STREAMTTO: EFFICIENT ONLINE TEST-TIME OPTIMIZATION FOR VIDEO DEPTH COMPLETION

Anonymous authors
Paper under double-blind review

ABSTRACT

Monocular depth foundation models generalize across diverse scenes, but recovering accurate metric depth consistent with a target sensor remains challenging under sensor variation and domain shift. Test-time optimization (TTO) guided by sparse depth addresses this limitation, yet independent per-frame adaptation incurs substantial computational cost. We propose StreamTTO, an online test-time optimization framework for video depth completion that reuses visual features and adaptation state. A frozen feature extractor with causal temporal attention incorporates past visual context into features cached for repeated decoder optimization. Causal Sliding-Window Optimization directly updates a shared depth decoder using cached features and sparse observations from recent frames, requiring only two passes over the window per incoming frame after initialization. Reusing recent observations and the adapted decoder substantially reduces optimization steps. We also introduce MS-Depth, comprising approximately 1.45M synchronized RGB–LiDAR frames from 40 hours of recordings. Its uninterrupted sequences capture transitions among indoor, underground, and outdoor environments under daytime and nighttime conditions, enabling evaluation of adaptation to changes in illumination, scene structure, and depth range. Experiments on KITTI, Bonn, NYUv2 Raw, TUM RGB-D, DDAD, and MS-Depth demonstrate competitive depth completion accuracy. On MS-Depth, StreamTTO achieves average processing speeds of approximately 15 FPS with VGGT and 25 FPS with MoGe-2 at an input resolution of 518×392.

1 INTRODUCTION

Accurate metric depth is important for autonomous driving, robotic navigation, and 3D reconstruction at physical scale (Rajapaksha et al., 2024; Geiger et al., 2013). Systems often obtain this information using range sensors such as LiDAR, but hardware and acquisition constraints limit measurement coverage and density (Ma & Karaman, 2018; Ma et al., 2019). Depth completion fills these gaps by estimating dense depth from sparse observations. RGB-guided methods learn to combine these measurements with visual cues about image boundaries and scene structure (Ma & Karaman, 2018; Cheng et al., 2018; Tang et al., 2020; Park et al., 2020; Zhang et al., 2023).

Meanwhile, monocular depth foundation models have substantially improved dense depth estimation across diverse scenes from RGB images alone (Yang et al., 2024; Piccinelli et al., 2025; Wang et al., 2026a). Beyond individual images, VGGT (Wang et al., 2025a) and π^3 (Wang et al., 2026b) reconstruct geometry from multiple views, while Video Depth Anything (Chen et al., 2025) incorporates temporal context for consistent video depth estimation. StreamVGGT (Zhuo et al., 2026) further enables online geometry reconstruction through causal attention and cached representations of past frames. Despite richer visual and temporal context, these models do not guarantee accurate physical distances. Image projections alone do not uniquely determine absolute scale, while learned scale cues can be inaccurate in unfamiliar environments (Wang et al., 2025a; 2026a). Sparse sensor observations provide a physical distance reference, while visual priors help reconstruct regions that sparse measurements alone cannot resolve. Recent depth completion methods exploit this complementarity by combining foundation-model priors with sparse observations (Lin et al., 2025; Wang et al., 2025c; Liu et al., 2024; Viola et al., 2025).

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.

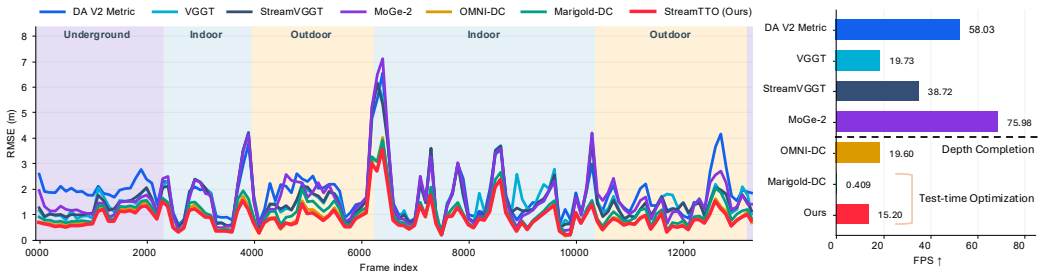

Figure 1: Depth accuracy and processing speed on MS-Depth TE05, with approximately 13,000 frames spanning indoor, outdoor, and underground environments. StreamTTO achieves the best accuracy among the evaluated methods at 15.2 FPS after initial adaptation.

However, applying this combination across sensors and environments requires accommodating variations in measurement density, spatial distribution, and noise caused by sensor configurations and processing pipelines. Although training-based methods have improved robustness to such variation (Zuo et al., 2025; Bartolomei et al., 2024), learned fusion can remain sensitive to observation conditions outside the training distribution (Viola et al., 2025; Ke et al., 2026). Test-time optimization (TTO) reduces this dependence by using sparse observations as direct supervision to optimize latent representations, prompts, or model parameters at inference time (Viola et al., 2025; Jeong et al., 2025; Ke et al., 2026). However, many existing methods optimize each image independently, repeating iterative adaptation across consecutive video frames and incurring substantial computational cost (Viola et al., 2025; Jeong et al., 2025). This limits their throughput and motivates reusing past computation and adaptation state to reduce the optimization required for each incoming frame.

Figure 1 illustrates the accuracy–speed trade-off among the evaluated methods, highlighting the computational burden of test-time optimization (TTO). We propose StreamTTO to address this bottleneck through an empirically validated design principle: jointly reuse visual computation and adaptation progress across frames. This reduces both per-update cost and the updates needed for online video depth completion.

A frozen causal feature extractor incorporates past visual context and computes features once per frame, preserving cache validity during decoder optimization. The shared decoder and its optimizer state retain earlier depth corrections. Causal Sliding-Window Optimization (CSWO) connects these forms of reuse, updating the decoder using cached features and sparse observations from current and recent frames. The framework performs only two window passes per incoming frame after initialization.

To evaluate adaptation under sustained environmental changes, we introduce Multi-Scene Depth (MS-Depth): 120 sequences of approximately 20 minutes each, totaling approximately 40 hours of synchronized RGB–LiDAR recordings. Test sequences span indoor, underground, and outdoor environments under daytime and nighttime conditions. We evaluate StreamTTO on KITTI (Geiger et al., 2013), Bonn (Palazzolo et al., 2019), NYUv2 Raw (Silberman et al., 2012), TUM RGB-D (Sturm et al., 2012), DDAD (Guizilini et al., 2020), and MS-Depth. Ablations and same-backbone comparisons support the components' complementary roles. StreamTTO achieves the lowest RMSE among the evaluated methods on KITTI, NYUv2 Raw, and DDAD, and averages approximately 15 FPS with VGGT and 25 FPS with MoGe-2 Wang et al. (2026a) on MS-Depth at 518×392 resolution.

2 Related Work

Training-based depth completion. Sparse-to-Dense (Ma & Karaman, 2018), CSPN (Cheng et al., 2018), GuideNet (Tang et al., 2020), NLSPN (Park et al., 2020), and CompletionFormer (Zhang et al., 2023) learn RGB–depth fusion or spatial propagation. To improve cross-domain generalization, VPP4DC (Bartolomei et al., 2024) reformulates completion as stereo matching, OMNI-DC (Zuo et al., 2025) combines synthetic training with multi-resolution depth processing, and LDCM (Yu et al., 2026) predicts 3D points using a transformer. PromptDA (Lin et al., 2025) and Any2Full (Zhou et al., 2026) condition Depth Anything on sparse measurements, while Prior Depth

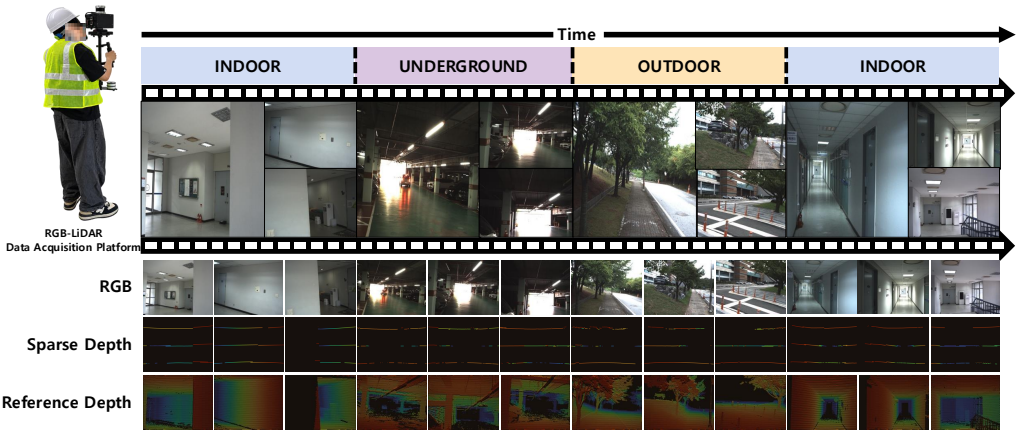

Figure 2: An example MS-Depth test sequence repeatedly traversing indoor, underground, and outdoor environments before returning indoors. Sequences average approximately 20 minutes and are recorded continuously with a calibrated handheld camera–LiDAR rig. RGB images are paired with sparse and dense depth maps.

Anything (Wang et al., 2025c) incorporates incomplete metric priors. With parameters fixed at inference, robustness to new sensors and scenes remains dependent on training-data coverage.

Test-time optimization for depth completion. TTO refines depth using sparse observations at inference. Park et al. (2024) and ETA (Chung et al., 2025) adapt pretrained completion models online using source-trained embedding and energy models, respectively, tying their learned guidance to source observation distributions. Foundation-model-based TTO uses target measurements without such auxiliary guidance: Marigold-DC (Viola et al., 2025) optimizes diffusion latents, scale, and shift, while TestPromptDC (Jeong et al., 2025) tunes visual prompts. However, image-wise prompt optimization incurs substantial computational overhead, as each update requires forward and backward passes through the full network. CAPA-LoRA (Ke et al., 2026) reduces the number of trainable parameters by updating only LoRA adapters in the image encoder. Nevertheless, its default adaptation procedure uses 100 optimization steps, requiring repeated encoder feature computation and backpropagation. StreamTTO adapts causally as frames arrive, freezing the feature extractor and optimizing only the decoder using target sparse observations, without separately trained auxiliary guidance. Reusing cached features with causal temporal context, recent observations in a sliding window, and the adapted decoder reduces both the number of optimization steps and the computational cost per incoming frame.

Continual depth completion. Continual depth adaptation includes selective module updates for throughput in MADNet (Tonioni et al., 2019), analyses of forgetting on revisited domains (Poggi et al., 2021), and frame replay for stable online self-supervised monocular adaptation (Kuznietsov et al., 2021; Vödisch et al., 2023). For depth completion, UnCLe (Chen et al., 2024) benchmarks unsupervised continual learning across datasets, while ProtoDepth (Rim et al., 2025) learns domain-specific prototypes over a frozen model to mitigate forgetting. These settings use photometric and sparse-depth losses for sequential learning, emphasizing retention of earlier domains for subsequent multi-domain reuse. StreamTTO continuously adapts during inference within a single sensor stream under a limited computational budget. Its adapted decoder serves only the current stream, with no adaptation state transferred to independent sequences. Preserving performance on earlier environments is not an explicit objective; the key criterion is depth accuracy in the current environment, including after environmental transitions, under a limited computational budget.

3 MS-Depth Benchmark

MS-Depth (Multi-Scene Depth) benchmarks video depth completion under continuous environmental changes. It comprises 120 approximately 20-minute sequences totaling approximately 40 hours

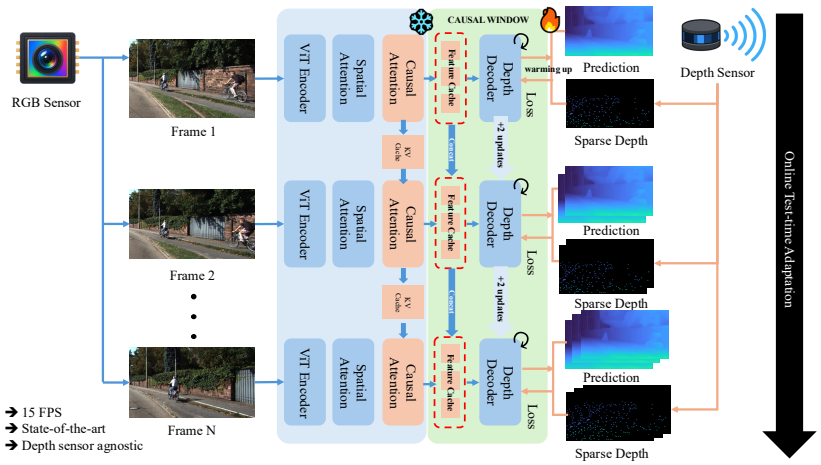

Figure 3: Overview of StreamTTO. A frozen ViT encoder with spatial and causal temporal attention extracts features using a key–value cache for past context. A separate feature cache stores the extracted features for batched optimization over a 3-frame causal window. Sparse depth observations supervise updates to the shared depth decoder, whose parameters are carried forward to subsequent frames. Only the decoder is optimized, avoiding repeated feature extraction.

of synchronized RGB–LiDAR recordings. The training, validation, and test splits contain 107, 1, and 12 sequences, respectively; the test split covers eight locations/routes.

A rigid handheld platform carries an Ouster OS-1-128-SR 128-channel LiDAR operating at 10 Hz and three identical Point Grey Research Blackfly BFLY-PGE-31S4C cameras. The central camera provides 2048×1536 RGB inputs; the other two form a stereo pair with a 20 cm baseline. All sensors are hardware-synchronized by a trigger board. Intrinsic and camera–LiDAR extrinsic calibration yield an image-plane calibration error of at most 0.9 pixels. Recordings exclude moving pedestrians. Reference depth is generated by projecting a single LiDAR scan into the central image, without multi-scan accumulation. We triangulate RoMa v2 Edstedt et al. (2025) correspondences to obtain stereo depth and discard LiDAR samples with missing correspondences, left–right consistency errors exceeding 1 pixel, or depth differences exceeding 10% of the LiDAR depth. Sparse inputs retain eight LiDAR rings at fixed ring-index intervals.

Each test sequence continuously traverses indoor and outdoor spaces before returning to its starting point; eight test sequences and the validation sequence also include underground segments (Figure 2). These recordings capture changes in illumination, scene structure, and depth range. MS-Depth provides RGB images, sparse inputs, reference depth maps, IMU measurements, sequence-level day/night labels, and temporal annotations of indoor, outdoor, and underground segments. We use the validation split for method development and the test split for final evaluation. Sparse observations supervise online adaptation, while reference depth is used to assess accuracy. The training split supports future research.

4 StreamTTO

4.1 Overview

We consider online video depth completion, where an RGB frame I_t and registered sparse metric depth S_t arrive sequentially. StreamTTO separates temporal representation from online adaptation: a frozen spatiotemporal extractor supplies reusable visual features, while a shared 2D depth decoder accumulates corrections from sparse observations (Figure 3). This decoder-only adaptation confines repeated forward and backward computation to the depth decoder, leaving cached representations unchanged by optimization.

