# Learning to Drive is a Free Gift: Large-Scale Label-Free Autonomy Pretraining from Unposed In-The-Wild Videos

## Supplementary Material

This supplementary material provides additional results and implementation details. We include full training configurations in Sec. A, planning fine-tuning and baseline descriptions in Sec. B, and extended qualitative visualizations: segmentation in Sec. C, motion in Sec. D, and depth in Sec. G.

## A. Training Details

For reproducibility, we share specific training details of our model. We train LFG on top of the pretrained $\pi^3$, keeping the DINOv2 encoder frozen, as well as the confidence, camera, and point decoders, including automatic mixed precision (bfloat16) to speed up training.

To obtain motion masks from **Grounded SAM2** and **CoTracker3**, we first query **Grounded DINO** using the object priors *car*, *vehicle*, and *person*, which yields an initial set of candidate instance masks. Each mask is then processed with **CoTracker3**, using a grid size of 80 and a motion threshold of $0.1$ in the normalized geometric space. An object is classified as dynamic if it exhibits motion in the majority of frames.

For the segmentation loss, we apply class-specific weighting across seven categories to address inherent frequency imbalances in driving scenes. Specifically, we assign weights of $0.5$ to *road*, $1.2$ to *vehicle*, $1.6$ to *person*, $1.8$ to both *traffic light* and *traffic sign*, $0.3$ to *sky*, and $0.2$ to *background/buildings*. These weights remain fixed throughout training and were found to provide a stable and effective balance across diverse urban environments.

We apply VGGT-style photometric augmentations during training. Color jittering perturbs brightness, contrast, and saturation by $\pm 40\%$ $(0.4)$ and hue by $\pm 10\%$ $(0.1)$. Random grayscale is applied with probability $0.1$. Additionally, we apply random Gaussian blur with probability $0.2$, using a sigma sampled uniformly from $[0.1, 2.0]$. We resize all images to $(294, 518)$, and train on the prior 3 images, predicting outputs for the next 3 images, but additionally train the motion head (final stage) on both the 3 prior images and 6 prior images. We vary the time between each image, randomly sampling from $2, 5, 10$Hz.

## B. Planning Fine-tuning and Baseline Details

We fine-tune all models on the NAVSIM planning benchmark using only the front-view camera over three consecutive frames to predict 4s future ego trajectories. Unless otherwise specified, all baseline vision encoders are kept frozen and we only train lightweight causal attention adapters and a shared anchor-based trajectory decoder.

**Common planning head.** For all methods (ours and baselines), we employ the same anchor-based trajectory decoder. Following DiffusionDrive [17], we adopt $K = 20$ trajectory anchors obtained by K-means clustering over ground-truth futures; however we omit the diffusion component and any iterative refinement to keep the architecture simple. After causal temporal aggregation from vision encoder's embedding, the decoder attends over trajectory anchors and across modes, and in a single forward pass predicts (i) confidence scores for each of the $K$ modes and (ii) coordinate offsets for each waypoint along each mode. At test time, the highest-confidence mode is selected as the final plan. All models predict 8 waypoints at 0.5s intervals (a 4s horizon) and are trained with a combination of focal loss (classification over modes) and L1 regression loss on waypoints.

**Temporal aggregation.** For each front-view frame, the pretrained encoder produces high-dimensional autonomy tokens encoding ego motion and scene context. To exploit temporal structure, we apply a small causal self-attention module across the three input frames' embeddings. The resulting aggregated features are passed into the trajectory decoder. With our method (LFG), since the encoder has already been pretrained with temporal reasoning, we use the last set of future autonomy tokens directly, which provides a temporally consistent representation for the planning head to condition on.

**Baselines and training protocol.** We evaluate three frozen encoders: PPGeo (geometric pre-training), DINOv3 (self-supervised ViT), and Pi3 (4D self-supervised learning). Each is followed by the same causal temporal adapter and shared anchor-based planning head. Our method (LFG) uses the pretrained temporal autoregressive encoder described in the main paper (also kept frozen), along with a lightweight multi-modal trajectory decoder. All models are optimized with AdamW and a cosine learning-rate schedule, and are trained under identical data-scaling regimes using 1%, 10%, and 100% of NAVSIM training data with learning rate 1e-4 to study data efficiency.

For DiffusionDrive, we follow the publicly released implementation (code available on GitHub) which uses three

Table A1. **Comparing PPGeo with different pretraining data-source.** PPGeo* indicates that the model is pretrained on the *same OpenDV dataset* used by LFG.

| Method | Input | 1% | 10% | 100% Data |
|---|---|---|---|---|
| PPGeo | 1Cam | 61.5 | 65.6 | 74.6 |
| PPGeo* | 1Cam | 59.8 | 70.0 | 76.4 |
| **LFG (Ours)** | 1Cam | **66.3** | **81.4** | **85.2** |

front-view cameras plus LiDAR input and their corresponding hyper parameters.

