Upsample Anything: A Simple and Hard to Beat Baseline for Feature Upsampling

Supplementary Material

7. Semantic Segmentation on Cityscapes

We evaluated feature upsampling and probability-map upsampling on Cityscapes using the official LoftUp segmentation codebase with a stronger training setup that includes 448×448 input resolution, 100 epochs, and a learning-rate scheduler. Under this configuration, all methods, including LoftUp and ours, produced almost the same mIoU as bilinear interpolation, which differs from the improvements reported in the LoftUp paper.

To ensure correctness, we carefully re-examined our implementation through an automated code audit with ChatGPT and a manual review by multiple authors. We found no inconsistencies or bugs.

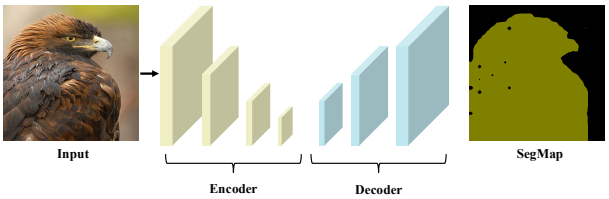
The quantitative results are summarized in Table 7. Across all methods, including feature-level and probability-level upsampling, the differences remain within a very narrow range. Cityscapes primarily contains large and regular structures, and its annotations are relatively coarse. With a sufficiently trained segmentation head, bilinear interpolation already performs near optimally, leaving little room for additional gains. In contrast, datasets such as COCO, PASCAL-VOC, and ADE20K include many small objects and complex boundaries, where upsampling delivers clear benefits.

Method	Cityscapes	
	mIoU (↑)	Acc. (↑)
Bilinear	57.90	90.59
FeatUp	57.92	90.61
LoftUp	57.89	90.60
JAFAR	57.91	90.58
AnyUp	57.93	90.62
Upsample Anything	57.92	90.63
Upsample Anything (prob.)	56.36	90.05

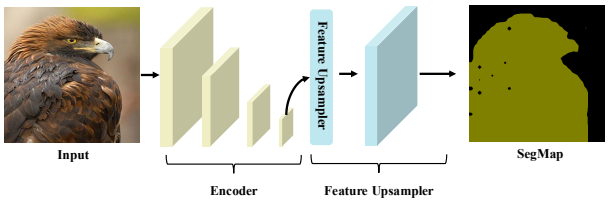
Table 7. Segmentation performance on the Cityscapes dataset using the official LoftUp evaluation pipeline.

8. Details of Probabilistic Map Upsampling in Table 1.

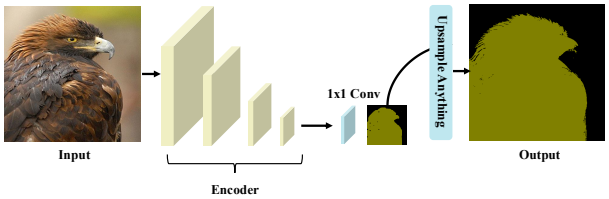
Figure 8-(c) corresponds to the *Upsample Anything (prob.)* configuration reported in Table 1. In this setting, the segmentation map is predicted from downsampled features using a lightweight 1×1 convolution, followed by our probabilistic upsampling to reconstruct high-resolution outputs. This simple setup already shows strong performance. Because the computation is performed on a small feature map, heavier or more complex decoders are expected to be feasible without large computational overhead. This suggests that the proposed segmentation pipeline has further potential, although designing such decoders is beyond the scope of this work.



(a) Conventional Segmentation pipeline



(b) Feature Upsampling for Segmentation Pipeline



(c) Upsample Anything Segmentation Pipeline

Figure 8. Comparison of segmentation pipelines. (a) Conventional segmentation pipeline with a Vision Foundation Model encoder and task-specific decoders such as DPT, UPerNet, SegFormer, or Mask2Former. (b) Feature upsampling pipeline using pretrained upsamplers such as FeatUP, LoftUP, JAFAR, or AnyUp, operating on feature maps. (c) Our proposed Upsample Anything, which performs test-time optimization and handles both feature and segmentation upsampling without additional training.

9. Details of Depth Estimation in Table 2.

We followed the DPT-based depth estimation setup used in prior works [10, 23]. Although DPT includes an internal up-

Algorithm 1 Depth estimation setting used in Table 2.