The framework maintains two forms of memory. *Representation memory* consists of a temporal key–value cache and a separate decoder-input feature cache. Let $\mathcal{F}$ denote the frozen RGB encoder

with spatial and causal temporal attention. Given the previous key–value memory $\mathcal{V}_{t-1}$, it produces

$$(Z_t, \mathcal{V}_t) = \mathcal{F}(I_t, \mathcal{V}_{t-1}), \tag{1}$$

where Z_t encodes the current image and past visual context. We cache Z_t so that recent observations can be revisited without repeating feature extraction.

Adaptation memory consists of $\Xi_t = (\theta_t, \mathcal{O}_t)$, where θ_t contains the parameters of the shared depth decoder $\mathcal{D}$ and $\mathcal{O}_t$ is its optimizer state. Causal Sliding-Window Optimization (CSWO) updates this state using recent cached features stacked along the batch dimension. Temporal reasoning remains in $\mathcal{F}$; the 2D decoder requires no additional temporal operations. The completed depth is

$$\hat{D}_t = \mathrm{Align}_{S_t}[\mathcal{D}(Z_t; \theta_t)], \tag{2}$$

where Align_{S_t} estimates global scale and shift from the sparse observations.

4.2 Sparse-Depth-Guided Decoder Adaptation

Let $D_t(\theta) = \mathcal{D}(Z_t; \theta)$ and $\Omega_t = \mathrm{dom}(S_t)$ denote the decoder prediction and valid sparse-observation locations, respectively. For each prediction, scale and shift are estimated by least-squares fitting:

$$(a_t^\star, b_t^\star) = \arg\min_{a,b} \sum_{p \in \Omega_t} \left(a\ \mathrm{sg}[D_t(\theta)(p)] + b - S_t(p)\right)^2, \tag{3}$$

where sg denotes stop-gradient. The frame-wise adaptation objective is

$$\mathcal{L}_t^{\mathrm{sp}}(\theta) = \frac{1}{|\Omega_t|} \sum_{p \in \Omega_t} \rho(a_t^\star D_t(\theta)(p) + b_t^\star - S_t(p)), \qquad \rho(e) = e^2 + \lambda|e|, \tag{4}$$

where λ weights the absolute residual penalty. Alignment is recomputed before each loss evaluation, but $a_t^\star$ and $b_t^\star$ are held constant during backpropagation. Gradients pass through the aligned prediction to all decoder parameters, not through the alignment solver. Thus, adaptation can correct spatially varying depth structure rather than only global scale and shift. We clip the global L_2 norm of decoder gradients before each optimizer update.

Warm-starting from the previous decoder reuses learned corrections, but fitting only the current observation set provides no explicit constraint on their compatibility with recent frames. CSWO retains these constraints in a causal buffer $\mathcal{B}_t$ containing up to $W = 3$ cached feature–depth pairs (Z_j, S_j). The buffer grows from the first frame and then replaces its oldest entry at each arrival.

Starting from pretrained decoder parameters and a fresh optimizer state, we perform $K_{\mathrm{boot}} = 20$ bootstrap updates on the first frame. Subsequent frames inherit the adapted decoder and optimizer state and perform $R = 2$ passes over the available window. Each pass processes the entire window as one batch and applies one optimizer update. The batch loss is the normalized weighted average of $\mathcal{L}_j^{\mathrm{sp}}$, assigning weight 2 to the current frame and 1 to each past frame.

Consecutive full windows share $W - 1$ observations, so the inherited decoder has already been adapted to most of the next objective. Revisiting these constraints supports retaining corrections across frames while incorporating new measurements. Unlike optimizing an available sequence jointly, CSWO uses only current and past observations, with bounded memory and two decoder updates per subsequent frame.

StreamTTO is designed to reduce both the cost of each optimization update and the number of updates required per incoming frame. Appendix D formalizes the consistency of cached causal representations and bounds the change in the decoder's backpropagated gradient across overlapping windows, providing a conditional rationale for warm-start adaptation with few updates.

5 Experiments

5.1 Experimental Setup

Datasets. We evaluate on 110-frame clips sampled from eight KITTI (1216×352), five Bonn RGB-D Dynamic, eight NYUv2 Raw (640×480), and eight TUM RGB-D (640×480) sequences,

Table 1: Quantitative comparison of depth foundation models, 3D foundation models, and depth completion methods on five public benchmarks. RMSE and MAE are reported in meters, excluding sparse-input observation locations from accuracy evaluation. Bold and underlined values indicate the best and second-best results across all methods, respectively, including ties.

Method	KITTI		Bonn		NYUv2 Raw		TUM RGB-D		DDAD		Speed
	RMSE↓	MAE↓	RMSE↓	MAE↓	RMSE↓	MAE↓	RMSE↓	MAE↓	RMSE↓	MAE↓	Avg. FPS↑
Depth Foundation Models											
Depth Anything V2 Metric	4.1902	2.1348	0.2278	0.1380	0.1660	0.1060	0.2705	0.1410	15.7768	8.7141	26.68
UniDepthV2	2.9397	1.2993	0.1832	0.0811	0.1327	0.0698	0.2674	0.1265	13.2580	5.7289	**37.47**
Video Depth Anything Metric (online)	4.3396	2.3805	0.1992	0.1125	0.2513	0.1380	0.2793	0.1490	15.4656	7.8687	35.53
3D Foundation Models											
VGGT	2.8888	1.2200	0.1856	0.0860	0.1861	0.0886	0.2191	0.1024	12.4564	6.1351	9.46
MoGe-2	2.8805	1.2522	0.1645	0.0775	0.1222	0.0707	0.2320	0.1083	10.4588	4.3600	<u>35.88</u>
StreamVGGT	2.9495	1.3631	0.1655	0.0781	0.1587	0.0794	0.2051	0.0962	12.7959	6.5728	17.47
ABot-Recon (Han et al., 2026)	2.5400	1.0041	0.2083	0.0935	0.2276	0.1174	0.1844	0.0915	9.3542	3.3378	20.27
Depth Completion Models											
PriorDA v1.1	1.8406	0.6121	0.0845	0.0268	0.0730	0.0306	0.0920	0.0242	8.4179	2.4995	23.75
OMNI-DC v1.1	1.3557	0.4219	0.0700	**0.0146**	0.0526	**0.0147**	0.0563	**0.0113**	7.6406	1.7980	9.07
Marigold-DC	1.7106	0.6890	0.0723	0.0249	0.0731	0.0315	0.0837	0.0273	9.9222	3.8292	0.221
TestPromptDC	1.1266	0.3441	<u>0.0514</u>	0.0190	0.0495	0.0229	**0.0514**	0.0163	8.2693	2.2167	0.256
CAPA-LoRA (offline)†	1.0902	**0.2684**	0.0566	0.0179	0.0546	0.0219	<u>0.0560</u>	<u>0.0144</u>	7.6213	<u>1.7592</u>	0.780
VGGT + Causal-Attn	3.0817	1.3334	0.1723	0.0792	0.1602	0.0832	0.2150	0.0988	12.7059	6.6541	17.52
VGGT + StreamTTO	<u>1.0223</u>	0.3920	0.0592	0.0210	<u>0.0403</u>	0.0220	0.0738	0.0274	<u>7.1448</u>	2.3483	7.14
MoGe-2 + Causal-Attn	2.8130	1.2086	0.1641	0.0793	0.1194	0.0732	0.2414	0.1075	10.2409	4.1452	33.28
MoGe-2 + StreamTTO	**0.7796**	<u>0.2763</u>	**0.0430**	<u>0.0155</u>	**0.0316**	<u>0.0173</u>	0.0586	0.0195	**5.2864**	**1.7124**	11.99

totaling 880, 550, 880, and 880 frames, respectively. For DDAD (1936×1216), we use all 3,950 front-camera frames from the 50 scenes in its official validation split. Details of public-benchmark sequence selection and clip construction are provided in Appendix C. For MS-Depth, we use the validation sequence VAL01 for method development, hyperparameter selection, and ablation studies. We evaluate on the 12 test sequences (TE01–TE12) using the hyperparameters selected on VAL01, without further tuning, and report the results in Table 2. These 12 sequences are excluded from method development and hyperparameter selection. MS-Depth RGB images are resized to 518×392 for model input. Further details of MS-Depth are provided in Appendix B.

Sparse-depth inputs. For KITTI, DDAD, and MS-Depth, we project measurements from at least 8 LiDAR rings onto the RGB image plane, keeping the ring selection fixed per sequence. For Bonn, NYUv2 Raw, and TUM RGB–D, we uniformly sample 10% of valid reference-depth pixels without replacement. NYUv2 Raw results at 3% and 1% density are provided in Appendix A.8. All methods requiring sparse depth receive identical observations; evaluation uses only the remaining valid reference-depth pixels.

Implementation details. We use official code and checkpoints, including VGGT (Wang et al., 2025a) and MoGe-2 (Wang et al., 2026a). CAPA-LoRA is evaluated offline using its official optimization settings. All other methods operate causally. Additional scale–shift alignment is applied only to depth and 3D foundation-model outputs. StreamTTO freezes the feature extractor and updates the full shared decoder using a 6-frame causal KV context and a 3-frame CSWO window. After $K_{\text{boot}} = 20$ first-frame updates, it performs $R = 2$ window passes per subsequent frame, retaining decoder and optimizer states. We use AdamW with learning rate 3×10^{-5}, $\lambda = 0.1$, and gradient-norm clipping at 1.0. We select W, R, and the learning rate on VAL01 and keep them fixed across all evaluation datasets. Although the MoGe-2 variant achieves higher accuracy and throughput in our main experiments, all ablations use VGGT with causal training following StreamVGGT (Zhuo et al., 2026). We choose VGGT to assess the proposed adaptation framework on a general-purpose 3D foundation model, beyond backbones specialized for monocular metric-depth estimation. All experiments run on NVIDIA B200 GPUs. Further implementation and ablation details are provided in Appendix A.2.

5.2 Quantitative Results

Public Benchmarks. Table 1 shows that most depth completion methods outperform foundation models calibrated with the same sparse observations. MoGe-2-based StreamTTO achieves the lowest RMSE on KITTI, Bonn, NYUv2 Raw, and DDAD. TestPromptDC leads in TUM RGB-D RMSE, while OMNI-DC achieves the lowest MAE on the three indoor benchmarks. MoGe-2 improves both

Table 2: Results on the 12 MS-Depth test sequences. RMSE and MAE (m) exclude shared sparse-input locations. Depth and 3D foundation models use sparse-depth-based scale–shift alignment; depth completion methods receive no additional alignment.

Method	MS-Depth								FPS↑
	Indoor				Outdoor				
	Underground		Aboveground		Day		Night		
	RMSE↓	MAE↓	RMSE↓	MAE↓	RMSE↓	MAE↓	RMSE↓	MAE↓	
Depth Foundation Models									
Depth Anything V2 Metric	2.3864	1.7842	1.7433	0.8399	2.7577	1.6428	2.4214	1.3059	59.52
UniDepthV2	1.4090	0.8418	1.6569	0.5920	3.6521	2.3448	2.9081	1.6813	**83.16**
Video Depth Anything Metric (online)	1.6474	1.0254	1.6382	0.6428	3.8411	2.6001	3.0138	1.8128	78.29
3D Foundation Models									
VGGT	1.5202	0.8582	1.7391	0.6478	2.2774	1.2286	2.5653	1.4303	20.39
MoGe-2	1.3088	0.7583	1.7428	0.6427	2.5355	1.4393	2.4911	1.3567	78.22
StreamVGGT	1.5370	0.9679	1.6938	0.6296	2.2854	1.2491	2.4262	1.3171	39.81
ABot-Recon (Han et al., 2026)	1.3468	0.7188	1.7619	0.6690	2.3350	1.2698	2.2844	1.1888	47.27
Depth Completion Models									
PriorDA v1.1	0.9434	0.3200	1.1660	0.2491	1.5273	0.5098	1.3949	0.4290	54.42
OMNI-DC v1.1	0.8746	**0.2046**	1.1042	0.1815	1.4891	0.3515	1.3572	0.2927	17.84
Marigold-DC	0.9806	0.3386	1.1964	0.2999	1.8212	0.6611	1.5550	0.5359	0.426
TestPromptDC	0.9487	0.2958	1.1235	0.2346	1.5795	0.4385	1.4509	0.4366	0.543
CAPA-LoRA (offline)	0.9062	0.2441	1.2494	0.2175	1.4502	0.3590	1.3923	0.3300	1.652
VGGT + Causal-Attn	1.5222	0.9521	1.8435	0.6425	2.3414	1.4592	2.3252	1.3433	39.84
VGGT + StreamTTO	0.8077	0.2610	1.0728	0.2211	1.3666	0.3668	1.2603	0.3497	15.73
MoGe-2 + Causal-Attn	1.5088	0.7783	1.9442	0.7325	2.7352	1.5499	2.7917	1.6565	70.25
MoGe-2 + StreamTTO	**0.5899**	0.2063	**0.8339**	**0.1647**	**0.9950**	**0.2779**	**0.8846**	**0.2558**	25.90

RMSE and MAE over the VGGT variant on all five benchmarks, potentially benefiting from its pretraining for metric geometry estimation (Wang et al., 2026a).

The VGGT and MoGe-2 variants achieve average throughputs of 7.14 and 11.99 FPS, respectively. Both use two window passes per frame after initialization, allowing the lighter MoGe-2 backbone to increase throughput with the same number of optimizer updates. Despite performing backpropagation and parameter updates, the MoGe-2 variant also outpaces training-based OMNI-DC (9.07 FPS), which keeps its parameters fixed at inference. It achieves approximately 54, 47, and 15 times the throughput of Marigold-DC, TestPromptDC, and offline CAPA-LoRA, respectively. Further results with VGGT and MoGe-2, including comparisons of adaptation strategies on the same backbone, are provided in Appendix A.7.

MS-Depth Dataset. MS-Depth reveals a different performance trend from the public benchmarks under continuous environmental changes (Table 2). TestPromptDC outperforms OMNI-DC in RMSE on four of the five public benchmarks, but this ranking reverses across all four MS-Depth conditions. StreamTTO maintains its accuracy advantage, with the MoGe-2 and VGGT variants achieving the lowest and second-lowest RMSE, respectively, in all four conditions. With only two window passes per incoming frame after initialization, the VGGT and MoGe-2 variants achieve throughputs of 15.73 and 25.90 FPS, respectively. Taken together with the public-benchmark results, these findings provide evidence that sparse-depth-guided test-time optimization can help depth foundation models adapt to diverse sensing conditions and changing environments. Further comparisons of adaptation strategies using the same VGGT and MoGe-2 backbones are provided in Appendix A.7.

5.3 Qualitative Results

Figure 4 compares baselines with VGGT-based StreamTTO. Predicted depth is back-projected into RGB-colored point clouds, with ground-truth points overlaid in gray. In the indoor staircase example, VGGT and StreamVGGT exhibit visible metric-scale discrepancies relative to the reference. UniDepthV2, MoGe-2, and TestPromptDC also show geometric misalignment, whereas OMNI-DC, CAPA-LoRA, and StreamTTO more closely reproduce the staircase geometry.

Outdoors, OMNI-DC exhibits severe vertical compression, while TestPromptDC and CAPA-LoRA distort the building height and facade. VGGT and StreamVGGT better preserve the overall building structure in this example. In these examples, StreamTTO is the only compared method that achieves close indoor metric alignment while preserving the outdoor building geometry.