Table A2. **DiffusionDrive comparison on NAVSIM (PDMS↑).**

| Method | Input | 1% | 10% | 100% |
|---|---|---|---|---|
| DiffusionDrive-DINOv2 | 3Cam+L | 57.3 | 74.4 | 81.5 |
| DiffusionDrive-DINOv2 | 1Cam | 57.5 | 73.0 | 79.7 |
| **LFG (Ours)** | **1Cam** | **66.3** | **81.4** | **85.2** |

For PPGeo, in Tab. 4 we use the publicly released ResNet-34 encoder from the PPGeo repository[1] pretrained with geometric self-supervision [30]. The original PPGeo encoder is pretrained using the YouTube driving video dataset introduced in the ACO project[2]. To isolate the impact of pre-training data source, we evaluate a variant, PPGeo*, where we replicate the same geometric pre-training procedure but restrict the pre-training corpus to exactly the data used by LFG. As shown in Tab. A1, PPGeo* slightly improves performance at higher label fractions but still under-performs LFG by a wide margin, highlighting that LFG's 4-D temporal pre-training paradigm provides inductive biases that align more directly with downstream planning.

**NAVSIM metrics** The NAVSIM benchmark uses a composite score called the Predictive Driver Model Score (PDMS) to evaluate planning performance. PDMS is computed in two phases: (i) two hard-multiplier subscores **No at-fault Collisions (NC)** and **Drivable Area Compliance (DAC)** that immediately zero the scenario score if violated; (ii) a weighted average of three performance subscores **Ego Progress (EP)**, **Time-to-Collision (TTC)**, and **Comfort (C)** — reflecting route progress, safety margin, and motion smoothness.

Formally,:

$$\text{PDMS} = \big(\text{NC} \times \text{DAC}\big) \times \frac{5\,\text{EP} + 5\,\text{TTC} + 2\,C}{5+5+2}$$

Here:

- NC = 1 if no at-fault collision, = 0.5 if a collision with a static object, = 0 otherwise.
- DAC = 1 if the ego vehicle remains within the drivable area for the entire rollout, = 0 if it leaves.
- EP is the ratio of actual route progress achieved to a safe upper bound (clipped to [0,1]).
- TTC = 1 if the minimum time-to-collision along the 4s horizon exceeds a fixed threshold, else = 0.
- C = 1 if all vehicle kinematic thresholds (acceleration, jerk) remain within comfort bounds, else = 0.

All metrics are evaluated via a non-reactive 4-second rollout in the benchmark simulator in the test set 12k samples.

## C. Segmentation Visualizations

We show segmentation visualizations on the OpenDV dataset, with sample unposed images, and the teacher SegFormer model outputs as in Fig. A6, on a 5hz scene. We find that LFG performs very competitively with its SegFormer teacher on the current frames, and future predicts the motion of the moving bus as it is about to pass the ego vehicle. LFG, however, suffers from a smoothing effect in the later frames. We posit that training LFG on more steps and the entire OpenDV dataset will improve this, as well as an edge aware point map loss to improve crispness of future frame predictions.

## D. Motion Visualizations

We demonstrate motion visualization on the OpenDV dataset with current frames to emphasize the performance trained from pseudo ground truth data, on 10Hz, but we show 3 frames spaced apart every other frame. LFG correctly predicts the moving cars in frame from only 2D images, with a small amount of frames. Future work entails demonstrating LFG's performance for constructing dynamic Gaussian Splats, where the motion masks can be freely obtained.

## E. Trajectory Prediction

Table A3. **Trajectory estimation results.** RelPos is split into rotation (deg) and translation (m).

| Dataset | Method | ATE | Rot | Trans |
|---|---|---|---|---|
| KITTI-360 | $\pi^3$ | 0.43 | 1.32 | 0.31 |
| | LFG | 1.00 | 2.30 | 0.31 |
| Waymo | $\pi^3$ | 0.02 | 0.98 | 0.44 |
| | LFG | 0.08 | 1.00 | 0.44 |

As our model predicts camera poses for 3 input frames and 3 future frames, we evaluate trajectory prediction on KITTI-360 and Waymo (200 sequences of 6 frames each),

[1] https://github.com/OpenDriveLab/PPGeo
[2] https://github.com/metadriverse/ACO

