CoordUp: Upsampling Any Feature Beyond the Megapixel Regime

Anonymous authors
Paper under double-blind review

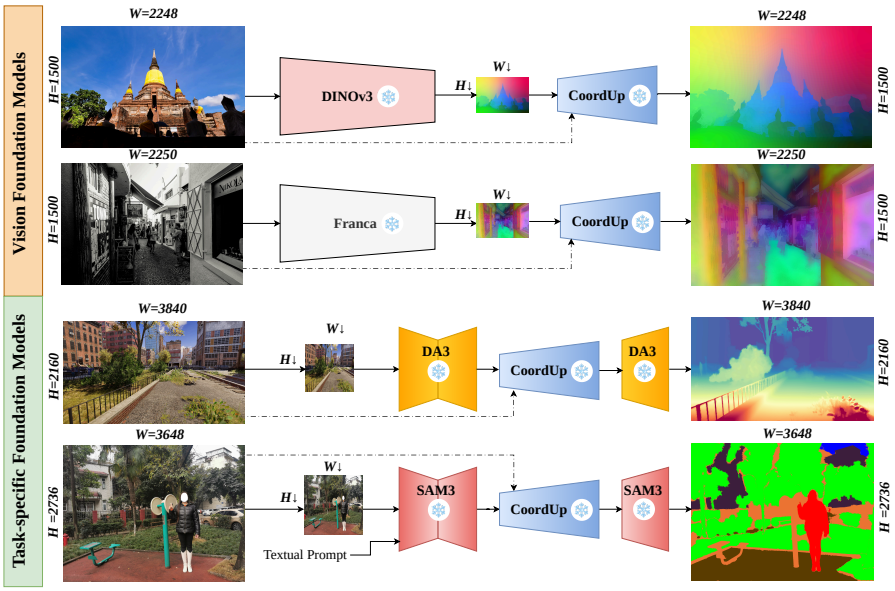


Figure 1: **CoordUp upsamples any features.** Top row: CoordUp upsamples *foundational features* from DINOv3 and Franca (3-channel PCA visualizations). Bottom row: CoordUp upsamples the *final activations* of DA3 and SAM3, enabling high-resolution dense predictions.

Abstract

Vision foundation models produce semantically rich but spatially coarse features, limiting fine-detail recovery in dense prediction tasks. Existing feature upsamplers tackle this problem but are often tied to specific backbones or require materializing dense feature grids, making high-resolution inference costly. We introduce CoordUp, a lightweight, feature-agnostic upsampler that formulates feature upsampling as a coordinate-conditioned operator. Across five frozen backbones and three high-resolution datasets for semantic segmentation and depth estimation, it consistently improves boundary-sensitive metrics over previous encoder-agnostic upsamplers. It also transfers zero-shot to the final activations of DA3 and SAM3, improving boundary quality without retraining. Finally, its coordinate-based formulation allows CoordUp to scale gracefully with output resolution, enabling, for example, an 80 Mpx output using only 8.6 GB of GPU memory.

1 Introduction

Thanks to their ability to provide semantically rich features that can be leveraged across a variety of downstream tasks, transformer-based backbones pre-trained on very large datasets – commonly referred to as Vision Foundation Models (VFMs) – have become standard components of modern computer vision architectures. Accordingly, VFMs such as DINOv2 (Oquab et al., 2024), DINOv3 (Siméoni et al., 2025), RADIOv2.5 (Heinrich et al., 2025), and Franca (Venkataramanan et al., 2026), can be paired with lightweight heads to address detection, segmentation, and other tasks.

However, although semantically rich, the foundational features provided by VFMs are spatially coarse, as they are produced on a patch-aligned grid that is much smaller than the input image. For

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.

example, with patch size 16×16, a 1024×1024 pixels image yields a feature grid of only 64×64. Hence, in dense prediction tasks such as semantic segmentation, matting, and depth estimation, the output must be generated at a substantially higher resolution than the feature grid. Yet, procedurally upsampling the feature grid —or the grid-resolution predictions produced by a task-specific head— using classical interpolation methods can result in substantial inaccuracies in fine-grained spatial details, such as sharp depth discontinuities and thin structures in segmentation maps.