Figure 4: Qualitative 3D comparisons for an indoor staircase and an outdoor building. Predicted depth is back-projected into 3D and colored using the corresponding RGB values. Reference geometry is shown in gray for comparison of scale and structure.

5.4 Ablation Studies

We conduct the component ablations in Table 3 with VGGT on the MS-Depth VAL01 validation sequence and report mean RMSE over 10 seeds. Unless otherwise specified, we use a 6-frame causal KV context, full-decoder adaptation, warm-start with two passes per frame, a CSWO window of $W = 3$, and 20 bootstrap updates.

Table 3a shows that warm-start adaptation benefits more from temporal context than from single-frame processing. With causal KV context, warm-start reduces RMSE from 1.3725 to 0.8694 m, reaching accuracy comparable to recomputed context (0.8718 m) at approximately one third of the FLOPs.

Tables 3b and 3c compare adaptation targets and parameterizations, including encoder-LoRA as used in CAPA (Ke et al., 2026). Full-decoder adaptation achieves lower RMSE than encoder-only updates or joint image-encoder and decoder updates at substantially lower computational cost. Decoder-LoRA slightly reduces computation but increases RMSE from 0.8694 to 0.9205 m, whereas full-LoRA requires $20.8\times$ as many FLOPs for similar accuracy.

Table 3d shows that additional updates have different effects under reset and warm-start adaptation. Increasing passes from 2 to 20 improves reset RMSE from 1.3725 to 0.8753 m, but worsens warm-start RMSE from 0.8694 to 0.9846 m. Warm-start with two passes therefore achieves accuracy comparable to reset with 20 passes at substantially lower cost. Appendix A.9 further evaluates warm-start adaptation in other TTO methods. Across the five public benchmarks, warm-starting reduces RMSE by an average of 2.50% for TestPromptDC and 5.72% for CAPA-LoRA relative to their original configurations. The complete VGGT-based StreamTTO framework achieves an average RMSE reduction of 63.35% relative to its causal backbone without test-time parameter updates. These averages are computed from the relative RMSE reduction on each benchmark.

Table 3e evaluates the CSWO window size. Increasing W from 1 to 3 reduces RMSE from 0.9314 to 0.8694 m. Expanding to $W = 6$ provides a further 1.6% reduction but increases FLOPs by 57.8%, while $W = 12$ worsens RMSE to 0.9544 m. We therefore use $W = 3$ to balance accuracy and computation.

Tables 3f and 3g show that extending KV context beyond 6 frames provides negligible accuracy gains, while 20–100 bootstrap updates yield similar sequence-level RMSE.

Table 3: StreamTTO ablations on MS-Depth VAL01 with the VGGT backbone. RMSE (m) is reported as mean ± standard deviation over 10 seeds. In the adaptation-target ablation, "full" denotes joint updates to the image encoder and depth decoder, while the spatial and causal temporal attention modules remain frozen. FLOPs (T/frame) include feature extraction and test-time adaptation for each configuration at an input resolution of 518×392. Default settings are highlighted in gray: 6-frame causal KV context, full-decoder adaptation, warm-start with 2 passes, $W = 3$, and 20 bootstrap updates.

(a) **Temporal context.**

	RMSE↓ (m)			FLOPs↓ (T/frame)
Context	Scale–shift	Reset	Warm-start	Warm-start
Single frame	1.4213	1.2778 ± 0.0352	1.1333 ± 0.0514	6.496
6 frames, recomputed	1.2883	1.1591 ± 0.0280	0.8718 ± 0.0401	20.727
6 frames, causal KV	1.5063	1.3725 ± 0.0189	0.8694 ± 0.0532	7.028

(b) **Adaptation target.**

target	#params	RMSE↓	FLOPs↓
encoder	304.372M	0.9306 ± 0.0287	149.538
decoder	32.655M	0.8694 ± 0.0532	7.028
full	337.026M	0.9271 ± 0.0492	150.892

(c) **Low-rank adaptation.**

target	#params	RMSE↓	FLOPs↓
encoder-LoRA	0.393M	0.8884 ± 0.0157	145.768
decoder-LoRA	0.357M	0.9205 ± 0.0536	5.719
full-LoRA	0.750M	0.8706 ± 0.0271	145.885

(d) **Initialization.**

init	passes	RMSE↓	FLOPs↓
reset	2	1.3725 ± 0.0189	7.028
reset	20	0.8753 ± 0.0526	43.595
warm-start	2	0.8694 ± 0.0532	7.028
warm-start	20	0.9846 ± 0.0232	43.595

(e) **CSWO window.**

W	RMSE↓	FLOPs↓
1	0.9314 ± 0.0102	4.320
3	0.8694 ± 0.0532	7.028
6	0.8551 ± 0.0272	11.091
12	0.9544 ± 0.0548	19.217

(f) **KV context length.**

KV context	RMSE↓	FLOPs↓
1	1.1333 ± 0.0514	6.496
3	0.8918 ± 0.0308	6.709
6	0.8694 ± 0.0532	7.028
12	0.8697 ± 0.0540	7.668

(g) **Bootstrap updates.**

K_{boot}	RMSE↓
20	0.8694 ± 0.0532
40	0.8727 ± 0.0572
60	0.8749 ± 0.0363
100	0.8684 ± 0.0130

Table 4: Post-transition RMSE (m, ↓) over $[0, 10)$ s on 51 transition clips from the 12 MS-Depth test sequences. Results are reported as mean ± standard deviation across 10 random seeds. W1 and W3 use matched decoder training FLOPs, which exclude frozen feature extraction.

Method	Indoor → Outdoor	Outdoor → Indoor	Indoor → Underground	Underground → Indoor	Outdoor → Underground	Underground → Outdoor	Overall	FPS↑	Training FLOPs↓ (T/frame)
W1 (6 updates)	1.0363 ± 0.0600	1.5833 ± 0.0469	1.1038 ± 0.0116	0.9515 ± 0.0435	0.8663 ± 0.0338	0.7513 ± 0.0368	1.2146 ± 0.0360	10.5	**4.063**
W3 (2 passes)	$\mathbf{0.9762 \pm 0.0241}$	$\mathbf{1.4970 \pm 0.0386}$	$\mathbf{1.0047 \pm 0.0268}$	$\mathbf{0.8789 \pm 0.0376}$	$\mathbf{0.7470 \pm 0.0424}$	$\mathbf{0.6459 \pm 0.0406}$	$\mathbf{1.1340 \pm 0.0286}$	**15.7**	**4.063**

Adaptation across environmental transitions. We compare W1 and W3 under matched decoder training FLOPs on 51 predefined transition clips from the 12 MS-Depth test sequences. Each clip is initialized approximately 4 seconds before the transition and processed until 10 seconds after it, retaining decoder parameters and optimizer state without resetting at $t = 0$. RMSE is evaluated only over the post-transition interval $[0, 10)$ seconds. Both configurations share the same frozen backbone and 6-frame causal KV context. W1 performs 6 current-frame updates, whereas W3 performs 2 passes over a 3-frame window, matching decoder training FLOPs at 4.063 T per incoming frame once the window is full.

Table 4 shows that W3 lowers RMSE across all six transition types, reducing overall RMSE from 1.2146 to 1.1340 m (6.6%) while increasing throughput from 10.5 to 15.7 FPS. These results support the benefit of reusing recent observations for post-transition depth accuracy under the same decoder training budget.

6 Conclusion

We presented StreamTTO, an online framework for video depth completion that exploits temporal redundancy to reuse both visual computation and depth corrections. By caching temporally contextualized features and updating a shared depth decoder through Causal Sliding-Window Optimization, StreamTTO replaces repeated image-wise optimization with brief, cumulative updates across a video. Experiments on five public benchmarks and MS-Depth demonstrate competitive depth accuracy, while our ablations show that combining temporal memory with window optimization can outperform independent 20-step TTO at substantially lower computational cost. We also introduced MS-Depth, a 40-hour dataset spanning diverse environments and continuous scene transitions, to support the study of long-horizon adaptation. Our results highlight that efficient video depth completion depends not only on reducing optimization steps, but also on making previous computation and corrections reusable across frames. Future work will focus on improving processing speed and adaptation robustness under prolonged environmental changes. Code and data will be released.

Ethics Statement

MS-Depth was collected by the authors in public places, with assistance from participants who were compensated for their time and effort. Faces and vehicle license plates were blurred to reduce privacy risks. The data collection and preprocessing procedures are described in Section 3 and Appendix B.

Reproducibility Statement

The proposed method and experimental setup are described in Section 4 and Section 5.1, respectively. Appendix A.2 provides additional implementation details, including offline fine-tuning, online adaptation, and runtime measurement. Dataset specifications and evaluation subset construction are documented in Appendices B and C. The assumptions and derivations underlying our analysis of cached online decoder adaptation are provided in Appendix D. The supplementary materials include selected RGB–LiDAR overlay visualizations from the MS-Depth validation sequence VAL01 and JSON annotations for VAL01 and all 12 test sequences, describing day/night conditions and environment transitions within each sequence. Figure 7 illustrates the temporal structure of the validation and test sequences. We will publicly release the source code and the full MS-Depth dataset upon acceptance.

AI Use Statement

We used AI tools to assist with language editing and translation, code cleanup, and dataset organization and cleaning, and to obtain feedback on manuscript drafts. The core research ideas, methodology, experiments, and implementation were developed by the authors. All final judgments and decisions were made by the authors, who take full responsibility for the final manuscript, code, and dataset.

References

Luca Bartolomei, Matteo Poggi, Andrea Conti, Fabio Tosi, and Stefano Mattoccia. Revisiting depth completion from a stereo matching perspective for cross-domain generalization. In *2024 international conference on 3D vision (3DV)*, pp. 1360–1370. IEEE, 2024.

Gilad Baruch, Zhuoyuan Chen, Afshin Dehghan, Tal Dimry, Yuri Feigin, Peter Fu, Thomas Gebauer, Brandon Joffe, Daniel Kurz, Arik Schwartz, et al. Arkitscenes: A diverse real-world dataset for 3d indoor scene understanding using mobile rgb-d data. *arXiv preprint arXiv:2111.08897*, 2021.

Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual kitti 2. *arXiv preprint arXiv:2001.10773*, 2020.

Sili Chen, Hengkai Guo, Shengnan Zhu, Feihu Zhang, Zilong Huang, Jiashi Feng, and Bingyi Kang. Video depth anything: Consistent depth estimation for super-long videos. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22831–22840. IEEE, 2025.

Xien Chen, Rit Gangopadhyay, Michael Chu, Patrick Rim, Hyoungseob Park, and Alex Wong. Uncle: Benchmarking unsupervised continual learning for depth completion. *arXiv preprint arXiv:2410.18074*, 2024.

Xinjing Cheng, Peng Wang, and Ruigang Yang. Depth estimation via affinity learned with convolutional spatial propagation network. In *European Conference on Computer Vision*, pp. 108–125. Springer, 2018.

Younjoon Chung, Hyoungseob Park, Patrick Rim, Xiaoran Zhang, Jihe He, Ziyao Zeng, Safa Cicek, Byung-Woo Hong, James S Duncan, and Alex Wong. Eta: Energy-based test-time adaptation for depth completion. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1–12. IEEE, 2025.

Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 2432–2443. IEEE, 2017.

Johan Edstedt, David Nordström, Yushan Zhang, Georg Bökman, Jonathan Astermark, Viktor Larsson, Anders Heyden, Fredrik Kahl, Mårten Wadenbäck, and Michael Felsberg. Roma v2: Harder better faster denser feature matching. *arXiv preprint arXiv:2511.15706*, 2025.

Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The international journal of robotics research*, 32(11):1231–1237, 2013.

Vitor Guizilini, Rares Ambrus, Sudeep Pillai, Allan Raventos, and Adrien Gaidon. 3d packing for self-supervised monocular depth estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2485–2494, 2020.

Jiarong Han, Jincheng Xiong, Yuzhou Liu, Linzhe Shi, Changjie Wu, Ning Guo, Mu Xu, Hang Zhang, and Ming Qian. Revisiting local context for long-horizon streaming 3d reconstruction. *arXiv preprint arXiv:2608.27529*, 2026.

Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. Deepmvs: Learning multi-view stereopsis. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2821–2830. IEEE, 2018.

Chanhwi Jeong, Inhwan Bae, Jin-Hwi Park, and Hae-Gon Jeon. Test-time prompt tuning for zero-shot depth completion. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9443–9454. IEEE, 2025.

Bingxin Ke, Qunjie Zhou, Jiahui Huang, Xuanchi Ren, Tianchang Shen, Konrad Schindler, Laura Leal-Taixé, and Shengyu Huang. Depth completion as parameter-efficient test-time adaptation. *arXiv preprint arXiv:2602.14751*, 2026.

Yevhen Kuznietsov, Marc Proesmans, and Luc Van Gool. Comoda: Continuous monocular depth adaptation using past experiences. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 2906–2916. IEEE, 2021.

Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2041–2050. IEEE, 2018.

Haotong Lin, Sida Peng, Jingxiao Chen, Songyou Peng, Jiaming Sun, Minghuan Liu, Hujun Bao, Jiashi Feng, Xiaowei Zhou, and Bingyi Kang. Prompting depth anything for 4k resolution accurate metric depth estimation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17070–17080. IEEE, 2025.

Zhiheng Liu, Ka Leong Cheng, Qiuyu Wang, Shuzhe Wang, Hao Ouyang, Bin Tan, Kai Zhu, Yujun Shen, Qifeng Chen, and Ping Luo. Depthlab: From partial to complete. *arXiv preprint arXiv:2412.18153*, 2024.

Fangchang Ma and Sertac Karaman. Sparse-to-dense: Depth prediction from sparse depth samples and a single image. In *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 4796–4803. IEEE, 2018.

Fangchang Ma, Guilherme Venturelli Cavalheiro, and Sertac Karaman. Self-supervised sparse-to-dense: Self-supervised depth completion from lidar and monocular camera. In *2019 international conference on robotics and automation (ICRA)*, pp. 3288–3295. IEEE, 2019.

Lukas Mehl, Jenny Schmalfuss, Azin Jahedi, Yaroslava Nalivayko, and Andrés Bruhn. Spring: A high-resolution high-detail dataset and benchmark for scene flow, optical flow and stereo. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4981–4991. IEEE, 2023.

Emanuele Palazzolo, Jens Behley, Philipp Lottes, Philippe Giguere, and Cyrill Stachniss. Refusion: 3d reconstruction in dynamic environments for rgb-d cameras exploiting residuals. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7855–7862. IEEE, 2019.

Hyoungseob Park, Anjali Gupta, and Alex Wong. Test-time adaptation for depth completion. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20519–20529. IEEE, 2024.

Jinsun Park, Kyungdon Joo, Zhe Hu, Chi-Kuei Liu, and In So Kweon. Non-local spatial propagation network for depth completion. In *European conference on computer vision*, pp. 120–136. Springer, 2020.

Luigi Piccinelli, Christos Sakaridis, Yung-Hsu Yang, Mattia Segu, Siyuan Li, Wim Abbeloos, and Luc Van Gool. Unidepthv2: Universal monocular metric depth estimation made simpler. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

Matteo Poggi, Alessio Tonioni, Fabio Tosi, Stefano Mattoccia, and Luigi Di Stefano. Continual adaptation for deep stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44 (9):4713–4729, 2021.