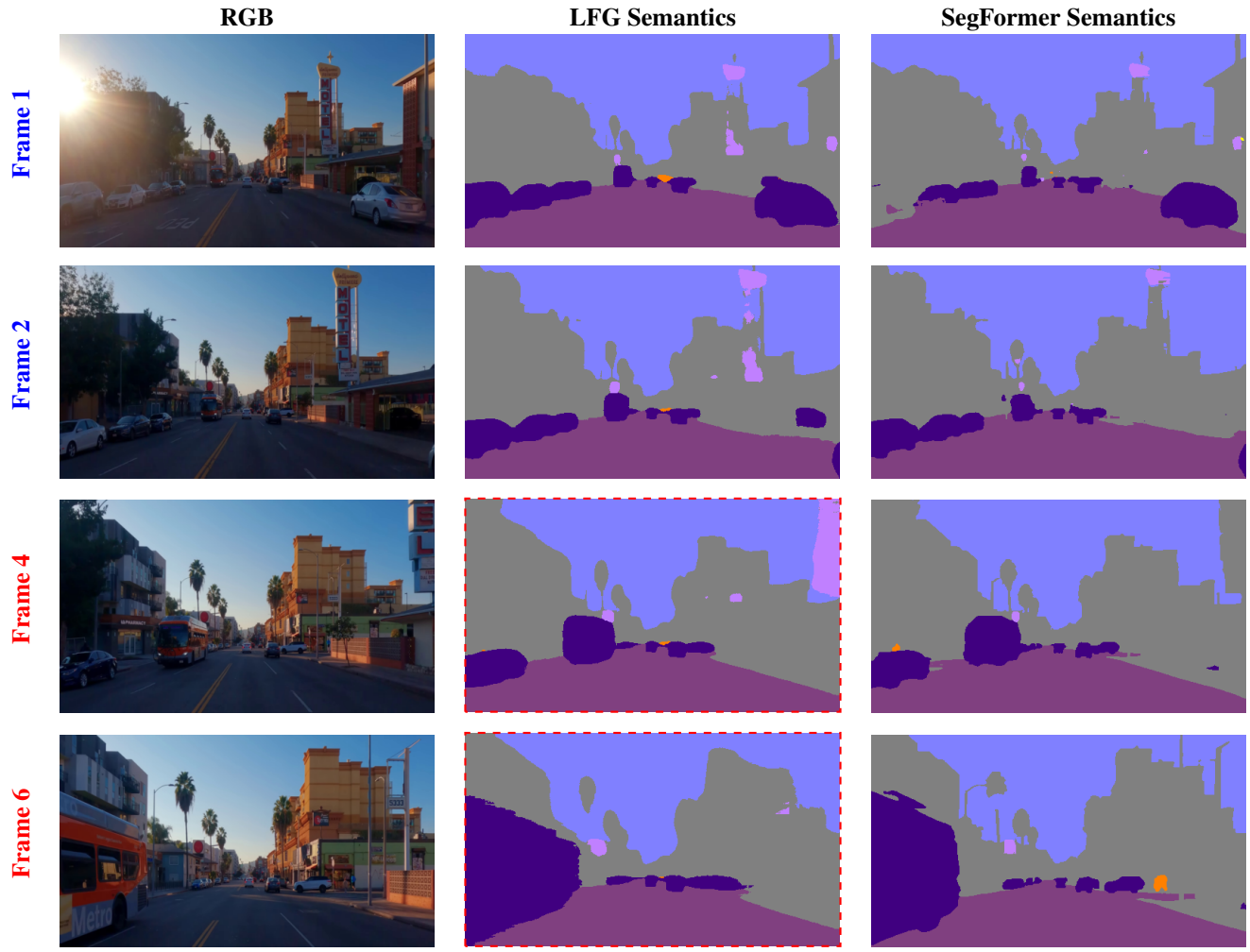


Figure A1. Qualitative comparison of semantic segmentation across RGB, LFG, and SegFormer for current frames 1 and 2 (with ground-truth input) and future frames 4 and 6. Dashed red outlines denote predicted frames with *no ground-truth image input*, produced solely from the model's future tokens.

comparing to $\pi^3$ with all 6 frames as input. We report Absolute Trajectory Error (ATE), rotation error (Rot, deg), and translation error (Trans, m). In Table A3, while the metrics are slightly worse than the teacher model, the result is competitive considering that our model does not have access to the last 3 frames.

## F. Point Cloud Reconstruction

Fig. A3 provides a qualitative comparison of full point cloud reconstructions from LFG and $\pi^3$ across three scenes, illustrating that LFG preserves geometric structure and camera motion even when predicting future frames.

## G. Depth Visualizations

We show depth visualizations of LFG compared to $\pi^3$ on validation images on our dataset, at a frequency of 5Hz, on Fig. A4. LFG performs comparable to $\pi^3$ on the seen frames, and while sharp edges are slowly lost in the future frames, LFG is able to understand dynamic and static objects, and the relative positioning of the other vehicles over time. Future work will crisp the point maps, and more results, including motion and semantic results, which are shown at the end of the supplementary.

