Supplementary Materials for SeeGroup: Multi-Layer Depth Estimation of Transparent Surfaces via Self-Determined Grouping

Hongyu Wen Jia Deng
Department of Computer Science, Princeton University
{hongyu.wen,jiadeng}@princeton.edu

1. Additional Details

1.1. Model Design

In a forward pass, we first extract features $\mathbf{F}_0$ from the input RGB image $\mathcal{I} \in \mathbb{R}^{H\times W\times 3}$ using a DINOv2 [2] backbone, following the pipeline of Depth Anything V2 [5]. Specifically, $\mathbf{F}_0$ consists of a sequence of feature maps with shape (B, N_0, H_0, W_0), where B is the batch size, N_0 is the channel dimension, and H_0, W_0 denote the spatial resolution in terms of patch counts along height and width. These feature maps are extracted from intermediate layers of the backbone. As a result, each map corresponds to a different stage of the network and captures information at a distinct scale, ranging from coarse to fine. In our implementation, the sequence length is $l = 4$.

The architecture of our recurrent decomposition module follows the overall design of Depth Anything V2 [5]. For the decomposer D, we first apply a 1×1 convolution to each backbone feature map, obtaining tensors of shape $(B, N_{1,i}, H_0, W_0)$, where $N_{1,i}$ depends on the backbone stage i. These tensors are then passed through either a ConvTranspose layer or a Conv layer (depending on the desired resolution) to produce feature maps of shape $(B, N_{1,i}, H_i, W_i)$. Here (H_i, W_i) and $N_{1,i}$ vary with i, with coarser stages generally having greater spatial resolution and fewer channels. Each of these feature maps is then mapped by another convolutional layer to a shared channel dimension, resulting in tensors of shape (B, N_2, H_i, W_i). The resulting sequence of multi-scale feature maps at step k forms $\mathbf{C}_k$ in the recurrent decomposition.

In the predictor P, the sequence $\mathbf{C}_k$ is first passed through two convolutional layers that align the spatial resolutions and channel dimensions of all feature maps. We then fuse these maps in a top-down manner, from the finest to the coarsest level, by combining adjacent pairs. For each pair of adjacent levels, we first upsample the finer-level feature using a convolutional block to match the resolution of the coarser one, sum the two features, and pass the result through another convolutional block. After all levels are fused, we obtain a single aggregated feature map of shape $(B, N_2, 2H_1, 2W_1)$. This feature map is further processed by two convolutional layers to produce the parameters $d_i, b_i \in \mathbb{R}_{>0}^{H\times W}$, where d_i is the center (depth) and b_i is the scale of the Laplace-shaped contribution.

N_0	$\{H_i\}$	$\{W_i\}$	$\{N_{1,i}\}$	N_2
1024	$\{148, 74, 37, 19\}$	$\{148, 74, 37, 19\}$	$\{256, 512, 1024, 1024\}$	256

Table 1. The exact values of architectural hyperparameters in recurrent decomposition module.

The remapper R is essentially the inverse of the decomposer D. Starting from the aggregated feature map, it first uses a convolutional layer to reconstruct, for each scale i, a feature map of shape $(B, N_{1,i}, H_i, W_i)$. A second convolutional layer then maps these features to a common resolution $(B, N_{1,i}, H_0, W_0)$, followed by a final convolution that projects them back to the original backbone feature space (B, N_0, H_0, W_0). The resulting feature map is subtracted from the previous residual feature map $\mathbf{F}_{k-1}$ to obtain the updated feature map $\mathbf{F}_k$. The exact values of N_0, $N_{1,i}$, N_2, and other architectural hyperparameters are listed in Tab. 1.

1.2. Loss Function

In this section, we provide proof for the following lemma:

Lemma 1.1. *Given an intensity function $\mathbf{\Lambda}$ on $[0, D_{\max}]$ and a set of ground-truth depths $\{d_1, d_2, \ldots, d_m\}$. Assuming the depths are independent, the probability of observing these depths at pixel (x, y) is given by:*

$$P(d_1, d_2, \ldots, d_m) = \mathbf{\Lambda}(d_1, x, y)\mathbf{\Lambda}(d_2, x, y) \ldots \mathbf{\Lambda}(d_m, x, y)$$

