learning a 3d occupancy world model for autonomous driving. In *ECCV (13)*, volume 15071 of *Lecture Notes in Computer Science*, pp. 55–72. Springer, 2024.

APPENDIX

A IMPLEMENTATION AND REPRODUCIBILITY DETAILS

This section provides the implementation details omitted from the main paper for concision. We first describe the common forecasting interface and experimental setup, followed by the module-specific architectures, training objectives, calibration procedures, and inference configurations of ForecastEdit. Unless otherwise stated, all components and controlled comparisons use the same frozen forecasting anchor and evaluation protocol.

A.1 COMMON SETUP AND FORECASTING INTERFACE

Frozen Forecasting Interface. All ForecastEdit variants are built on the same frozen SparseWorld forecasting checkpoint. For each input sequence, the anchor produces the reference semantic occupancy forecast B together with the intermediate query/refine states, semantic logits, regression and velocity predictions, and renderer outputs required by the editing modules. EGBE additionally reads the point-to-voxel provenance retained by the frozen renderer, while ESE operates on the corresponding forecasting states and camera-derived predicted object tracks. The forecasting backbone and renderer remain unchanged throughout all ForecastEdit training and evaluation. We choose SparseWorld, the SOTA model within the query-based 4D occupancy forecasting branch, as the sole anchor to isolate the effect of forecast editing from variations across forecasting models. This controlled setting directly evaluates whether structured editing can further improve an already strong world forecast.

Shared Evaluation Protocol. We follow the SparseWorld protocol on Occ3D-nuScenes and evaluate future occupancy at 1, 2, and 3 s horizons. All ForecastEdit variants, ablations, and controlled correction baselines use the same reference forecasts, validation samples, renderer, and metric implementation. Semantic performance is computed from the 17 non-empty occupancy classes using per-horizon mIoU, while occupied IoU evaluates class-agnostic occupancy quality. Unless explicitly stated otherwise, all semantic forecasting comparisons cover the same 4,219 validation samples from 150 scenes.

Implementation Environment. All experiments are implemented with Python 3.9.25 and PyTorch 2.0.1 under CUDA 11.8. The detection and 3D perception stack uses MMCV 1.7.0, MMDetection 2.28.0, and MMDetection3D 1.0.0rc4. Training, intermediate prediction export, and supplementary measurements are conducted on NVIDIA L20 GPUs with 48 GB of memory.

A.2 EVIDENCE-GUIDED BOUNDARY EDITING DETAILS

This section provides the implementation details of Evidence-Guided Boundary Editing (EGBE) omitted from the main paper for concision. Following Sec. 4.2, EGBE operates on occupied states in the frozen reference forecast and determines whether each state should be preserved or retracted using renderer provenance and local spatial support.

Provenance and Structural Evidence. For each occupied candidate voxel, we retain the refine points that either directly contribute to the rendered state or provide valid same-class support through the frozen renderer. Each contributor preserves its frozen semantic response and local geometry relative to the candidate-voxel center. The semantic representation includes compressed class responses, the response associated with the candidate class, the top response, the top-1/top-2 margin, semantic entropy, and the maximum logit. Geometric evidence consists of the relative 3D displacement and Euclidean distance from the contributor to the candidate center. At the candidate level, we additionally retain the predicted semantic class and a 16-dimensional structural descriptor that summarizes contributor support, direct provenance, local neighborhood consistency, and source confidence. The complete evidence schema is summarized in Table 5.

The candidate class is represented by a learned semantic embedding. EGBE does not use raw image features, sparse-query features, absolute point or voxel coordinates, or horizon embeddings. This design keeps the retraction decision focused on local provenance and spatial support rather than introducing additional scene-level prediction signals.

Table 5: Evidence representation used by EGBE. Point-level features characterize frozen renderer provenance, while candidate-level features summarize the local structural support of each occupied reference voxel.