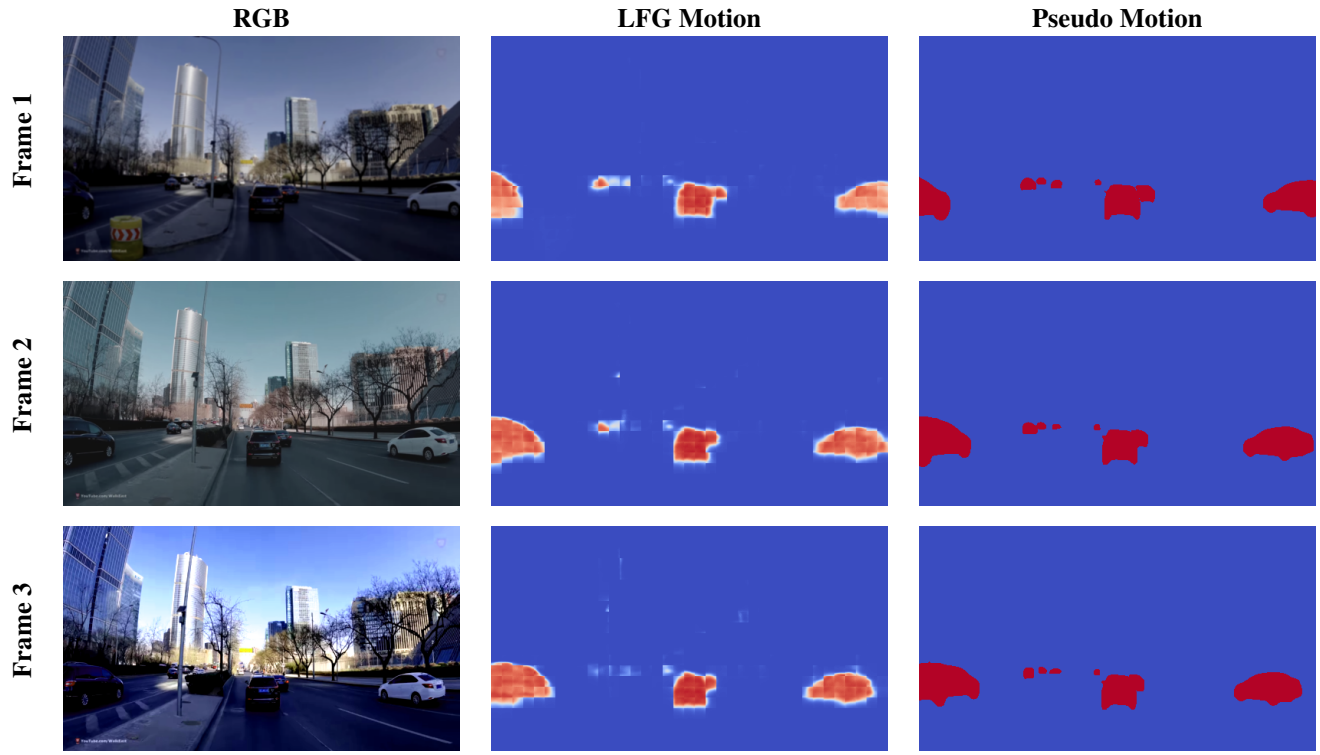


Figure A2. Qualitative comparison of motion predictions (LFG vs Pseudo GT motion) with corresponding RGB frames. We show results on non-future frames to demonstrate the motion map precision on a few images. The ego vehicle is moving on a road with three nearby vehicles moving.


Figure A3. Qualitative comparison of full point cloud reconstructions of LFG vs. $\pi^3$. The current camera poses are in blue, and future poses in red. LFG point maps retain overall geometric quality, even on future frames, and the predicted camera motion remains precise. **Dashed red** outlines denote predicted frames with *no ground-truth image input*, produced solely from the model's future tokens.