Require: Input image x
Ensure: Predicted depth map D
Extract multi-scale features F using a pretrained Vision Foundation Model encoder (e.g., DINOv2)
Pass F through the DPT head:
  $F_1 \leftarrow \mathrm{Conv}(F,\ k{=}3,\ c{\to}c/2)$
  $F_2 \leftarrow \mathrm{Conv}(F_1,\ k{=}3,\ c/2{\to}32)$
  $F_3 \leftarrow \mathrm{ReLU}(F_2)$
  $D_{inv} \leftarrow \mathrm{Conv}(F_3,\ k{=}1,\ 32{\to}1)$
  $D_{inv} \leftarrow \mathrm{ReLU}(D_{inv})$ (if non-negative)
if `invert` is True **then**
  $D \leftarrow \frac{1}{\mathrm{clip}(s \cdot D_{inv} + t,\ 1\mathrm{e}{-8},\ \infty)}$
else
  $D \leftarrow D_{inv}$
end if
Output the final depth prediction D

sampling, its exact implementation details are not provided in the paper or codebase. Therefore, we reimplemented the head as described in Algorithm 1 and used it consistently for all our depth estimation experiments.

10. From 2D Low-Resolution Feature Maps to 3D High-Resolution Feature Volumes

We extend our test-time optimization (TTO) framework to reconstruct dense 3D feature volumes directly from low-resolution 2D feature maps. Starting with an RGB–Depth (RGB-D) pair, we first downsample the RGB image by a factor of s to simulate a low-resolution feature space. The corresponding high-resolution RGB-D map is used as the guide signal for optimization. During TTO stage, we train only the pixel-wise anisotropic Gaussian kernel parameters $(\sigma_x, \sigma_y, \sigma_z, \theta, \sigma_r)$ so that the 3D Upsample Anyting can accurately project the low-resolution RGB features to their high-resolution RGB-D counterparts. Once optimized, these learned kernels are frozen and reused in upsample stage to upsample semantic features extracted from 2D LR feature into full 3D feature volumes. This process allows each low-resolution feature token to be expanded not only spatially along the x–y plane, but also along the depth axis z, guided by the HR depth map. The resulting tensor $\mathbf{F}_{3D} \in \mathbb{R}^{D_h \times C \times H_h \times W_h}$ captures the local geometric and appearance-aware structure of the scene. We visualize these 3D feature maps using PCA on each depth slice, revealing how distinct depth layers retain meaningful semantic separation while smoothly transitioning across depth. Figure 9 shows that even without explicit 3D supervision, our Full3DJBU reconstructs volumetric features that align with depth continuity, edges, and object boundaries—demonstrating that our framework can generalize from 2D low-resolution feature inputs to 3D high-resolution representations at test time.

11. Gaussian Blob Visualization

To better understand what our learned anisotropic kernels capture, we visualized the Gaussian blobs of our model, as shown in Fig. 10. (a) shows the original high-resolution RGB image, and (b) overlays the learned Gaussian blobs on the corresponding low-resolution image. Although the visualization can be difficult to interpret directly, certain spatial regions such as the eyes, nose, and corners exhibit overlapping or consistently oriented blobs, which suggests that nearby kernels capture semantically similar local structures. This indicates that the learned kernels adaptively encode meaningful directional features rather than behaving randomly. However, similar to other methods that rely on Gaussian Splatting or 2DGS, not every blob is fully interpretable, and some visual noise appears due to overparameterization and kernel redundancy.

12. Segment-then-Upsample Pipeline Visualization Results

This section presents the visualization results of our *segment-then-upsample* pipeline, corresponding to the method in Fig. 8-(c). In this configuration, we perform semantic segmentation on the low-resolution feature maps first and subsequently upsample the segmentation logits by a factor of $16\times$. Despite the large upsampling ratio, our Upsample Anything produces visually sharp and semantically consistent results, as illustrated in Fig. 11. Compared to conventional bilinear interpolation, the recovered boundaries and fine structures are significantly clearer.

13. Formal Relation Between Joint Bilateral Upsampling and Gaussian Splatting

The purpose of this section is not to claim that Joint Bilateral Upsampling (JBU) and Gaussian Splatting (GS) are mathematically equivalent. Instead, we aim to show why the GS framework provides a useful foundation for our formulation. By reinterpreting JBU through the perspective of GS, we reveal a common idea based on continuous and differentiable Gaussian kernels, which motivates our use of GS-style parameter learning in the **Upsample Anything (GSJBU)** framework. In short, this section clarifies the conceptual link between the two views and explains why GS-based test-time optimization naturally applies to feature upsampling.