Level	Evidence Group	Included Information
Point	Semantic response	Compressed class responses, candidate-class response, top response, top-1/top-2 margin, entropy, and maximum logit.
Point	Relative geometry	Relative 3D displacement and distance to the candidate-voxel center.
Candidate	Contributor support	Contributor count and selected-slot coverage.
Candidate	Direct provenance	Direct same-class source, nearest support distance, and renderer-support indicator.
Candidate	Local structure	Same-class ratio, occupied ratio, and semantic diversity at radii $r = 1$ and $r = 2$.
Candidate	Source confidence	Maximum, mean, and standard deviation of source-class responses, together with contributor margins.

Table 6: Architecture and training configuration of EGBE. The boundary editor is trained exclusively from frozen renderer evidence while the underlying forecasting model remains fixed.

Setting	Value
Contributor / candidate dimension	96
Provenance projection dimension	24
Evidence grid	$200 \times 200 \times 16$
Local 3D dilation rates	$1, 2, 4$
Training patch	$48 \times 48 \times 16$
Supervised core	$16 \times 16 \times 16$
Training scenes	560
Temporal anchors per scene	8
Epochs	1
Optimizer	AdamW
Learning rate	1×10^{-4}
Weight decay	1×10^{-4}
Candidate batch size	4,096
Gradient clipping	5.0
Utility-BCE weight	1.0
Pairwise-ranking weight	1.0
Max. samples per ranking side	64

Evidence Aggregation and Local 3D Context. Each contributor is encoded into a 96-dimensional feature e_i, while the candidate statistics and class embedding are mapped to a 96-dimensional query q_v. The candidate-conditioned aggregation in Eq. (5) therefore uses attention dimension $d = 96$. We additionally introduce a learnable null contributor to provide a stable aggregation reference when renderer support is sparse. The resulting provenance representation is projected to 24 dimensions using a LayerNorm–Linear–SiLU block and scattered onto a $200 \times 200 \times 16$ evidence grid. An additional occupancy indicator distinguishes a zero-valued candidate representation from grid locations without an EGBE candidate.

The local context network starts with a $3 \times 3 \times 3$ convolutional stem followed by three residual blocks with dilation rates 1, 2, and 4. A final $1 \times 1 \times 1$ convolution produces the voxel-wise retraction score, with normalization applied only along the channel dimension. During training, the network processes $48 \times 48 \times 16$ context patches, while only the central $16 \times 16 \times 16$ core contributes supervised candidates. The surrounding region provides spatial context without adding additional training targets. At inference, the same local network is applied to the complete evidence grid.

Retraction Learning. EGBE is trained on 560 source-training scenes with eight temporal anchors deterministically sampled from each scene. Only occupied candidates inside the supervised core contribute to the utility-aware retraction objective in Eq. (8). The class-dependent retraction consequences derived in the main paper are normalized within each semantic class before weighting the

Table 7: Frozen class-specific execution routes used by EGBE. Routes and retraction thresholds are determined exclusively from source-side reliability evidence before formal evaluation.

Route	Execution Policy	Assigned Classes
`warmstart`	Fully fitted boundary editor	All semantic classes not listed below.
`reference`	Conservative frozen checkpoint	Bicycle, construction vehicle, motorcycle, pedestrian.
`identity`	Always preserve reference	Traffic cone.

Table 8: Object representation and structured dynamics architecture used by ESE. Camera-derived tracks are temporally encoded, relationally contextualized, and conditioned on ontology states before multimodal future decoding.

Group	Component	Specification
Input	Box center	$(x/40,\ y/40,\ z/6.4)$
Input	Box size	$(l/10,\ w/5,\ h/4)$
Input	Yaw / velocity	$(\sin\theta, \cos\theta)$ and $(v_x/20,\ v_y/20,\ v_z/10)$
Input	Confidence	Detection and class probabilities
Input	Class / age / time	class_id$/9$, clip(age$, 0, 20)/20$, and clip$(\Delta t/2.5, -1, 0)$
Input	Pairwise relation	8-D relative position, BEV distance, planar velocity, class, and identity features
Encoder	Track encoder	Two-layer MLP, $16 \to 256$, with LayerNorm and GELU
Encoder	Temporal Transformer	3 layers, 8 heads, with observation masking
Encoder	Relation Graph	4 layers, 8 heads, with pairwise features as attention bias
Structured states	Ontology heads	4 object-state heads and 16 pairwise-relation heads
Decoder	Future dynamics	6 motion modes, 3-layer MLP, and 6 future steps
Decoder	Spatial support	Differentiable oriented-box renderer and local flow head

binary objective. The pairwise-ranking term additionally encourages beneficial retractions to receive higher scores than candidates that should be preserved. For each ranking side, at most 64 candidates are sampled. The main architecture and optimization settings are summarized in Table 6.

