Exposing the Long-tail in Embodied Urban Navigation via Scalable Learning from In-the-Wild Videos

Supplementary Material

Contents

A. Data Sources, Human Review, Licensing, and Privacy . . . 1
B. Setup of WILD-LongTail. . . . 1
C. Feature representations and Rarity . . . 2
D. Reflection-Based Failure Attribution . . . 4
E. Additional Experimental Results . . . 11

A. Data Sources, Human Review, Licensing, and Privacy

A.1. Data Sources

Our dataset is constructed from publicly accessible urban walking videos hosted on YouTube, following the web-scale data collection paradigm adopted by CityWalker. These videos capture diverse real-world urban environments and provide rich visual observations for embodied navigation research.

A.2. Human Review and Data Quality

To improve data quality, we commissioned a professional external data-annotation company to review video-derived navigation episodes and their planning annotations. Reviewers inspected temporally ordered first-person frames and recorded why a proposed plan was inconsistent with the visible scene, together with a suggested correction. Figures 1 and 2 show representative review records. The first identifies a turn that would direct the agent into a vehicle lane; the second identifies a reversing plan that overlooks pedestrians and vehicles behind the agent.

A.3. Data Release and Privacy

To respect the rights of original content creators and reduce potential privacy risks, we do not redistribute the original videos or raw video frames. We will open-source the annotation tools used across the entire dataset. We will publicly release 10,000 human-reviewed WILD-LongTail episodes in the form of derived navigation annotations and source-video links/IDs. The original videos remain hosted on their respective platforms under the control of their content owners. Our release does not include copies of the source videos.

B. Setup of WILD-LongTail.

B.1. Dataset Partition

WILD-LongTail is constructed from our large-scale in-the-wild video collection as a diagnostic benchmark for systematically characterizing the long-tail distribution of urban navigation. It is disjoint from the conventional video-learning splits at the source-video level and comprises two components: a Long-Tail Mining Set containing 287,820 video-derived navigation episodes and a held-out Long-Tail Test Set containing 104,655 episodes with fixed identifiers. The remaining video episodes are used as additional training data for policy learning or model adaptation. The Long-Tail Mining Set is used to estimate perception-motion feature frequencies, identify distributionally rare patterns and model-dependent high-error cases, conduct reflection-based failure analysis, and construct the incremental fine-tuning subsets. Specifically, `+Random` contains 15K randomly sampled mining episodes, `+Hard` contains 7.5K mined hard episodes, and `+Hard+Random` combines the same 7.5K hard episodes with 7.5K randomly sampled replay episodes, yielding a total of 15K episodes. All incremental variants are fine-tuned for one epoch. The Long-Tail Test Set is reserved exclusively for evaluation: Test-all denotes the complete set of 104,655 held-out episodes, whereas Test-hard is a fixed subset of Test-all selected once using the frozen WILD-Nav-inst model before any incremental fine-tuning. Test-hard is defined as the union of episodes ranked by the ADE and FDE error. The resulting episode identifiers remain unchanged across all subsequent evaluations, ensuring that every model is compared on exactly the same difficult cases. Therefore, Test-hard is an evaluation subset of Test-all rather than an additional independent split, and no episode from the Long-Tail Test Set is used for fine-tuning or hyperparameter selection.

B.2. Trajectory Reconstruction

The reconstruction is performed at 3 Hz and a 32-second temporal window to preserve short-term motion dynamics. We define 20 meta-actions to discretize continuous future trajectories into interpretable navigation primitives. Each trajectory is transformed into an ego-centric coordinate system centered at the current frame and represented by five future waypoints in forward–left coordinates. The taxonomy was informed by exploratory trajectory clustering but uses deterministic geometric rules for stable annotation. Valid trajectories are then categorized as stop, backward, or mov-

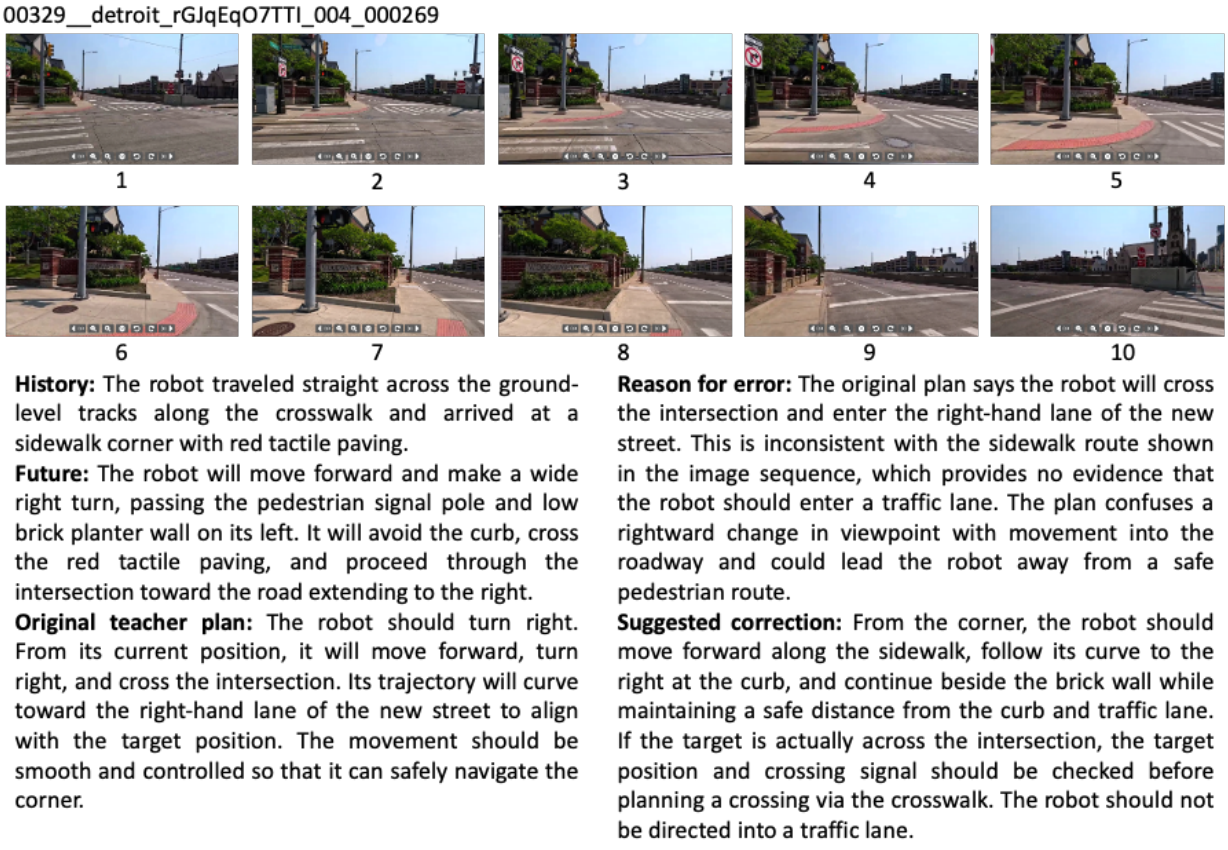


Figure 1. Example of human review at an intersection. The ordered first-person frames appear above the review record, which identifies an unsafe turn into a vehicle lane and suggests continuing along the sidewalk.

ing actions, where stop requires less than 1.2 m net displacement and 1.8 m path length. Moving trajectories are divided into slow, normal, and fast tiers using mean-step thresholds of 0.75 and 1.05 m and path-length thresholds of 3.75 and 5.25 m. And they are further classified by geometry as straight motion, left/right-biased forward motion, left/right turns, or lateral translations. Overall, the taxonomy is dominated by speed-aware forward and turning actions, while stopping, backward motion, lateral translation, and ambiguous trajectories form a small set of specialized classes.

C. Feature representations and Rarity

C.1. Feature representations and clustering dimensionality

We first construct a unified sparse representation from the aligned perception annotation after labeling by Grounding-DINO and Qwen3.6VL. After label normalization, each sample is encoded by 5 atomic features: `obj` denotes an observed object category; `count` denotes category multiplicity of at least two, capped at four; `loc` and `depth` attach the category to its image-plane and depth positions; `co` denotes an unordered pair of co-occurring categories, capped at 32 pairs per sample. For `loc` and `depth`, image-plane position is discretized into five horizontal regions from left to right, while vertical and depth information are represented by five additional regions. These atomic features form the common input vocabulary from which the context representation and joint perception–motion representation are derived.

The joint representation is then constructed by crossing the relevant atomic perception components with the rule-based meta-action labels. `joint_obj_motion` combines each `obj` category with the canonical `meta_action` and each of the four geometry tokens. `joint_slot_geometry` combines the same category together with its `loc/depth` slot and each geometry token. The `count` and `co` remain perception context features rather than entering this cross-product, preventing excessive vocabulary growth. Across all aligned samples, the resulting vocabulary contains 32,662 observed dimensions: 4,249 object–motion, 22,271 slot–geometry, 185 count, and 5,957 co-occurrence tokens. These di-

