One Look is Enough: Seamless Patchwise Refinement for Zero-Shot Monocular Depth Estimation on High-Resolution Images

Supplementary Material

Base	Methods	Booster		ETH3D		Middle14			DIS	CE ↓
		AbsRel↓	δ_1 ↑	AbsRel↓	δ_1 ↑	AbsRel↓	δ_1 ↑	D^3R ↓	BR↑	
MiDaS v3.1[2]	MiDaS v3.1 [2]	0.0538	0.973	0.0655	0.965	0.0399	0.991	0.1914	0.028	-
	PatchFusion $_{P=16}$ [23]	0.0613	0.964	0.0826	0.952	0.0465	0.989	0.1435	**0.100**	0.456
	PatchRefiner $_{P=16}$ [24]	0.0520	**0.977**	0.0594	0.971	0.0377	**0.993**	0.1396	0.058	0.461
	PRO	**0.0513**	**0.977**	**0.0578**	**0.972**	**0.0370**	**0.993**	**0.1261**	0.062	**0.077**
DepthAnythingV1 [46]	DepthAnythingV1 [46]	**0.0482**	0.977	0.0489	0.981	0.0315	0.995	0.1546	0.023	-
	PatchFusion $_{P=16}$ [23]	0.0621	0.962	0.0672	0.970	0.0410	0.993	0.1246	**0.081**	0.393
	PatchRefiner $_{P=16}$ [24]	0.0507	0.977	0.0480	0.982	0.0301	**0.996**	0.1240	0.043	0.409
	PRO	0.0500	**0.978**	**0.0455**	**0.983**	**0.0294**	**0.996**	**0.1193**	0.052	**0.073**

Table 5. Generality across various base models. Quantitative comparison of depth estimation methods on Booster [29], ETH3D [35], Middlebury 2014 [34], and DIS-5K [28] datasets. **Bold** indicates the best performance in each metric.

Meth	BFM	Booster		ETH3D		Middle14			DIS
		AbsRel↓	δ_1 ↑	AbsRel↓	δ_1 ↑	AbsRel↓	δ_1 ↑	D^3R ↓	BR↑
PF [23]		0.0504	0.985	0.0735	0.956	0.0450	0.989	0.1124	0.206
	✓	0.0446	0.991	0.0761	0.956	0.0447	0.991	0.1149	0.198
PR [24]		0.0348	0.989	0.0435	0.985	0.0292	0.995	0.0830	0.151
	✓	0.0307	0.994	0.0432	0.985	0.0290	0.995	0.0791	0.151

Table 6. Effect of Bias Free Masking (BFM) on training of PatchFusion (PF) [23] and PatchRefiner (PR) [24].

A. Extra Experiments

Generality across Base Models. Replacing the base model (BM) requires retraining. To show the generality of our method, we provide additional comparisons using MiDaS v3.1 [2] and DepthAnythingV1 (DA1) [46] as base models. All evaluations are conducted in the same way in the Section 4.3. As shown in the Table 5, the MiDaS v3.1- and DA1-based PRO consistently outperform the previous SOTA models. However, all DA1-based methods show inferior AbsRel on the Booster compared to the DA1. This is because, as shown in the Fig. 6, DA1 yields poor predictions on transparent objects, thus leading the other methods to further degraded performance.

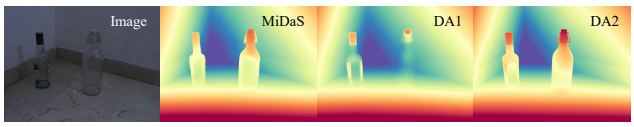

Figure 6. Depth estimation results from different base models (MiDaS v3.1 [2], DepthAnythingV1 [46], and DepthanythingV2 [47].

Effect of Bias Free Masking (BFM). To further demonstrate the effectiveness of BFM, we apply it to both PatchFusion [23] and PatchRefiner [24] during training. The consistent performance improvements observed across most metrics validate the generalizability of our approach, as summarized in Table 6.