Proof. Fix small, disjoint intervals $I_1, I_2, \ldots, I_m \subset [0, D_{\max}]$ such that each I_i is centered at d_i and has length $|I_i| = \epsilon$. Let $N(I_i)$ denote the number of observed depths in I_i, and let

$$A \coloneqq \{N(I_1) \geq 1, N(I_2) \geq 1, \ldots, N(I_m) \geq 1\}$$

be the event that we observe at least one depth in each in-

terval. We can write

$$\mathbb{P}(A) = \mathbb{P}\big(N(I_1) \geq 1, \ldots, N(I_m) \geq 1\big) = \sum_{n_1=1}^{\infty} \cdots \sum_{n_m=1}^{\infty} \mathbb{P}\big(N(I_1) = n_1, \ldots, N(I_m) = n_m\big).$$

Assume the depths form an inhomogeneous Poisson process on $[0, D_{\max}]$ with intensity function $\Lambda(\cdot)$ along the depth axis. For disjoint intervals, the counts $N(I_1), \ldots, N(I_m)$ are independent, and each $N(I_i)$ is Poisson distributed with mean

$$\mu_i \coloneqq \int_{I_i} \Lambda(u) \mathrm{d}u$$

Thus,

$$\mathbb{P}\big(N(I_i) = n\big) = e^{-\mu_i} \frac{\mu_i^n}{n!}, \qquad n = 0, 1, 2, \ldots$$

is the Poisson probability mass function for interval I_i.

By independence of the counts on disjoint intervals, we have

$$\begin{aligned}\mathbb{P}\big(N(I_1) = n_1, \ldots, N(I_m) = n_m\big) &= \prod_{i=1}^{m} \mathbb{P}\big(N(I_i) = n_i\big) \\ &= \prod_{i=1}^{m} e^{-\mu_i} \frac{\mu_i^{n_i}}{n_i!}.\end{aligned}$$

Plugging this into the expression for $\mathbb{P}(A)$ and rearranging sums and products, we obtain

$$\begin{aligned}\mathbb{P}(A) &= \sum_{n_1=1}^{\infty} \cdots \sum_{n_m=1}^{\infty} \prod_{i=1}^{m} e^{-\mu_i} \frac{\mu_i^{n_i}}{n_i!} \\ &= \prod_{i=1}^{m} \left(\sum_{n_i=1}^{\infty} e^{-\mu_i} \frac{\mu_i^{n_i}}{n_i!} \right) \\ &= \prod_{i=1}^{m} \left(1 - e^{-\mu_i}\right),\end{aligned}$$

Next, note that each interval I_i has length ϵ, so as $\epsilon \to 0$,

$$\mu_i = \int_{I_i} \Lambda(u)\, du \approx \Lambda(d_i)\, \epsilon \to 0.$$

Using the first-order Taylor expansion $1 - e^{-x} = x + O(x^2)$ as $x \to 0$, we get

$$1 - e^{-\mu_i} = \mu_i + O(\mu_i^2).$$

Hence,

$$\begin{aligned}\mathbb{P}(A) &= \prod_{i=1}^{m} \left(1 - e^{-\mu_i}\right) \\ &= \prod_{i=1}^{m} \left(\mu_i + O(\mu_i^2)\right) \\ &= \left(\prod_{i=1}^{m} \mu_i \right) \left(1 + o(1)\right) \\ &= \left(\prod_{i=1}^{m} \Lambda(d_i)\, \epsilon \right) \left(1 + o(1)\right) \quad \text{as } \epsilon \to 0.\end{aligned}$$

In particular, up to a multiplicative factor that does not depend on the locations $d_1, \ldots, d_m$, we have

$$\mathbb{P}(A) \propto \prod_{i=1}^{m} \Lambda(d_i).$$

□

1.3. Training

During training, we follow the scale-invariant normalization of [3]. Given a depth map d, we normalize it to $\tilde{d}$ as

$$\begin{aligned}t &= \mathrm{median}(d), \\ s &= \mathrm{mean}(|d - t|), \\ \tilde{d} &= \frac{d - t}{s}.\end{aligned}$$

The scale parameters b of the Laplace distributions are clipped to the range $[1, 10]$. We clip the gradient norm to 0.1 to stabilize training. We use the AdamW [1] optimizer with an initial learning rate of 1×10^{-5} for the main network and 1×10^{-6} for pretrained the DINOv2 backbone. The momentum parameters are set to $(0.9, 0.99)$, and the weight decay is set to 0.01. The learning rate multiplier at training step k is given by $\left(1 - \frac{k}{\text{total steps}}\right)^{0.9}$. All images are resized to 518×518 during training. For the gradient matching loss, all layers are weighted by $(1.2, 1.0, 1.0, 1.0)$. During inference, we first resize the image so that its shorter side is 518 pixels while preserving the aspect ratio, and then round the longer side to the nearest multiple of 14. The predicted depth maps are finally bilinearly upsampled back to the original image resolution.