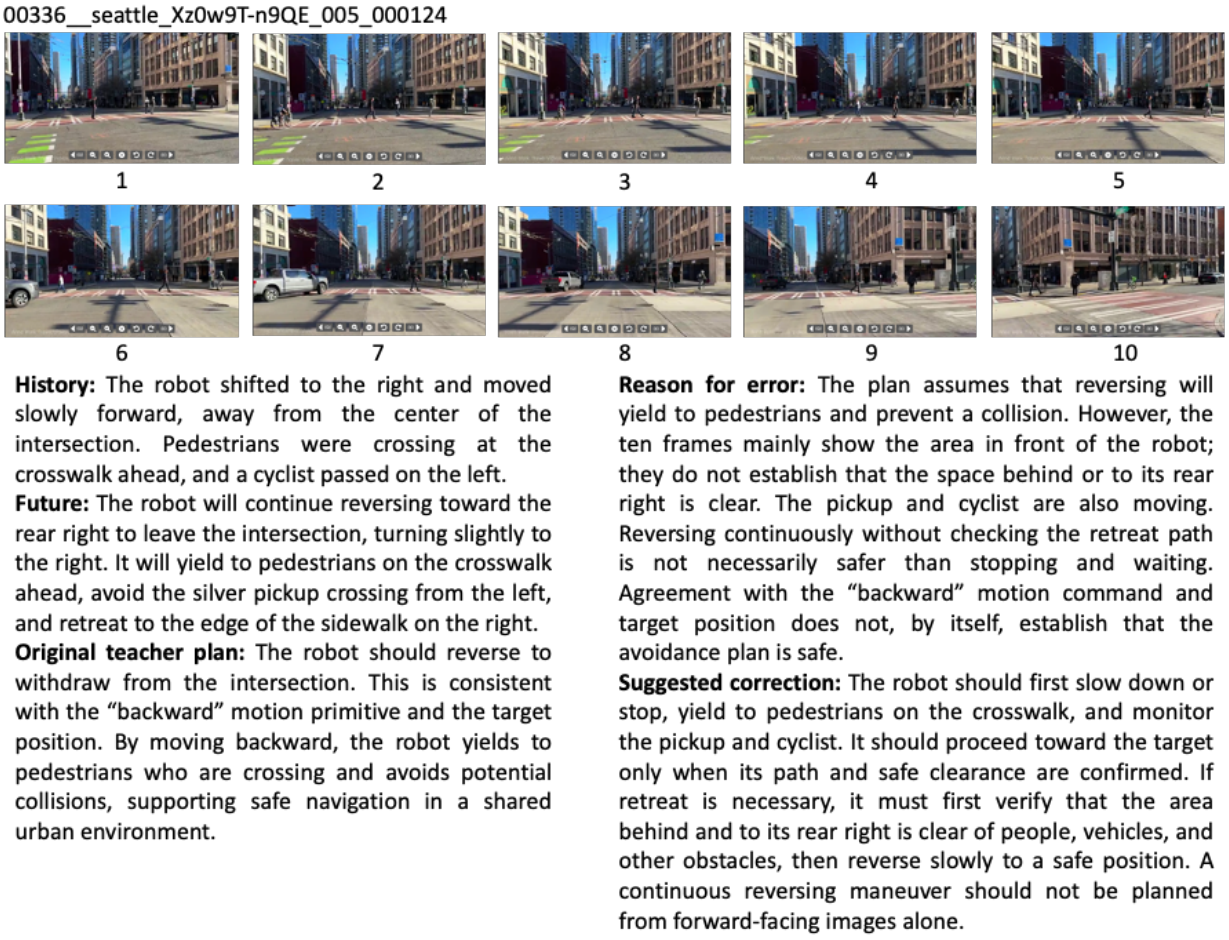


Figure 2. Example of human review near a crosswalk. The review record identifies pedestrians and vehicles behind the agent that the original reversing plan overlooked, and recommends checking for safe clearance before moving.

mensions describe the observed token vocabulary after sample alignment, rather than a predefined dense feature length. Rarity is measured by the length-normalized sum of token-level inverse document frequency (IDF), and the highest-scoring 5% of samples are selected as rare cases.

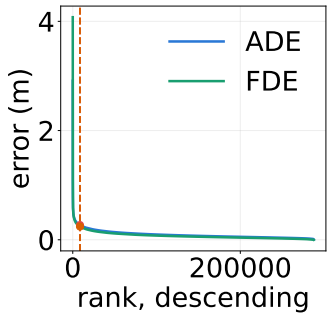
Figure 3 provides the subset coverage and the score distributions used to select the mining thresholds. As illustrated in Fig. 3, the majority of episodes belong to the common–easy region, while rare–hard episodes represent only a small fraction of the benchmark but correspond to cases that are simultaneously underrepresented and challenging for the navigation policy. The hard and rare subsets are selected from their complete descending score distributions, independently of the subsequent reflection labels. For hard mining, the top 3% corresponds to 14,160 unique hard samples before alignment, with cutoffs of 0.270 m for ADE and 0.242 m for FDE. For rarity mining, the top 5% of samples are selected at a score cutoff of 34.94. Within the 287,456 episodes shared by the two results, there are 14,143 hard and 14,373 rare episodes. Their intersection contains 2,423 episodes (17.13% of the hard set and 16.86% of the rare set); 11,720 are hard only and 11,950 are rare only. In both rank curves, the selected boundary lies beyond the steepest high-score head and before the broad low-score body. The fixed 3% and 5% ratios therefore provide transparent tail-focused operating points while retaining enough samples for cluster- and reflection-level analysis; they are distribution-supported selection rules rather than thresholds optimized on downstream failure labels. The two tails capture related but non-equivalent phenomena. Thus, underrepresented perception–motion combinations account for a measurable minority of difficult predictions, but most difficult cases are not distributionally rare and most rare cases do not cause large trajectory errors. This partial overlap is consistent with the residual domain gap reported after real-world fine-tuning: a smaller real-world training set can improve average trajectory accuracy while still leaving uncommon visual–motion combinations insufficiently cov-

Aligned episodes ($n = 287{,}456$)

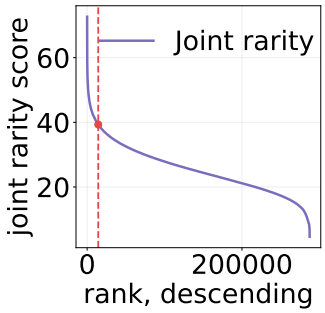
	Rare	Not rare
Hard	**2,423**	11,720
Not hard	11,950	261,363

17.13% of hard are rare
16.86% of rare are hard

(a)



(b)



(c)

Figure 3. Each episode is categorized according to whether it belongs to the top-5% rarity subset and the hard subset identified by ADE/FDE ranking. The rare–hard region represents navigation cases that are both underrepresented in the dataset and difficult for the reference policy. (a) Contingency table for the independently selected hard and rare subsets on their shared episodes. (b) Descending ADE and FDE ranks. (c) Descending joint-rarity scores.

ered. The overlap is associative evidence and does not by itself establish rarity as the cause of prediction error.

D. Reflection-Based Failure Attribution

D.1. Reflection Protocol

We analyze samples from the fixed hard set. For each sample, a teacher receives the student CoT and predicted trajectory together with historical frames, the current frame, ego-history, and the navigation goal. Privileged future frames and the ground-truth future trajectory are available only for verification. The teacher returns a schema containing five analysis fields (prediction, navigation goal, feasible area, rules, and collision risks), a planning rationale, and analysis/planning/trajectory corrections. Failure labels are derived deterministically from this schema after reflection. The action-mismatch label is computed separately by comparing predicted and reference meta-actions. Labels are multi-label: a sample can simultaneously exhibit, for example, prediction, collision-risk, planning, and trajectory-shape errors. A mutually exclusive primary label is retained only for summary; its fixed priority is Action > Predict > Goal > Feasible > Rule > Risk > Plan > Shape > Unexplained > Uncertain. It should not be used to infer the absence of secondary failures.

Fig. 4 reports the multi-label incidence. `meta_action_mismatch` is computed after reflection by comparing the predicted action with the withheld reference meta-action. `trajectory_shape_error` is assigned when the teacher provides a non-empty trajectory correction. `unexplained_hard_sample` denotes a schema-valid reflection with no explicit failure signal despite membership in the fixed hard set; it is a coverage limitation, not evidence that the prediction is correct.

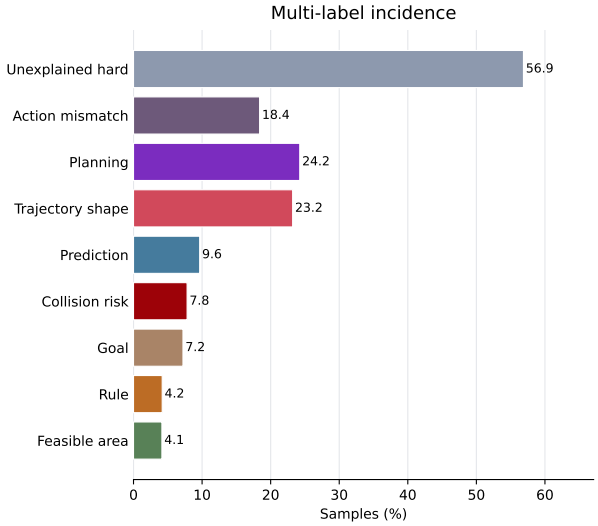


Figure 4. Multi-label failure distribution. Co-occurring planning and trajectory failures are visible here but suppressed when only one primary label is selected.

D.2. Qualitative cluster-conditioned cases

For the clustering experiment in Sec. 5 of the main paper, SVD reduces 3,597 tokens to 80 dimensions (51.96% explained variance), followed by L2 normalization and cosine UMAP to 20 dimensions. A separate 2D projection is used only for visualization, not cluster assignment. Figs. 5–10 show two non-noise cases per highlighted cluster, each with its displayed failure label. The panels include temporal frames, ego-centric history and reference/predicted trajectories, the student prediction, and the complete teacher reflection. Together, they illustrate Action, Rule, Predict, Risk, Plan, and Shape failures without estimating their prevalence.

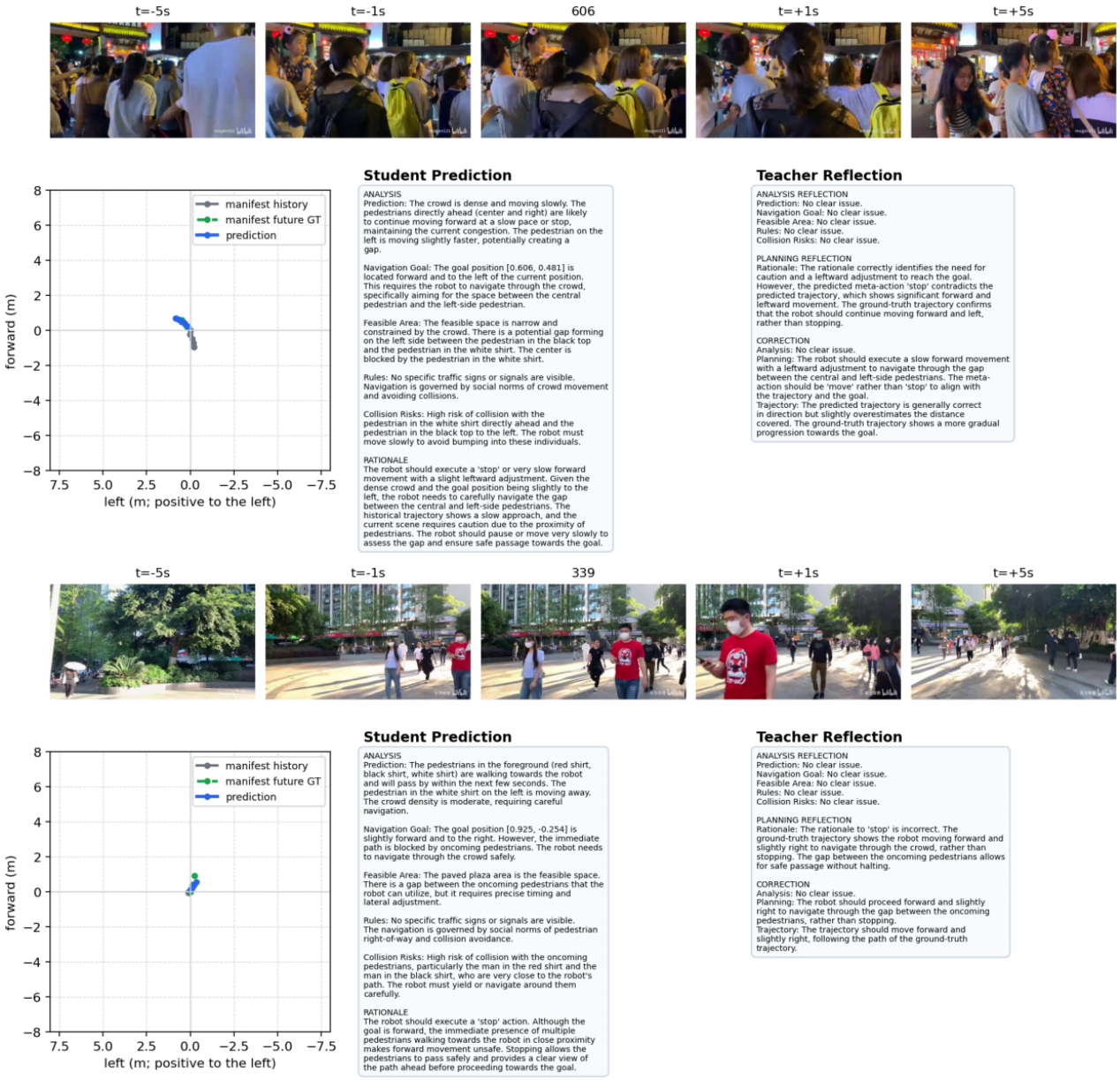


Figure 5. Dense-pedestrian stopping cases in C18. Both cases show planning and trajectory-shape errors: the student selects or justifies stopping although the verified trajectory permits controlled forward progress.