P	Booster		ETH3D		Middle14			DIS
	AbsRel↓	δ_1 ↑	AbsRel↓	δ_1 ↑	AbsRel↓	δ_1 ↑	D^3R ↓	BR↑
3×3	0.0305	0.994	0.0423	0.985	0.0288	0.995	0.0805	0.125
4×4	0.0304	0.994	0.0422	0.985	0.0287	0.996	0.0803	0.156
5×5	0.0305	0.994	0.0423	0.985	0.0288	0.996	0.0797	0.162

Table 7. Effect of patch numbers on the performance.

Base	CE ↓	Booster		ETH3D		Middle14			DIS
		AbsRel↓	δ_1 ↑	AbsRel↓	δ_1 ↑	AbsRel↓	δ_1 ↑	D^3R ↓	BR↑
Diag	0.075	0.0302	0.994	0.0423	0.986	0.0288	0.996	0.0787	0.171
GPCT	0.049	0.0304	0.994	0.0422	0.985	0.0287	0.996	0.0803	0.156

Table 8. Quantitative comparisons between regularizing consistency diagonally (Diag) and GPCT.

Effect of Patch numbers on Performance. To investigate the effect of the number of patches during inference, we conduct experiments with the PRO model by partitioning the input images into 3×3, 4×4, and 5×5 patches, and compare their performance. As we can see in the Table 7, depth metrics remain stable across patch numbers, while edge metrics improve with more patches. This allows users to adjust patch counts at inference to balance speed and detail without retraining.

Ablation on GPCT vs Diagonal consistency. We additionally provide comparison between regularizing 2-diagonal patches and our GPCT in the Table 8. While the depth metrics show marginal differences, GPCT achieves significantly better consistency in CE metric qualitatively as shown in the Fig.7.

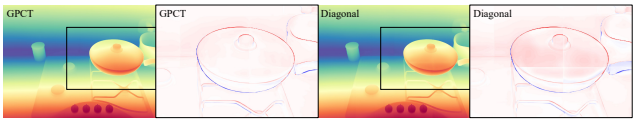

Figure 7. Qualitative comparisons between Diagonal approach and GPCT.

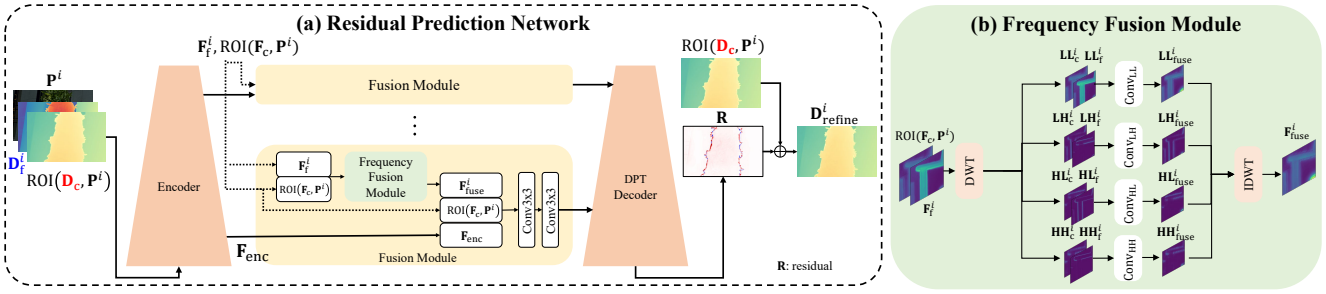

Figure 8. **Architecture of the Residual Prediction Network and Frequency Fusion Module (FFM). (a) Residual Prediction Network.** The Residual Prediction Network comprises an encoder, a decoder, and the Fusion Module. **(b) Frequency Fusion Module (FFM).** We utilize Discrete Wavelet Transform (DWT) to decompose the input features into four frequency components. Then, each frequency component is processed independently using convolutional layers.