Uchitha Rajapaksha, Ferdous Sohel, Hamid Laga, Dean Diepeveen, and Mohammed Bennamoun. Deep learning-based depth estimation methods from monocular image and videos: A comprehensive survey. *ACM computing surveys*, 56(12):1–51, 2024.

Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10881–10891. IEEE, 2021.

Patrick Rim, Hyoungseob Park, Suchisrit Gangopadhyay, Ziyao Zeng, Younjoon Chung, and Alex Wong. Protodepth: Unsupervised continual depth completion with prototypes. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6304–6316. IEEE, 2025.

Mike Roberts, Jason Ramapuram, Anurag Ranjan, Atulit Kumar, Miguel Angel Bautista, Nathan Paczan, Russ Webb, and Joshua M Susskind. Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10892–10902. IEEE, 2021.

Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *European conference on computer vision*, pp. 746–760. Springer, 2012.

Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of rgb-d slam systems. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 573–580. IEEE, 2012.

Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2443–2451. IEEE, 2020.

Jie Tang, Fei-Peng Tian, Wei Feng, Jian Li, and Ping Tan. Learning guided convolutional network for depth completion. *IEEE Transactions on Image Processing*, 30:1116–1129, 2020.

Yifu Tao, Miguel Ángel Muñoz-Bañón, Lintong Zhang, Jiahao Wang, Lanke Frank Tarimo Fu, and Maurice Fallon. The oxford spires dataset: Benchmarking large-scale lidar-visual localisation, reconstruction and radiance field methods. *The International Journal of Robotics Research*, 45 (6):839–857, 2026.

Alessio Tonioni, Fabio Tosi, Matteo Poggi, Stefano Mattoccia, and Luigi DI Stefano. Real-time self-adaptive deep stereo. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 195–204, 2019.

Massimiliano Viola, Kevin Qu, Nando Metzger, Bingxin Ke, Alexander Becker, Konrad Schindler, and Anton Obukhov. Marigold-dc: Zero-shot monocular depth completion with guided diffusion. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5359–5370. IEEE, 2025.

Niclas Vödisch, Kürsat Petek, Wolfram Burgard, and Abhinav Valada. Codeps: Online continual learning for depth estimation and panoptic segmentation. *arXiv preprint arXiv:2303.10147*, 2023.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5294–5306. IEEE, 2025a.

Qin Wang, Olga Fink, Luc Van Gool, and Dengxin Dai. Continual test-time domain adaptation. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7191–7201. IEEE, 2022.

Renhao Wang, Yu Sun, Arnuv Tandon, Yossi Gandelsman, Xinlei Chen, Alexei A Efros, and Xiaolong Wang. Test-time training on video streams. *Journal of Machine Learning Research*, 26(9): 1–29, 2025b.

Ruicheng Wang, Sicheng Xu, Yue Dong, Yu Deng, Jianfeng Xiang, Zelong Lv, Guangzhong Sun, Xin Tong, and Jiaolong Yang. Moge-2: Accurate monocular geometry with metric scale and sharp details. *Advances in Neural Information Processing Systems*, 38:35928–35959, 2026a.

Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. π^3: Permutation-equivariant visual geometry learning. In *International Conference on Learning Representations*, volume 2026, pp. 10481–10497, 2026b.

Zehan Wang, Siyu Chen, Lihe Yang, Jialei Wang, Ziang Zhang, Hengshuang Zhao, and Zhou Zhao. Depth anything with any prior. *arXiv preprint arXiv:2505.10565*, 2025c.

Hexiang Wei, Jianhao Jiao, Xiangcheng Hu, Jingwen Yu, Xupeng Xie, Jin Wu, Yilong Zhu, Yuxuan Liu, Lujia Wang, and Ming Liu. Fusionportablev2: A unified multi-sensor dataset for generalized slam across diverse platforms and scalable environments. *The International Journal of Robotics Research*, 44(7):1093–1116, 2025.

Tong Wu, Jiarui Zhang, Xiao Fu, Yuxin Wang, Jiawei Ren, Liang Pan, Wayne Wu, Lei Yang, Jiaqi Wang, Chen Qian, et al. Omniobject3d: Large-vocabulary 3d object dataset for realistic perception, reconstruction and generation. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 803–814. IEEE, 2023.

Hongchi Xia, Yang Fu, Sifei Liu, and Xiaolong Wang. Rgbd objects in the wild: Scaling real-world 3d object learning from rgb-d videos. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22378–22389. IEEE, 2024.

Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *Advances in neural information processing systems*, 37:21875–21911, 2024.

Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1787–1796. IEEE, 2020.

Zhu Yu, Runmin Zhang, Lingteng Qiu, Kejie Qiu, Yisheng He, Siyu Zhu, Zilong Dong, Si-Yuan Cao, Hui-liang Shen, et al. Large depth completion model from sparse observations. In *International Conference on Learning Representations*, volume 2026, pp. 119433–119471, 2026.

Youmin Zhang, Xianda Guo, Matteo Poggi, Zheng Zhu, Guan Huang, and Stefano Mattoccia. Completionformer: Depth completion with convolutions and vision transformers. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18527–18536. IEEE, 2023.

Yuhao Zhang, Mihai Bujanca, and Mikel Luján. Ngd-slam: Towards real-time dynamic slam without gpu. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3467–3473. IEEE, 2025.

Yang Zheng, Adam W Harley, Bokui Shen, Gordon Wetzstein, and Leonidas J Guibas. Pointodyssey: A large-scale synthetic dataset for long-term point tracking. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 19798–19808. IEEE, 2023.

Zhiyuan Zhou, Ruofeng Liu, Taichi Liu, Weijian Zuo, Shanshan Wang, Zhiqing Hong, and Desheng Zhang. Any to full: Prompting depth anything for depth completion in one stage. In *European Conference on Computer Vision*, pp. 442–462. Springer, 2026.

Dong Zhuo, Wenzhao Zheng, Jiahe Guo, Yuqi Wu, Jie Zhou, and Jiwen Lu. Streaming visual geometry transformer. In *International Conference on Learning Representations*, volume 2026, pp. 88055–88072, 2026.

Yiming Zuo, Willow Yang, Zeyu Ma, and Jia Deng. Omni-dc: Highly robust depth completion with multiresolution depth integration. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9287–9297. IEEE, 2025.

Appendix

A Implementation Details and Additional Experiments

A.1 Broader implications

Test-time optimization (TTO) allows a deployed depth completion system to adapt using sparse sensor measurements without revisiting its original training data. A central practical challenge is fitting iterative adaptation within the computational budget of a continuous perception system. Even when updates are restricted to input prompts or low-rank adapters, repeated optimization can remain expensive. StreamTTO addresses this cost by reusing both visual representations and accumulated adaptation across frames.

The underlying design principle is simple: freeze a causal feature extractor, cache its representations, and carry a shared decoder and its optimizer state forward while adapting to a short window of recent observations. This reduces both repeated feature computation and the additional optimization required for each incoming frame. During decoder adaptation, cached features remain consistent with recomputation under the same causal execution policy. At fixed decoder parameters, the change between consecutive window gradients is bounded by differences in the relevant per-frame gradients. When the preceding window gradient is small and the relevant per-frame gradients vary little, the inherited decoder also has a small gradient on the new window. This provides a conditional explanation for reusing adaptation across overlapping windows.

The experiments show the practical value of this reuse. On the VAL01 validation sequence, warm-start adaptation with a three-frame window and two passes achieves accuracy comparable to reset adaptation with 20 passes, using one tenth as many optimizer updates per subsequent frame. The diagnostic experiments further show that additional updates can improve the fit to sparse observations while degrading accuracy at unobserved locations. These findings support a small update budget once adaptation state is carried across frames.

On MS-Depth at 518×392 resolution, StreamTTO achieves 15.73 FPS with VGGT and 25.90 FPS with MoGe-2 on an NVIDIA B200 GPU. The MoGe-2 variant achieves approximately $47.7\times$, $60.8\times$, and $15.7\times$ the throughput of TestPromptDC, Marigold-DC, and offline CAPA-LoRA, respectively, while attaining the lowest RMSE among the evaluated methods across all four MS-Depth conditions. These results support the feasibility of sustained sparse-depth-guided TTO for streaming perception under the tested resolution and hardware. More broadly, jointly reusing representations and adaptation state may benefit other tasks with causal features, efficiently adaptable prediction heads, and sparse online supervision. Online 3D reconstruction with intermittent range measurements and pose estimation from sparse correspondences are promising directions for testing this principle.

A.2 StreamTTO Implementation Details

Implementation Details for Adaptation Ablations. For the adaptation-target and parameterization ablations in Tables 3b and 3c, we tune the learning rate separately for each variant on the MS-Depth VAL01 validation sequence. This includes encoder-only, decoder-only, and joint image-encoder and decoder parameter updates, as well as their LoRA counterparts.

For LoRA, we closely follow the CAPA-LoRA (Ke et al., 2026) configuration used with VGGT, using AdamW with rank $r = 4$ and scaling parameter $\alpha = 8$. The selected learning rate is 10^{-3} for the encoder-LoRA, decoder-LoRA, and full-LoRA variants in Table 3c. Direct updates to pretrained parameters use smaller learning rates, selected separately for each adaptation target; the default decoder-only configuration uses 3×10^{-5}. The LoRA variants retain the causal online adaptation protocol used in our ablations, with the same sparse observations, CSWO window size, and number of optimization passes. The comparisons therefore use separately tuned learning rates while keeping the data and evaluation protocol consistent.

Offline fine-tuning. For our VGGT-based implementation, we follow the causal fine-tuning procedure of StreamVGGT (Zhuo et al., 2026), starting from the public VGGT-1B checkpoint (Wang et al., 2025a). The VGGT backbone uses $L = 24$ spatial and temporal attention layers and

FlashAttention-2 for efficient inference. We freeze the image encoder and depth decoder and fine-tune the attention components for causal processing with cached keys and values. We train for 10 epochs using AdamW, with a 0.5-epoch linear warm-up followed by cosine decay and a peak learning rate of 10^{-6}. Each iteration randomly samples a batch of 10 frames from diverse training scenes, preserving variable aspect ratios while resizing the longest image edge to 518 pixels. Fine-tuning the VGGT-based model takes approximately one day on 16 NVIDIA B200 GPUs.

For MoGe-2 (Wang et al., 2026a), we insert causal attention layers between the pretrained image encoder and depth decoder. These layers aggregate current and past frame features using causal attention, with key–value (KV) caching for efficient streaming inference, and pass the resulting features to the depth decoder. During offline fine-tuning, we freeze both the pretrained image encoder and depth decoder and train only the newly added causal attention layers, using the same optimization settings and training data as the VGGT-based variant.

Training data. For both backbones, we use the same 13-dataset collection as StreamVGGT (Zhuo et al., 2026): Co3Dv2 (Reizenstein et al., 2021), BlendedMVS (Yao et al., 2020), ARKitScenes (Baruch et al., 2021), MegaDepth (Li & Snavely, 2018), WildRGB-D (Xia et al., 2024), ScanNet (Dai et al., 2017), HyperSim (Roberts et al., 2021), OmniObject3D (Wu et al., 2023), MVS-Synth (Huang et al., 2018), PointOdyssey (Zheng et al., 2023), Virtual KITTI (Cabon et al., 2020), Spring (Mehl et al., 2023), and Waymo (Sun et al., 2020). The collection covers indoor and outdoor environments and diverse temporal scales, balancing synthetic data and real-world captures. During subsequent online adaptation, each variant freezes its entire feature extractor, including the image encoder and causal attention layers, and optimizes only its depth decoder, shared across frames, using sparse observations.

MoGe-2 Implementation Details. For MoGe-2, we insert three spatial attention layers followed by three causal temporal attention layers between the pretrained image encoder and depth decoder. All six layers use eight attention heads and preserve the feature dimension of the encoder output. The added layers are randomly initialized, and we introduce no additional positional encoding in these layers. The causal temporal layers incorporate current and past visual context using a 6-frame context, comprising the current frame and five preceding frames, with key–value caching for streaming inference. During offline fine-tuning, we freeze the pretrained image encoder and depth decoder and train only the six added attention layers using the original MoGe-2 training loss. We use the same optimization settings and training data as the VGGT-based variant. During online test-time adaptation, we freeze the image encoder and all attention layers and update only the shared depth decoder using the sparse-depth objective described in Section 4.2.

Runtime measurement. All runtimes are measured on an NVIDIA B200 GPU. For each dataset, FPS is computed as the total number of evaluated frames divided by the total processing time, including StreamTTO's initial 20 bootstrap updates. Table 1 reports the average FPS across the five public benchmarks at their dataset-specific input resolutions, including 1216×352 for KITTI and 1936×1216 for DDAD. Table 2 reports FPS on MS-Depth at 518×392. The higher-resolution inputs in the public benchmarks contribute to the lower average FPS in Table 1 compared with Table 2.

A.3 Qualitative Results

Figures 5 and 6 present qualitative results on MS-Depth at 100-frame intervals. StreamTTO recovers scene geometry consistent with the reference depth across indoor, underground, and outdoor environments, despite changes in illumination and depth range. These examples complement the quantitative results, illustrating the robustness of online adaptation under changing observation conditions.

A.4 Interaction between Window Size and Optimization Passes

We extend the initialization and window-size ablations in Tables 3d and 3e by jointly varying the CSWO window size W and optimization passes R on MS-Depth VAL01 using the VGGT backbone. We compare reset adaptation, which reinitializes the decoder and optimizer at every incoming frame, with warm-start adaptation, which retains both states across frames after initial bootstrap adaptation.

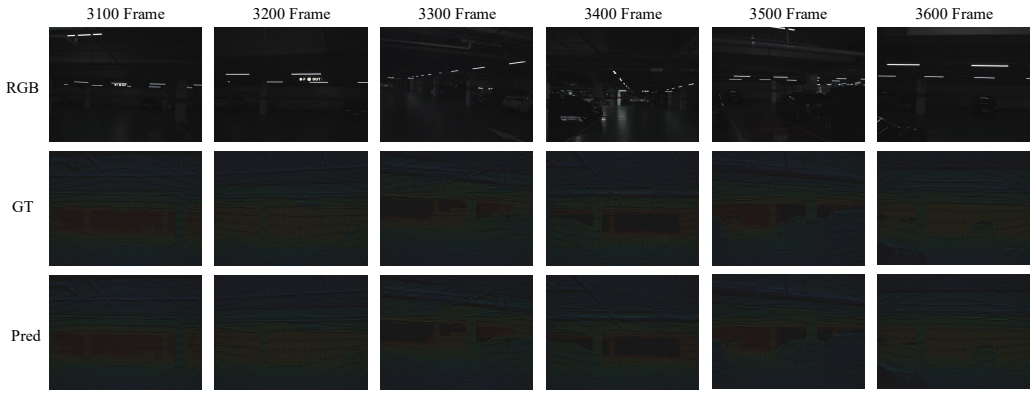

Figure 5: Qualitative results on MS-Depth at 100-frame intervals, showing RGB images, ground-truth depth, and StreamTTO predictions in that order.