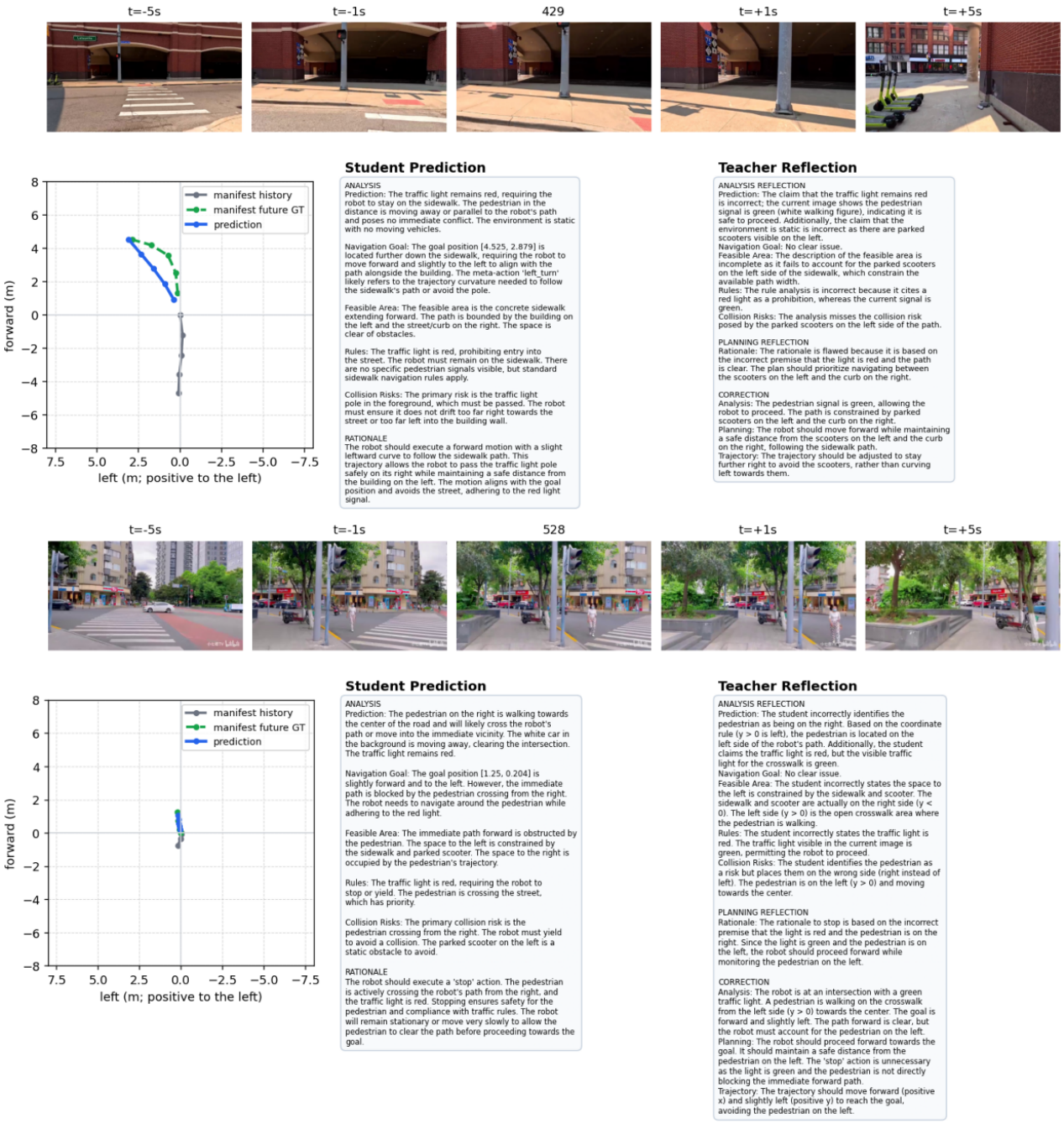


Figure 6. Scooter-sign crossing cases in C47. The cases combine action mismatch with rule interpretation errors around traffic signals and sidewalk versus vehicle-lane semantics.

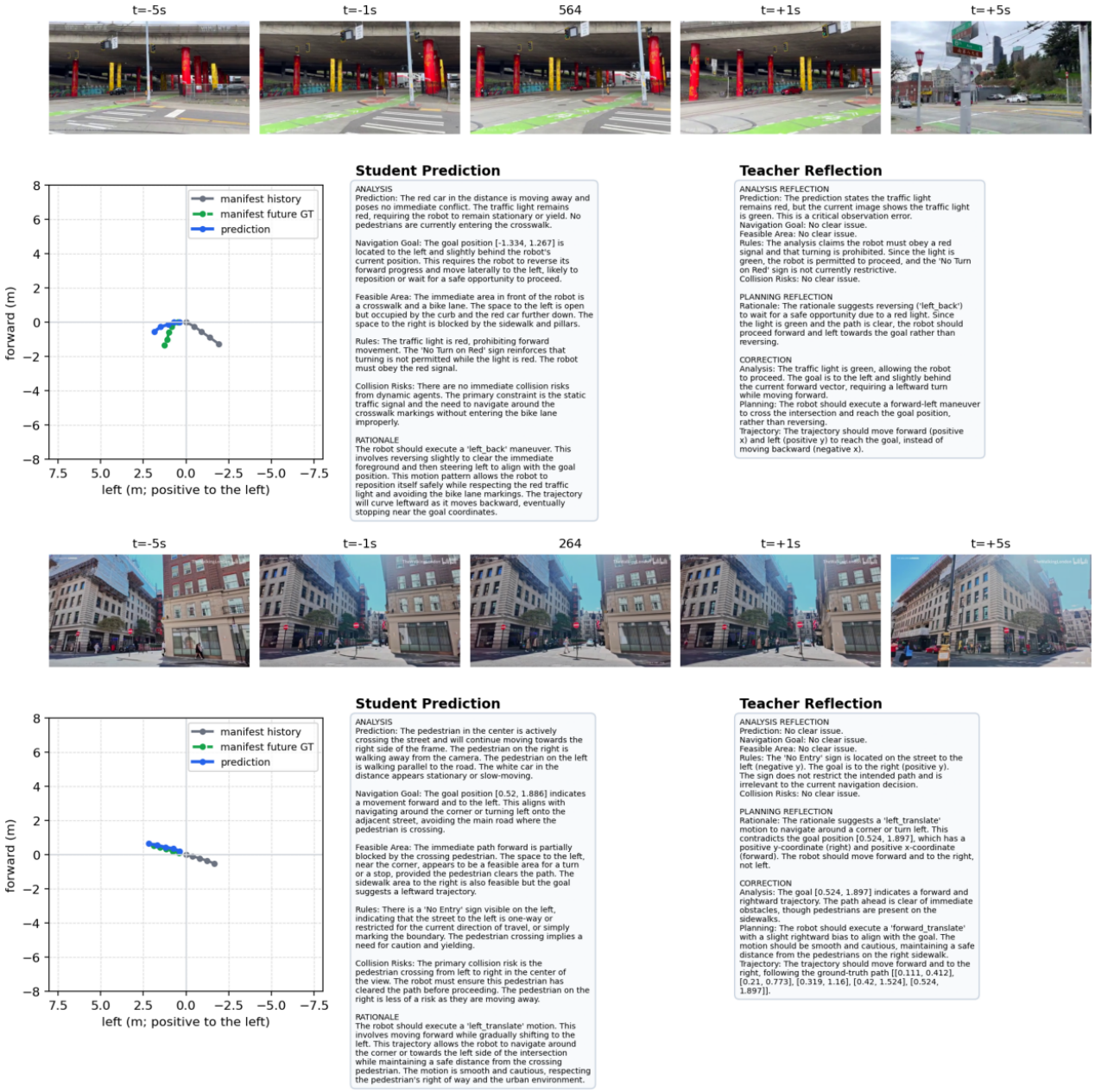


Figure 7. Traffic-sign context cases in C60. The teacher identifies rule errors caused by treating signs or signals outside the intended pedestrian path as binding constraints, or by misreading their current state.