Models	FLOPs (Mac)	Parameter	DIS	UHRSD	Middle14			ETH3D		Booster		NuScenes	
			BR↑	BR↑	AbsRel↓	δ_1 ↑	D^3R ↓	AbsRel↓	δ_1 ↑	AbsRel↓	δ_1 ↑	AbsRel↓	δ_1 ↑
Conv	31664G	63M	0.127	0.071	0.0290	0.995	0.0842	0.0428	0.984	0.0306	0.993	0.106	0.882
FFM (Ours)	20250G	51M	**0.156**	**0.083**	**0.0287**	**0.996**	**0.0803**	**0.0422**	**0.985**	**0.0304**	**0.994**	**0.104**	**0.883**

Table 9. Ablation study of the Frequency Fusion Module (FFM) on DIS-5K [28], UHRSD [45], Middlebury 2014 [34], ETH3D [35], Booster [29], and NuScenes [4]. **Bold** indicates the best performance in each metric.

B. Architecture of the Fusion Module

The encoder takes $\mathbf{P}^i$, $\mathbf{D}_\text{f}^i$, and $\text{ROI}(\mathbf{D}_\text{c}, \mathbf{P}^i)$ as inputs and produces a set of five intermediate features, denoted as $\mathbf{F}_\text{enc} = \{\mathbf{f}_{\text{enc},j}^i\}_{j=1}^5$ following [23]. Then, the fused feature map $\mathbf{F}_\text{fuse}^i$ is obtained through the frequency fusion module (FFM), defined as $\mathbf{F}_\text{fuse}^i = \text{FFM}(\text{concat}(\mathbf{F}_\text{f}^i, \text{ROI}(\mathbf{F}_\text{c}, \mathbf{P}^i)))$. Subsequently, $\text{concat}(\mathbf{F}_\text{fuse}^i, \text{ROI}(\mathbf{F}_\text{c}, \mathbf{P}^i), \mathbf{F}_\text{enc})$ is processed through two consecutive layers, each consisting of a 3×3 convolution, batch normalization, and ReLU activation. Finally, the resulting feature is fed into the DPT decoder [31] to obtain the residual map $\mathbf{R}$.

Architecture of the Frequency Fusion Module (FFM) To obtain accurate depth values from the coarse depth and preserve fine details from the fine depth, we design a Frequency Fusion Module (FFM) that effectively extracts and integrates edge information. We utilize Discrete Wavelet Transform (DWT) to decompose the input features into four frequency components: LL, LH, HL, and HH, which represent the low-frequency and high-frequency information. Each component is processed with its own dedicated convolution to capture scale-specific features. Finally, the components are recombined using the Inverse Discrete Wavelet Transform (IDWT), resulting in fused features that retain both global depth consistency and enhanced edge details. Overall process is described in Fig. 8-(b). To describe this process in more detail, we first decompose $\text{ROI}(\mathbf{F}_\text{c}, \mathbf{P}^i)$ and $\mathbf{F}_\text{f}^i$ into four frequency sub-bands $\mathbf{X}$ ($\forall \mathbf{X} \in \{\mathbf{LL}, \mathbf{LH}, \mathbf{HL}, \mathbf{HH}\}$) using DWT. Each sub-band is then fused using a corresponding convolution $\text{Conv}_\mathbf{X}$. Finally, the fused feature map $\mathbf{F}_\text{fuse}^i$ is obtained through IDWT.

$$\mathbf{X}_\text{c}^i, \mathbf{X}_\text{f}^i = \text{DWT}(\text{ROI}(\mathbf{F}_\text{c}, \mathbf{P}^i)), \text{DWT}(\mathbf{F}_\text{f}^i) \tag{8}$$

$$\mathbf{X}_\text{fuse}^i = \text{Conv}_\mathbf{X}(\text{concat}(\mathbf{X}_\text{c}^i, \mathbf{X}_\text{f}^i)) \tag{9}$$

$$\mathbf{F}_\text{fuse}^i = \text{IDWT}(\mathbf{LL}_\text{fuse}^i, \mathbf{LH}_\text{fuse}^i, \mathbf{HL}_\text{fuse}^i, \mathbf{HH}_\text{fuse}^i) \tag{10}$$

