High Frame Rate, Dense Depth Stream by a Low Frame Rate, Sparse Active Sensor

Andrea Conti [1], Matteo Poggi [1], Valerio Cambareri [2], and Stefano Mattoccia [1]

[1]University of Bologna, Italy
[2]Sony Depthsensing Solutions, Brussels, Belgium
https://andreaconti.github.io/projects/depth_on_demand

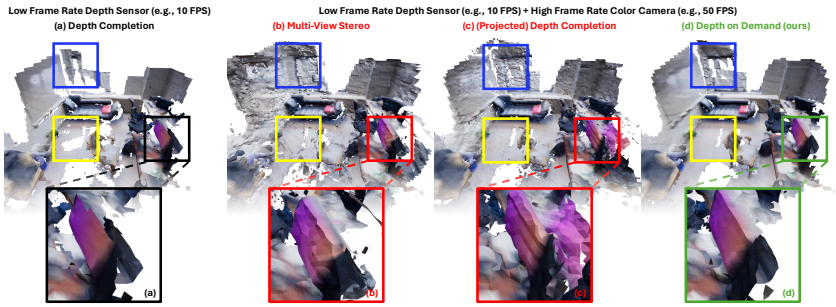


Fig. 1: 3D reconstruction with a low frame rate, sparse depth sensor. Running depth completion (a) on low FPS, sparse depth maps generate holes in the final reconstruction. Adding a higher FPS color camera allows for obtaining depth from Multi-View Stereo (b) or projecting depth to nearby color views and running completion (c), with unsatisfactory results. Our framework (d) performs *temporal* completion using two views and one sparse depth frame, yielding denser and more accurate meshes.

Abstract. High frame rate and accurate depth estimation plays an important role in several tasks crucial to robotics and automotive perception. To date, this can be achieved through ToF and LiDAR devices for indoor and outdoor applications, respectively. However, their applicability is limited by low frame rate, energy consumption, and spatial sparsity. Depth on Demand (DoD) allows for accurate temporal and spatial depth densification achieved by exploiting a high frame rate RGB sensor coupled with a potentially lower frame rate and sparse active depth sensor. Our proposal jointly enables lower energy consumption and denser shape reconstruction, by significantly reducing the streaming requirements on the depth sensor thanks to its three core stages: i) multi-modal encoding, ii) iterative multi-modal integration, and iii) depth decoding. We present extended evidence assessing the effectiveness of DoD on indoor and outdoor video datasets, covering both environment scanning and automotive perception use cases.

1 Introduction

This abstract refers to an already published paper, "Depth on Demand: Streaming Dense Depth from a Low Frame Rate Active Sensor" by the same authors [7], to appear at ECCV 2024. The full paper is attached to this submission as supplementary material.

RGB-D camera systems have become prominent in fields such as robotics, automotive and augmented reality and are not available on handheld devices such as the Apple Ipad. Among these sensors, Time-of-Flight (ToF) cameras infer the distance by emitting modulated infrared light into the scene and measuring irs return time [1, 2, 20]. On the other hand, Light Detection And Ranging (LiDAR) sensors allow for long-range measurements up to hundreds of meters with or without sunlight at much higher energy consumption and footprint.

ToF sensors – used mainly in mobile – can achieve a high frame rate but imply high energy consumption and overheating. Usually, a drastic reduction of their frame rate is applied to fit consumption with the limited available battery. On the other hand, LiDAR sensors are bulky devices used in autonomous driving, characterized by a low frame rate due to moving mechanical components. Finally, both these technologies manifest spatial sparsity, generating predictions only for specific spatial locations.