Notation. Let $F_{\mathrm{lr}} : \mathcal{Q} \to \mathbb{R}^C$ be a low-resolution feature map on a discrete grid $\mathcal{Q} \subset \mathbb{Z}^2$, and let $I : \Omega \to \mathbb{R}^d$ be an HR guidance signal ($d{=}1$ for grayscale, $d{=}3$ for RGB,

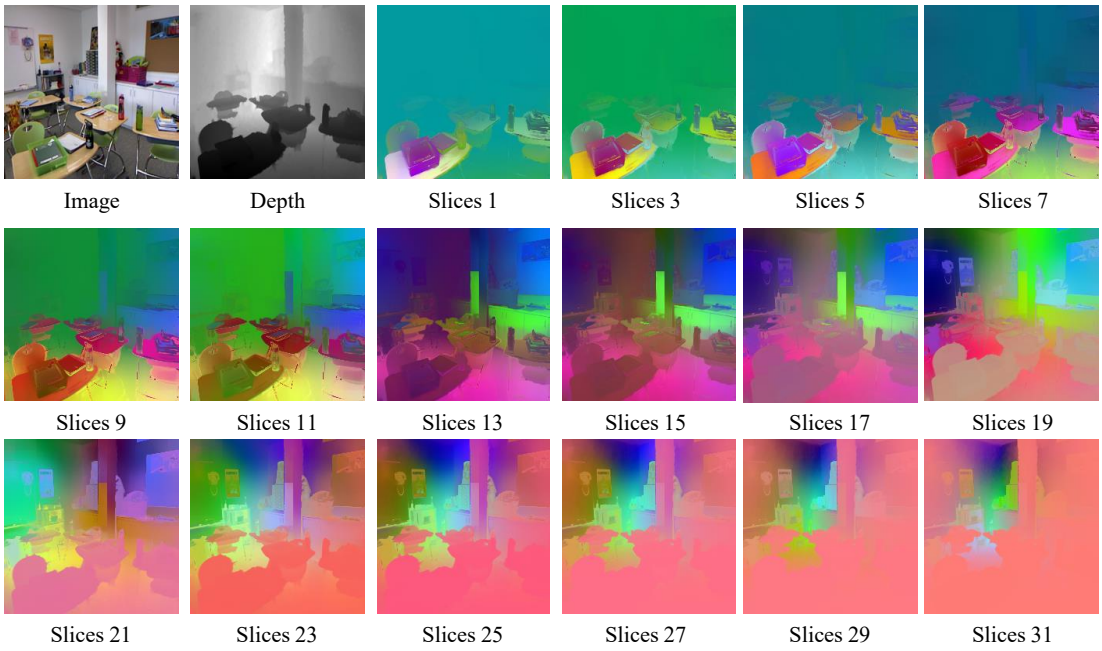
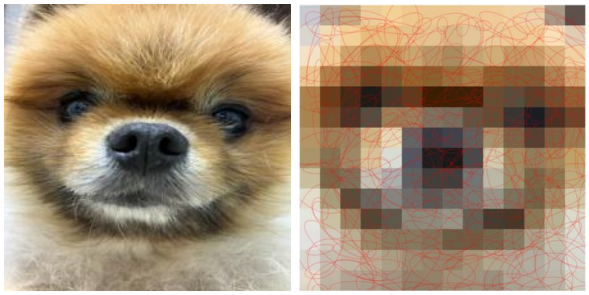


Figure 9. Visualization of our 3D feature upsampling results. The first two columns show the RGB image and its corresponding depth map. The remaining panels depict representative depth slices from the reconstructed 3D high-resolution feature volume obtained using our Upsample Anything. Each slice is visualized via PCA projection into RGB space. Notice that the recovered 3D feature layers exhibit smooth transitions along the depth axis while preserving fine object boundaries and geometric continuity.