Motivated by this observation, recent works have explored task-agnostic modules that upsample the low-resolution features of a VFM to the input resolution, guided by the RGB image. FeatUp (Fu et al., 2024) pioneered this direction with learned kernels inspired by joint bilateral upsampling (Kopf et al., 2007). LoftUp (Huang et al., 2025), with a coordinate-based formulation, and JAFAR (Couairon et al., 2025) replaced these kernels with cross-attention, while UPLiFT (Walmer et al., 2026) introduced a local operator to avoid its quadratic complexity. All these modules, however, are tied to hyper-parameters of the target encoder, primarily its number of feature channels, and thus cannot generalize across VFMs. Recent methods such as AnyUp (Wimmer et al., 2026) and NAF (Chambon et al., 2026) lifted this limitation by proposing *feature-agnostic* upsamplers: by making cross-attention agnostic to the dimensionality of the value vectors, they can be applied to any VFM, including unseen backbones with different channel counts and patch sizes.

A further gap concerns image resolution: existing upsamplers are typically deployed and evaluated on relatively low-resolution images, generally below 1 Mpx. We argue that high-resolution images, e.g., in the 8–12 Mpx range, are instead the regime in which feature upsampling matters most. After all, the very purpose of this research direction is to capture fine-grained details, and with their abundance of small objects, thin structures, and minute details, high-resolution images are its most natural testbed. This regime is out of reach because existing feature-agnostic upsamplers materialize intermediate tensors at the target resolution, so their memory grows with the number of output pixels: on a 48 GB GPU, AnyUp and NAF already run out of memory at 4 and 16 Mpx, respectively (Sec. 4.4).

We introduce CoordUp (Fig. 1), a feature-agnostic module that upsamples features by learning a continuous function of spatial coordinates. Given a query that combines continuous spatial coordinates with high-resolution RGB information, CoordUp predicts the feature at the queried location through a cross-attention mechanism that, as in AnyUp and NAF, is agnostic to the dimensionality of the value vectors. This formulation allows arbitrary features to be queried at any output resolution. Moreover, it is particularly memory-efficient, since the high-resolution feature map never needs to be fully materialized: coordinates can be queried in batches, processed in parallel on the GPU without a substantial runtime penalty. As a result, unlike previous approaches, CoordUp enables feature upsampling well beyond 1 Mpx, on images of 8–12 Mpx and above.

We apply CoordUp to upsample foundational features from different VFMs (Fig. 1, top). Using DINOv2, DINOv3, RADIOv2.5, and Franca with a standard linear probing protocol, we evaluate semantic segmentation on MaSS13K (Xie et al., 2025) and monocular depth estimation on UnrealStereo4K (Tosi et al., 2021) and Synth4K (Yu et al., 2026), all high-resolution datasets with pixel-accurate ground truth.

Moreover, we observe that popular Task-specific Foundation Models (TFMs) designed and trained to be deployed zero-shot on any data, such as Depth Anything 3 (Lin et al., 2026) and Segment Anything 3 (Carion et al., 2026), tend to struggle with high resolution images. On the one hand, their dense decoders must materialize large activations, easily exceeding the GPU memory. On the other hand, the models may turn out less accurate when they are pushed beyond their training regime. Motivated by these observations, we propose to explore the adoption of feature-agnostic modules to enable zero-shot inference on high-resolution images by upsampling the final activations of a TFM operating on a downscaled input consistent with its training (Fig. 1, Bottom). *We will publicly release the code upon acceptance.* To summarize our contributions:

- We propose CoordUp, a novel *feature-agnostic* feature upsampler conditioned on continuous query coordinates. Our design provides explicit control over memory usage, avoiding the need to materialize full-resolution feature maps and enabling efficient parallel processing of coordinate batches. As a result, CoordUp unlocks inference at arbitrary output resolutions, including resolutions well above the 1 Mpx regime.
- We explore feature upsampling beyond VFMs, demonstrating how *feature-agnostic* upsamplers can be deployed with TFMs for zero-shot inference on high-resolution images.