Source-Side Calibration and Class Routing. After training, class-specific retraction thresholds are fitted on an independent set of 2,107 source-side calibration records. Training and calibration are kept separate, and both the learned boundary editor and all calibrated thresholds are frozen before formal evaluation. To account for class-dependent reliability, the final EGBE policy assigns each semantic class to one of three frozen execution routes: `warmstart`, `reference`, or `identity`. The `warmstart` route uses the boundary editor fitted on the full source-training split, whereas the `reference` route uses a previously frozen, more conservative boundary-editor checkpoint. The `identity` route disables retraction and always preserves the reference prediction for that class. All routing decisions are determined from scene-disjoint source-side class reliability evidence before evaluation and are never selected using validation performance.

The final class assignments are summarized in Table 7. Traffic cone uses the `identity` route; bicycle, construction vehicle, motorcycle, and pedestrian use the `reference` route; all remaining semantic classes use `warmstart`. Together with the class-specific thresholds, these routes remain fixed throughout evaluation and inference.

A.3 Executable Semantic Editing Details

This section provides the implementation details of Executable Semantic Editing (ESE) omitted from the main paper for concision. We follow the computation flow from camera-derived object tracks to ontology-conditioned future dynamics, voxel-level edit candidates, utility-aware selection, and semantic commitment.

Object Representation and Ontology. The object pathway operates on camera-derived predicted tracks rather than ground-truth object trajectories. For each temporal anchor, we retain at most 64 tracked objects, use five historical frames, and predict six future steps. Historical boxes are trans-

Table 9: Ontology vocabulary used by ESE. Object-state variables directly condition multimodal future dynamics, while pairwise relations provide structured relational supervision.

Group	States / Relations
Existence	Observed, Temporarily Missing, Exited, Uncertain
Kinematics	Static, Moving, Accelerating, Decelerating, Constant Velocity, Turning Left, Turning Right, Reversing, Uncertain
Interaction	Following, Overtaking, Merging, Cutting In, Crossing, Yielding, Stopping for Agent, Free Motion, Uncertain
Visibility	Fully Visible, Partially Occluded, Heavily Occluded, Uncertain
Spatial relations	front-of, behind, left-of, right-of, near, overlapping
Motion relations	approaching, moving-away, relative-static, heading-aligned, heading-opposed
Interaction relations	following, cutting-in-front-of, trajectory-conflict, blocks-corridor, yielding-to

formed into the current ego-coordinate frame using the corresponding ego poses before feature construction. Each object state is represented by a 16-dimensional feature vector containing normalized geometry, motion, confidence, semantic identity, observation age, and temporal offset. A binary observation mask distinguishes valid observations from missing states in the track history. For each directed object pair, we additionally construct an 8-dimensional relation feature containing relative 3D position, BEV distance, relative planar velocity, a same-class indicator, and an identity-edge indicator. The input representation and structured object architecture are summarized in Table 8.

The Temporal Transformer first encodes the historical evolution of each object, after which the Relation Graph incorporates directed interactions with surrounding agents. Ontology heads operate on the resulting relational representation and predict four object-state distributions—existence, kinematics, interaction, and visibility—together with pairwise spatial, motion, and interaction relations. The ontology vocabulary used throughout ESE is summarized in Table 9. As defined in Eq. (10), the four object-state distributions are projected back into the continuous representation before future decoding, so they directly condition predicted object dynamics rather than serving only as auxiliary supervision.