(a) *Image* (b) *Gaussian Blob*

Figure 10. Visualization of learned Gaussian blobs. (a) shows the original image and (b) displays the Gaussian blobs overlaid on the low-resolution input. The blobs reveal locally coherent directions and magnitudes, indicating that the learned kernels adapt to the underlying structure of the scene.

etc.). For $p \in \Omega \subset \mathbb{R}^2$, classical JBU is

$$\hat{F}_{\mathrm{hr}}(p) = \frac{\sum_{q\in\Omega(p)} F_{\mathrm{lr}}(q)\, \exp\!\big(-\tfrac{\|p-q\|^2}{2\sigma_s^2}\big)\, \exp\!\big(-\tfrac{\|I(p)-I(q)\|^2}{2\sigma_r^2}\big)}{\sum_{q\in\Omega(p)} \exp\!\big(-\tfrac{\|p-q\|^2}{2\sigma_s^2}\big)\, \exp\!\big(-\tfrac{\|I(p)-I(q)\|^2}{2\sigma_r^2}\big)}. \tag{8}$$

Joint spatial–range lifting. Define the lifted embedding

$$\phi : \Omega \to \mathbb{R}^{2+d}, \qquad \phi(x) := \begin{bmatrix} x \\ I(x) \end{bmatrix},$$

and the block-diagonal covariance

$$\Lambda(\sigma_s, \sigma_r) := \mathrm{diag}\big(\sigma_s^2 I_2,\ \sigma_r^2 I_d\big) \in \mathbb{R}^{(2+d)\times(2+d)}.$$

For $u, v \in \mathbb{R}^{2+d}$, let

$$\mathcal{G}_\Lambda(u, v) := \exp\Big(-\tfrac{1}{2}\,(u-v)^\top \Lambda^{-1}(u-v)\Big).$$

Theorem 1 (JBU as a normalized Gaussian mixture in the joint domain)**.** *Fix $\sigma_s{>}0$, $\sigma_r{>}0$ and let $\Lambda = \Lambda(\sigma_s, \sigma_r)$. Then for any $p \in \Omega$,*

$$\hat{F}_{\mathrm{hr}}(p) = \frac{\sum_{q\in\Omega(p)} F_{\mathrm{lr}}(q)\, \mathcal{G}_\Lambda\big(\phi(p),\, \phi(q)\big)}{\sum_{q\in\Omega(p)} \mathcal{G}_\Lambda\big(\phi(p),\, \phi(q)\big)}. \tag{9}$$

In particular, JBU coincides with evaluating a normalized *Gaussian mixture in the lifted space $\mathbb{R}^{2+d}$ whose centers are $\{\phi(q)\}_{q\in\Omega(p)}$ and whose (isotropic-by-block) covariance is Λ.*

Proof. By construction, $\|\phi(p) - \phi(q)\|^2_{\Lambda^{-1}} = (p-q)^\top(\sigma_s^{-2} I_2)(p-q) + (I(p)-I(q))^\top(\sigma_r^{-2} I_d)(I(p)-I(q))$. Thus $\mathcal{G}_\Lambda(\phi(p),\phi(q)) = \exp\big(-\tfrac{\|p-q\|^2}{2\sigma_s^2}\big)\exp\big(-\tfrac{\|I(p)-I(q)\|^2}{2\sigma_r^2}\big)$, and substituting this identity in (1) yields (9). □

Figure 11. Visualization of the *segment-then-upsample* pipeline. The segmentation logits are first generated at low resolution and then upsampled by 16× using our method. The results exhibit remarkably sharp object boundaries and preserve semantic coherence, highlighting the effectiveness of our upsampling approach.

Corollary 1 (Discrete GS view in the joint domain)**.** *Let* $\mu_q := \phi(q)$ *and* $f_q := F_{\mathrm{lr}}(q)$*. Then Theorem 1 states that JBU equals*

$$\hat{F}_{\mathrm{hr}}(p) = \frac{\sum_q f_q \, \exp\big(-\tfrac{1}{2}(\phi(p)-\mu_q)^\top \Lambda^{-1}(\phi(p)-\mu_q)\big)}{\sum_q \exp\big(-\tfrac{1}{2}(\phi(p)-\mu_q)^\top \Lambda^{-1}(\phi(p)-\mu_q)\big)}, \tag{10}$$

i.e., a Gaussian Splatting *evaluation in* $\mathbb{R}^{2+d}$ *with fixed block-diagonal covariance and centers on the lifted LR grid.*