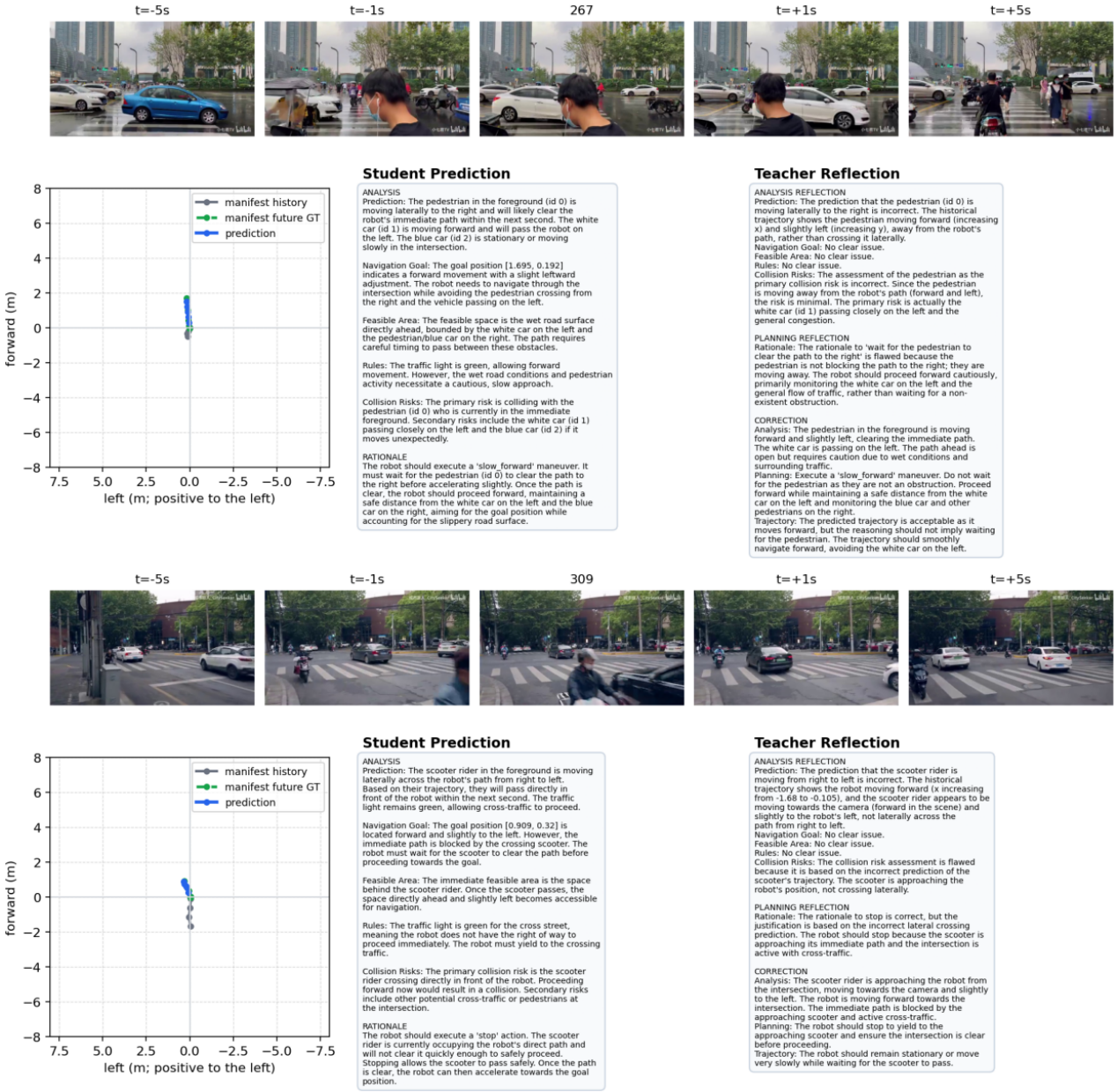


Figure 8. Scooter-signal interaction cases in C105. The student mispredicts the motion or relevance of dynamic pedestrians and scooters, leading to incorrect near-term trajectory reasoning.



Figure 9. Cyclist-interaction cases in C141. The examples show under- or over-estimated collision risks caused by incorrect motion interpretation of cyclists, pedestrians, or cross-traffic.

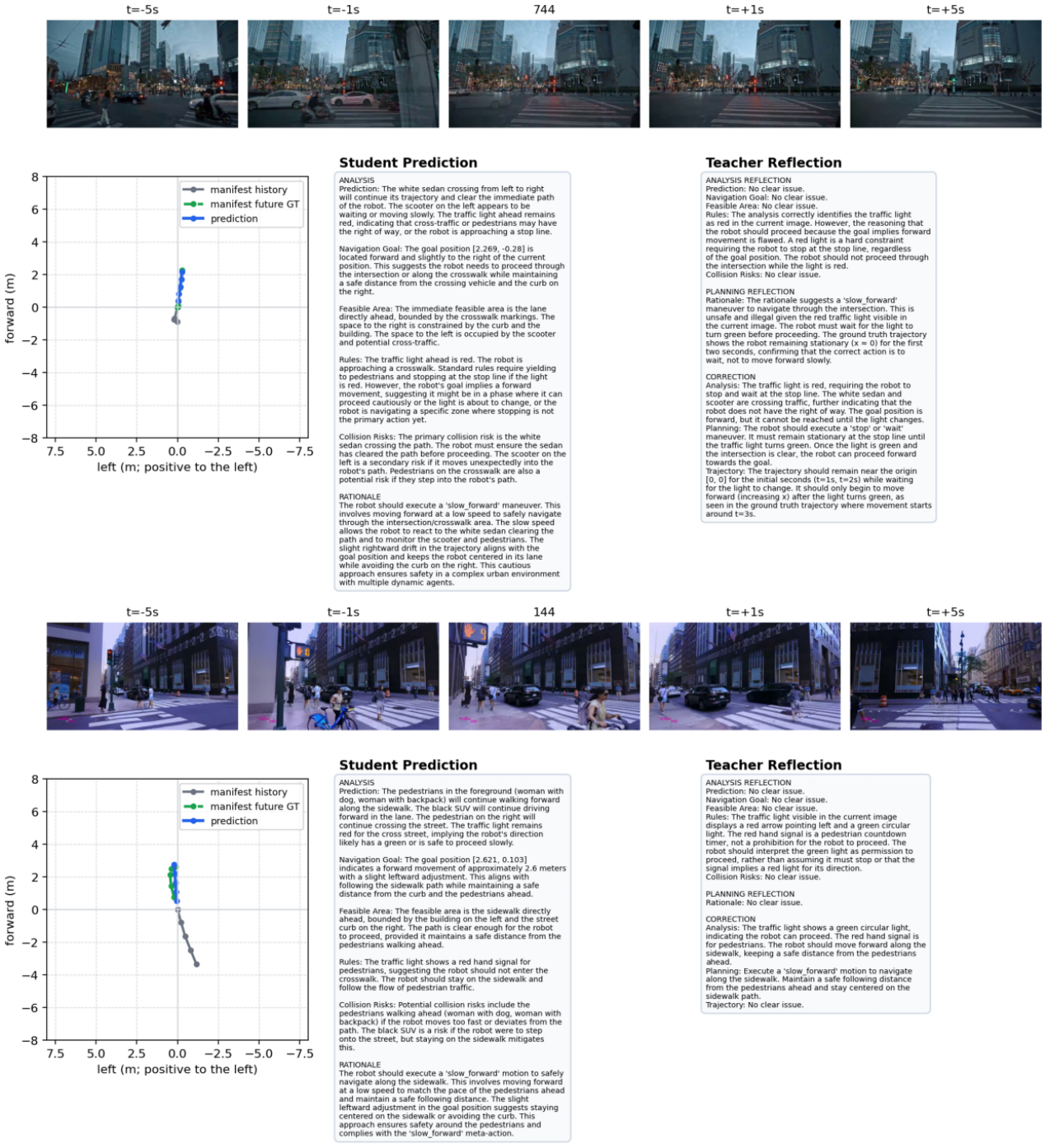


Figure 10. Crosswalk-control cases in C155. The cases illustrate traffic-control reasoning errors, including incorrectly treating a signal as a hard constraint or misinterpreting pedestrian signal semantics.

The selection includes at least two examples for each of the Action, Rule, Predict, Risk, Plan, and Shape failure labels. The appendix supports a scene-conditioned interpretation: dense-pedestrian stopping contexts are associated with planning and trajectory-shape failures; scooter/sign and traffic-control contexts with action or rule failures; and scooter/cyclist interactions with prediction or collision-risk failures. The added hard–rare analysis shows that only a subset of these failure contexts is also underrepresented in the joint perception–motion distribution. These are statistically supported associations between independently generated reflection labels, fixed hard membership, and independently computed rarity; they are not exhaustive causal categories, particularly because a substantial portion of the fixed hard set has no explicit teacher-identified issue.

E. Additional Experimental Results

E.1. Closed-Loop Simulation Validation

We conduct closed-loop point-goal navigation experiments in two simulated environments. ABotN-PointBench provides photorealistic reconstructions of outdoor urban scenes, while we build ten additional urban-navigation scenes as USD assets in NVIDIA Isaac Sim and use them for closed-loop tests. Figure 11 presents the ABotN outdoor scenes and our Isaac Sim assets. Table 3 of the main paper aggregates quantitative results across both environments under a shared evaluation protocol.

ABotN-PointBench does not include an explicit robot model in its reconstructed scenes. We therefore complement its scene-based evaluation with Isaac Sim experiments that instantiate a simulated robot in our urban assets, allowing us to assess closed-loop behavior with an embodied robot.

We evaluate all 225 outdoor point-goal episodes in ABotN-PointBench and 85 point-goal episodes across the ten Isaac Sim scenes. In both environments, each method receives an RGB observation and a local point goal, predicts a short-horizon trajectory, executes the next local motion, and replans from the following observation. All compared methods use the same start and goal within an episode. Runs terminate upon arrival or at the episode's step limit. The four ABotN and six Isaac Sim episodes illustrated below are qualitative examples of these tests.

We use the same collision-aware criterion in both environments: an episode succeeds only when the agent finishes within 0.5 m of the goal with fewer than three path-collision steps. For each episode, let s indicate this success, T be the number of executed steps, C the number of path-collision steps, P the traveled distance, B the distance traveled on collided path segments, and L the shortest-path distance. The episode scores for $\mathrm{SR}_{<3\mathrm{col}}$, SPL, TCR, and DCR are respectively s, $sL/\max(P, L)$, $s(T-C)/T$, and $s(P-B)/P$. Table 3 in the main paper reports each score averaged over the combined ABotN-PointBench and Isaac Sim episodes.

Figures 12–15 show four matched outdoor point-goal episodes in ABotN-PointBench. Within each episode, WILD-Nav-think, WILD-Nav-inst, CityWalker, UrbanNav, and SocialNav navigate from the same start to the same goal. Each method's row shows a first-person view, a third-person view, and the executed trajectory on a top-down map, together with its success or failure status.

Figures 16–21 show six additional closed-loop episodes in the custom Isaac Sim scenes with an instantiated robot. Each row again compares the two WILD-Nav variants and three baselines using first-person observations, third-person robot views, and executed paths on a top-down walkability map.



Figure 11. Simulation scenes used for closed-loop validation. Top: 15 photorealistic outdoor scenes from ABotN. Bottom: ten urban-navigation USD scenes built in NVIDIA Isaac Sim, including streets, parks, plazas, and building complexes.