B.1. Ablation study of FFM

To validate the effectiveness of the Frequency Fusion Module (FFM), we conduct an ablation study by replacing the FFM with a simple convolutional block consisting of Conv-ReLU-Conv layers. To ensure that any performance gain is not simply due to an increase in the number of parameters or FLOPs, we design the simple convolutional block to have more parameters and FLOPs than the FFM. This allows us to attribute the performance improvement to the design of FFM itself, rather than computational complexity. As shown in the Table 9, our method not only achieves the best performance in standard depth metrics, but also yields significant improvements in edge accuracy. Specifically, it achieves a 22.5% improvement on the DIS-5K dataset and a 16.9% improvement on the UHRSD dataset in the Boundary Recall (BR) metric. In addition, we observe a 4.6% improvement in the edge quality metric (D^3R). It demonstrates that the proposed FFM effectively integrates edge information through the use of Discrete Wavelet Transform (DWT), which enables selective enhancement of high-frequency details without sacrificing global structure. This highlights the benefit of frequency-domain processing in depth refinement tasks.

C. Additional Implementation Details

Details in training. Before training, we first filter out unnecessary training samples based on the unreliable mask, $\mathbf{M}_{\text{unreliable}}$, described in Section 3.3. Specifically, we load each $\mathbf{M}_{\text{unreliable}}$ using NumPy and discard the corresponding training sample if the mean value of $\mathbf{M}_{\text{unreliable}}$ exceeds 0.5, indicating excessive unreliable regions. During training, we randomly crop a region (e.g., 846×1505) from a 2160×3840 input to generate 2×2 overlapping patches (540×960 each), which are resized to 518×518 for refinement. The overlap size (234×415) corresponds to $\frac{224}{518} \approx 43\%$ of the patch dimensions, where 224 is the empirically chosen overlap value from Section 4.4.

Details in inference. At inference, the input is divided into a 4×4 non-overlapping grid, and the refined 518×518 patches are reassembled into a 2072×2072 depth map. The final depth map is then upsampled to the original resolution via bilinear interpolation. This inference procedure is consistent with $\text{PF}_{\text{P=16}}$ and $\text{PR}_{\text{P=16}}$. As demonstrated in Table 7, the number of patches can be flexibly adjusted depending on the inference setting.

D. Qualitative Results

Ablation Study In the ablation study (Section 4.4), we analyze the effect of Grouped Patch Consistency Training (GPCT) and Bias Free Masking (BFM) quantitatively. In this section, we analyze the effect of GPCT and BFM with qualitative results. As shown in the Fig. 9, the model trained without GPCT shows remarkable depth discontinuity problem on the grids. On the other hand, our PRO model trained with GPCT alleviates the depth discontinuity problem. Likewise, as shown in the Fig. 10 , the model trained without BFM exhibits artifacts on transparent surfaces, such as glass windows, as well as reflective surfaces like TV screens. In contrast, our model trained with BFM effectively refines only the edge regions while preventing artifacts on transparent objects.

Additional Qualitative Results We provide additional qualitative comparisons of BoostingDepth [26], PatchFusion [23], PatchRefiner [24], and PRO (Ours) on the UHRSD [45] dataset and on internet images (e.g., from Unsplash[1] and Pexels[2]), as shown in Fig.11 and Fig.12.