This paper proposes Depth on Demand (DoD), a framework addressing the three major issues related to active depth sensors in streaming dense depth maps – *i.e.* spatial sparsity, energy consumption, and limited frame rate. DoD allows coupling together an active depth sensor and an RGB camera to stream dense depth at the RGB camera frame rate, which may be much higher than the former. On the one hand, it allows the adaptation of active depth sensor energy consumption to the task specifications, thus meeting the energy constraints of the ToF use case. On the other, it unlocks frame rates higher than the maximum attainable by the active depth sensor itself, tackling effectively the LiDAR use case. Decoupling frame rates benefits also 3D scene reconstruction as it reduces energy consumption and allows for denser reconstructions. This is showcased in Figure 1: by performing depth completion only at a low frame rate (a) several holes appear in the mesh. Integrating information from a higher frame rate RGB camera (b-d) produces denser meshes. A simple solution to achieve the latter would be relying on Multi-View Stereo algorithms without using depth sensor data (b), or performing depth completion by projecting previous sparse depth points (c). Both these approaches introduce several artifacts as in the highlighted boxes. DoD produces denser and more accurate reconstructions (d).

2 Depth on Demand

Our approach consists of exploiting the higher framerate of an RGB camera to increase the temporal resolution of an active depth sensor. This is carried out by leveraging multi-view geometry on the RGB video stream and estimating

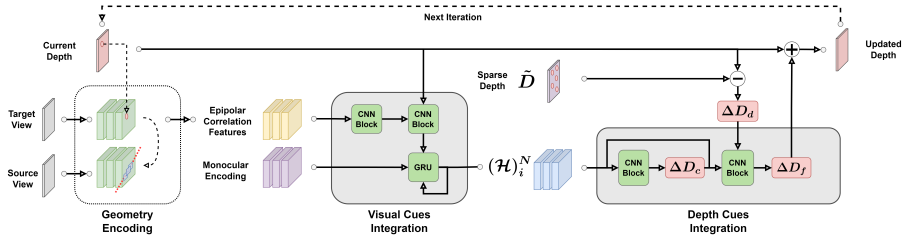


Fig. 2: Depth on Demand Framework Overview. We provide a high-level overview of DoD, level-wise architectural details are provided in the supplementary material. DoD embeds multi-view cues and monocular features in the Visual Cues Integration, then integrates sparse depth updates in the Depth Cues Integration. To properly exploit both these information these stages are applied iteratively in the form of depth updates.

depth for any RGB frame, both those for which measurements from the active sensor are available and those for which are not. To this aim, we make use of the minimum amount of information needed to exploit geometry – *i.e.* for each RGB view on which we seek to compute depth (the *target* view) we retain a previously collected RGB frame (*source* view) and sparse depth points (*source* depth). In this section we divide our framework into a set of three sequential steps and provide a brief description of each, please refer to the full paper [7] for further details.

Multi-Modal Encoding Our framework exploits information from different *modalities* to perceive 3D structures – *i.e.*, multi-view geometry, monocular cues and sparse depth. We separately compute domain-appropriate features and delegate integration to the next step. In figure 2 we specify the encoding of each domain. A shared encoder [13] extracts features $\mathcal{F}^t$,$\mathcal{F}^s$ at $\frac{1}{8}$ spatial resolution from target and source views. Given the predicted depth at a specific coordinate of the target view D_{u_t,v_t}, camera intrinsic parameters K and relative pose P, per-point correlation cues $\mathcal{C}$ can be computed by finding matching coordinates (u_t, v_t) and (u_s, v_s) as $\mathcal{C} = \frac{1}{\sqrt{F}} \sum_{f=1}^{F} \mathcal{F}^t_{u_t v_t f} \mathcal{F}^s_{u_s v_s f}$. Moreover, we sample a set of correlation patches along the epipolar line defined by D_{u_t,v_t} and relative pose P to increase the distinctiveness of each sampling. Monocular data is important in all the areas not in the intersection between the target and source views. Purposely, we integrate a monocular encoder exploiting the first layers of a ResNet34 [13] to output multi-scale feature maps $\tilde{\mathcal{F}}^t_2$, $\tilde{\mathcal{F}}^t_4$ and $\tilde{\mathcal{F}}^t_8$ at respectively $\frac{1}{2}$, $\frac{1}{4}$, $\frac{1}{8}$ resolution out of the target view alone. Finally, we assume the availability of sparse depth data obtained from an active sensor captured at a previous time instant. We project such sparse depth points onto the target view using pose information, obtaining a coarse depth map $\tilde{D}$ that will be exploited for both initialization and iterative multi-modal fusion. Since projecting at a lower resolution may lead to inaccurate positioning when building the sparse depth map, we propagate sub-pixel projection coordinates too. Unlike the depth completion task, projected sparse depth is characterized by errors on moving ob-

jects and occlusion, making it more difficult to exploit as a source. We delegate their management to the integration phase, where the exploitation of multiple modalities ameliorates such issues.