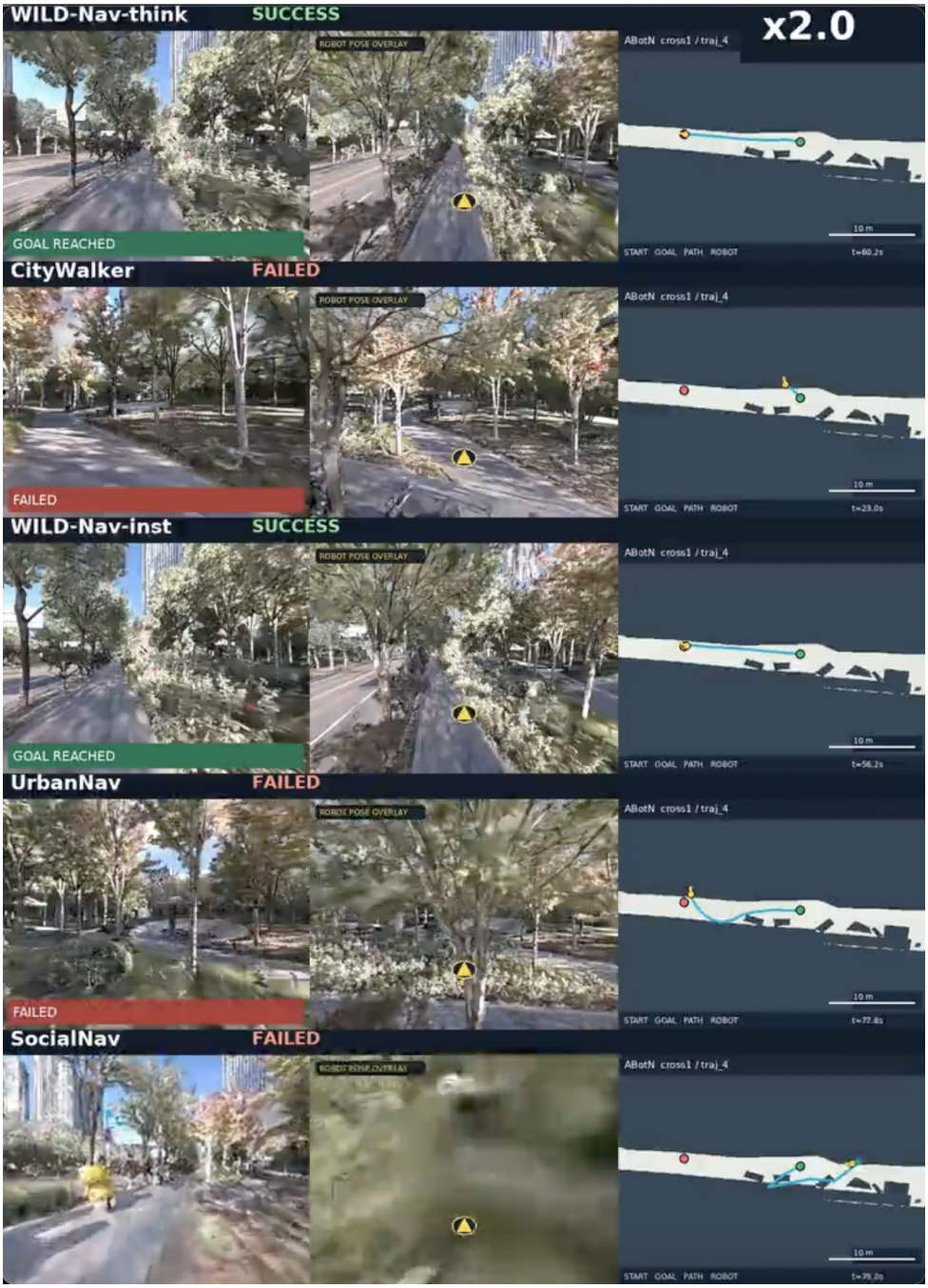


Figure 12. Closed-loop navigation on a tree-lined pedestrian path. WILD-Nav-think and WILD-Nav-inst reach the goal, while CityWalker, UrbanNav, and SocialNav fail in this episode.



Figure 13. Closed-loop navigation along a sidewalk with roadside obstacles. Only WILD-Nav-think reaches the goal in this episode.

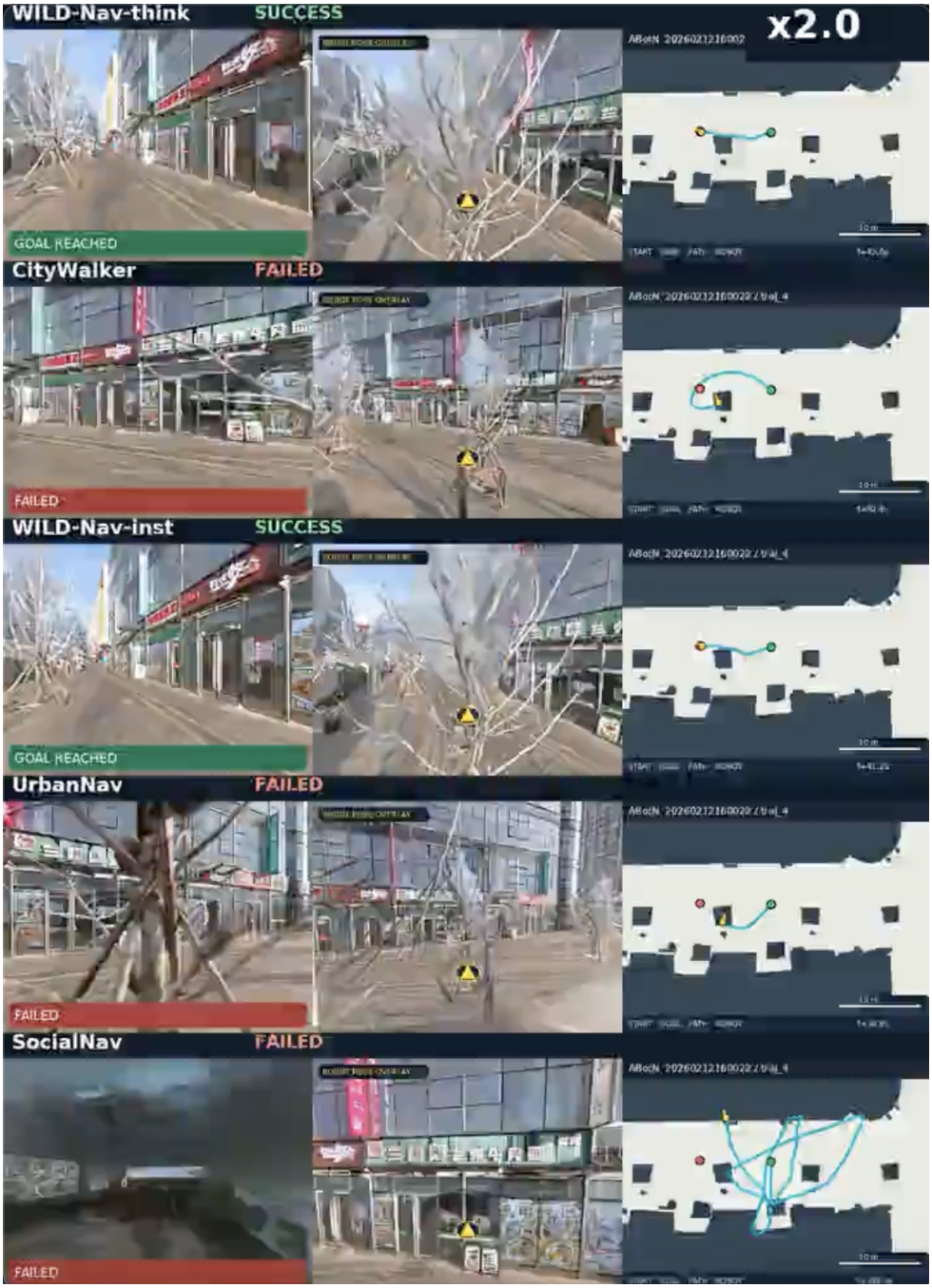


Figure 14. Closed-loop navigation through a commercial streetscape. WILD-Nav-think and WILD-Nav-inst reach the goal, while all three baselines fail in this episode.

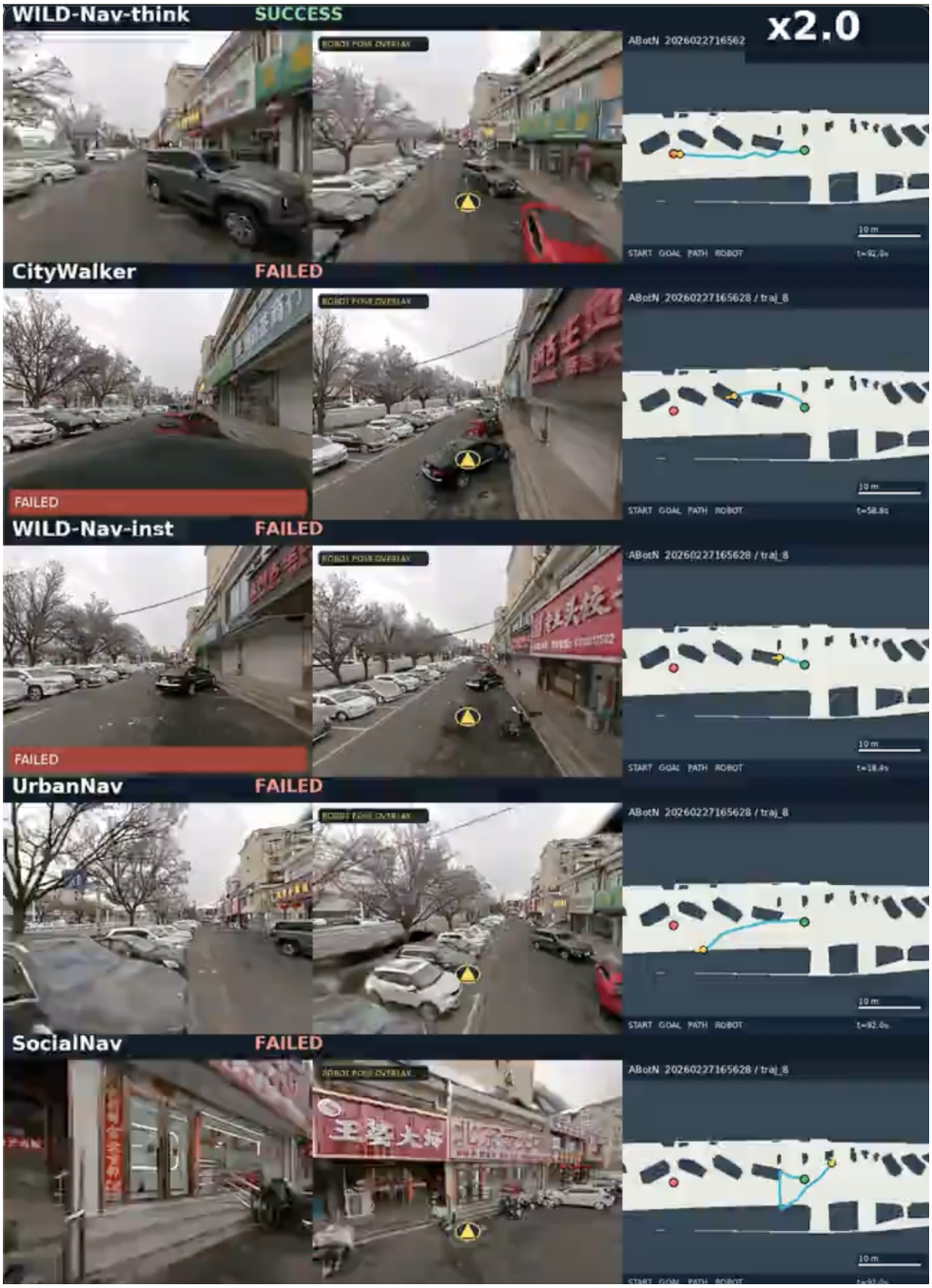


Figure 15. Closed-loop navigation on a street lined with parked vehicles. Only WILD-Nav-think reaches the goal in this episode.