[1] https://unsplash.com
[2] https://www.pexels.com

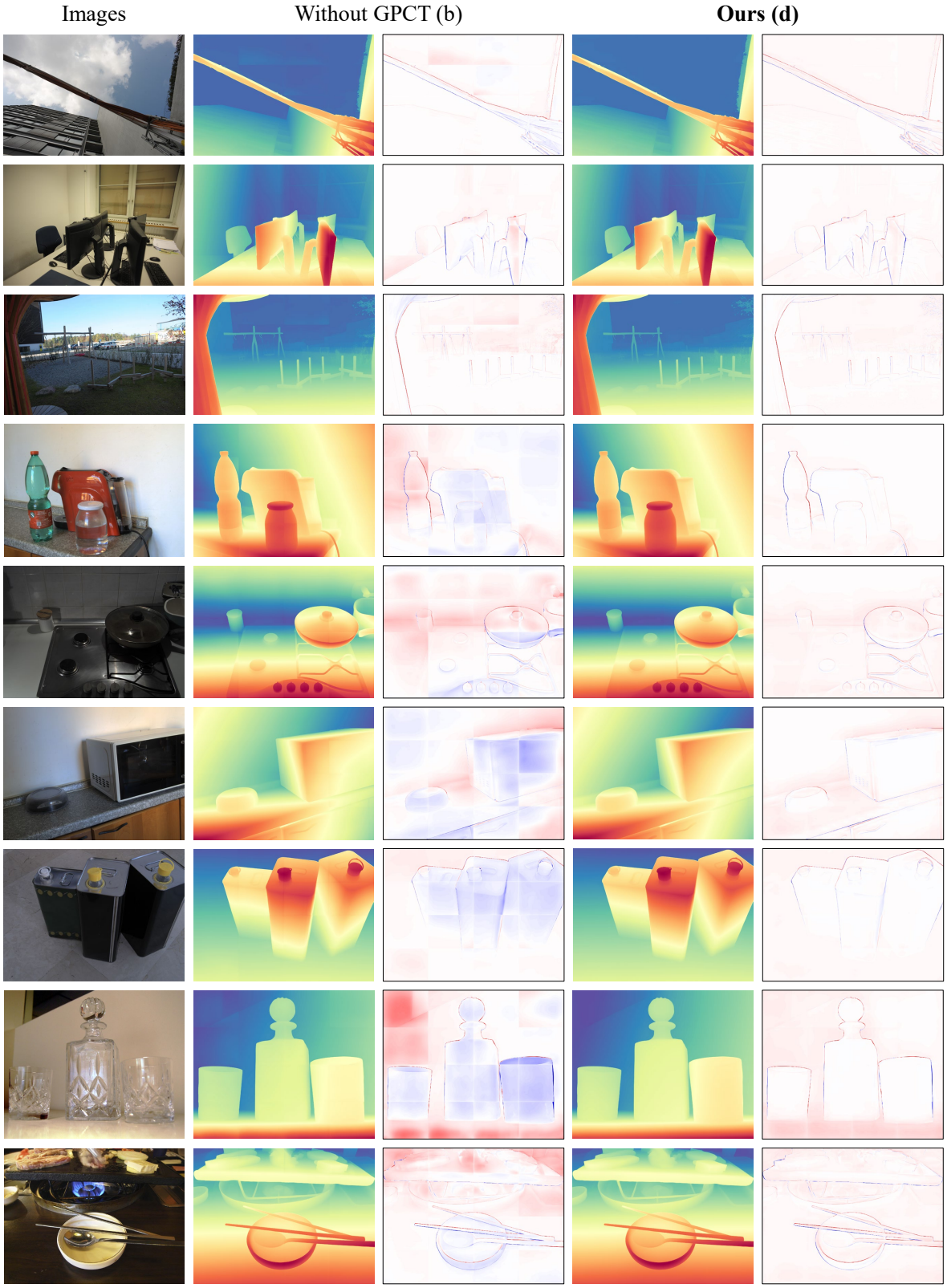

Figure 9. **Qualitative comparisons of GPCT's impact on ETH3D [35], Booster [29], and DIS-5k [28].** We compare PRO (Ours (d)) with the model trained without Grouped Patch Consistency Training (GPCT) (b). (b) and (d) represent the model index in Table 3. We also visualize the residuals to highlight the presence of artifacts more effectively. Zoom in for details.