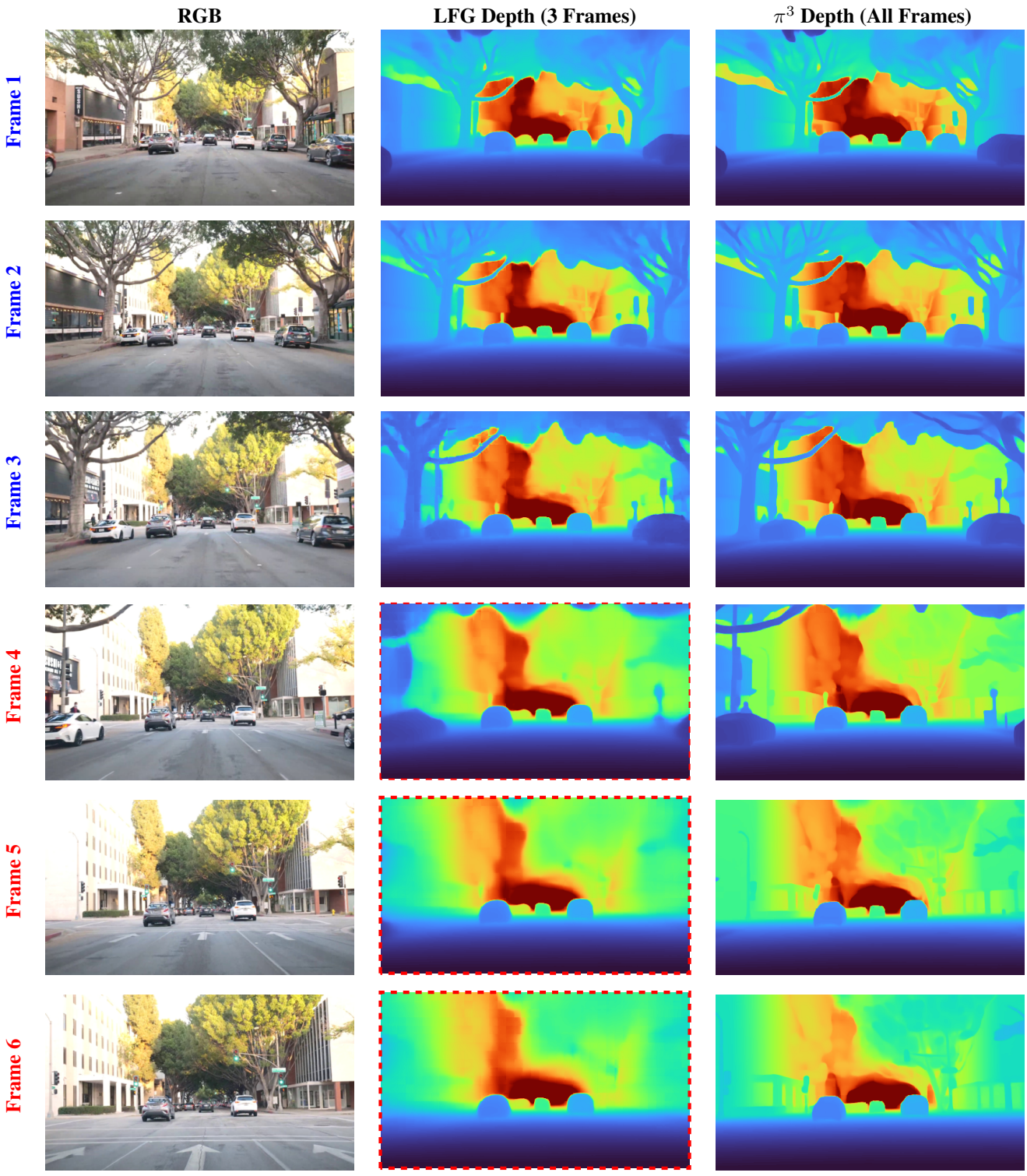


Figure A4. Qualitative comparison of depth prediction for six frames (first three frames are blue, the future three frames are red. LFG is able to decouple static and dynamic objects as it continues along the road, and future work will improve the sharpness of the last frames' predictions. **Dashed red** outlines denote predicted frames with *no ground-truth image input*, produced solely from the model's future tokens.

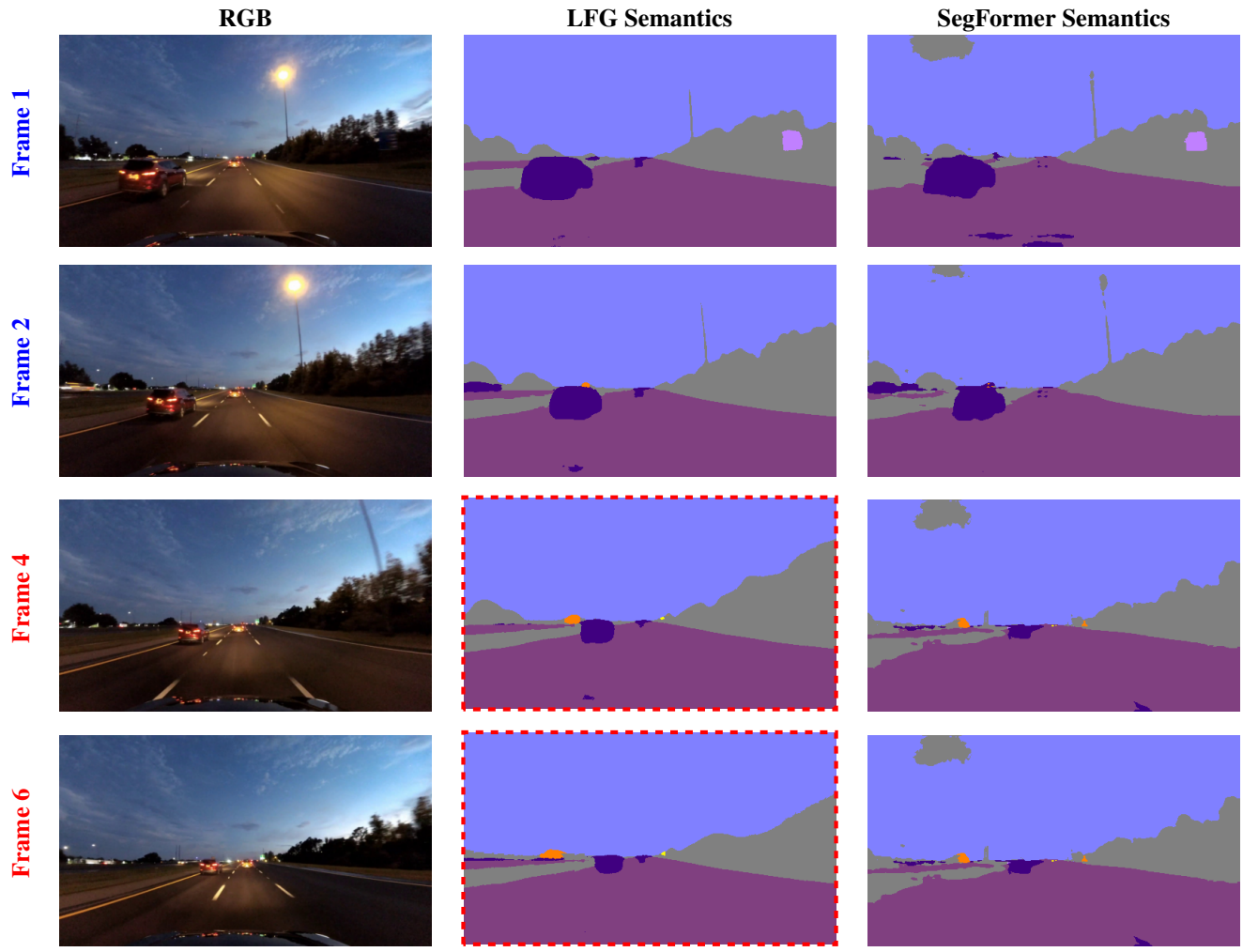


Figure A5. Additional qualitative comparison of semantic segmentation across RGB, LFG, and SegFormer for current frames 1 and 2 (with ground-truth input) and future frames 4 and 6. **Dashed red** outlines denote predicted frames with *no ground-truth image input*, produced solely from the model's future tokens. LFG on current frames enjoys crisper predictions even than its teacher.