Connection to standard 2D GS. Standard (2D) GS writes, for $p \in \mathbb{R}^2$,

$$F(p) = \frac{\sum_i \alpha_i \, \exp\big(-\tfrac{1}{2}(p-\tilde{\mu}_i)^\top \tilde{\Sigma}_i^{-1}(p-\tilde{\mu}_i)\big)\, \tilde{f}_i}{\sum_j \alpha_j \, \exp\big(-\tfrac{1}{2}(p-\tilde{\mu}_j)^\top \tilde{\Sigma}_j^{-1}(p-\tilde{\mu}_j)\big)}. \tag{11}$$

The range term in JBU can be *absorbed* by lifting to the joint domain (Theorem 1), or, equivalently, by keeping the domain 2D and letting the amplitude be query-dependent, $\alpha_i(p) = \exp\big(-\frac{\|I(p)-I(\tilde{\mu}_i)\|^2}{2\sigma_r^2}\big)$. The former is strictly *query-independent* and thus mathematically cleaner; the latter matches common GS implementations with view-dependent weights.

Theorem 2 (Specialization of GSJBU to JBU (isotropic limit))**.** *Consider the anisotropic per-center model:*

$$F(p) = \frac{\sum_q f_q \, \exp\big(-\tfrac{1}{2}(p-q)^\top \Sigma_q^{-1}(p-q)\big)\, \beta_q(p)}{\sum_q \exp\big(-\tfrac{1}{2}(p-q)^\top \Sigma_q^{-1}(p-q)\big)\, \beta_q(p)}, \tag{12}$$
$$\beta_q(p) := \exp\big(-\tfrac{\|I(p)-I(q)\|^2}{2\sigma_r^2(q)}\big).$$

If $\Sigma_q \to \sigma_s^2 I_2$ *and* $\sigma_r(q) \to \sigma_r$ *for all* q*, then (12) reduces exactly to JBU (1).*

Proof. Substitute $\Sigma_q = \sigma_s^2 I_2$ and $\sigma_r(q) = \sigma_r$ into (12) to recover the numerator/denominator of (1). □

Proposition 1 (Discrete-to-continuous convergence)**.** *Assume* F_{lr} *admits a bandlimited (or Lipschitz-continuous) interpolation* $\tilde{F} : \Omega \to \mathbb{R}^C$*, and let the LR grid spacing be* Δx*. Then as* $\Delta x \to 0$*,*

$$\sum\nolimits_q \tilde{F}(q)\, \mathcal{G}_\Lambda\big(\phi(p),\phi(q)\big)\, (\Delta x)^2 \;\longrightarrow\; \int_\Omega \tilde{F}(x)\, \mathcal{G}_\Lambda\big(\phi(p),\phi(x)\big)\, dx, \tag{13}$$

and the corresponding normalized ratios converge as well. Hence, discrete JBU converges to its continuous lifted-domain GS counterpart.

Sketch. $\mathcal{G}_\Lambda(\phi(p),\phi(\cdot))$ is bounded and continuous for fixed p. Under the stated regularity, Riemann sums converge to the integral

Parameter	Symbol	Default	Role	Rationale
Spatial sigma (x)	σ_x	init = scale (e.g., 16)	Controls major-axis smoothing; receptive-field size	Initialized proportional to upsampling factor to provide a wide prior; refined by TTO.
Spatial sigma (y)	σ_y	Same as σ_x	Controls minor-axis smoothing	Same reasoning as σ_x; enables anisotropy to emerge during TTO.
Orientation	θ	0	Rotation of the anisotropic Gaussian	Zero-init avoids directional bias; TTO discovers optimal orientation.
Range sigma	σ_r	0.12	Sensitivity to appearance/color similarity	Moderate color differences ($\Delta I \approx 0.2 \sim 0.3$) are significantly downweighted; acts as soft bilateral prior.
Support radius (max)	$R_{\max}$	4–8	Upper bound on spatial Gaussian support	Balances context capture and cost ($\mathcal{O}((2R_{\max}+1)^2)$); too small truncates optimal kernels.
Dynamic multiplier	α_{dyn}	2.0	Converts σ_{eff} to effective support radius	Ensures coverage of $\sim 95\%$ Gaussian mass (2σ rule); prevents under-coverage early in TTO.
Center mode	–	`nearest`	Determines LR anchor for each HR pixel	Nearest-center alignment improves stability and avoids aliasing for large upsampling factors.