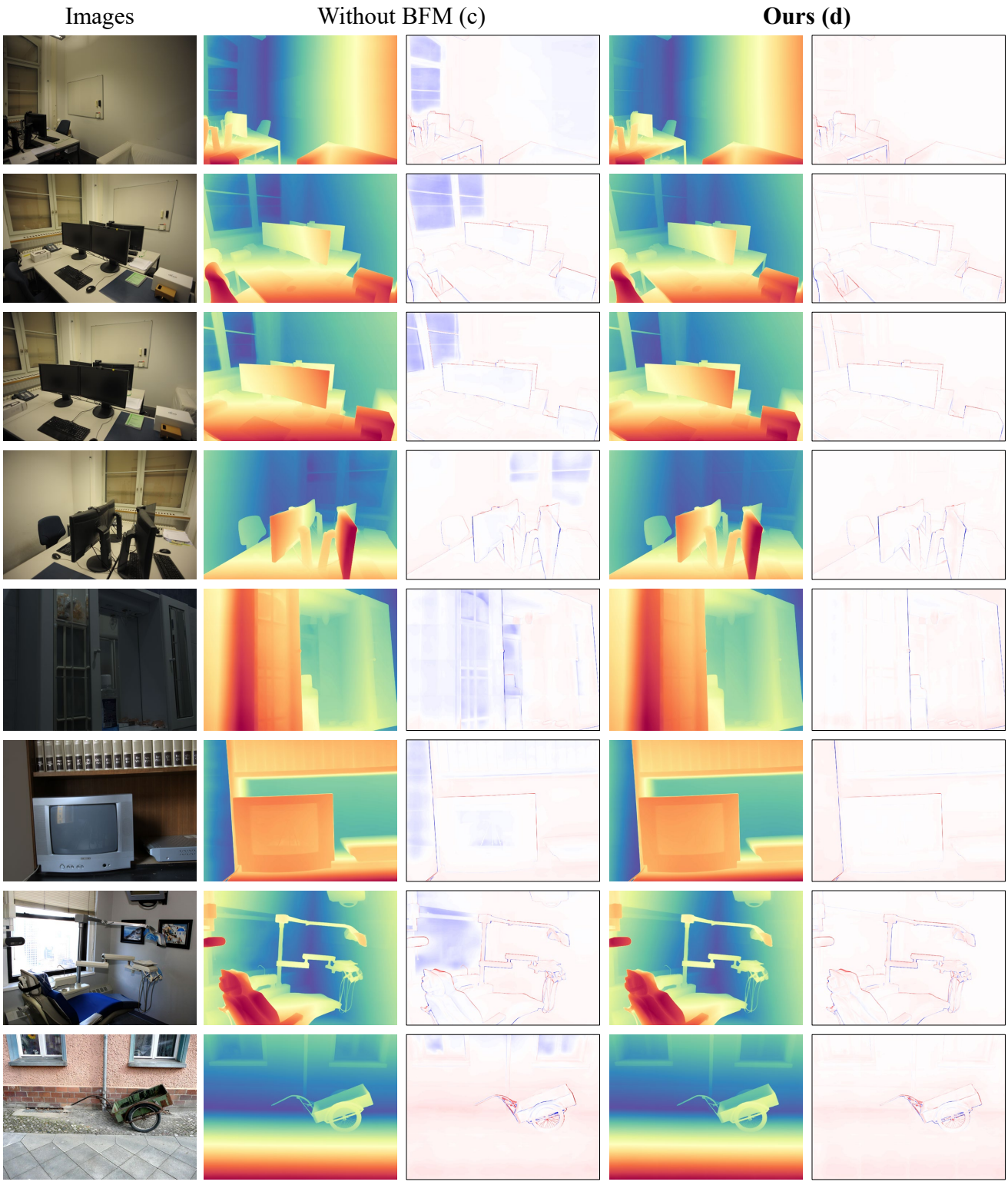

Figure 10. **Qualitative comparisons of BFM's impact on ETH3D [35], Booster [29], and DIS-5k [28].** We compare PRO (Ours (d)) with the model trained without Bias Free Masking (BFM) (c). (c) and (d) represent the model index in Table 3. We also visualize the residuals to highlight the presence of artifacts more effectively. Zoom in for details.