Ontology-Conditioned Multimodal Future Dynamics. The future decoder predicts six motion modes for each tracked object. For every mode and future step, it predicts incremental center and yaw, velocity, size residuals, existence probability, uncertainty, local occupancy, and flow. The predicted log-scale uncertainty is clipped to $[-5, 4]$.

We additionally impose soft consistency constraints over the ontology predictions. For each implication rule, the penalty is

$$\mathcal{L}_{\text{rule}} = \max(0,\, p_{\text{premise}} - p_{\text{consequence}})\,. \tag{14}$$

The direct future-dynamics objective is

$$\mathcal{L}_{\text{dyn}} = \mathcal{L}_{\text{traj}} + 0.5\mathcal{L}_{\text{exist}} + \mathcal{L}_{\text{occ}} + 0.5\mathcal{L}_{\text{flow}}, \tag{15}$$

where $\mathcal{L}_{\text{traj}}$ is the multimodal trajectory mixture NLL, $\mathcal{L}_{\text{exist}}$ is the existence BCE, $\mathcal{L}_{\text{occ}}$ is the local occupancy BCE, and $\mathcal{L}_{\text{flow}}$ is a Smooth-L_1 flow loss. Ontology supervision is defined as

$$\mathcal{L}_{\text{onto}} = 0.2\mathcal{L}_{\text{state}} + 0.2\mathcal{L}_{\text{rel}} + 0.05\mathcal{L}_{\text{rule}} + 0.1\mathcal{L}_{\text{cal}}, \tag{16}$$

where $\mathcal{L}_{\text{state}}$ and $\mathcal{L}_{\text{rel}}$ supervise ontology states and relations, while $\mathcal{L}_{\text{cal}}$ improves probability calibration. The temporal encoder, relation graph, ontology heads, state projector, and multimodal decoder are jointly optimized within the object pathway.

Voxel Support and Utility-Aware Edit Selection. For each predicted motion mode, the future center, size, and yaw are projected onto the 0.4 m Occ3D voxel grid using a differentiable oriented-box renderer. Signed oriented distance to the predicted box is converted through a sigmoid function into local occupancy support. Candidate construction combines rendered object support with query-level evidence, historical support, all six motion modes, and temporal evidence across future horizons. Each atomic record stores the sample identifier, future horizon, voxel index, and proposed semantic class.

A candidate may induce a fill, restoration, local deletion, or foreground relabeling operation. Here, *restoration* denotes writing a foreground state back to a voxel previously retracted by EGBE and

Table 10: Evidence used for utility-aware executable edit selection. The selector combines proposal-level, local spatial, reference-forecast, and temporal evidence before deciding whether a candidate should overwrite the current state.

Evidence Family	**Representative Information**
Proposal evidence	Proposal rank, support length, priority, and proposal-level confidence
Cell support	Support count, prefix support, score statistics, and competing semantic states
Local geometry	Relative object coordinates and local new-class, old-class, and free-space ratios
Forecast support	Query hit and score together with the current reference state
Action semantics	Proposal family, edit type, and multimodal motion support
Cross-model reliability	Agreement and support consistency between ontology-conditioned and generic dynamics
Temporal consistency	Cross-horizon occurrence and neighboring support across future steps

therefore does not constitute an additional primitive relative to the original reference forecast B. The evidence supplied to the edit selector is summarized in Table 10.

For each candidate, source-side supervision assigns one of the three editing consequences introduced in Sec. 4.3: *repair*, *damage*, or *remain-wrong*. Class-conditional source weights encode the relative consequences of these outcomes for different semantic classes and are fixed after training. At each voxel–horizon coordinate, candidates are compared using the utility rule in Eq. (11), with the no-op action serving as the preservation baseline. When multiple candidates address the same coordinate, only the highest-utility admissible state is committed; otherwise, the selector executes a no-op. Consequently, each voxel–horizon pair receives at most one atomic edit before the selected revisions are scattered onto F to produce H.