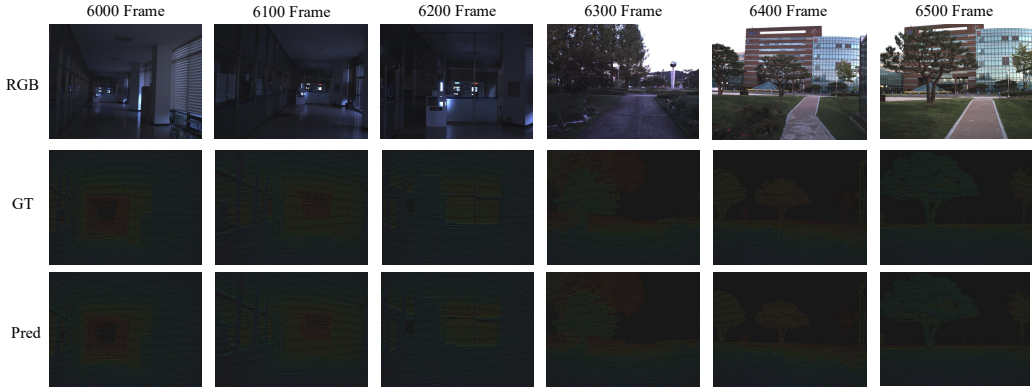

Figure 6: Qualitative results on MS-Depth at 100-frame intervals, showing RGB images, ground-truth depth, and StreamTTO predictions in that order.

Each pass processes the available causal window as one batch and applies one optimizer update. Tables 5 and 6 report RMSE for the two initialization strategies.

Under reset adaptation, larger windows improve the best attainable accuracy when sufficient optimization is allowed (Table 5). The lowest RMSE among the evaluated pass counts decreases from 0.9245 m at $W = 1$ to 0.8600 m at $W = 12$. These minima occur at $R = 20$ for $W \in \{1, 2, 3, 6\}$ and at $R = 40$ for $W = 12$. In contrast, warm-start adaptation reaches its lowest RMSE with substantially fewer passes (Table 6). The best-performing pass count decreases as the window grows: $R = 5$ for $W \in \{1, 2, 3\}$, $R = 2$ for $W = 6$, and $R = 1$ for $W = 12$. At $W = 3$, warm-start adaptation with two passes achieves 0.8694 m, comparable to the 0.8753 m obtained by reset adaptation with 20 passes, using one tenth as many optimizer updates per subsequent frame.

Increasing the update budget beyond these best-performing pass counts degrades warm-start accuracy. For every evaluated multi-frame window ($W \geq 2$), RMSE increases at each tested pass count beyond the row-wise minimum. For example, at $W = 2$, RMSE increases from 0.8869 m at $R = 5$ to 1.5399 m at $R = 20$ and 1.8283 m at $R = 40$. At $W = 12$, the lowest RMSE is 0.8922 m with a single pass, increasing to 0.9544 m with two passes and 1.5117 m with 40 passes. Thus, under the tested settings, additional optimization does not necessarily improve the accuracy of a decoder whose adaptation state is already carried across frames.

These trends are consistent with a reduced need for additional optimization when accumulated adaptation and overlapping window constraints are reused. Consecutive full windows share $W - 1$ obser-

Table 5: Window size and optimization passes under reset adaptation on MS-Depth VAL01 with the VGGT backbone. The decoder and optimizer are reinitialized at every frame. Values are RMSE in meters (lower is better); underlined entries indicate the lowest RMSE in each row.

	Optimization passes R					
W	1	2	5	10	20	40
1	1.4609	1.3786	1.1952	0.9908	<u>0.9245</u>	0.9623
2	1.4583	1.3734	1.1847	0.9723	<u>0.8895</u>	0.9233
3	1.4577	1.3725	1.1831	0.9681	<u>0.8753</u>	0.9039
6	1.4588	1.3745	1.1883	0.9728	<u>0.8615</u>	0.8769
12	1.4623	1.3812	1.2028	0.9927	0.8645	<u>0.8600</u>

Table 6: Window size and optimization passes under warm-start adaptation on MS-Depth VAL01 with the VGGT backbone. The decoder and optimizer states are retained across frames, with R passes per subsequent frame after initial bootstrap adaptation. Values are RMSE in meters (lower is better); underlined entries indicate the lowest RMSE in each row.

	Optimization passes R					
W	1	2	5	10	20	40
1	0.9733	0.9314	<u>0.9038</u>	0.9781	0.9770	0.9758
2	0.9481	0.9004	<u>0.8869</u>	0.9514	1.5399	1.8283
3	0.9233	0.8694	<u>0.8594</u>	0.9704	0.9846	1.2465
6	0.9048	<u>0.8551</u>	0.8812	0.9746	1.0659	1.2872
12	<u>0.8922</u>	0.9544	1.0273	1.2603	1.4860	1.5117

vations, so the inherited decoder has already been adapted to much of the next window's supervision. Window size also changes the balance between current and past observations. Under the weighting scheme in Section 4.2, the current frame receives weight two and each past frame receives weight one. For a full window, the current frame therefore has a coefficient of $2/(W+1)$ in the normalized loss. Larger windows consequently place relatively less weight on the current frame, and additional updates may favor fitting recent observations without improving current-frame accuracy. Accumulated adaptation, overlapping supervision, and this change in relative weighting provide possible explanations for the observed interaction between W and R.

We select $W = 3$ and $R = 2$ as the default configuration to balance accuracy and computational cost. At $W = 3$, increasing R from two to five reduces RMSE from 0.8694 to 0.8594 m, an improvement of approximately 1.15%, but requires $2.5\times$ as many optimizer updates per frame. Similarly, increasing W from three to six at $R = 2$ reduces RMSE from 0.8694 to 0.8551 m, an improvement of approximately 1.6%, while increasing total FLOPs per frame by 57.8% (Table 3e). The default configuration therefore achieves near-best accuracy in this sweep with a smaller per-frame computational budget.

A.5 Diagnostic Analysis of Optimization Passes and Window Size

We investigate the degradation associated with 20-pass warm-start adaptation and a larger optimization window on the full MS-Depth VAL01 validation sequence. Table 7 compares ten configurations, including learning-rate changes, periodic resets, and current-frame reweighting. For these diagnostic experiments, the baseline assigns equal weight to all frames in the optimization window. We report mean per-frame RMSE at sparse-input locations and at the remaining valid evaluation pixels separately.

At $W = 3$, increasing the pass count from 2 to 20 reduces observed RMSE from 0.8162 to 0.2623 m but increases unobserved RMSE from 0.8696 to 0.9773 m (Table 7). This is consistent with excessive fitting to sparse observations that does not generalize to unobserved locations. A lower constant learning rate partially mitigates the degradation, reducing unobserved RMSE to 0.9321 m. The

Table 7: Diagnostic experiments on the full MS-Depth VAL01 validation sequence. Observed and unobserved RMSE are evaluated at sparse-input locations and at the remaining valid evaluation pixels, respectively; underlined values indicate the lowest RMSE in each column. All window frames receive weight 1, except in the W12 reweighting variant, which assigns weight 5.5 to the current frame and 1 to each past frame. Periodic resets are applied every 250 frames throughout the sequence, immediately before adaptation at zero-based frame indices $250, 500, 750, 1000, \ldots$. Optimizer-only reset clears the AdamW state while retaining the decoder; full reset additionally restores the pretrained decoder. Both retain the KV cache, cached features, and optimization window, without additional bootstrap updates. All frames, including those immediately following resets, contribute to the reported mean per-frame RMSE.

			RMSE (m) ↓	
W	R	Configuration	Observed	Unobserved
3	2	Baseline	0.8162	<u>0.8696</u>
3	20	No modification	<u>0.2623</u>	0.9773
3	20	Lower constant LR	0.7454	0.9321
3	20	LR decay	0.4113	0.9955
3	20	Optimizer-only reset	0.2740	0.9869
3	20	Decoder and optimizer reset	0.2823	0.9603
12	2	No modification	0.8696	0.9251
12	2	LR decay	0.8926	0.9309
12	2	Decoder and optimizer reset	0.8664	0.9151
12	2	Current-frame weight 5.5	0.7907	0.8734

tested learning-rate decay and optimizer-only reset do not improve accuracy, while resetting both the decoder and optimizer provides a modest improvement to 0.9603 m.

For $W = 12$, learning-rate decay does not improve accuracy, and resetting both the decoder and optimizer provides only a modest improvement from 0.9251 to 0.9151 m. In contrast, increasing the current-frame weight to 5.5 reduces unobserved RMSE to 0.8734 m. For a full window, this changes the current-frame coefficient in the normalized loss from $1/12$ to $5.5/(5.5 + 11) = 1/3$, matching the W3 baseline used in these diagnostic experiments. The resulting gap to the baseline is 0.0038 m, recovering approximately 93.2% of the original RMSE gap. This supports the contribution of current-frame weighting to the degradation observed with the larger window under the tested conditions.

The accuracy recovery at $W = 12$ retains the computational cost of processing a larger window. The computational analysis in Table 3e shows that increasing W from 3 to 12 at $R = 2$ raises FLOPs per frame from 7.028 to 19.217 T, approximately a $2.73\times$ increase. Reweighting does not reduce the number of frames processed by the decoder. On VAL01, the reweighted W12 configuration also does not improve upon the W3 baseline RMSE of 0.8696 m. These results support our choice of $W = 3$ and $R = 2$, which provides comparable accuracy at substantially lower computational cost.

A.6 Long-Horizon Adaptation Stability

We evaluate the effect of accumulated adaptation on the 12 MS-Depth test sequences. Both experiments use the VGGT backbone with a 6-frame causal KV context, a CSWO window of $W = 3$, and $R = 2$ window passes per incoming frame after initialization. Each sequence begins with $K_{\text{boot}} = 20$ bootstrap updates. Accuracy is evaluated at valid reference-depth pixels, excluding the sparse-input observation locations.

Environmental transitions after accumulated adaptation. We examine whether adaptation accumulated earlier in a recording remains effective after environmental transitions. Using the same 51 transition clips as in Table 4, we compare two initialization strategies. *Local initialization* resets the decoder and optimizer approximately four seconds before each transition and performs 20 bootstrap updates. *Full-stream warm-start* processes the recording from its first frame and carries

Table 8: Post-transition RMSE (m, ↓) on 51 transition clips from the 12 MS-Depth test sequences. Local initialization resets the decoder and optimizer approximately four seconds before each transition and performs 20 bootstrap updates. Full-stream warm-start retains adaptation state from the beginning of each recording. Both configurations use identical visual features and sparse observations.

Initialization	$[0, 1)$ s	$[1, 3)$ s	$[3, 10)$ s	$[0, 10)$ s
Local reset + bootstrap	1.26	1.16	1.11	1.14
Full-stream warm-start	1.24	1.10	1.07	1.09

Table 9: Long-horizon adaptation on the 12 complete MS-Depth test sequences. RMSE (m, ↓) is reported by elapsed time and over the complete recordings. Periodic resets reinitialize the decoder and optimizer every 250 frames while retaining the KV and feature caches. The bootstrap variant performs 20 additional updates at each reset. All configurations use $W = 3$ and $R = 2$ after bootstrap.

Adaptation strategy	0–5 min	5–10 min	10–15 min	≥15 min	Overall
Continuous warm-start	1.06	1.14	1.05	1.11	1.09
Periodic reset	1.15	1.24	1.16	1.22	1.19
Periodic reset + bootstrap	1.05	1.16	1.08	1.14	1.11

the decoder and optimizer states through all subsequent frames without resetting at transitions. Both configurations use the same visual features extracted from the full recording and the same sparse observations. Local initialization retains the KV and feature caches, isolating the effect of the decoder and optimizer adaptation history. We report RMSE over the post-transition intervals $[0, 1)$, $[1, 3)$, and $[3, 10)$ seconds, together with the overall result over $[0, 10)$ seconds.

Table 8 shows that full-stream warm-start achieves lower RMSE in all three post-transition intervals, reducing overall RMSE from 1.14 to 1.09 m. The improvement is present during the first second after the transition and persists over the subsequent intervals. These results suggest that corrections accumulated earlier in the stream remain useful after environmental transitions.

Continuous adaptation versus periodic resets. We further compare continuous warm-start with periodic initialization throughout each of the 12 complete recordings. The continuous configuration retains the decoder and optimizer states for the entire sequence. The periodic-reset configurations reinitialize both states every 250 frames while retaining the KV cache, cached features, and optimization window. We evaluate periodic resets both without additional bootstrap updates and with 20 additional bootstrap updates at each reset. All configurations otherwise use the same $W = 3$ and $R = 2$ adaptation settings. All frames, including those immediately following resets, contribute to the reported metrics. We report RMSE in elapsed-time intervals of 0–5, 5–10, 10–15, and at least 15 minutes, as well as over the complete recordings.

As shown in Table 9, continuous warm-start achieves an overall RMSE of 1.09 m, compared with 1.19 m for periodic resets without additional bootstrap and 1.11 m with bootstrap. Although periodic resets with bootstrap yield slightly lower RMSE during the first five minutes, continuous adaptation performs better in all subsequent intervals. Because scene difficulty changes throughout each recording, we assess accumulated adaptation through comparisons with reset baselines on the same frames. The relative performance of continuous adaptation remains competitive in the later intervals, supporting the retention of adaptation state throughout the evaluated recordings. Together with the transition results in Table 8, these findings support the effectiveness of full-stream warm-start under the evaluated recording durations and environmental changes.

A.7 Contribution Beyond the Backbone

We examine the contribution of StreamTTO through two complementary comparisons: adaptation strategies on the same causal backbone, and CSWO with and without causal temporal attention.

All variants receive identical RGB frames and sparse observations and follow the same evaluation protocol.

First, we compare TestPromptDC (Jeong et al., 2025) and StreamTTO using the same causal VGGT and MoGe-2 backbones. Within each backbone group, the two methods start from the same pre-trained causal-backbone checkpoint and use a 6-frame causal KV context, comprising the current frame and five preceding frames. The corresponding Causal-Attn row reports the causal backbone without test-time parameter updates.

For TestPromptDC, we use the official implementation[1] and retain its original optimization configuration for both causal backbones. At each frame, an additive three-channel visual prompt is initialized to zero and optimized for 50 updates using a fresh AdamW optimizer with a learning rate of 2×10^{-3}. The objective is the sum of the L1 and RMSE losses at sparse observation locations, with least-squares scale–shift alignment recomputed at each iteration. Only the input prompt is updated; all backbone weights remain fixed. StreamTTO instead freezes the feature extractor and updates the shared depth decoder using CSWO with $W = 3$ and $R = 2$ after bootstrap, retaining decoder parameters and optimizer state across frames.

Second, the CSWO-only variants apply the same window-based decoder adaptation without causal temporal attention. They extract visual features independently for each frame, without a temporal KV cache, while retaining the feature buffer used for CSWO. They use the same bootstrap and decoder adaptation settings as StreamTTO, including $W = 3$, $R = 2$, and the retention of decoder parameters and optimizer state across frames. Thus, CSWO still reuses recent observations and adaptation state, but the visual features do not incorporate past frames through causal temporal attention.

Tables 10 and 11 show that StreamTTO's advantages persist when compared with TestPromptDC on the same causal backbone. On the public benchmarks, StreamTTO achieves lower RMSE than TestPromptDC on three of five datasets with VGGT and four of five datasets with MoGe-2, with throughput speedups of $35.5\times$ and $117.5\times$, respectively. On MS-Depth, StreamTTO achieves lower RMSE and MAE in all four environmental conditions with both backbones, with throughput speedups of $25.9\times$ for VGGT and $60.5\times$ for MoGe-2. These results show that backbone choice alone does not explain the observed accuracy and efficiency gains.