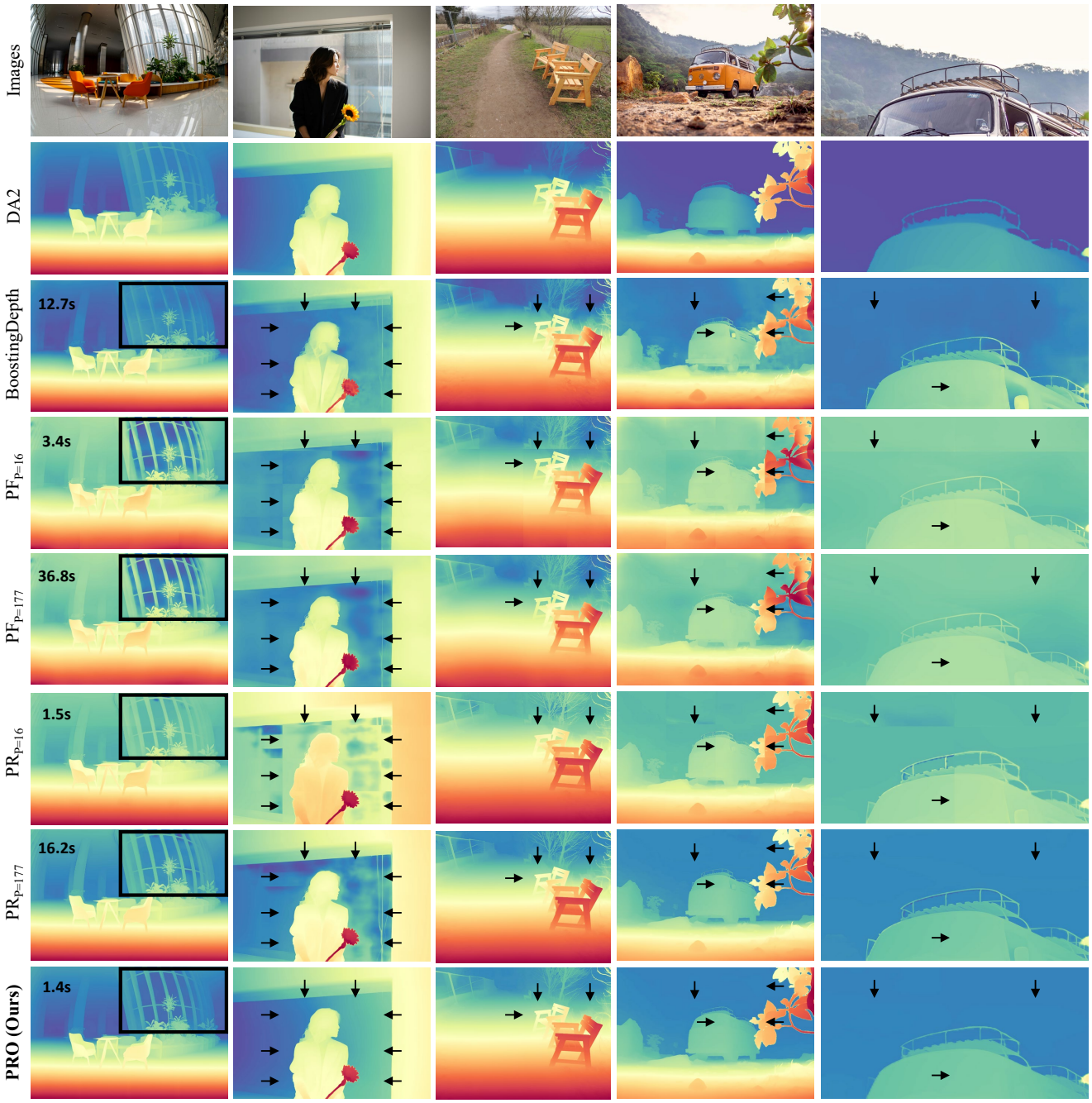

Figure 11. **Qualitative comparisons for patch-wise DE methods on UHRSD [45] and images from the internet.** We compare PRO (Ours) with BoostingDepth [26], PatchFusion (PF) [23], and PatchRefiner (PR) [24]. The time displayed in the leftmost depth column represents the inference time. Black rectangle area represents the transparent object region. Black arrows indicate patch boundaries. Zoom in for details.