Semantic Commitment and Differentiable Training. The semantic commitment branch is parameterically separated from the object pathway and operates exclusively on detached states from the frozen SparseWorld forecaster. Its input concatenates the future query feature, original semantic logits, regression and velocity offsets, metric point, and horizon embedding. The resulting 297-dimensional representation is processed by a LayerNorm–Linear–SiLU–Linear–SiLU residual head that outputs a 17-dimensional semantic correction. The residual is bounded as

$$\Delta s = 6\tanh(z/6), \qquad s' = \mathrm{stopgrad}(s) + \Delta s. \tag{17}$$

The final projection layer is zero-initialized so that training begins from $\Delta s = 0$ and initially reproduces the frozen semantic prediction. The branch contains 147,155 trainable parameters, while all parameters of the forecasting anchor remain frozen.

Because inference uses the original hard renderer, semantic commitment is trained with the straight-through bridge

$$q_{\mathrm{ST}} = q_{\mathrm{soft}} + \mathrm{stopgrad}\left(q_{\mathrm{hard}} - q_{\mathrm{soft}}\right), \tag{18}$$

where q_{hard} and q_{soft} denote the corresponding hard and differentiable rendered quantities. The forward pass therefore exactly follows the hard renderer, while the soft renderer supplies the backward gradient. The renderer temperature is set to 24, and geometric coordinates, voxel indexing, and morphological operations are excluded from the surrogate backward path.

Class-conditional supervision is constructed exclusively from the source-training population. For each future horizon, we accumulate an 18×18 transition matrix between frozen reference predictions and source targets. These statistics determine fixed positive coefficients for recovering each target class and penalties for incorrectly expanding other states into that class. The coefficients are frozen before residual-head optimization and influence semantic commitment only through the learned correction parameters.

To reduce concentration on a narrow subset of scene patterns, the 700 source scenes are deterministically partitioned into eight fixed groups of sizes $88/88/88/88/87/87/87/87$. After each complete group cycle, the group-level excess risk relative to the frozen reference is updated as

$$\bar{e}_g \leftarrow 0.9\bar{e}_g + 0.1\left(e_g - \mathrm{mean}_{g'}\, e_{g'}\right), \tag{19}$$

Table 11: Architecture and training configuration of semantic commitment. All inputs are detached from the frozen forecaster, and only the lightweight residual head is optimized in this stage.

Group	Setting	Value
Input	Frozen query feature	256
Input	Semantic logits	17
Input	Regression / velocity offsets	3 / 2
Input	Metric point / horizon embedding	3 / 16
Input	Total dimension	297
Architecture	Residual head	LN–Linear–SiLU–Linear–SiLU
Architecture	Trainable parameters	147,155
Architecture	Residual bound	$6\tanh(z/6)$
Training	Source scenes / scene groups	700 / 8
Training	Epochs / updates	1 / 2,472
Training	Global batch size	8
Training	Random seed	17
Training	Optimizer / learning rate	AdamW / 1×10^{-4}
Training	Weight decay / gradient clipping	0.01 / 5.0
Training	Group temperature / EMA	$0.25/0.9$
Training	Renderer temperature	24.0

Table 12: Stage-wise optimization of ForecastEdit. The forecasting anchor remains frozen throughout, while each editing stage is optimized against fixed forecasting states before formal evaluation.

Stage	Trainable Component	Primary Supervision	Output
Reference forecast	None	Frozen SparseWorld prediction	B
EGBE	Boundary editor	Retraction utility and pairwise ranking	F
ESE object path	Object dynamics and edit selector	Dynamics, ontology, and edit utility	H
Semantic commitment	Residual semantic head	Class-conditional semantic supervision	P
Composition	None	Deterministic reference-aware rule	C

where e_g denotes the current excess risk of group g. The sampling distribution for the following cycle is

$$w_g = \mathrm{softmax}_g\left(\bar{e}_g/0.25\right). \tag{20}$$

The architecture and optimization settings of semantic commitment are summarized in Table 11.

After training, the class-conditional coefficients, group-sampling policy, semantic residual checkpoint, and rendering configuration are frozen. Inference executes only the learned editing modules and the original hard renderer, without maintaining source-population statistics or updating model parameters online.