Full StreamTTO also achieves lower RMSE and MAE than the CSWO-only variants on every public benchmark and in every MS-Depth environmental condition, for both backbones. Although the CSWO-only variants run faster, incorporating causal temporal attention substantially improves depth accuracy. These findings support combining temporal visual context with the reuse of recent observations and decoder adaptation state: under the evaluated settings, CSWO alone does not recover the accuracy of the complete framework.

A.8 Robustness to Reduced Observation Density

We evaluate robustness to reduced sparse-depth supervision on NYUv2 Raw using observation densities of 3% and 1% of valid reference-depth pixels. We evaluate the VGGT-based StreamTTO variant and follow the main evaluation protocol: all methods receive identical sparse observations at each density, and accuracy is evaluated at the remaining valid reference-depth locations.

Table 12 shows that StreamTTO maintains strong accuracy as the available observations become sparser. Reducing the observation density from 3% to 1%, a 66.7% reduction in observations, increases its RMSE from 0.0541 to 0.0593 m, a 9.6% increase. Its MAE increases from 0.0237 to 0.0254 m, or 7.2%. Under the same density reduction, the RMSE of TestPromptDC and OMNI-DC increases by 25.5% and 32.9%, respectively.

StreamTTO ties for the lowest RMSE with TestPromptDC at 3% observation density and achieves the lowest RMSE among all evaluated methods at 1%. At 1% density, its RMSE of 0.0593 m is 12.7% lower than that of the next-best method, TestPromptDC (0.0679 m). OMNI-DC retains the lowest MAE, while StreamTTO achieves the second-lowest MAE at this density. These results show that the VGGT-based StreamTTO remains effective with substantially fewer sparse observations, with its relative RMSE advantage strengthening over the evaluated density range.

[1] https://github.com/JinhwiPark/TestPromptDC/blob/main/main.py

Table 10: Contribution of the adaptation strategy and causal temporal attention across five public benchmarks. All variants receive identical RGB frames and sparse observations and follow the same evaluation protocol. Within each backbone group, Causal-Attn, Causal-Attn + TestPromptDC, and StreamTTO share the same pretrained causal-backbone initialization and 6-frame causal KV context. Causal-Attn performs no test-time parameter updates; TestPromptDC optimizes an input visual prompt independently per frame. CSWO applies window-based decoder adaptation without causal temporal attention or a temporal KV cache. Both CSWO and StreamTTO use $W = 3$ and $R = 2$ after bootstrap and retain decoder parameters and optimizer state across frames. CAPA-LoRA is excluded because it uses an offline clip-level protocol in our evaluation. RMSE and MAE are reported in meters, excluding sparse-input locations, following Table 1. FPS is averaged over the five benchmarks at their dataset-specific input resolutions on one B200. Bold and underlined values indicate the best and second-best results within this table, including ties.

Method	KITTI		Bonn		NYUv2 Raw		TUM RGB-D		DDAD		Speed
	RMSE↓	MAE↓	RMSE↓	MAE↓	RMSE↓	MAE↓	RMSE↓	MAE↓	RMSE↓	MAE↓	Avg. FPS↑
VGGT + Causal-Attn	3.0817	1.3334	0.1723	0.0792	0.1602	0.0832	0.2150	0.0988	12.7059	6.6541	<u>17.52</u>
VGGT + Causal-Attn + TestPromptDC	1.2449	0.3768	<u>0.0559</u>	<u>0.0208</u>	0.0584	0.0254	**0.0550**	**0.0190**	9.3939	2.6467	0.201
VGGT + CSWO	2.2404	0.8587	0.1330	0.0543	0.0992	0.0621	0.1425	0.0782	8.9450	4.2453	9.491
VGGT + StreamTTO	<u>1.0223</u>	0.3920	0.0592	0.0210	<u>0.0403</u>	<u>0.0220</u>	0.0738	0.0274	7.1448	<u>2.3483</u>	7.14
MoGe-2 + Causal-Attn	2.8130	1.2086	0.1641	0.0793	0.1194	0.0732	0.2414	0.1075	10.2409	4.1452	**33.28**
MoGe-2 + Causal-Attn + TestPromptDC	1.2212	<u>0.3744</u>	0.0602	0.0213	0.0533	0.0247	<u>0.0577</u>	<u>0.0195</u>	9.2037	2.6379	0.102
MoGe-2 + CSWO	1.8369	0.7469	0.1122	0.0500	0.0795	0.0446	0.1550	0.0745	<u>6.4210</u>	2.7939	14.83
MoGe-2 + StreamTTO	**0.7796**	**0.2763**	**0.0430**	**0.0155**	**0.0316**	**0.0173**	0.0586	<u>0.0195</u>	**5.2864**	**1.7124**	11.99

Table 11: Contribution of the adaptation strategy and causal temporal attention on the 12 MS-Depth test sequences. All variants receive identical RGB frames and sparse observations and follow the same evaluation protocol. Within each backbone group, Causal-Attn, Causal-Attn + TestPromptDC, and StreamTTO share the same pretrained causal-backbone initialization and 6-frame causal KV context. Causal-Attn performs no test-time parameter updates; TestPromptDC optimizes an input visual prompt independently per frame. CSWO applies window-based decoder adaptation without causal temporal attention or a temporal KV cache. Both CSWO and StreamTTO use $W = 3$ and $R = 2$ after bootstrap and retain decoder parameters and optimizer state across frames. CAPA-LoRA is excluded because it uses an offline clip-level protocol in our evaluation. RMSE and MAE are reported in meters, excluding sparse-input locations; FPS is measured at 518×392 on one B200. Bold and underlined values indicate the best and second-best results within this table, including ties.

Method	MS-Depth								FPS↑
	Indoor				Outdoor				
	Underground		Aboveground		Day		Night		
	RMSE↓	MAE↓	RMSE↓	MAE↓	RMSE↓	MAE↓	RMSE↓	MAE↓	
VGGT + Causal-Attn	1.5222	0.9521	1.8435	0.6425	2.3414	1.4592	2.3252	1.3433	<u>39.84</u>
VGGT + Causal-Attn + TestPromptDC	1.0995	0.3174	1.2347	0.2536	1.8212	0.5253	1.7266	0.4693	0.608
VGGT + CSWO	1.1234	0.6227	1.4582	0.4388	1.6999	0.8930	1.7881	0.8691	19.46
VGGT + StreamTTO	<u>0.8077</u>	<u>0.2610</u>	<u>1.0728</u>	<u>0.2211</u>	<u>1.3666</u>	<u>0.3668</u>	<u>1.2603</u>	<u>0.3497</u>	15.73
MoGe-2 + Causal-Attn	1.5088	0.7783	1.9442	0.7325	2.7352	1.5499	2.7917	1.6565	**70.25**
MoGe-2 + Causal-Attn + TestPromptDC	1.1233	0.3490	1.2752	0.2717	1.7801	0.4670	1.6584	0.4842	0.428
MoGe-2 + CSWO	1.0184	0.5783	1.2210	0.5757	1.8025	1.1066	1.7783	1.2788	33.22
MoGe-2 + StreamTTO	**0.5899**	**0.2063**	**0.8339**	**0.1647**	**0.9950**	**0.2779**	**0.8846**	**0.2558**	25.90

A.9 Effect of Warm-Start Adaptation on Baseline Methods

We evaluate how warm-start adaptation affects existing test-time optimization methods on the five public benchmarks. Table 13 compares TestPromptDC and CAPA-LoRA with their warm-start variants under the same evaluation settings as Table 1. We also include VGGT + Causal-Attn and VGGT + StreamTTO to show the combined effect of our complete adaptation framework on the causal backbone.

Warm-start adaptation improves both RMSE and MAE on all five benchmarks for TestPromptDC and CAPA-LoRA. Averaged over the benchmark-wise relative reductions, RMSE improves by

Table 12: Depth completion under reduced observation density on NYUv2 Raw. Observation density is the fraction of valid reference-depth pixels provided as input. RMSE and MAE are reported in meters; lower is better. Input observation locations are excluded from evaluation at each density. StreamTTO uses the VGGT backbone. Bold and underlined values indicate the best and second-best results in each column, respectively, including ties.

	Observations 3%		Observations 1%	
Method	RMSE↓	MAE↓	RMSE↓	MAE↓
Depth foundation models				
Depth Anything V2 Metric	0.1660	0.1060	0.1661	0.1061
UniDepthV2	0.1327	0.0698	0.1328	0.0698
Video Depth Anything Metric (online)	0.2514	0.1380	0.2516	0.1381
3D foundation models				
MoGe-2	0.1222	0.0707	0.1223	0.0708
VGGT	0.1861	0.0886	0.1862	0.0888
StreamVGGT	0.1587	0.0794	0.1588	0.0795
ABot-Recon	0.2276	0.1174	0.2277	0.1175
Depth completion models				
PriorDA v1.1	0.0786	0.0316	0.0919	0.0360
OMNI-DC v1.1	<u>0.0583</u>	**0.0163**	0.0775	**0.0230**
Marigold-DC	0.0783	0.0323	0.0976	0.0368
TestPromptDC	**0.0541**	<u>0.0235</u>	<u>0.0679</u>	0.0265
CAPA-LoRA (offline)	0.0680	0.0302	0.0694	0.0308
Our variants				
VGGT + Causal-Attn	0.1687	0.0858	0.1714	0.0886
VGGT + StreamTTO (Ours)	**0.0541**	0.0237	**0.0593**	<u>0.0254</u>

Table 13: Effect of warm-start adaptation on baseline methods, with StreamTTO included for comparison. All experiments follow the same evaluation settings as Table 1.

	KITTI		Bonn		NYUv2 Raw		TUM RGB-D		DDAD	
Method	RMSE↓	MAE↓	RMSE↓	MAE↓	RMSE↓	MAE↓	RMSE↓	MAE↓	RMSE↓	MAE↓
TestPromptDC	1.1266	0.3441	0.0514	0.0190	0.0495	0.0229	0.0514	0.0163	8.2693	2.2167
TestPromptDC + warm-start	1.0994	0.3368	0.0507	0.0175	0.0460	0.0196	0.0512	0.0161	8.1666	2.1951
Error reduction	2.41%	2.12%	1.36%	7.89%	7.07%	14.41%	0.39%	1.23%	1.24%	0.97%
CAPA-LoRA (offline)	1.0902	0.2684	0.0566	0.0179	0.0546	0.0219	0.0560	0.0144	7.6213	1.7592
CAPA-LoRA + warm-start	1.0076	0.2374	0.0540	0.0156	0.0488	0.0185	0.0532	0.0123	7.5613	1.7009
Error reduction	7.58%	11.55%	4.59%	12.85%	10.62%	15.53%	5.00%	14.58%	0.79%	3.31%
VGGT + Causal-Attn	3.0817	1.3334	0.1723	0.0792	0.1602	0.0832	0.2150	0.0988	12.7059	6.6541
VGGT + StreamTTO	1.0223	0.3920	0.0592	0.0210	0.0403	0.0220	0.0738	0.0274	7.1448	2.3483
Error reduction	66.83%	70.60%	65.64%	73.48%	74.84%	73.56%	65.67%	72.27%	43.77%	64.71%

2.50% for TestPromptDC and 5.72% for CAPA-LoRA. These results demonstrate that retaining adaptation state benefits existing TTO methods. The comparison between VGGT + Causal-Attn and VGGT + StreamTTO measures the effect of the complete framework, including decoder adaptation and CSWO, and therefore does not isolate warm-starting.

Read together with the component ablations, these results highlight the complementary roles of StreamTTO's components. Causal temporal context strengthens warm-start adaptation, while CSWO improves the accuracy–computation trade-off by reusing recent sparse observations. Freezing the feature extractor enables these observations to be revisited through cached features, and retaining decoder parameters and optimizer state carries adaptation progress across frames. Their coordinated use makes both visual computation and learned depth corrections reusable, enabling accurate online adaptation with few updates per incoming frame. The component ablations and complete-framework results jointly support the value of this integrated design for improving depth accuracy and computational efficiency.

B Dataset Specifications

We summarize the acquisition characteristics and temporal coverage of the datasets used in our experiments, followed by the recording-level structure of MS-Depth. Table 14 reports dataset-level statistics, whereas Table 15 describes the evaluation subsets.

Table 14: Dataset specifications and video volume. Available group or split statistics are shown in parentheses. All video durations are approximate, using reported durations or estimates from frame counts and capture rates.

Dataset	Environment	Sensing	RGB Resolution	Rate	Total Video Duration / Frame Count
KITTI	Outdoor Driving	Stereo RGB; 64-Channel LiDAR	Sequence-Dependent[2]	10 Hz	1.3 h / 47,937 frames
Bonn	Indoor	ASUS Xtion Pro LIVE RGB-D	640×480	30 Hz	17 min / 30,022 frames
NYUv2 Raw	Indoor	Kinect RGB-D	640×480	20–30 Hz	407,024 frames
TUM RGB-D	Indoor	Kinect / ASUS Xtion	640×480	30 Hz	1 h 16 min / 129,824 frames (Public Pose GT: 49 min / 81,413 frames; Validation: 27 min / 48,411 frames)
DDAD	Outdoor Driving	6 RGB Cameras; Luminar-H2 LiDAR	1936×1216	10 Hz	28 min / 16,600 frames per camera (Train: 21 min / 12,650 frames; Validation: 7 min / 3,950 frames) Test: 3,080 frames
MS-Depth	Indoor / Outdoor / Underground	RGB; 128-Channel LiDAR; IMU	2048×1536	10 Hz	40 h / 1.45 M frames (Train 36 h / 1.3 M frames; Validation 17 min / 10,041 frames; Test 4 h / 138,074 frames)

B.1 Dataset Overview

Public datasets. KITTI provides stereo RGB and LiDAR observations for outdoor driving and reports approximately 6 h of data collection (Geiger et al., 2013). However, its public Raw release is limited to sequences with 3D object annotations or those included in the odometry benchmark's training set. Consequently, the publicly available footage is substantially shorter than the reported 6 h. Direct counts of the public synchronized release yield 47,937 left-camera frames and 47,932 right-camera frames. We report the larger count, corresponding to approximately 1.3 h at 10 Hz.

Bonn provides RGB-D recordings of indoor environments with moving people and objects (Palazzolo et al., 2019). Direct counts of the official release yield 30,022 frames, corresponding to approximately 17 min at the reported capture rate of 30 Hz (Zhang et al., 2025).

NYUv2 Raw contains recordings from 464 indoor scenes (Silberman et al., 2012). We report its officially listed count of 407,024 unlabeled frames. Video duration is omitted because the capture rate varies over time and the official documentation specifies a range of 20–30 Hz.

TUM RGB-D supports tracking and SLAM evaluation across indoor sequences with varying motion, scene texture, and dynamics (Sturm et al., 2012). Our statistics cover 47 sequences with public camera-trajectory ground truth and 33 validation sequences, excluding nine calibration sequences. Direct counts of their released RGB recordings yield 81,413 and 48,411 frames, respectively. The officially listed durations of the 47 sequences sum to approximately 49 min. The validation sequences provide public RGB and depth images, while their reference camera trajectories are withheld and their durations are not listed. We estimate their duration as approximately 27 min from 48,411 recorded frames at 30 Hz. Together, the 80 sequences contain 129,824 frames and cover approximately 1 h 16 min.