Multi-Modal Integration We employ a fusion module to combine encoded cues into a target-aligned depth map, iteratively refined by a fixed number of steps N. Our integration module is depicted in Figure 2 and can be logically divided into two sequential components. The first stage extracts depth-related features by visual cues only. We embed monocular and multi-view cues in the hidden state $(\mathcal{H})_i^N$ of a Gated Recurrent Unit, where $(\cdot)_i^N$ indicates a sequence of tensors across a set of iterations from $i = 0$ to $i = N - 1$. We initialize $(\mathcal{H})_{i=0}^N$ with a deep convolutional module fed with monocular features, specified in the supplementary material. The second stage takes into account sparse depth data availability. First, a branch predicts a depth update ΔD_c from the embedded visual features $\mathcal{H}_i$. Then, a sparse depth update ΔD_d is computed pixel-wise versus the current prediction as

$$\Delta D_d = \begin{cases} \tilde{D}_{i,j} - D_{i,j} & \text{if } \tilde{D}_{i,j} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Finally, updates fusion is carried out by a further branch predicting ΔD_f, used to update the current depth prediction. This allows for filtering the sparse depth which is likely to contain outliers due to reprojection. It is worth noticing that missing values can be integrated as zero updates, effectively dealing with the varying sparsity problem often affecting depth completion methods [8]. The previously described multi-modal updating strategy is applied multiple times, generating at each iteration a refined depth map that is then used at the subsequent iteration to improve the multi-view correlation samples and the sparse depth update. Accordingly, an initial depth state is required: we choose to initialize the depth for the first iteration with the sparse depth data, filling the missing coordinates with the mean value of the valid ones. In case no projected sparse depth points are available in the target view we initialize with a reasonable depth value of 3 meters, if not otherwise specified.

Depth Decoding The multi-modal integration module outputs a sequence of incrementally refined depth maps $(D)_i^N$ at $\frac{1}{8}$ resolution. We exploit a learned procedure inspired by convex upsampling [23] to perform upsampling. Given the depth map at $\frac{1}{8}$ resolution, we employ a set of three modules $\theta_s(\cdot)$ $s \in \{2, 4, 8\}$ performing a $2\times$ resolution upsampling composed of two convolutional layers with the aim to embed fine-grain monocular contextual information and enforcing locally smooth and consistent depth propagation. Please refer to the full paper [7] for further details.

3 Experiments

We evaluate our framework on indoor video sequences, aerial scenes and automotive environments to evaluate its accuracy both in single domains, and generalization. To each target frame I_i we associate a buffer of previous N frames

Table 1: Results on ScanNetV2. On top, (a) 2D and (b) 3D performance by DoD and competing approaches. At the bottom, (c) 3D performance by DoD with low/high temporal resolution. The best, second-best and third-best are highlighted.