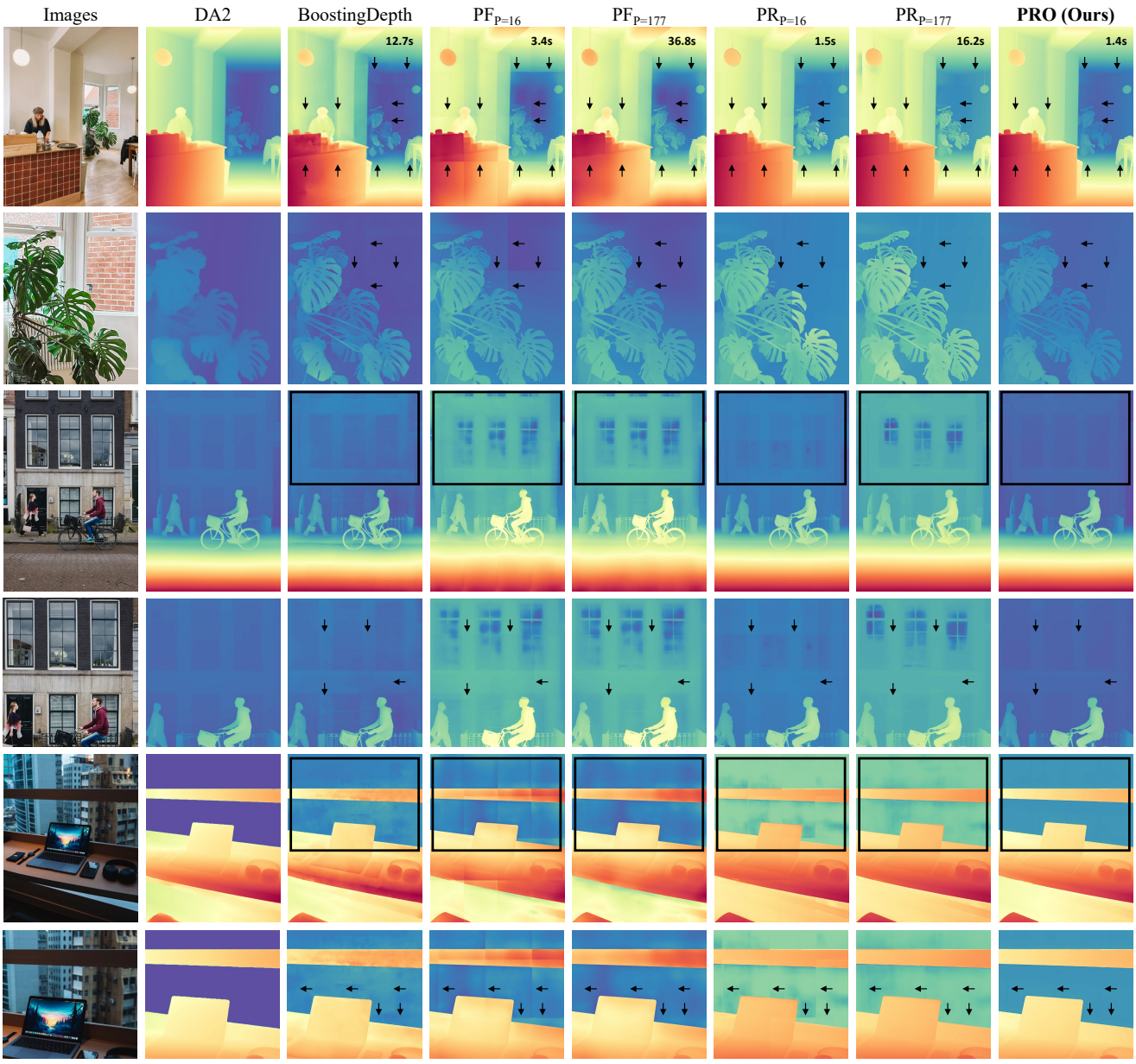

Figure 12. **Qualitative comparisons for patch-wise DE methods on images from the internet.** We compare PRO (Ours) with BoostingDepth [26], PatchFusion (PF) [23], and PatchRefiner (PR) [24]. The time displayed in the top depth row represents the inference time. Black rectangle area represents the transparent object region. Black arrows indicate patch boundaries. Zoom in for details.