DDAD provides camera and LiDAR observations for outdoor driving (Guizilini et al., 2020). The current public release contains 150 training scenes and 50 validation scenes, each approximately 5 s or 10 s long. These splits contain 16,600 frames per camera over approximately 28 min, obtained by summing the reported scene lengths. Six synchronized cameras capture different views of the same scenes, providing 99,600 frames across all views. This combined frame count remains below the 138,074 frames in the MS-Depth test split. DDAD also provides 3,080 test images. Their count is listed separately in Table 14, but they are excluded from its video duration totals because the release documentation does not specify their source video sequences or frame sampling intervals.

[2]KITTI's rectified RGB resolutions are 1224×370, 1226×370, 1238×374, 1241×376, and 1242×375, as specified in the date-specific calibration files of the official raw release (Geiger et al., 2013).

Table 15: Evaluation subsets and frame counts used in our experiments. For each dataset, we report the selected sequences or scenes and the corresponding total frame count. The MS-Depth count covers the 12 test sequences; VAL01 is reserved for validation.

Dataset	Evaluation Subset	Frames
KITTI	8 Sequences $\times$ 110 Frames	880
Bonn	5 Sequences $\times$ 110 Frames	550
NYUv2 Raw	8 Sequences $\times$ 110 Frames	880
TUM RGB-D	8 Sequences $\times$ 110 Frames	880
DDAD	50 Val. Scenes (Front Camera)	3,950
MS-Depth	12 Test Recordings	138,074

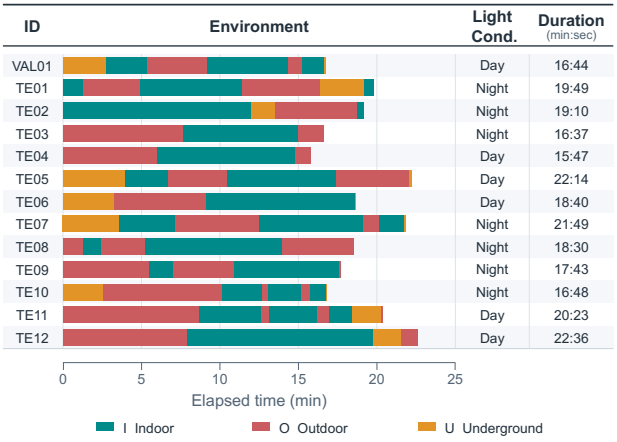

Figure 7: Environment transitions in the MS-Depth validation and test recordings. Each bar shows one continuous recording, with colors indicating environment types and segment widths representing durations on a shared time axis.

MS-Depth. All MS-Depth videos are captured from an egocentric perspective using a handheld sensor platform, in contrast to the vehicle-mounted viewpoints of KITTI and DDAD (Geiger et al., 2013; Guizilini et al., 2020). Every MS-Depth recording contains at least indoor and outdoor segments within a single uninterrupted video, with some also including underground segments. Each recording follows a route that returns to its starting point. These recordings preserve temporal continuity across environmental transitions, enabling evaluation of online adaptation as the observer moves between indoor, outdoor, and underground spaces. The platform combines high-resolution RGB imagery, 128-channel LiDAR, and IMU measurements. The RGB camera has a resolution of 2048×1536, and the Ouster OS-1-128-SR LiDAR operates at 10 Hz. A trigger board synchronizes sensor acquisition. The training split contains 107 recordings, approximately 1.3 million frames, and approximately 36 h of video. The validation split contains one recording with 10,041 frames and approximately 17 min of video. The test split contains 12 recordings, 138,074 frames, and approximately 4 h of video. Our online depth-completion evaluation uses the test split.

B.2 Evaluation Subsets

As described in Section 5.1, we construct sparse depth inputs for KITTI, DDAD, and MS-Depth using a fixed selection of LiDAR scan rings within each sequence, retaining at least 8 rings. For Bonn, NYUv2 Raw, and TUM RGB-D, we sample 10% of valid reference-depth pixels. All methods using sparse depth receive the same observations, and evaluation excludes the observed input locations.

B.3 Temporal Structure of MS-Depth Validation and Test Splits

Recording coverage. The validation and test splits contain one and 12 recordings, respectively, covering eight labeled locations/routes in total. Together, these 13 recordings include six daytime and seven nighttime sequences. Five location/route labels appear under two different capture conditions, supporting comparisons across times of day within the same labeled location or route. Figure 7 shows the environment transitions and duration of each validation and test recording. Individual recordings last approximately 16–23 min.

Environment transitions. Every validation and test recording contains both indoor and outdoor segments, and nine also contain underground segments. The annotations partition all retained frames in the validation and test splits into 70 contiguous environment segments without gaps or overlaps. These segments define 57 within-recording transitions, where a transition is a change in the environ-

Table 16: Dataset characteristics relevant to continuous adaptation. Duration is per recording. Fractions denote qualifying/total sequences; Day/Night gives capture-time counts. I/O denotes indoor–outdoor recordings. MS-Depth statistics cover the validation and test splits.

Dataset	Platform	RGB resolution	Rate	LiDAR channels	Duration (min/seq.)	Seq. $\geq$15 min	I/O seq.	Day / Night	Depth data	Env. info.
FusionPortableV2	Multiple	1024×768	20 Hz	128	1.3–11.6	0/27	6/27	24 / 3	Subset + tools	Sequence
Oxford Spires	Human-carried	1440×1080	5–20 Hz	64	3.5–21.3	3/24	11/24	24 / 0	Subset + tools	Sequence
MS-Depth (val.+test)	Handheld	2048×1536	10 Hz	128	15.8–22.6	13/13	13/13	6 / 7	Input + reference	Temporal

ment label between adjacent segments. The directed transition counts are 18 indoor-to-outdoor, 21 outdoor-to-indoor, 7 indoor-to-underground, 4 underground-to-indoor, 2 outdoor-to-underground, and 5 underground-to-outdoor. Each recording returns to its starting point after traversing multiple environment types.

Implications for online adaptation. MS-Depth enables evaluation of sparse-depth-guided adaptation along continuous recordings with repeated changes between indoor, outdoor, and underground environments. Maintaining the adaptation state across these transitions makes it possible to examine whether updates remain useful after a scene change, whether errors introduced by earlier updates persist, and how predictions recover when an environment type reappears. The segment annotations support evaluation before and after transitions and upon returning to an earlier environment type.

B.4 Comparison with SLAM and 3D Reconstruction Datasets

In this subsection, we compare MS-Depth with FusionPortableV2 (Wei et al., 2025) and Oxford Spires (Tao et al., 2026), which also provide calibrated RGB–LiDAR recordings with indoor–outdoor transitions, to clarify their differences in supporting long-horizon online adaptation for depth completion.

FusionPortableV2 (FPv2) targets SLAM generalization across platforms, and Oxford Spires benchmarks localization, 3D reconstruction, and novel-view synthesis. Both provide calibrated RGB–LiDAR recordings across varied environments, while MS-Depth is designed to evaluate long-horizon online adaptation guided by sparse depth across environmental changes. Table 16 summarizes their acquisition settings and adaptation-relevant characteristics. We report sequences lasting at least 15 minutes to match the final elapsed-time interval in our long-horizon adaptation analysis. Oxford recording durations were computed from front-camera image timestamps.

The FPv2 paper identifies short recording durations as a limitation, with all recordings lasting at most 11.6 minutes and the six handheld recordings spanning 1.9–6.3 minutes. Oxford Spires includes longer raw recordings, but only 3 of 24 last at least 15 minutes, limiting the number of sequences available for long-horizon evaluation. To support long-horizon evaluation across multiple sequences, we constructed the MS-Depth validation and test splits with 13 recordings lasting 15.8–22.6 minutes each, reserving one for validation and using the remaining 12 for test evaluation.

FPv2 and Oxford Spires contain indoor–outdoor transitions in a subset of their recordings. Because FPv2's official release lacks explicit I/O sequence labels, we inspected its public RGB previews and identified six I/O recordings. We included parking interiors and enclosed road tunnels but excluded open-sided canopies. All six were captured using legged robots, UGVs, or vehicles. For Oxford Spires, six recordings are officially tagged as both Outdoor and Indoor (proper), and eleven as both Outdoor and Indoor (limited). The latter tag indicates short indoor coverage or challenging lighting. Inspection of all 11 limited recordings identified 5 with indoor scenes, excluding open-to-sky spaces and very short tunnel-like passageways as ambiguous cases. Together with the 6 proper recordings, these yield 11/24 I/O recordings. Neither release provides explicit day/night labels. We therefore used recording timestamps and RGB inspection, identifying daytime and nighttime recordings in FPv2 and only daytime recordings in Oxford Spires. MS-Depth includes indoor–outdoor transitions in every handheld test recording and provides separate daytime and nighttime recordings of shared

Table 17: Public-benchmark evaluation subsets. Each sequence contributes one evaluation clip; DDAD clips contain all front-camera frames from their respective validation scenes.

Dataset	Sequences	Pairs per sequence	Total pairs
KITTI	8	110	880
Bonn	5	110	550
NYUv2 Raw	8	110	880
TUM RGB-D	8	110	880
DDAD	50	50 or 100	3,950
Total	79		7,140

locations. Beyond recording-level environment descriptions and tags, MS-Depth provides temporal labels for analyzing adaptation across transitions.

FPv2 and Oxford Spires, primarily developed for SLAM and 3D reconstruction, provide precomputed depth images for selected recordings; others require processing the released RGB–LiDAR data and calibration with the provided tools. In contrast, MS-Depth is captured with a 128-channel LiDAR and configured for depth completion: only eight selected channels are provided as sparse input, and depth at unobserved pixels must be predicted. For every recording, we provide calibrated RGB–depth pairs with precomputed input and reference depth, alongside the raw sensor data.

C Evaluation Subset Construction for Public Benchmarks

We construct temporally continuous RGB–depth evaluation subsets from KITTI, Bonn, NYUv2 Raw, TUM RGB-D, and DDAD. For KITTI, Bonn, NYUv2 Raw, and TUM RGB-D, we use one 110-pair clip from each selected sequence. This choice standardizes the clip length across these four datasets under a fixed evaluation budget. The dataset-specific sequence-selection and clip-extraction rules are described below. Clip extraction uses data availability, temporal continuity, and RGB–depth association criteria, without using model predictions or evaluation errors. For DDAD, we use every front-camera frame from all 50 scenes in the official validation split. The resulting public-benchmark evaluation comprises 79 sequences and 7,140 RGB–depth pairs (Table 17). All methods are evaluated on the same selected pairs, and each pair contributes to the accuracy evaluation exactly once.

Frame identification and interval boundaries. All start and end frames listed below are included in the evaluation. For Bonn, NYUv2 Raw, and TUM RGB-D, we identify evaluated pairs by their depth timestamps. The timestamp bounds specify the first and last selected depth frames; the exact RGB–depth associations are recorded in frame-level manifests. KITTI retains its original frame IDs, while DDAD retains the original timestamp tokens together with the within-scene sample order.

KITTI. We examine all 13 drives in the official KITTI depth-completion validation split and retain all eight drives that provide at least 110 consecutive frame IDs with both RGB images and corresponding annotated depth maps (Table 18). The remaining five drives do not contain a sufficiently long consecutive RGB–depth interval. For each eligible drive, we identify the longest consecutive run of frame IDs shared by the RGB images and annotated depth maps. If multiple runs have the same maximum length, we select the earliest one. We then use the first 110 frames of the selected run. For all eight drives, the resulting interval is `0000000005`–`0000000114`, inclusive.

Bonn. We adopt the five-sequence Bonn evaluation list used by Video Depth Anything (VDA) and DepthCrafter, as listed in Table 19. Within each sequence, we select the earliest 110-frame depth interval in which the time between successive depth frames does not exceed 0.05 s and every depth frame has an RGB match satisfying the association threshold. Each depth frame is paired with the temporally nearest RGB frame, with an absolute RGB–depth timestamp difference of at most 0.025 s.

Table 18: KITTI evaluation sequences and inclusive frame-ID intervals.

Sequence ID	Start frame	End frame	Pairs
2011_09_26_drive_0005_sync	0000000005	0000000114	110
2011_09_26_drive_0013_sync	0000000005	0000000114	110
2011_09_26_drive_0023_sync	0000000005	0000000114	110
2011_09_26_drive_0036_sync	0000000005	0000000114	110
2011_09_26_drive_0095_sync	0000000005	0000000114	110
2011_09_29_drive_0026_sync	0000000005	0000000114	110
2011_09_30_drive_0016_sync	0000000005	0000000114	110
2011_10_03_drive_0047_sync	0000000005	0000000114	110

Table 19: Bonn evaluation sequences and inclusive depth-timestamp intervals.

Sequence ID	Start depth timestamp	End depth timestamp	Pairs
rgbd_bonn_synchronous	1548340301.91798	1548340305.55515	110
rgbd_bonn_person_tracking	1548265882.51176	1548265886.14241	110
rgbd_bonn_crowd2	1548339894.59618	1548339898.23325	110
rgbd_bonn_crowd3	1548339959.43347	1548339963.07062	110
rgbd_bonn_balloon2	1548266533.17712	1548266536.81403	110

NYUv2 Raw. We use the eight Raw video sequences specified by the official VDA evaluation helper, as listed in Table 20. Within each sequence, we select the earliest 110-frame depth interval in which successive depth timestamps differ by at most 0.05 s and every depth frame has an RGB match within 0.025 s. Each depth frame is paired with its temporally nearest RGB frame. This evaluation uses the specified Raw video clips and is distinct from evaluation on the 654-image NYUv2 test split.

TUM RGB-D. We use a custom subset of eight sequences, listed in Table 21, fixed before evaluating methods on TUM RGB-D. The subset includes sequences from Freiburg 1, 2, and 3. RGB and depth frames are associated using the official TUM greedy one-to-one timestamp-matching procedure with a maximum absolute timestamp difference of 0.02 s. After association, we select the earliest interval of 110 matched pairs for which successive timestamps differ by at most 0.075 s in both the matched RGB stream and the matched depth stream.

DDAD. We use all 50 scenes in the official validation split, 000150–000199, and evaluate all frames from the front camera, CAMERA_01, in their original sample order. The 21 scenes listed in the first row of Table 22 contain 50 frames each, with zero-based sample indices 0–49. The remaining 29 scenes contain 100 frames each, with sample indices 0–99. This gives $21 \times 50 + 29 \times 100 = 3{,}950$ evaluated RGB–depth pairs.

D Analysis of Cached Online Decoder Adaptation

We analyze how representation caching and overlapping observation windows support efficient online decoder adaptation. Building on causal KV caching and decoder-only test-time adaptation, we formalize two properties of their use in StreamTTO: the consistency of cached representations during adaptation, and the change in the backpropagated gradient when the observation window advances. The first property concerns the cost of repeated optimization; the second provides a conditional explanation for reusing optimization progress.

Setup. Let ϕ denote the frozen feature-extractor parameters and θ the adapted decoder parameters. We assume deterministic feature extraction, causal temporal operations, and fixed positional assignments. The attention mask and KV retention policy for a frame depend only on information available when that frame arrives. The decoder processes each cached frame independently, with

Table 20: NYUv2 Raw evaluation sequences and inclusive depth-timestamp intervals.