	Method	Views	2D Metrics					3D Metrics					
			MAE↓	RMSE↓	Abs Rel↓	Sq Rel↓	$\sigma < 1.05$↑	Comp↓	Acc↓	Chamfer↓	Prec↑	Recall↑	F-Score↑
MVS	SimpleRecon [21]	8	0.093	0.151	0.047	0.016	0.717	0.062	0.056	0.059	0.702	0.646	0.671
MVS	PatchMatch-Net [24]	8	0.184	0.270	0.102	0.048	0.437	0.106	0.086	0.096	0.511	0.433	0.467
MVS	CAS-MVSNet [12]	8	0.170	0.254	0.091	0.044	0.507	0.086	0.082	0.084	0.545	0.498	0.519
MVS	UCS-Net [3]	8	0.167	0.252	0.088	0.042	0.512	0.084	0.082	0.083	0.547	0.502	0.522
MVS + Depth	Guided PatchMatch-Net [24]+ [19]	8	0.183	0.267	0.102	0.048	0.437	0.106	0.085	0.095	0.512	0.432	0.467
MVS + Depth	Guided CAS-MVSNet [12]+ [19]	8	0.124	0.203	0.068	0.029	0.635	0.064	0.061	0.062	0.667	0.634	0.649
MVS + Depth	Guided UCS-Net [3]+ [19]	8	0.133	0.210	0.074	0.030	0.576	0.070	0.065	0.068	0.616	0.578	0.595
MVS + Depth	Guided PatchMatch-Net [24]+ [19]	2	0.291	0.384	0.160	0.096	0.284	0.135	0.125	0.130	0.406	0.315	0.353
MVS + Depth	Guided CAS-MVSNet [12]+ [19]	2	0.286	0.388	0.154	0.094	0.304	0.099	0.109	0.104	0.447	0.419	0.431
MVS + Depth	Guided UCS-Net [3]+ [19]	2	0.258	0.353	0.148	0.093	0.328	0.103	0.099	0.101	0.451	0.385	0.414
Depth	SpAgNet [8]	1	0.069	0.138	0.039	0.016	0.824	0.046	0.037	0.042	0.836	0.789	0.810
Depth	NLSPN [18]	1	0.067	0.137	0.037	0.017	0.847	0.046	0.035	0.041	0.851	0.799	0.822
Depth	CompletionFormer [26]	1	0.075	0.149	0.041	0.019	0.829	0.047	0.037	0.042	0.846	0.795	0.818
	DoD (ours)	2	0.041	0.103	0.022	0.008	0.899	0.039	0.025	0.032	0.904	0.845	0.871

(a) (b)

Method	3D Metrics					
	Comp↓	Acc↓	Chamfer↓	Prec↑	Recall↑	F-Score↑
DoD – Low Temporal Resolution	0.064	0.014	0.039	0.961	0.778	0.856
DoD – High Temporal Resolution	0.039	0.025	0.032	0.904	0.845	0.871

(c)

$\{(I_j,\ \tilde{D}_j)\ :\ j \in [i-N, i-1]\}$. Then – at training time – we randomly select a frame from such buffer as the source one. For each sample, we collect a sequence of progressively refined depth maps $(D)_i^N$ supervised using an exponentially decayed ℓ_1-loss, as described in Equation 2 with decaying factor $\nu = 0.8$.

$$L = \sum_{i=1}^{N} \nu^{N-i} ||(D)_i^N - D_{\text{gt}}||_1 \tag{2}$$

While testing, we suppose an RGB video stream with a higher frame rate than the active depth sensor. Thus, given a sequence of RGB frames $[I_0, \ldots, I_n]$ only a few of them will be associated with a depth frame $\{\tilde{D}_0, \ldots, \tilde{D}_m\}$. We link each RGB view I_i to the immediately preceding RGB image coupled with a depth frame $(I_j,\ \tilde{D}_j)$, $j \leq i$ and feed our and competing methods with such data. We select a wide range of competitor methods and evaluate each one in our setting. Concerning depth completion, we compare with state-of-the-art frameworks [6, 18, 26] projecting sparse depth points from the source frame onto the target view I_i. Concerning Multi-View Stereo methods, we select [19] as a natural competitor since it enables standard MVS frameworks [3, 12, 24] to exploit both sparse depth and multi-view data natively. Since Multi-View Stereo methods usually exploit a large number of views, we train and evaluate with both 2 and 8 input views. To perform a fair comparison, we retrain each competitor following the authors' guidelines but applying the previously described training protocol as the original authors' pre-trained model performs worse in our setting.

3.1 Indoor Scenario

ToF sensors are mainly used for indoor applications, even though their low spatial resoluyion and short working range. In this setting, temporal resolution is limited to reduce power consumption and overheating. We train on ScanNetV2 [9] and test on ScanNetV2 and 7Scenes [11], randomly sampling 500

Table 2: Results on 7Scenes. 2D performance by DoD and competing approaches in generalization on 7Scenes. The best , second -best and third -best are highlighted.