Figure 16. Isaac Sim closed-loop rollout in Scene 6 (case 1). Both WILD-Nav variants reach the goal, while CityWalker, UrbanNav, and SocialNav fail.

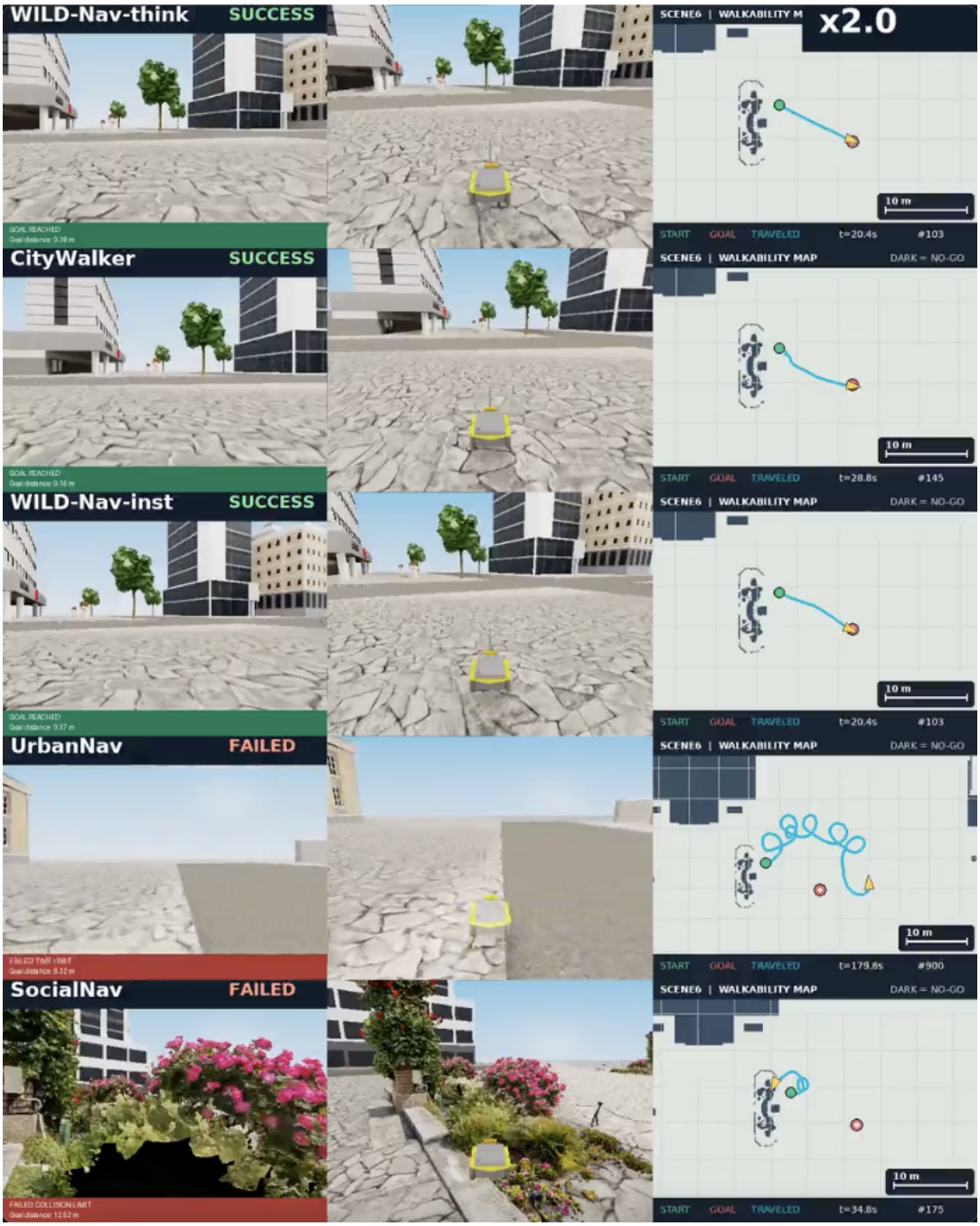


Figure 17. Isaac Sim closed-loop rollout in Scene 6 (case 2). Both WILD-Nav variants and CityWalker reach the goal; UrbanNav and SocialNav fail.



Figure 18. Isaac Sim closed-loop rollout in Scene 8 (case 1). Both WILD-Nav variants, CityWalker, and SocialNav reach the goal; UrbanNav fails.



Figure 19. Isaac Sim closed-loop rollout in Scene 8 (case 2). Both WILD-Nav variants reach the goal, while all three baselines fail.

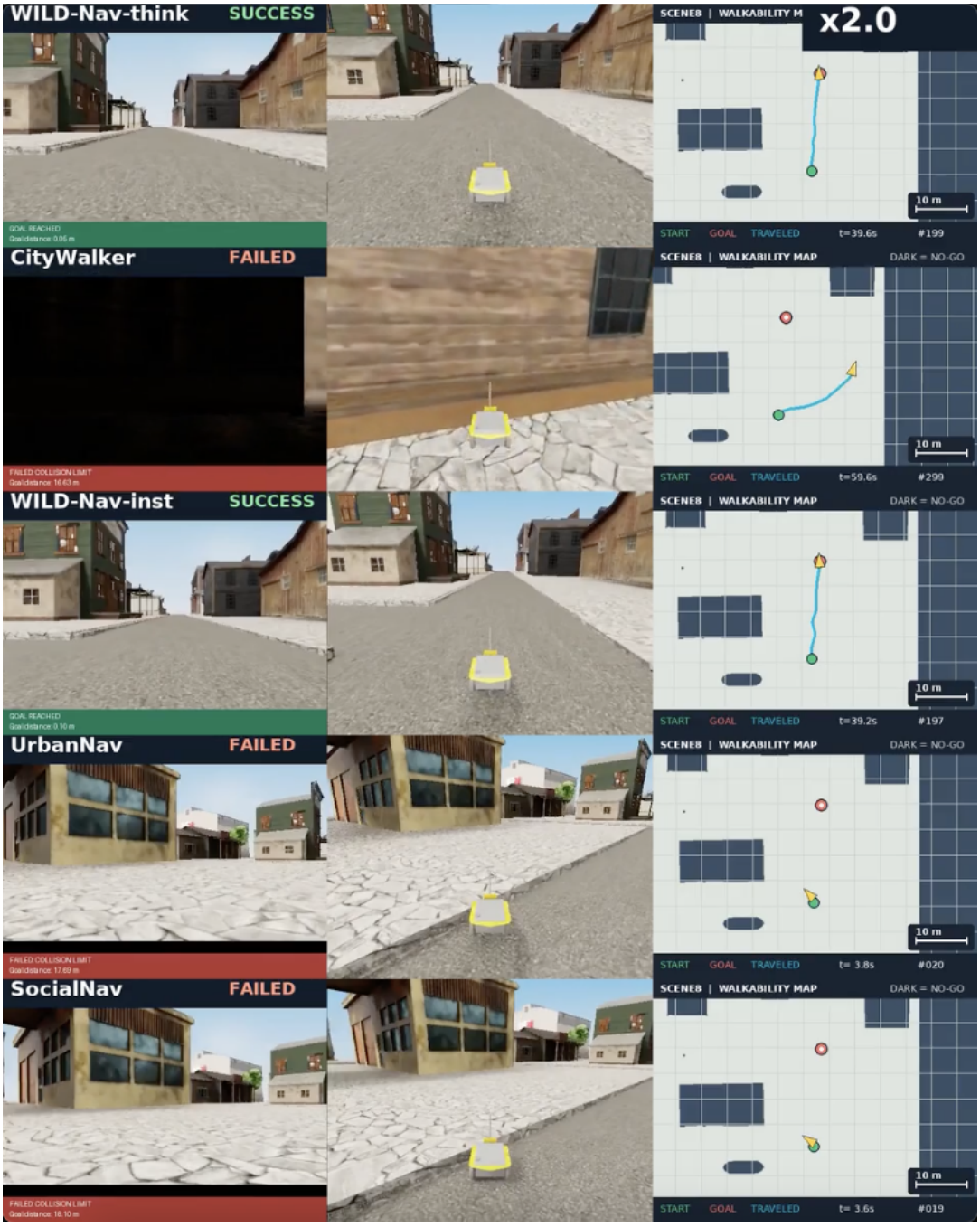


Figure 20. Isaac Sim closed-loop rollout in Scene 8 (case 3). Both WILD-Nav variants reach the goal, while all three baselines fail.

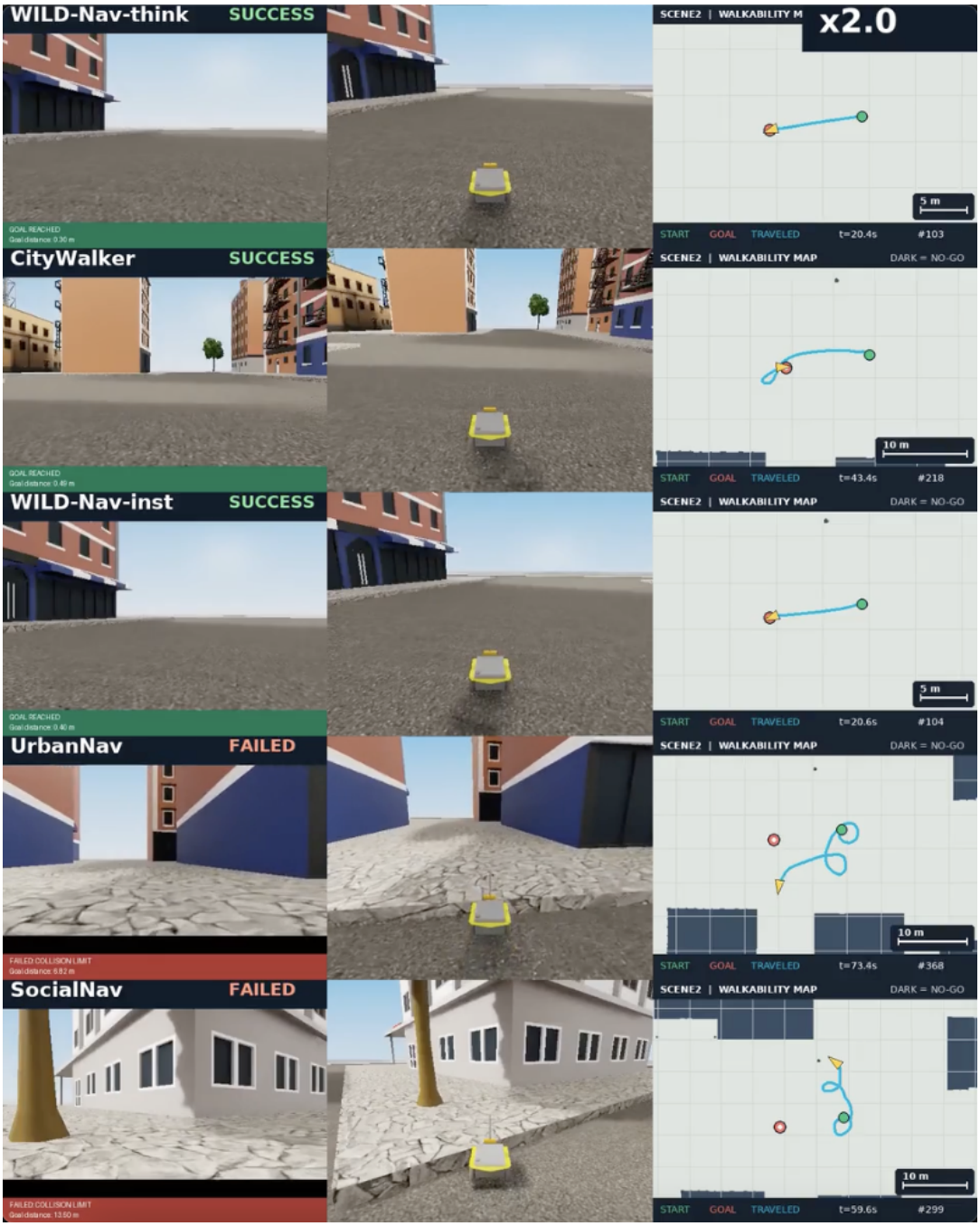


Figure 21. Isaac Sim closed-loop rollout in Scene 2. Both WILD-Nav variants and CityWalker reach the goal; UrbanNav and SocialNav fail.