Sequence ID	Start depth timestamp	End depth timestamp	Pairs
basement_0001a	1316653609.773176	1316653613.405163	110
bookstore_0001a	1295525659.331413	1295525662.963355	110
cafe_0001a	1295527093.873179	1295527097.504396	110
classroom_0001a	1315404991.101447	1315404994.732725	110
kitchen_0003	1294887072.518566	1294887076.168351	110
office_0004	1294851491.914890	1294851495.546016	110
playroom_0002	1315169148.541866	1315169152.154410	110
study_0002	1315161478.295818	1315161481.920261	110

Table 21: TUM RGB-D evaluation sequences and inclusive depth-timestamp intervals.

Sequence ID	Start depth timestamp	End depth timestamp	Pairs
rgbd_dataset_freiburg1_desk	1305031453.374112	1305031457.342954	110
rgbd_dataset_freiburg1_desk2	1305031526.688356	1305031530.656445	110
rgbd_dataset_freiburg1_room	1305031910.771502	1305031914.407339	110
rgbd_dataset_freiburg2_desk	1311868164.373557	1311868168.115940	110
rgbd_dataset_freiburg2_desk_with_person	1311870427.207687	1311870431.247317	110
rgbd_dataset_freiburg2_large_with_loop	1311875572.417883	1311875576.219311	110
rgbd_dataset_freiburg3_long_office_household	1341847980.723020	1341847984.542793	110
rgbd_dataset_freiburg3_walking_xyz	1341846313.592088	1341846317.442587	110

parameters shared across frames and no batch-dependent coupling. All comparisons below use the same input history, initial cache state, and causal computation policy.

Proposition 1 (Consistency of cached representations). Let $Z_j^{[t]}$ denote the representation at frame j in a reference execution on the prefix $I_{1:t}$ under the preceding assumptions. For every $t \geq j$, and for any subsequent decoder parameter updates,

$$Z_j^{[t]} = Z_j^{[j]} \equiv Z_j, \qquad \frac{\partial Z_j}{\partial \theta} = 0. \tag{5}$$

Consequently, evaluating the decoder and its sparse-depth loss using cached Z_j gives the same result as recomputing that representation under the same reference execution.

Proof. At each causal attention layer, the representation of frame j depends only on states available at or before j. Appending frames with indices greater than j therefore does not change its attention inputs or mask. Induction over the layers establishes the first equality. Since the extractor is frozen and has no dependence on decoder parameters, decoder updates leave Z_j unchanged, establishing the second equality. Substituting the same Z_j into the decoder and loss gives identical mathematical computations at any fixed θ. □

This result justifies reusing temporally contextualized features during decoder adaptation without repeated feature extraction. The reference execution uses the same causal KV policy as streaming inference; it is not a fresh execution on an isolated sliding clip with its earlier state discarded. Updates to parameters that affect cached representations generally require refreshing those representations to match a replay under the updated parameters. Decoder adaptation preserves cache consistency structurally, although other updates that leave all cached quantities unchanged can also do so.

Window gradients under the implemented loss. For a retained pair (Z_j, S_j), let $g_j(\theta)$ denote the gradient backpropagated to the decoder from the frame-wise sparse-depth loss, before global norm clipping and the AdamW update. As in the main method, scale and shift are recomputed at each loss evaluation and held constant during backpropagation. Thus, g_j follows the implemented stop-gradient convention; it need not equal the total derivative through the alignment solver. Decoder predictions, alignment coefficients, and gradients are reevaluated at the current θ; only the features and observations are reused.

Table 22: Complete DDAD validation-scene coverage for CAMERA_01. Sample-index intervals are zero-based and inclusive.

Frames per scene	Sample indices	Scene IDs
50	0–49	000151, 000152, 000154, 000157, 000159, 000162, 000166, 000172, 000174, 000177, 000182, 000183, 000191, 000192, 000193, 000194, 000195, 000196, 000197, 000198, 000199
100	0–99	000150, 000153, 000155, 000156, 000158, 000160, 000161, 000163, 000164, 000165, 000167, 000168, 000169, 000170, 000171, 000173, 000175, 000176, 000178, 000179, 000180, 000181, 000184, 000185, 000186, 000187, 000188, 000189, 000190

Once the window is full, let its size be $W \geq 2$ and assign weight $\alpha \geq 1$ to the current frame and weight one to each past frame. Writing $c = W - 1 + \alpha$, the normalized window gradient is

$$G_t(\theta) = \frac{1}{c}\left[\alpha g_t(\theta) + \sum_{j=t-W+1}^{t-1} g_j(\theta)\right]. \tag{6}$$

This expression uses the same frame-wise loss normalization as the main method and therefore remains valid when observation counts differ between frames.

Proposition 2 (Change across overlapping windows). For two consecutive full windows, $t \geq W + 1$, evaluated at the same decoder parameters θ,

$$G_t(\theta) - G_{t-1}(\theta) = \frac{\alpha g_t(\theta) - (\alpha - 1)g_{t-1}(\theta) - g_{t-W}(\theta)}{c}. \tag{7}$$

In particular, define

$$\delta_{a,b}(\theta) = \|g_a(\theta) - g_b(\theta)\|_2 . \tag{8}$$

Then

$$\|G_t(\theta) - G_{t-1}(\theta)\|_2 \leq \frac{\alpha\delta_{t,t-1}(\theta) + \delta_{t-1,t-W}(\theta)}{c}. \tag{9}$$

Proof. Proposition 1 makes each retained pair (Z_j, S_j) independent of the subsequent window arrival time. At a common θ, its backpropagated gradient is therefore the same in both windows. Subtracting their normalized sums cancels the terms with indices $t - W + 1, \ldots, t - 2$. The remaining terms correspond to the new frame, the previous current frame whose weight changes from α to one, and the removed frame. Rewriting their numerator as $\alpha(g_t - g_{t-1}) + (g_{t-1} - g_{t-W})$ and applying the triangle inequality proves the bound. □

Implication for warm-start adaptation. Let θ_{t-1} be the decoder state inherited from the previous frame, and set $\varepsilon_{t-1} = \|G_{t-1}(\theta_{t-1})\|_2$. Proposition 2 gives

$$\|G_t(\theta_{t-1})\|_2 \leq \varepsilon_{t-1} + \frac{\alpha\delta_{t,t-1}(\theta_{t-1}) + \delta_{t-1,t-W}(\theta_{t-1})}{c}. \tag{10}$$

If previous adaptation has reduced the preceding window gradient and the relevant frame gradients vary little, the inherited decoder also has a small backpropagated gradient on the new window. The default weights $(W, \alpha) = (3, 2)$ yield

$$G_t(\theta) - G_{t-1}(\theta) = \frac{2g_t(\theta) - g_{t-1}(\theta) - g_{t-3}(\theta)}{4}. \tag{11}$$

For $\alpha = 1$, the change instead reduces to $(g_t - g_{t-W})/W$, covering the equal-weight diagnostic experiments. The identity concerns gradients evaluated at a common parameter state; the gradients themselves are not cached across optimizer updates.

Table 23: Decoder-gradient diagnostics on MS-Depth VAL01 before current-frame adaptation. The ratio q_t compares warm-start and pretrained initializations, while d_t and s_t measure gradient difference and cosine similarity between consecutive windows evaluated at the same decoder parameters.

W	Mean q_t	Median q_t	Frames with $q_t < 1$	Mean d_t	Mean s_t
1	0.58	0.30	84%	13.9	0.55
3 (default)	0.55	0.29	86%	9.4	0.78
6	0.57	0.30	85%	6.1	0.86
12	1.05	0.46	70%	4.3	0.91

Empirical gradient diagnostics. We examine the gradient behavior underlying warm-start adaptation on the MS-Depth VAL01 validation sequence with window sizes $W \in \{1, 3, 6, 12\}$, keeping the remaining adaptation settings fixed. For each incoming frame, we evaluate gradients before its first adaptation update. Let θ_{t-1} denote the inherited decoder parameters and θ_{pre} the pretrained decoder parameters. Using the implemented sparse-depth loss, we measure

$$
\begin{aligned}
q_t &= \frac{\|G_t(\theta_{t-1})\|_2}{\|G_t(\theta_{\mathrm{pre}})\|_2 + \epsilon}, \\
d_t &= \|G_t(\theta_{t-1}) - G_{t-1}(\theta_{t-1})\|_2, \\
s_t &= \cos(G_t(\theta_{t-1}), G_{t-1}(\theta_{t-1}))\,,
\end{aligned}
$$

where $\epsilon > 0$ is a small constant for numerical stability. The ratio q_t compares the two initializations on the same current window using identical cached features, sparse observations, and loss weights. Both gradients used in d_t and s_t are reevaluated at the same inherited parameters θ_{t-1}. All gradients are measured before clipping. We aggregate statistics over the same post-bootstrap frame indices for which both consecutive windows are full for every tested W. The $W = 1$ setting provides a single-frame reference.

At the default window size $W = 3$, the mean and median gradient norm ratios are 0.55 and 0.29, respectively, and warm-start initialization yields a smaller gradient on 86% of frames. Across the tested window sizes, the mean gradient difference decreases from 13.9 to 4.3, while the mean cosine similarity increases from 0.55 to 0.91. Thus, larger windows exhibit smaller gradient changes and greater directional alignment between consecutive objectives in these runs.

The warm-start gradient norm ratio does not improve monotonically with window size. At $W = 12$, its median remains below one at 0.46 and 70% of frames have $q_t < 1$, but its mean increases to 1.05. This distinguishes consistency between consecutive window gradients from the reduction obtained relative to pretrained initialization. The result is consistent with the conditional analysis: a small change between windows alone does not ensure a small gradient on the new window, which also depends on the preceding window gradient at the inherited decoder state.

Computational implications. Let $C_{\mathcal{F}}$ be the cost of extracting the incoming frame's features with the fixed KV context, $C_{\mathrm{upd}}(W)$ the cost of one complete window update, including decoder forward and backward computation, alignment, clipping, and the optimizer step, and C_{out} the cost of producing the final current-frame output. After bootstrap, the model-computation cost is

$$
C_{\mathrm{frame}}(W, R) = C_{\mathcal{F}} + R\, C_{\mathrm{upd}}(W) + C_{\mathrm{out}}. \tag{12}
$$

Feature extraction is paid once per incoming frame and is not multiplied by the number of optimization passes. With fixed KV context, W, input resolution, and model size, both per-frame model computation and persistent state memory are bounded independently of the total stream length.

Interpretation and scope. The analysis separates exact representation reuse from the conditional reuse of optimization progress. Causality and freezing make cached features consistent, while window overlap permits cancellation of unchanged gradient terms. Temporal continuity alone does not guarantee that the gradient differences in equation 10 are small, particularly at environmental transitions. The bound concerns the backpropagated gradient and does not establish closeness to a minimizer, a bound on AdamW parameter displacement, or convergence in a prescribed number of

updates. Those claims would require additional assumptions and analysis of the optimizer dynamics.

The empirical ablations assess whether the proposed reuse is effective under the evaluated conditions. The initialization ablation shows that two warm-start passes achieve accuracy comparable to twenty reset passes. The compute-matched transition experiment further shows a 6.6% reduction in post-transition RMSE when recent observations are reused in a three-frame window. These results support the practical choice of a small update budget; the analysis provides its structural and conditional rationale without asserting that $R = 2$ is universally sufficient or optimal.

E Additional Discussion and Clarifications

Q1. How does MS-Depth capture variation in LiDAR observation density, and what are the limitations of its reference depth? MS-Depth includes depths up to approximately 30 m, with LiDAR observation density varying substantially across environments and capture conditions. In some frames, fewer than 50 LiDAR points are available. Including these frames enables the evaluation of online depth completion across conditions ranging from abundant observations to extremely sparse measurements. This variability reflects practical operating conditions in which uniform sensor coverage cannot be guaranteed as environments change, making the benchmark useful for evaluating adaptation to changes in observation density.

Reference depth is generated by projecting a single LiDAR scan into the central camera image, without accumulating multiple scans. Stereo correspondences are used to verify the consistency of projected LiDAR measurements, and depth values that pass the filter retain their original LiDAR measurements. Points are discarded if correspondences are unavailable, if the left–right consistency error exceeds one pixel, or if the difference between stereo and LiDAR depth exceeds 10% of the LiDAR depth.

This filtering may reduce coverage more strongly in dark or distant regions. Including diverse observation densities therefore does not eliminate selection bias introduced by the filter, and reported accuracy should be interpreted as performance at reference locations that pass the filter. All methods receive identical sparse inputs and are evaluated at the same locations, excluding the observation locations provided as input.

The eight LiDAR rings used for sparse input and the reference depth are obtained from the same scan. Excluding input locations from evaluation prevents the supplied measurements themselves from contributing directly to the scores, but does not eliminate shared sensor and calibration errors. Accordingly, MS-Depth evaluates how accurately methods complete depth at unobserved locations against a filtered LiDAR reference, subject to these measurement and coverage limitations.

Q2. What does StreamTTO contribute beyond existing test-time adaptation and causal caching? StreamTTO shows that efficient online depth completion depends on *where adaptation occurs*, not merely how many parameters are updated. By freezing the causal feature extractor, adapting the shared depth decoder on cached features, and retaining decoder and optimizer states across frames, StreamTTO reduces both per-step computation and the additional updates needed for each frame.

Tables 3(b)–3(c) illustrate this distinction: full-decoder adaptation updates 32.655 million parameters, approximately 83 times the 0.393 million updated by encoder-LoRA, yet requires 95.2% fewer FLOPs (7.028 T versus 145.768 T per frame) and achieves lower RMSE (0.8694 m versus 0.8884 m). Both variants follow the same online protocol, with separately selected learning rates. Encoder adaptation requires recomputing representations and propagating gradients through the encoder, whereas decoder adaptation preserves expensive features for reuse across optimization steps.

Prior methods adapt encoders through reconstruction objectives for video tasks (Wang et al., 2025b) or low-rank adapters for depth completion (Ke et al., 2026). StreamTTO instead examines how adaptation location enables computational reuse. It also differs from CoTTA (Wang et al., 2022), which uses teacher–student consistency on unlabeled inputs: StreamTTO optimizes against sensor-measured sparse depth and predicts after updating on current and recent observations. Causal attention and KV reuse follow streaming geometry models (Zhuo et al., 2026); StreamTTO combines

them with decoder-input feature caching, causal sliding-window optimization (CSWO), and persistent adaptation states.

Appendix D formalizes when cached representations remain consistent during decoder updates and how gradients change between overlapping observation windows. The latter provides a conditional rationale for warm-starting when both the previous window's gradient and the inter-window gradient change are small.

The ablations validate these complementary benefits. In Table 3(d), two warm-start updates achieve 0.8694 m RMSE, comparable to 0.8753 m with 20 updates after resetting every frame. Under matched decoder training FLOPs, a three-frame window processed twice reduces post-transition RMSE by 6.6% compared with six current-frame updates (1.1340 m versus 1.2146 m; Table 4). Tables 10–11 further isolate adaptation strategies with the same causal backbone and include CSWO-only comparisons for VGGT and MoGe-2. Together, these analyses establish a design principle that jointly considers adaptation location, representation reuse, and accumulated adaptation state.