	Method	Views	2D Metrics				
			MAE↓	RMSE↓	Abs Rel↓	Sq Rel↓	$\sigma < 1.05$↑
MVS	SimpleRecon [21]	8	0.121	0.169	0.068	0.021	0.536
	PatchMatch-Net [24]	8	0.193	0.268	0.112	0.048	0.390
	CAS-MVSNet [12]	8	0.177	0.251	0.101	0.041	0.421
	UCS-Net [3]	8	0.176	0.250	0.099	0.040	0.428
MVS + Depth	Guided PatchMatch-Net [24]+ [19]	8	0.191	0.264	0.112	0.047	0.391
	Guided CAS-MVSNet [12]+ [19]	8	0.120	0.192	0.071	0.024	0.587
	Guided UCS-Net [3]+ [19]	8	0.141	0.209	0.083	0.028	0.484
	Guided PatchMatch-Net [24]+ [19]	2	0.267	0.345	0.158	0.080	0.268
	Guided CAS-MVSNet [12]+ [19]	2	0.250	0.338	0.141	0.069	0.303
	Guided UCS-Net [3]+ [19]	2	0.228	0.306	0.139	0.063	0.303
Depth	SpAgNet [8]	1	0.068	0.139	0.040	0.014	0.806
	NLSPN [18]	1	0.061	0.134	0.037	0.014	0.842
	CompletionFormer [26]	1	0.067	0.144	0.039	0.015	0.827
	DoD (ours)	2	0.043	0.106	0.025	0.008	0.896

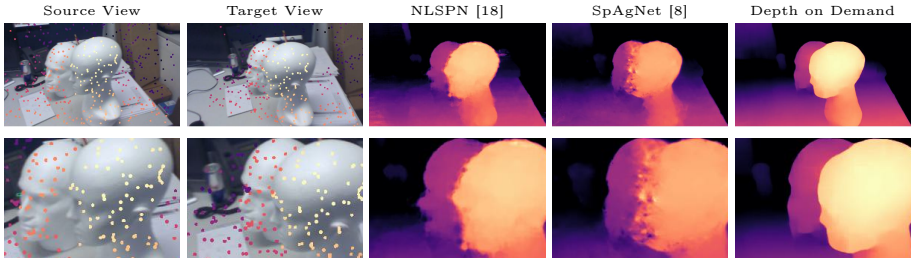


Fig. 3: Qualitative results on 7Scenes. From left to right: source view with sparse depth points, the target view with projected sparse depth points, and predictions by DoD and existing methods.

sparse depth points consistently with the depth completion literature [4, 15, 18]. For testing, we sparsify depth over time with 1 depth frame out of 5.

In Table 1 (a) we show the 2D performance on ScanNetV2 [9]. At the top, we report the performance of RGB-only methods [3, 12, 21, 24] with 8 views in input. Below, we show the performance of [19] with either 8 or 2 input views and projected sparse depth. In such methods, integrating sparse depth in our scenario provides a small improvement due to their specific design being poor at processing sequential frames. On the contrary, depth completion methods [8, 18, 26] relying only on the target view and projected sparse depth result in being the most competitive solution for temporal depth stream densification in the existing literature. Our framework indisputably outperforms completion models on any metric. This superior accuracy translates also into more accurate, dense 3D reconstructions shown in Table 1 (b). Table 1 (c), instead, highlights the effect of increasing the temporal resolution at which depth is estimated. We can notice how keeping a low temporal resolution – i.e., the same as the depth sensors – yields slightly accurate reconstructed meshes, while a higher temporal resolution trades accuracy to increase completeness. Nonetheless, maintaining a high temporal resolution yields better F-Scores overall.

We assess the generalization capabilities on 7Scenes [22] testing the models trained on ScanNetV2 [9]. Results are collected in Table 2, where our framework

Table 3: Results on TartanAir. 2D performance of our and competing approaches on TartanAir [25]. The best, second -best and third -best are highlighted.