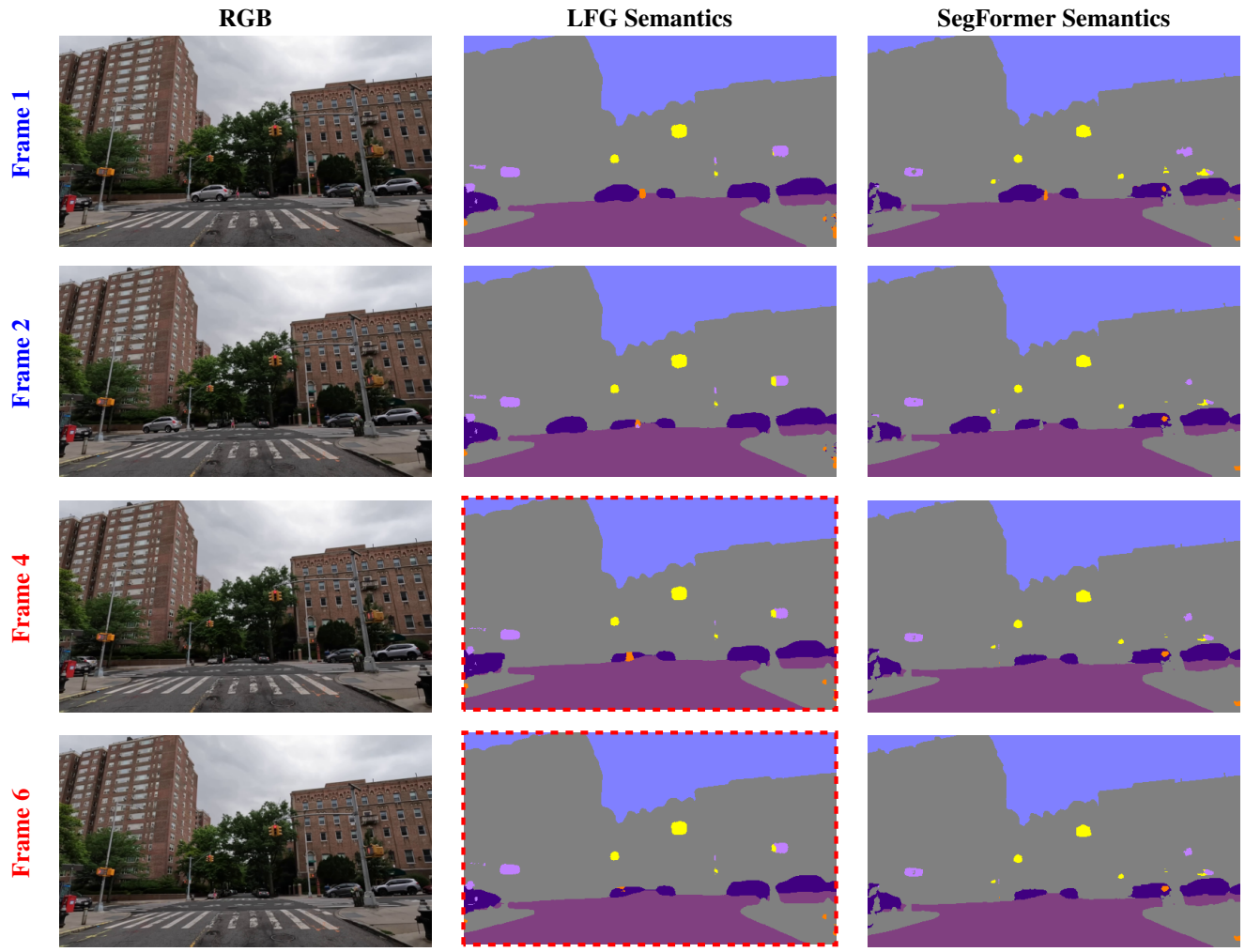


Figure A6. Additional qualitative comparison of semantic segmentation across RGB, LFG, and SegFormer for current frames 1 and 2 (with ground-truth input) and future frames 4 and 6. **Dashed red** outlines denote predicted frames with *no ground-truth image input*, produced solely from the model's future tokens. LFG retains accurate predictions of cars, road, buildings, sky, traffic lights and signs, and even a person.