- Throughout our experimental evaluation, which focuses on semantic segmentation and depth estimation on high-resolution images in both the VFM and TFM settings, CoordUp consistently outperforms previous *feature-agnostic* feature upsamplers on the key metrics that capture the ability to recover fine-grained image structures.

2 Related Work

Feature upsamplers. Feature upsamplers are task-agnostic modules that turn coarse backbone representations into dense ones. FeatUp (Fu et al., 2024) adapts the spatial-range kernel of joint bilateral upsampling (Kopf et al., 2007), computed from the guidance image, into a learned kernel for deep features. LiFT (Suri et al., 2024) instead learns a lightweight transform directly in feature space, without guidance. JAFAR (Couairon et al., 2025) aligns image queries with feature keys through cross-attention, whereas LoftUp (Huang et al., 2025) formulates upsampling as a coordinate mapping, trained on mask-derived high-resolution pseudo-ground truth (Kirillov et al., 2023). UPLiFT (Walmer et al., 2026) replaces LiFT's refinement stage with local attentional pooling, avoiding the quadratic cost of global attention. All these approaches depend on backbone-specific hyper-parameters, primarily the channel count, and must be retrained for each encoder. More recent methods remove this dependence. AnyUp (Wimmer et al., 2026) maps incoming representations into a unified feature space before applying attention, while retaining the original encoder features as values, so a single set of weights transfers to unseen encoders with different channel counts and patch sizes. Upsample Anything (Seo et al., 2026) requires no dataset-level training and instead fits, at test time, a per-pixel anisotropic Gaussian kernel over spatial and range cues for each input image; this per-image optimization, however, limits throughput at high resolution and scale. NAF (Chambon et al., 2026) returns to dataset-level training and predicts filtering weights from the high-resolution image using local cross-scale attention. Both AnyUp and NAF, however, materialize intermediate tensors at the target resolution, so memory scales with output size and exceeds GPU capacity at high resolutions. CoordUp shares the feature-agnostic design of these methods, producing each output feature as a combination of encoder tokens, but conditions it on continuous query coordinates. Unlike LoftUp, whose coordinate-based formulation is tied to a specific backbone, CoordUp thus combines arbitrary-resolution querying with transfer to unseen features, without ever materializing the full dense feature map. Moreover, we show that feature-agnostic upsamplers, and CoordUp in particular, can upsample the final activations of TFMs, enabling their zero-shot inference on high-resolution images.

Coordinate-based inference. Continuous coordinate representations are widely used in neural fields (Mildenhall et al., 2020; Sitzmann et al., 2020; Tancik et al., 2020) and arbitrary-resolution dense prediction. LIIF (Chen et al., 2021) decodes images from latent codes at continuous coordinates, while CoordFormer (Curti et al., 2026) and InfiniDepth (Yu et al., 2026) extend this idea to very-high-resolution segmentation and depth estimation. These methods condition task-specific outputs on coordinates, refining predictions computed from coarse features. CoordUp instead conditions the feature upsampling operator itself, supplying task-independent features that are already accurate at boundaries, and is thus complementary to them.