	Method	Views	2D Metrics				
			MAE↓	RMSE↓	Abs Rel↓	Sq Rel↓	$\sigma < 1.05$↑
MVS + Depth	Guided PatchMatch-Net [24]+ [19]	8	2.353	5.259	0.234	2.285	0.470
	Guided CAS-MVSNet [12]+ [19]	8	1.296	3.753	0.126	1.270	0.647
	Guided UCS-Net [3]+ [19]	8	1.231	3.624	0.115	1.106	0.675
	Guided PatchMatch-Net [24]+ [19]	2	3.629	6.564	0.438	3.669	0.230
	Guided CAS-MVSNet [12]+ [19]	2	1.985	4.845	0.185	1.794	0.492
	Guided UCS-Net [3]+ [19]	2	1.804	4.513	0.177	1.526	0.486
Depth	SpAgNet [8]	1	0.841	2.273	0.090	0.561	0.718
	NLSPN [18]	1	0.941	2.327	0.113	0.623	0.613
	CompletionFormer [26]	1	0.961	2.411	0.106	0.608	0.625
	DoD (ours)	2	0.648	2.230	0.056	0.490	0.832

Fig. 4: KITTI Setup. On KITTI, we project the 360° LiDAR point cloud over the target point of view. If the camera is moving forward – as usually happens – the furthest scan lines are used only, leading to noisy and spaced depth values on the target view. However, the FoV of the target image is usually fully covered.

shows remarkable performance. In Figure 3 we provide a comparison of handling erroneous sparse depth points due to occlusion where DoD can disregard outliers by exploiting multi-view cues.

3.2 Outdoor Scenario

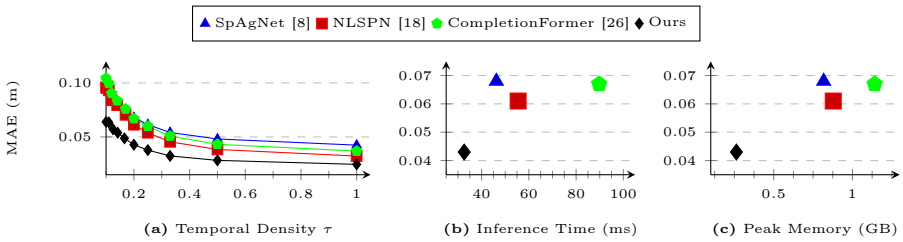
3D reconstruction in outdoor environments poses significantly challenges: larger depth ranges and depth measurements scattering. We exploit TartanAir [25] and KITTI [10] in this scenario. Table 3 shows our framework performance on TartanAir [25]. As for the indoor case, completion models largely outperform competitor networks, confirming their limitations in dealing with our setup. Again, our architecture shines in accuracy, achieving the lowest errors by a notable margin. Table 4 shows the results achieved by our framework on KITTI [10], by simulating different temporal sparsification levels – *i.e.* RGB camera at 10Hz and the LiDAR sensor at respectively 1Hz and 0.5Hz. We exploit an off-the-shelf keypoint matcher [16], perspective-n-points [14], and locally-optimized RANSAC [5] to estimate accurate pose, as already done in the depth completion literature [17]. Despite the more challenging setting, our framework still outperforms any existing alternative.

3.3 Temporal Sparsification Study

We study the sensitivity of our approach to different *temporal* densities, *i.e.*, frame rate imbalances between the RGB and depth sensor (that is, $\tau = f_{\mathrm{D}}/f_{\mathrm{RGB}}$). In Figure 5(a) we report the Mean Absolute Error (MAE) on the 7Scenes test

Table 4: Results on KITTI. 2D performance by DoD and competing approaches. The best, second-best and third-best are highlighted.