Table 8. **Hyperparameter settings for *Upsample Anything*.** All parameters act as soft priors; the effective kernel shape is governed by test-time optimization of pixelwise anisotropic Gaussians.

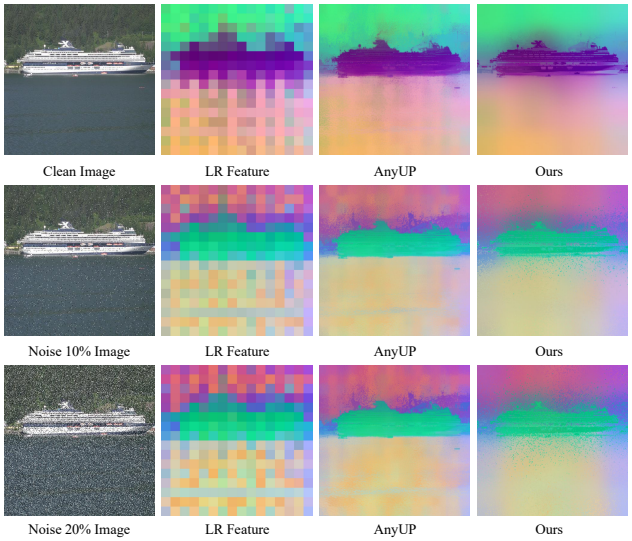


Figure 12. Qualitative comparison under low-SNR and noise corruption. From left to right: RGB input, low-resolution feature, upsampled feature by AnyUp, and ours (Upsample Anything). From top to bottom: clean image, 10% noise, and 20% noise. AnyUp remains stable under noise, while our TTO-based method overfits to noisy pixels, revealing its limitation when directly optimizing on corrupted inputs.

and the denominators stay strictly positive (finite kernel mass). The ratio convergence follows by standard arguments (e.g., dominated convergence and continuity of division on $\mathbb{R} \setminus \{0\}$). □

Consequences. (i) *Equivalence in the joint domain* (Thm. 1) shows that JBU is a GS evaluation on $(x, I(x))$ with block-diagonal covariance. (ii) *Anisotropic generalization* (12) recovers JBU in the isotropic limit (Thm. 2), and enables per-center covariance learning (our GSJBU). (iii) *Discrete-to-continuous consistency* (Prop. 1) justifies replacing sums by integrals when refining the sampling grid.

Implementation note. In practice we adopt (12) with test-time optimization of $(\Sigma_q, \sigma_r(q))$. For stability, $\Sigma_q \succ 0$ is parameterized via $R(\theta_q)\mathrm{diag}(\sigma_x^2(q), \sigma_y^2(q))R(\theta_q)^\top$ with $\sigma_x, \sigma_y > 0$.

14. Hyperparameter Table

The hyperparameters in Table 8 function primarily as soft priors for test-time optimization. Since all spatial and range parameters are refined during the 50 optimization steps, the final performance depends only weakly on their initial values. A well-chosen initialization simply accelerates convergence, whereas suboptimal values are eventually corrected by the optimization itself. The table therefore summarizes practical initialization rules rather than strict hyperparameter requirements. These rules are based on the expected receptive-field size, the dynamic range of the guidance image, and the desired locality prior, and they lead to stable and fast convergence.

15. Limitation under Low-SNR or Corrupted Inputs

Although our method performs robustly across diverse datasets and even under moderate perturbations such as those in ImageNet-C, it exhibits a clear limitation when applied to images with extremely low signal-to-noise ratios or severe corruption.

Because our framework performs test-time optimization (TTO) by reconstructing the input image itself, the optimization process inherently assumes that the image contains a clean and reliable signal. When the input is degraded by noise—such as salt-and-pepper artifacts or heavy sensor perturbations—the model tends to overfit to these corruptions rather than recovering the underlying structure. Figure 12 illustrates this effect: the first row shows results on a clean image, while the second and third rows demonstrate increasing corruption levels of 10% and 20%, respectively.

Despite the noise, pretrained Vision Foundation Models still produce reasonable feature embeddings, and AnyUp remains stable by directly upsampling feature maps. In contrast, our TTO-based Upsample Anything reconstructs the noisy signal faithfully, which unintentionally amplifies noise in both the reconstructed RGB and upsampled feature domains.

This limitation is not unique to our method but is common across all TTO-based image restoration approaches that optimize directly on corrupted inputs. While one could incorporate a denoising stage before optimization to alleviate this issue, we consider it outside the current scope.