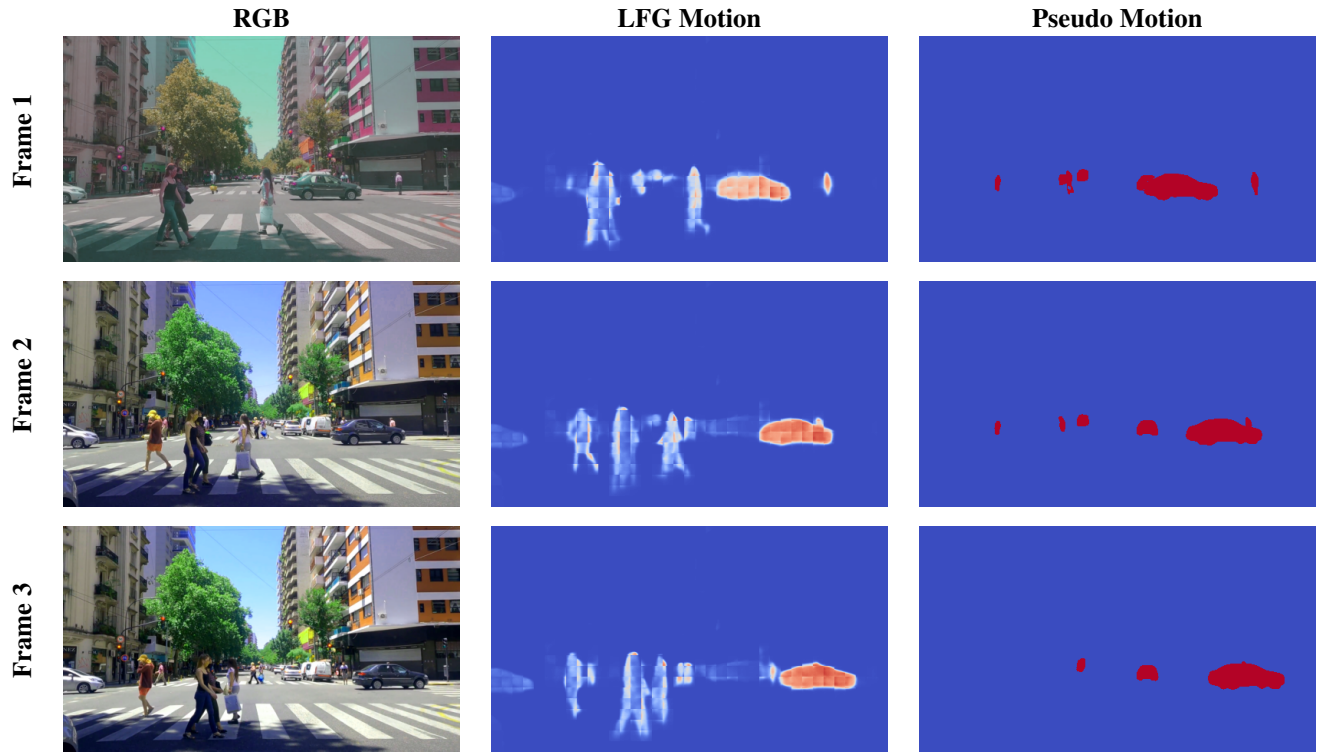


Figure A7. More qualitative comparison of motion predictions (LFG vs Pseudo) with corresponding RGB frames. In this scene, LFG predicts the moving car across the intersection, but also the close pedestrians, demonstrating that the pretrained point decoders of $\pi^3$ improve the predictions.