	Method	Views	2D Metrics – **LiDAR 1Hz**					2D Metrics – **LiDAR 0.5Hz**				
			MAE↓	RMSE↓	Abs Rel↓	Sq Rel↓	$\sigma < 1.05$↑	MAE↓	RMSE↓	Abs Rel↓	Sq Rel↓	$\sigma < 1.05$↑
MVS + Depth	Guided PatchMatch-Net [24]+ [19]	8	2.649	5.149	0.216	3.090	0.453	2.809	5.353	0.232	3.330	0.416
	Guided CAS-MVSNet [12]+ [19]	8	0.608	2.126	0.034	0.229	0.888	0.873	2.501	0.052	0.350	0.786
	Guided UCS-Net [3]+ [19]	8	0.575	1.930	0.034	0.229	0.881	0.828	2.321	0.050	0.303	0.789
	Guided PatchMatch-Net [24]+ [19]	2	1.898	4.165	0.117	0.863	0.496	2.282	4.564	0.145	1.154	0.404
	Guided CAS-MVSNet [12]+ [19]	2	0.676	2.203	0.035	0.225	0.872	0.916	2.562	0.052	0.328	0.773
	Guided UCS-Net [3]+ [19]	2	0.545	1.859	0.030	0.146	0.885	0.837	2.330	0.049	0.277	0.779
Depth	SpAgNet [8]	1	0.532	1.626	0.027	0.095	0.879	0.687	1.865	0.037	0.133	0.808
	NLSPN [18]	1	0.426	1.282	0.023	0.069	0.902	0.614	1.591	0.035	0.121	0.827
	CompletionFormer [26]	1	0.348	1.299	0.019	0.085	0.939	0.555	1.695	0.031	0.150	0.868
	DoD (ours)	2	0.347	1.288	0.017	0.061	0.944	0.492	1.544	0.025	0.094	0.890



(a) Temporal Density τ **(b)** Inference Time (ms) **(c)** Peak Memory (GB)

Fig. 5: Memory and Time Study. We analyze time and memory footprint in evaluation on a single RTX 3090 GPU of our and competing methods 7Scenes.

split with the testing protocol described in Section 3, while varying the temporal sparsification τ from 0.1 – *i.e.* one out of ten frames – to 1. Actually, when $\tau = 1$ the source view always matches with the target view, and sparse points are aligned with it. Thus, $\tau = 1$ is equivalent to the well-studied depth completion case. This may also occur in real use cases where the camera is static. We report a sensitivity study to different *spatial* densities in the supplementary material.

3.4 Memory and Time Analysis

Figures 5 (b) and 5 (c) report the memory footprint and execution time of the main methods involved in our experiments. We measure the peak memory and computation time the model requires for inference when processing 640×480 inputs. All measurements are based on a single RTX 3090 GPU and 32-bit floating-point precision. Our approach excels in terms of both.

4 Conclusion

In this paper, we faced the temporal sparsification of a video RGB-D stream when *temporally* reducing the number of depth frames used for accurate 3D reconstruction and we proposed an approach to integrate depth, monocular, and multi-view cues in an effective framework, as confirmed by the extensive validation over various datasets. We provide further details, experiments and ablation studies in the full paper [7].