E.2. Details of Real-world Deployment

We conducted two representative real-world experiments outdoors: street-side navigation involving an intersection crossing, and long-distance navigation within a university campus. The platform used was an AgileX Scout 2.0 mobile robot. All computation ran on a laptop equipped with an NVIDIA RTX 4090 GPU (16 GB), using only an Intel RealSense D455 camera as the perceptual input. The model achieved an average inference time of approximately 0.76 seconds. For navigation, we pre-sampled target points every 4 m on a pre-built point cloud map and sequentially fed these points as navigation goals to the model; the point cloud map itself was not exploited by the online navigation algorithm. Given the current camera image and the target point, the model inferred a local trajectory for the next 5 seconds. A PD controller then drove the robot to track the first waypoint on this planned trajectory in real time, thereby closing the control loop. More detailed results are provided in the supplementary video, which simultaneously shows three views: the camera's first-person view, a third-person view, and the LiDAR localization trajectory recorded by the FAST-LIO algorithm.

In the long-distance campus experiment, we compare CityWalker and WILD-Nav-inst on the same approximately 0.7-km point-goal route from the College of Engineering to a sandwich shop (Fig. 22). Both methods navigate the same route on the AgileX Scout 2.0 platform using the local point goals and trajectory-tracking controller described above. CityWalker repeatedly deviates from the pedestrian route into the adjacent vehicle lane and collides before reaching the destination. WILD-Nav-inst completes the route, including the crossing and roadside interactions shown in the selected frames. The route maps show the outcome directly: CityWalker's path stops at the collision site, while WILD-Nav-inst's path reaches the marked goal.

E.3. Hyperparameters for Model and Training.

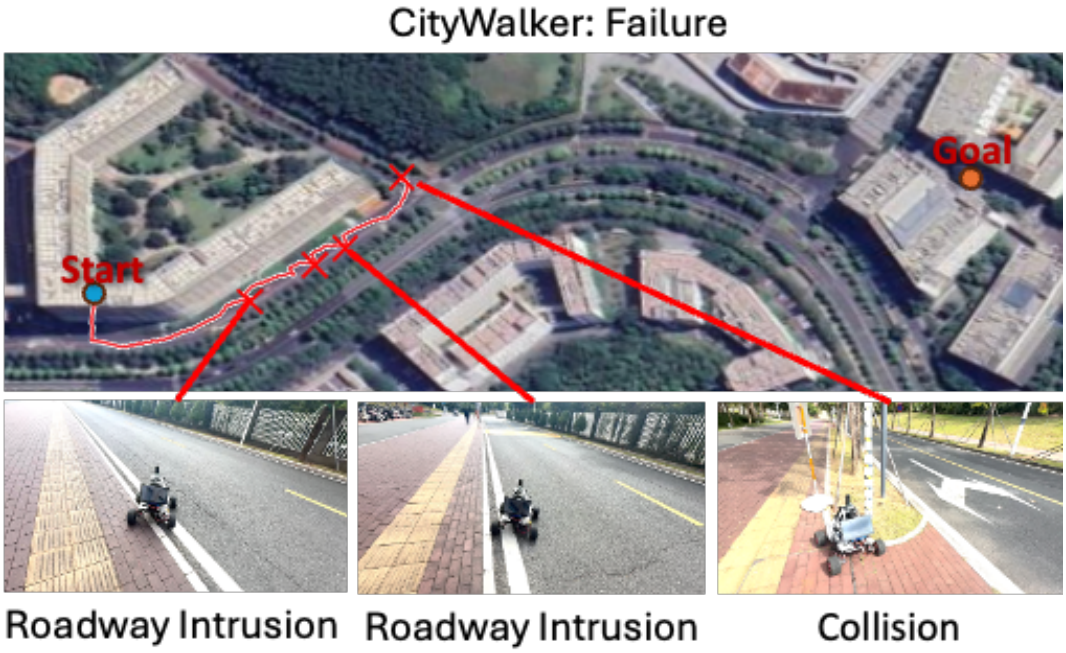
We fine-tune Qwen3.5VL-4B for WILD-Nav. The vision encoder is frozen; the LLM, vision–language merger, and trajectory head are trained. Each input contains five frames (the current frame and four historical frames sampled at 1 Hz) at 640×360 resolution, without a fixed center crop. The vision processor uses a dynamic-resolution pixel budget of 100,352–602,112 pixels per frame. The model predicts 5 future 2D waypoints via probe token <PLAN_QUERY>. Training runs for 1 epoch ($\sim$7.2k steps) on 8 GPUs (Nvidia RTX A6000) with DeepSpeed ZeRO-2, effective batch size 64, AdamW (LR 2×10^{-5} for backbone, 5×10^{-4} for action head), cosine schedule with 3% warmup, bf16, and gradient checkpointing.

E.4. Qualitative Comparison of Trajectory Planning

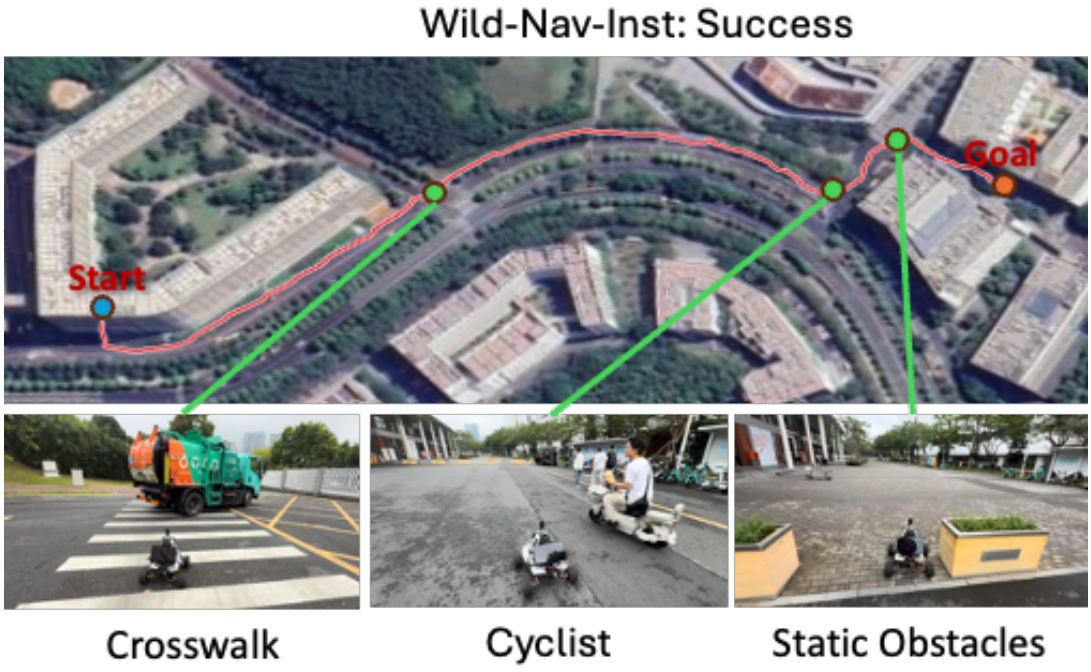
Figures 23–25 compare predicted trajectories on first-person videos. Figure 23 covers structured scenes such as open park paths, commercial pedestrian streets, and clear sidewalks. In these examples, WILD-Nav-think, WILD-Nav-inst, and UrbanNav plan trajectories that closely follow the recorded future path.

Figure 24 shows more constrained urban scenes. The WILD-Nav predictions remain closer to the recorded future path, while UrbanNav deviates substantially in the illustrated examples.

Figure 25 compares WILD-Nav-think with WILD-Nav-inst in scenes involving pedestrians and vehicles and displays the intermediate reasoning text. In these examples, WILD-Nav-inst produces larger trajectory deviations, whereas WILD-Nav-think uses its analysis and planning stages to account for moving obstacles and navigable space, producing trajectories closer to the recorded future path.



(a) CityWalker: repeated excursions into the vehicle lane followed by a collision before reaching the goal.



(b) WILD-Nav-inst: successful navigation to the goal along the same route.

Figure 22. Real-world closed-loop comparison on an approximately 0.7-km route from the College of Engineering to a sandwich shop. The top panel of each subfigure shows the executed path; the lower panels show selected robot views along that path.