2. Additional Results

To help readers better visually assess the effectiveness of our datasets, we provide additional qualitative results on real world benchmark LayeredDepth [4], as shown in Fig. 1. Our method produces noticeably sharper predictions with fewer artifacts.

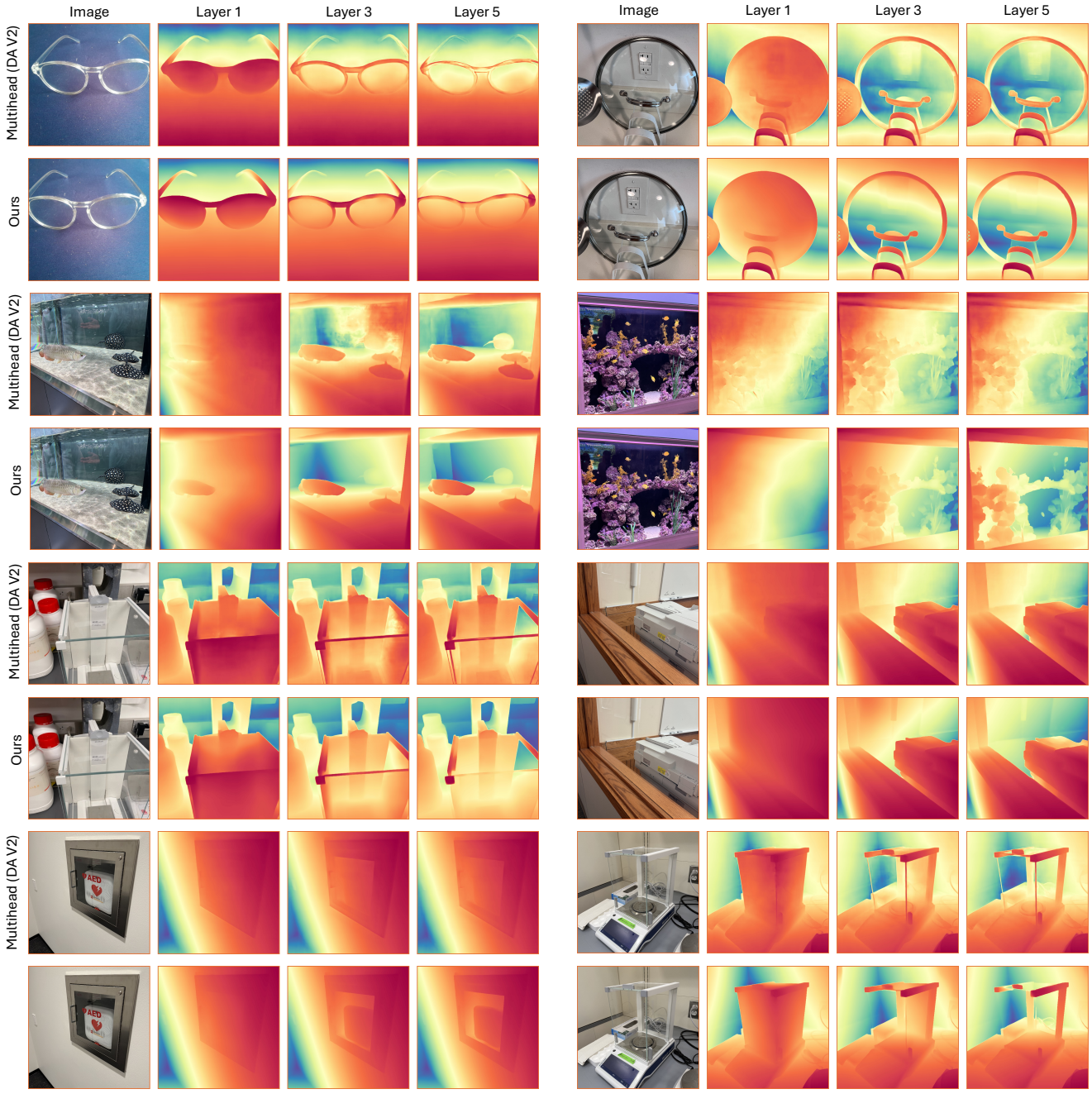

Figure 1. Additional qualitative results on LayeredDepth.

References

[1] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 2

[2] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 1

[3] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer, 2020. 2

[4] Hongyu Wen, Yiming Zuo, Venkat Subramanian, Patrick Chen, and Jia Deng. Seeing and seeing through the glass: Real and synthetic data for multi-layer depth estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6715–6725, 2025. 2

[5] 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, 2025. 1