References

1. Bamji, C., Godbaz, J., Oh, M., Mehta, S., Payne, A., Ortiz, S., Nagaraja, S., Perry, T., Thompson, B.: A review of indirect time-of-flight technologies. IEEE Transactions on Electron Devices **69**(6), 2779–2793 (2022). https://doi.org/10.1109/TED.2022.3145762
2. Bhandari, A., Raskar, R.: Signal processing for time-of-flight imaging sensors: An introduction to inverse problems in computational 3-d imaging. IEEE Signal Processing Magazine **33**(5), 45–58 (2016). https://doi.org/10.1109/MSP.2016.2582218
3. Cheng, S., Xu, Z., Zhu, S., Li, Z., Li, L.E., Ramamoorthi, R., Su, H.: Deep stereo using adaptive thin volume representation with uncertainty awareness. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2524–2534 (2020)
4. Cheng, X., Wang, P., Yang, R.: Depth estimation via affinity learned with convolutional spatial propagation network. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 103–119 (2018)
5. Chum, O., Matas, J., Kittler, J.: Locally optimized ransac. In: DAGM-Symposium (2003), https://api.semanticscholar.org/CorpusID:15181392
6. Conti, A., Poggi, M., Aleotti, F., Mattoccia, S.: Unsupervised confidence for lidar depth maps and applications. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (2022), iROS
7. Conti, A., Poggi, M., Cambareri, V., Mattoccia, S.: Depth on demand: Streaming dense depth from a low frame-rate active sensor. In: European Conference on Computer Vision (ECCV) (October 2024)
8. Conti, A., Poggi, M., Mattoccia, S.: Sparsity agnostic depth completion. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). pp. 5871–5880 (January 2023)
9. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: Proc. Computer Vision and Pattern Recognition (CVPR), IEEE (2017)
10. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2012)
11. Glocker, B., Izadi, S., Shotton, J., Criminisi, A.: Real-time rgb-d camera relocalization. In: International Symposium on Mixed and Augmented Reality (ISMAR). IEEE (October 2013)
12. Gu, X., Fan, Z., Zhu, S., Dai, Z., Tan, F., Tan, P.: Cascade cost volume for high-resolution multi-view stereo and stereo matching. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2495–2504 (2020)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 770–778 (2015), https://api.semanticscholar.org/CorpusID:206594692
14. Lepetit, V., Moreno-Noguer, F., Fua, P.: Epnp: An accurate o(n) solution to the pnp problem. International Journal of Computer Vision **81**, 155–166 (2009)
15. Lin, Y., Cheng, T., Zhong, Q., Zhou, W., Yang, H.: Dynamic spatial propagation network for depth completion. vol. 36 (2022). https://doi.org/10.1609/aaai.v36i2.20055

16. Lindenberger, P., Sarlin, P.E., Pollefeys, M.: LightGlue: Local Feature Matching at Light Speed. In: ICCV (2023)
17. Ma, F., Cavalheiro, G.V., Karaman, S.: Self-supervised sparse-to-dense: Self-supervised depth completion from lidar and monocular camera. 2019 International Conference on Robotics and Automation (ICRA) pp. 3288–3295 (2018)
18. Park, J., Joo, K., Hu, Z., Liu, C.K., Kweon, I.S.: Non-local spatial propagation network for depth completion. In: Proc. of European Conference on Computer Vision (ECCV) (2020)
19. Poggi, M., Conti, A., Mattoccia, S.: Multi-view guided multi-view stereo. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (2022), iROS
20. Qiao, X., Poggi, M., Deng, P., Wei, H., Ge, C., Mattoccia, S.: Rgb guided tof imaging system: A survey of deep learning-based methods. Int. J. Comput. Vis. (2024), https://link.springer.com/article/10.1007/s11263-024-02089-5
21. Sayed, M., Gibson, J., Watson, J., Prisacariu, V., Firman, M., Godard, C.: Simplerecon: 3d reconstruction without 3d convolutions. In: Proceedings of the European Conference on Computer Vision (ECCV) (2022)
22. Shotton, J., Glocker, B., Zach, C., Izadi, S., Criminisi, A., Fitzgibbon, A.W.: Scene coordinate regression forests for camera relocalization in rgb-d images. 2013 IEEE Conference on Computer Vision and Pattern Recognition pp. 2930–2937 (2013), https://api.semanticscholar.org/CorpusID:8632684
23. Teed, Z., Deng, J.: Raft: Recurrent all-pairs field transforms for optical flow. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M. (eds.) Computer Vision – ECCV 2020. pp. 402–419. Springer International Publishing, Cham (2020)
24. Wang, F., Galliani, S., Vogel, C., Speciale, P., Pollefeys, M.: Patchmatchnet: Learned multi-view patchmatch stereo. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14194–14203 (2021)
25. Wang, W., Zhu, D., Wang, X., Hu, Y., Qiu, Y., Wang, C., Hu, Y., Kapoor, A., Scherer, S.: Tartanair: A dataset to push the limits of visual slam (2020)
26. Zhang, Y., Guo, X., Poggi, M., Zhu, Z., Huang, G., Mattoccia, S.: Completionformer: Depth completion with convolutions and vision transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 18527–18536 (June 2023)