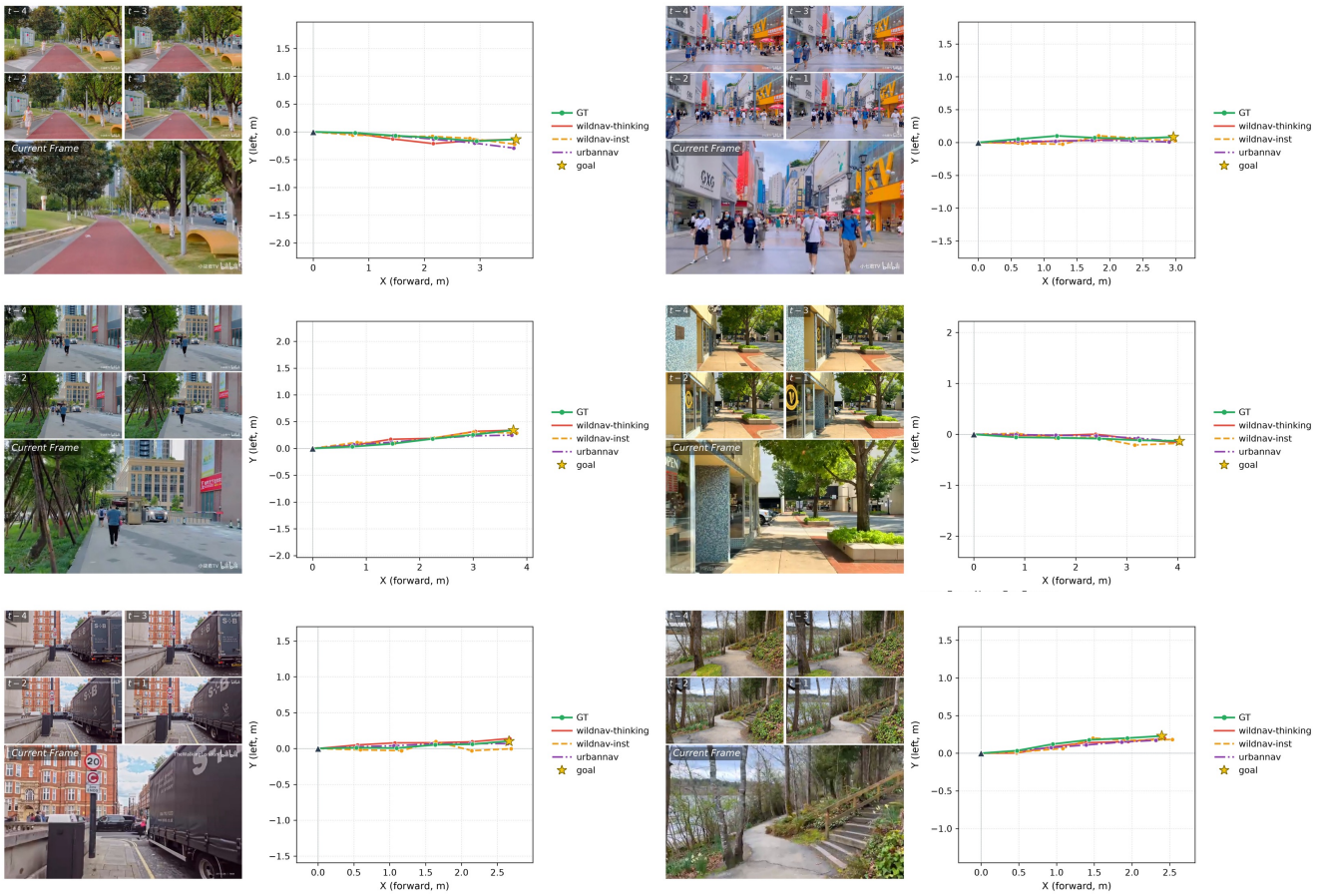


Figure 23. Qualitative comparison in standard navigation scenarios. In these relatively structured environments, both our proposed methods and the baseline (UrbanNav) demonstrate robust performance, generating trajectories that closely align with the ground truth (GT).

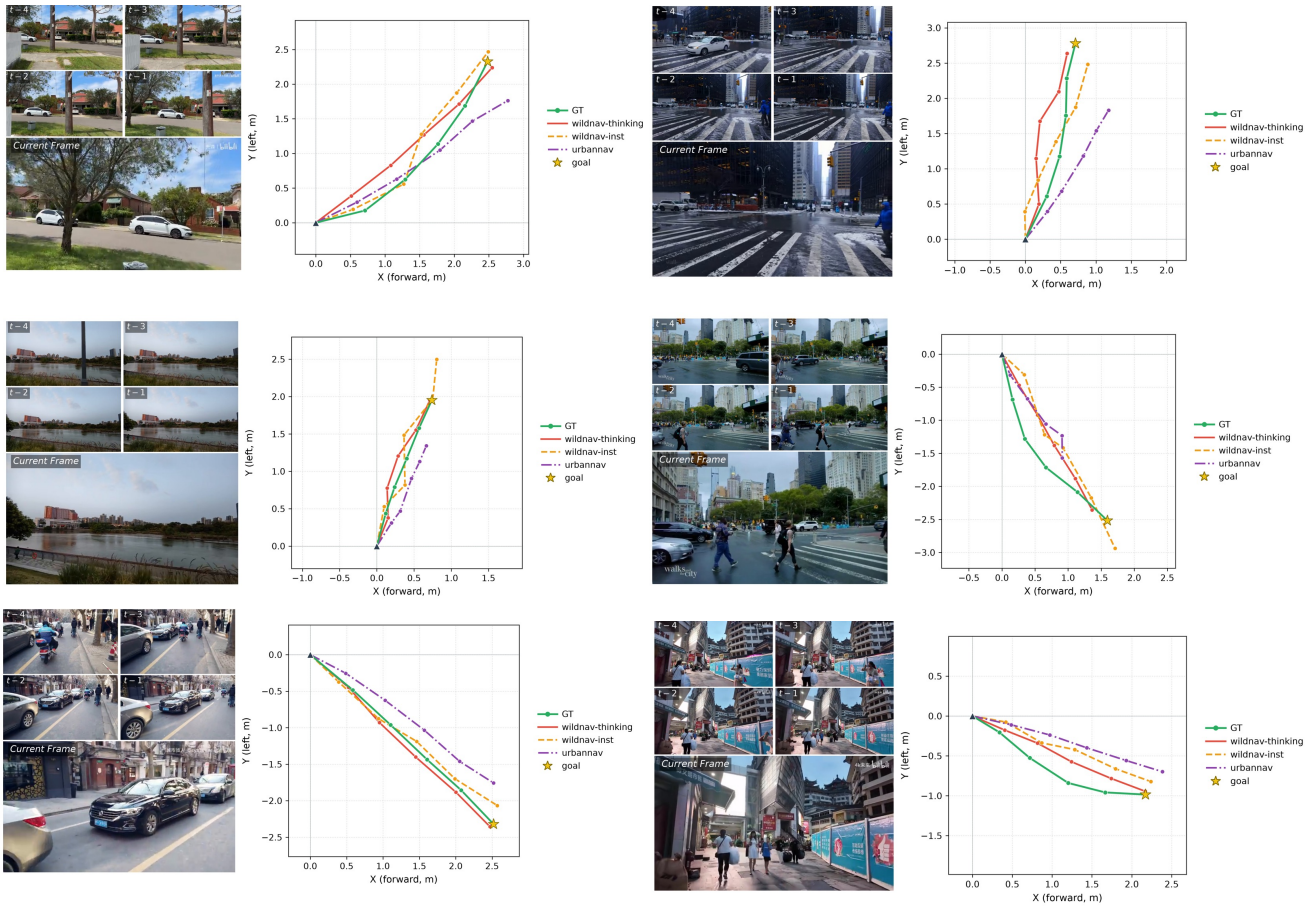


Figure 24. Qualitative comparison across diverse urban environments. Across various challenging scenarios, our proposed methods maintain accurate trajectory planning that aligns closely with the ground truth (GT). In contrast, the baseline (UrbanNav) exhibits significant deviations from the target path.

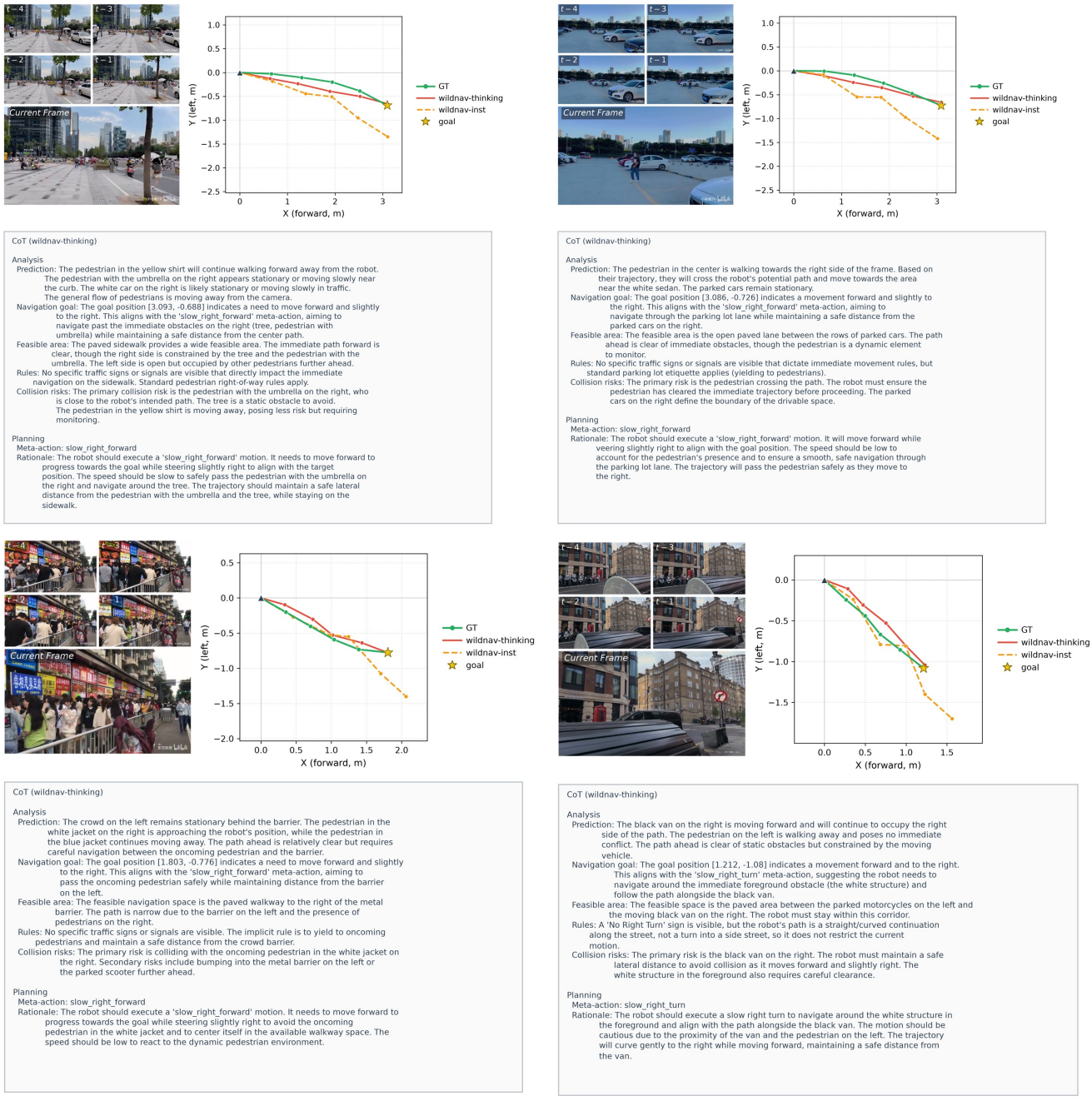


Figure 25. Qualitative comparison of WILD-Nav-think and WILD-Nav-inst with the intermediate analysis and planning text. In these examples, WILD-Nav-think predicts trajectories closer to the recorded future path.