A.4 Training, Calibration, and Inference Details

Stage-Wise Optimization. ForecastEdit is optimized stage-wise while the underlying SparseWorld forecasting model remains frozen throughout all stages. EGBE is trained first from frozen renderer provenance using the utility-aware retraction objective in Eq. (8). After EGBE training and calibration, its boundary model, class-specific thresholds, and execution routes are fixed. The ESE object pathway is then optimized to model temporal and relational representations, ontology-conditioned multimodal dynamics, local occupancy support, and utility-aware edit selection. Finally, semantic commitment is trained on detached forecasting states, with only the lightweight residual head updated. The resulting stages therefore learn complementary revisions relative to the same frozen reference forecast without end-to-end finetuning of the forecasting anchor. Table 12 summarizes the optimization dependencies and outputs of the complete framework.

Table 13: Source-side quantities used for training and calibration. All quantities are estimated without validation/test feedback and are either frozen for inference or used only during training.

Quantity	Source / Purpose	Evaluation-Time Status
EGBE utility weights	Source-training class statistics	Encoded in trained editor
EGBE retraction thresholds	Independent source calibration	Frozen and applied
EGBE class routes	Scene-disjoint source reliability	Frozen and applied
ESE consequence weights	Source-side edit outcomes	Fixed with edit selector
Semantic class coefficients	Source transition statistics	Training supervision only
Scene-group sampling state	Source-training scene groups	Training only

Source-Only Calibration and Freezing. All learned parameters, source-derived statistics, calibration thresholds, and routing decisions are determined before formal evaluation, without validation- or test-set feedback. For EGBE, training uses the source-training population, while class-specific retraction thresholds are fitted separately on 2,107 source-side calibration records. The class-routing policy is determined from scene-disjoint source-side reliability evidence, and both routes and thresholds are frozen before evaluation. For executable edit selection, the three editing consequences—*repair*, *damage*, and *remain-wrong*—and their class-conditional weights are constructed from source-side supervision and fixed after training. For semantic commitment, an 18×18 transition matrix is accumulated independently at each future horizon from frozen reference predictions and source targets. These transition statistics define the class-conditional supervision used to train the residual head and do not require evaluation-set statistics.

Scene-group reweighting is used only during semantic-commitment training. The 700 source scenes are deterministically partitioned into eight fixed groups, and their sampling weights are updated from excess risk relative to the frozen reference according to Eqs. (19) and (20). After optimization, the resulting residual checkpoint and all associated source-derived training quantities are fixed. Table 13 summarizes the role and evaluation-time status of the principal source-side quantities.

Inference Protocol. At inference, the frozen SparseWorld model first produces the reference forecast B together with the intermediate states required by the editing modules. EGBE applies its frozen provenance encoder, local 3D network, class route, and calibrated threshold to retract unsupported occupancy and obtain F. The ESE object pathway then predicts ontology-conditioned future dynamics, constructs voxel-level candidates, resolves competing edits at each voxel–horizon coordinate, and applies the selected revisions to produce H. In parallel, semantic commitment processes detached forecasting states with the frozen residual head and uses the original hard renderer to produce the semantic candidate P. The straight-through bridge in Eq. (18) is used only for training and is not required during inference. The final edited forecast C is obtained from B, H, and P using the deterministic composition rule in Eq. (13).

All editing parameters, class routes, and calibration thresholds remain fixed throughout inference. The scene-group reweighting mechanism and source-population statistics used during training are not maintained at test time. ForecastEdit therefore requires neither target annotations nor online calibration, source-statistic updates, or parameter adaptation during inference.

B Additional Results and Analysis

B.1 Controlled Comparison with Generic Correction

Matched Generic Correction Baselines. We construct three generic correction baselines to test whether the gains of the structured executable editing introduced in Sec. 4.3 can be explained by additional post-hoc correction capacity or conventional semantic supervision. All variants use the same frozen SparseWorld anchor, fixed EGBE output, object-candidate source, renderer, data split, and evaluation protocol. They further share the same generic object pathway and semantic-head architecture, while differing only in the supervision applied to the semantic correction branch. The reported prediction corresponds to the complete generic object pathway together with its semantic correction head rather than an isolated residual module.