In summary, AnyUp demonstrates higher robustness under corrupted or low-SNR conditions, whereas our Upsample Anything excels when inputs are visually clean or when handling multi-modal signals such as RGB-D or 3D features.

References

[1] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021. 2

[2] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016. 1

[3] Paul Couairon, Loick Chambon, Louis Serrano, Jean-Emmanuel Haugeard, Matthieu Cord, and Nicolas Thome. Jafar: Jack up any feature at any resolution. *arXiv preprint arXiv:2506.11136*, 2025. 2, 3, 5

[4] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2

[5] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 1

[6] Stephanie Fu, Mark Hamilton, Laura E. Brandt, Axel Feldmann, Zhoutong Zhang, and William T. Freeman. Featup: A model-agnostic framework for features at any resolution. In *The Twelfth International Conference on Learning Representations*, 2024. 2, 3, 5

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2

[8] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022. 2

[9] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 conference papers*, pages 1–11, 2024. 7

[10] Haiwen Huang, Anpei Chen, Volodymyr Havrylov, Andreas Geiger, and Dan Zhang. Loftup: Learning a coordinate-based feature upsampler for vision foundation models, 2025. 2, 3, 5, 1

[11] Bingxin Ke, Anton Obukhov, Shengyu Huang, Nando Metzger, Rodrigo Caye Daudt, and Konrad Schindler. Repurposing diffusion-based image generators for monocular depth estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9492–9502, 2024. 1

[12] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 3

[13] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4015–4026, 2023. 2

[14] Johannes Kopf, Michael F Cohen, Dani Lischinski, and Matt Uyttendaele. Joint bilateral upsampling. *ACM Transactions on Graphics (ToG)*, 26(3):96–es, 2007. 3, 4

[15] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 1

[16] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11976–11986, 2022. 2

[17] 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. 2

[18] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021. 2

[19] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12179–12188, 2021. 2

[20] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *European conference on computer vision*, pages 746–760. Springer, 2012. 1

[21] Shuangbing Song, Fan Zhong, Tianju Wang, Xueying Qin, and Changhe Tu. Guided linear upsampling. *ACM Transactions on Graphics (TOG)*, 42(4):1–12, 2023. 6, 7

[22] Saksham Suri, Matthew Walmer, Kamal Gupta, and Abhinav Shrivastava. Lift: A surprisingly simple lightweight feature transform for dense vit descriptors. In *European Conference on Computer Vision*, pages 110–128. Springer, 2024. 2, 3, 5

[23] Thomas Wimmer, Prune Truong, Marie-Julie Rakotosaona, Michael Oechsle, Federico Tombari, Bernt Schiele, and Jan Eric Lenssen. Anyup: Universal feature upsampling. *arXiv preprint arXiv:2510.12764*, 2025. 2, 3, 5, 7, 1

[24] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *Proceedings of the European conference on computer vision (ECCV)*, pages 418–434, 2018. 2

[25] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and

efficient design for semantic segmentation with transformers. *Advances in neural information processing systems*, 34: 12077–12090, 2021. 2

[26] 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. 1

[27] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11975–11986, 2023. 2

[28] Xinjie Zhang, Xingtong Ge, Tongda Xu, Dailan He, Yan Wang, Hongwei Qin, Guo Lu, Jing Geng, and Jun Zhang. Gaussianimage: 1000 fps image representation and compression by 2d gaussian splatting. In *European Conference on Computer Vision*, pages 327–345. Springer, 2024. 3

[29] Yunxiang Zhang, Bingxuan Li, Alexandr Kuznetsov, Akshay Jindal, Stavros Diolatzis, Kenneth Chen, Anton Sochenov, Anton Kaplanyan, and Qi Sun. Image-gs: Content-adaptive image representation via 2d gaussians. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pages 1–11, 2025. 3

[30] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 127(3):302–321, 2019. 1

[31] Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. ibot: Image bert pre-training with online tokenizer. *arXiv preprint arXiv:2111.07832*, 2021. 2

[32] Lingting Zhu, Guying Lin, Jinnan Chen, Xinjie Zhang, Zhenchao Jin, Zhao Wang, and Lequan Yu. Large images are gaussians: High-quality large image representation with levels of 2d gaussian splatting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 10977–10985, 2025. 3