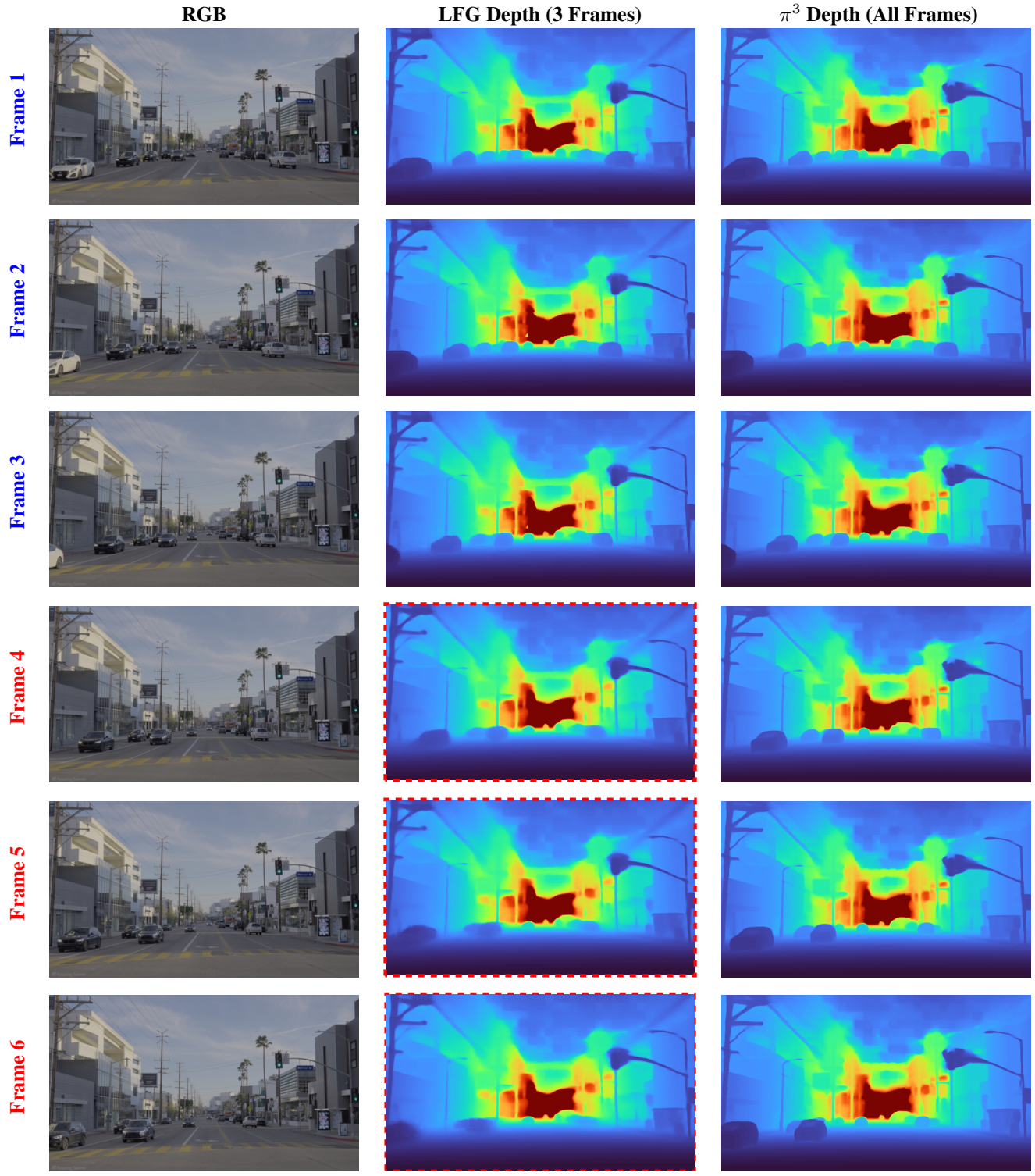


Figure A8. Qualitative comparison of depth prediction for six frames. LFG is able to decouple static and dynamic objects as it continues along the road, and future work will improve the sharpness of the last frames' predictions. **Dashed red** outlines denote predicted frames with *no ground-truth image input*, produced solely from the model's future tokens

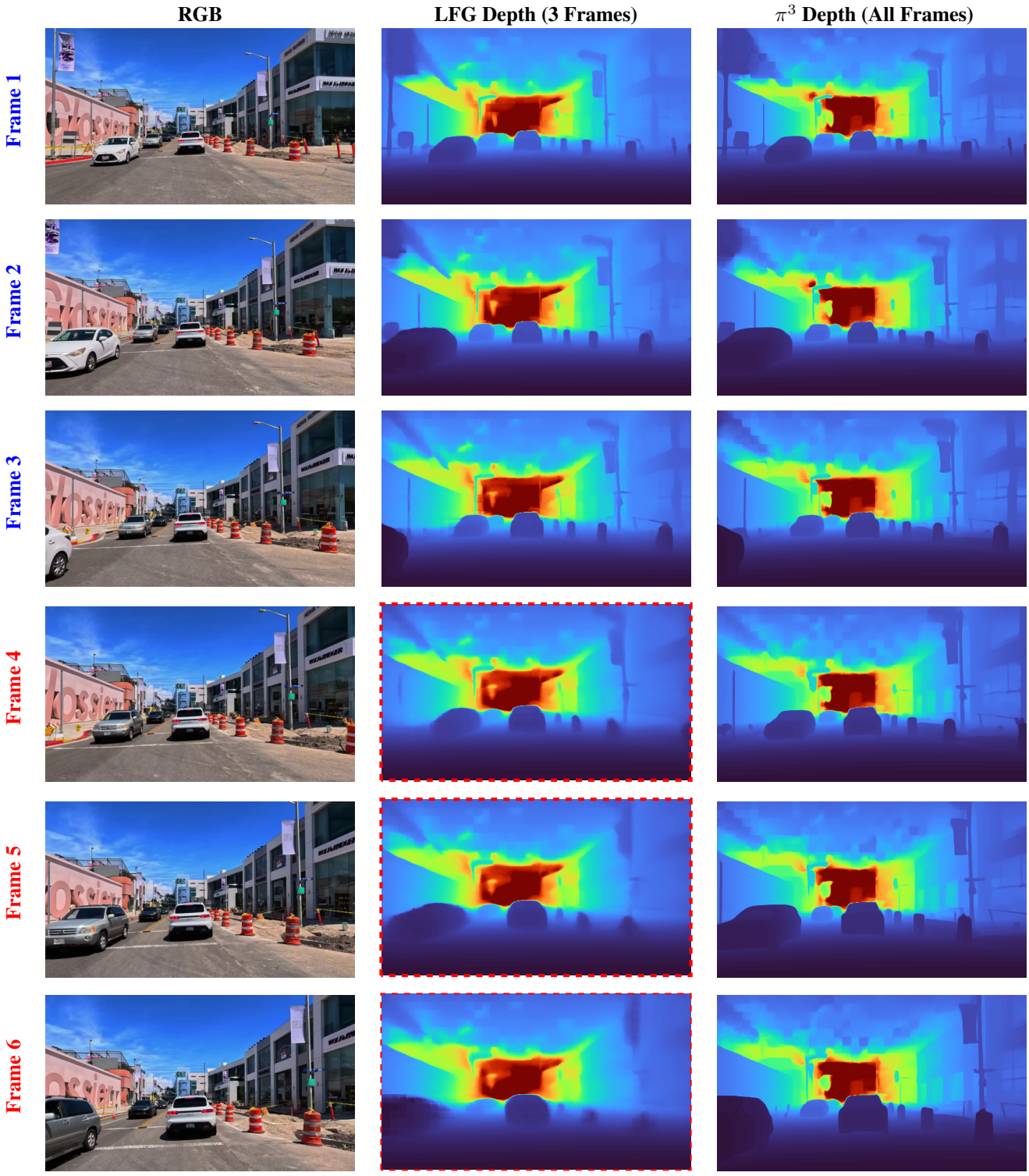


Figure A9. More qualitative comparison of depth prediction for six frames. **Dashed red** outlines denote predicted frames with *no ground-truth image input*, produced solely from the model's future tokens