Table 14: Matched generic correction baselines. All variants share the same forecasting anchor, EGBE output, correction architecture, training budget, and evaluation protocol, while differing in semantic supervision.

Variant	Semantic Supervision	Controlled Factor
Point-level supervision	Original prediction objectives without additional voxel-level cross-entropy	Generic correction capacity
Voxel CE	Standard cross-entropy on rendered voxel predictions with uniform weighting	Direct voxel supervision
Class-weighted voxel CE	Voxel cross-entropy weighted by normalized inverse-square-root source class frequency	Static class balancing

Both voxel-level variants supervise the 17 foreground classes together with the `free` state, while evaluation follows the same 17-class semantic mIoU used throughout the paper. For voxel CE, the global loss coefficient is calibrated from source-side gradient scales and fixed before formal training. For class-weighted voxel CE, class weights are computed independently at each horizon from source-training frequencies and are likewise fixed before evaluation. No validation-set class frequencies or evaluation results are used to tune either baseline.

The three semantic heads use the same 297-dimensional input and 147,155 trainable parameters as the semantic commitment branch. They also share 700 source scenes, 2,472 parameter updates, a global batch size of 8, random seed 17, and the remaining optimization settings. Consistent with the training protocol in Sec. 4.5, the forecasting anchor remains frozen and all source-derived quantities are fixed before formal evaluation. The comparison controls for model capacity, optimization budget, forecasting anchor, and evaluation protocol while varying the semantic learning objective.

Controlled Comparison. As reported in Sec. 5.3, all three generic correctors improve over the frozen anchor, confirming that post-hoc correction itself is beneficial. Direct voxel supervision and class weighting provide additional gains, showing that both final-state supervision and class imbalance are relevant to semantic forecasting. However, the strongest generic alternative remains below ForecastEdit under the matched setting. This residual gap indicates that the improvement cannot be explained by correction capacity, direct voxel supervision, or static class balancing alone, motivating the component-level analysis below.

B.2 Component and Operation-Wise Analysis

We further examine where the advantage of structured executable editing arises by decomposing ESE at both the component and operation levels. The former isolates the contributions of the object-editing pathway and semantic commitment, while the latter examines which reference-relative editing operations benefit most from the structured design.

Component Replacement. The generic-correction comparison in Appendix B.1 replaces multiple ESE components simultaneously. We therefore perform controlled component replacement while fixing the SparseWorld anchor, EGBE output, and final composition rule. Starting from the full ESE, we replace either the ontology-conditioned object pathway or semantic commitment while retaining the other trained component. This isolates the contribution of each pathway without changing the reference forecast or output-composition mechanism.

As shown in Table 15, replacing the ontology-conditioned object pathway reduces the gain from $+1.62$ to $+1.42$ mIoU, showing that structured object identity, relations, and multimodal dynamics contribute beyond a generic proposal pathway. Conversely, retaining the ESE object pathway while replacing semantic commitment yields at most $+1.37$ mIoU, even with class-weighted voxel supervision. Thus, both pathways contribute independently, and neither generic object modeling nor static class-balanced semantic correction fully reproduces the complete ESE gain. The larger degradation from replacing semantic commitment further indicates that final reference-conditioned write decisions are important in addition to constructing structured object-level proposals.

Operation-Wise Decomposition. We next group committed revisions by their net transition relative to the original reference forecast B into four operation families: *Preserve*, *Retract*, *Fill*, and *Relabel*. Preserve retains the reference semantic state, including restoring a foreground state that

Table 15: Component replacement analysis of ESE. With the SparseWorld anchor, EGBE output, and final composition fixed, we replace either the object-editing pathway or semantic commitment while retaining the other trained component. All values report ΔmIoU relative to the SparseWorld anchor.

Variant	Replaced Component	ΔmIoU
ForecastEdit	None	+1.62
Generic Object Path + ESE Semantic	Object pathway	+1.42
ESE Object Path + Point Sup.	Semantic commitment	+1.29
ESE Object Path + Voxel CE	Semantic commitment	+1.30
ESE Object Path + Weighted Voxel CE	Semantic commitment	+1.37

Table 16: Operation-wise decomposition of forecast editing. Starting from the fixed EGBE prediction, we replay only one reference-relative operation family at a time while suppressing the others. All values report ΔmIoU relative to the SparseWorld anchor.

Editing Mechanism	EGBE	Fill	Preserve	Retract	Relabel
Generic Corrector: Point Sup.	+0.97	+1.16	+1.08	+1.04	+1.15
Generic Corrector: Voxel CE	+0.97	+1.17	+1.08	+1.04	+1.15
Generic Corrector: Weighted Voxel CE	+0.97	+1.21	+1.08	+1.05	+1.19
ESE (Ours)	+0.97	**+1.37**	+1.08	+1.05	**+1.35**

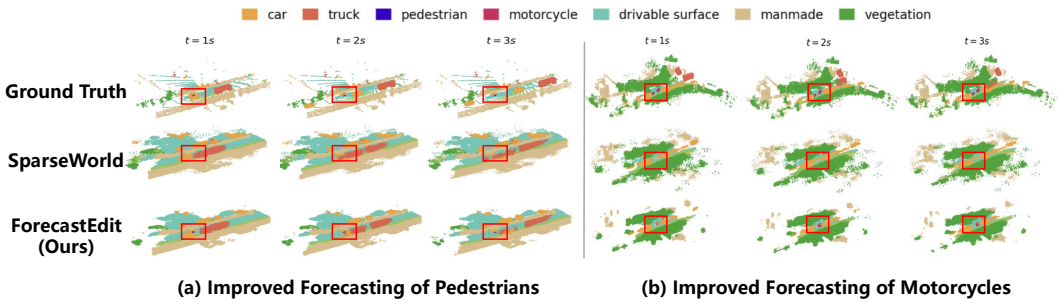

Figure 5: Qualitative recovery of small dynamic objects. Compared with SparseWorld, ForecastEdit maintains more coherent pedestrian (a) and motorcycle (b) occupancy across 1–3 s horizons. Red boxes highlight the corresponding future object regions.

was temporarily retracted by EGBE; Retract changes occupied reference states to `free`; Fill introduces foreground occupancy at reference-free locations; and Relabel changes one foreground semantic class to another. This reference-relative definition is consistent with the editing primitives used throughout the main paper and avoids treating restoration as an additional primitive.

Starting from the fixed EGBE prediction, we replay each operation family independently while suppressing the others. Therefore, all methods share the same boundary-edited input, isolating differences in how each semantic operation is constructed and committed.

As shown in Table 16, the largest separation between ESE and generic correction occurs for *Fill* and *Relabel*, where introducing or revising semantic content requires supported alternatives to the reference forecast. In contrast, *Retract* remains largely determined by the preceding EGBE stage, while *Preserve* exhibits little variation across correction mechanisms. This pattern is consistent with the intended division of labor: EGBE primarily removes unsupported occupancy, whereas ESE contributes structured recovery and semantic reassignment. Together with the component replacement results, the analysis shows that ForecastEdit gains arise from both structured proposal construction and selective semantic commitment rather than from a single correction operation.

B.3 Evidence Grounding and Qualitative Analysis

The analysis in Fig. 4 shows that ForecastEdit grounds different revisions in complementary evidence sources, including boundary provenance, object-level future support, and semantic residuals. Furthermore, we complement this mechanism-level analysis with qualitative examples that illustrate how such structured revisions translate into persistent improvements in future occupancy.

Small-Object Forecast Recovery. Fig. 5 presents representative examples for pedestrians and motorcycles, two small dynamic categories for which forecasting errors become increasingly pronounced over longer horizons. SparseWorld tends to weaken or miss the corresponding foreground occupancy as the prediction horizon increases, whereas ForecastEdit maintains more coherent states that better align with the ground truth across 1–3 s. The improvement is temporally persistent rather than confined to an isolated horizon, consistent with ESE using structured object dynamics to propagate supported foreground states into the future. These cases complement the class-wise analysis in the main paper by showing how gains on small dynamic categories arise from localized and temporally coherent forecast revisions.