3 Method

Let $\mathcal{M}$ be a frozen VFM or TFM, which maps an image $I \in \mathbb{R}^{3\times H\times W}$ to a coarse feature map with stride s, $F_{in} = \mathcal{M}(I) \in \mathbb{R}^{C\times\lfloor H/s\rfloor\times\lfloor W/s\rfloor}$. The goal is to upsample F_{in} to the spatial resolution of the input image, $H \times W$. To address this, we propose CoordUp, a learnable upsampler Γ_θ that formulates feature upsampling as a *pointwise* operation over a continuous spatial domain. Given the coarse feature map F_{in}, a coordinate pair $(x, y) \in [0, W-1] \times [0, H-1]$, the corresponding RGB values, $I(x, y) \in \mathbb{R}^3$, and a downsampled copy of I, denoted by I_{key}, CoordUp predicts a feature vector at the specified coordinates:

$$\hat{f}(x, y) = \Gamma_\theta(F_{in}, (x, y), I(x, y), I_{key}),\ \hat{f}(x, y) \in \mathbb{R}^C. \tag{1}$$

A dense output at any resolution can be obtained by querying Eq. (1) at every desired output coordinate pair. CoordUp predicts a feature $\hat{f}(x, y)$ through a cross-attention operation involving a spatial *Query*, computed by a *Query Encoder* ($\mathcal{Q}$) from (x, y) and $I(x, y)$; a set of *Values*, drawn from the feature vectors in F_{in}; and a set of *Keys*, computed by a *Key Encoder*, ($\mathcal{K}$) from I_{key}. Hence, each output

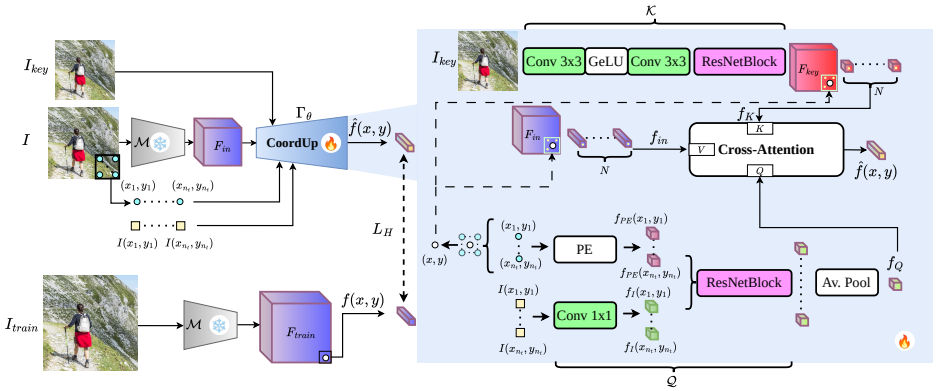


Figure 2: **Overview of CoordUp architecture (right) and training procedure (left).** CoordUp consists of a *Query Encoder* ($\mathcal{Q}$), shown on the bottom-right, a *Key Encoder* ($\mathcal{K}$), on the top-right, and a single cross-attention layer. As shown on the left, CoordUp is trained to upsample the features extracted by a foundation model $\mathcal{M}$. This is achieved by providing as inputs the feature computed by $\mathcal{M}$ on a downsampled version, I, of each training image, I_{train}, and as targets those computed by $\mathcal{M}$ on the full resolution image I_{train}.

feature is a convex combination of a subset of the feature vectors in F_{in}, making the operation agnostic to C. Moreover, the only learnable parameters in our architecture belong to $\mathcal{Q}$ and $\mathcal{K}$. In Sec. 3.1 and 3.2 we present the architecture and training protocol of CoordUp with the help of Fig. 2, whereas inference is described in Sec. 3.3. Additional details are presented in App. A.1.

3.1 Architecture

Query Encoder. The *Query* encodes the spatial location where the upsampled feature must be predicted, (x, y), along with the appearance of the high-resolution image at that location, $I(x, y)$. As shown in Fig. 2 (bottom-right), after normalization in $[0, 1] \times [0, 1]$ the spatial location (x, y) is encoded using a log-spaced sinusoidal positional embedding, $f_{PE} \in \mathbb{R}^{C_{PE}}$ (Tancik et al., 2020), whereas $I(x, y)$ is expanded into an appearance embedding, f_I, by a 1×1 convolution ($\mathbb{R}^3 \to \mathbb{R}^{C_I}$). In our experiments we set $C_{PE} = 42, C_I = 128$. The positional and appearance embeddings, f_{PE} and f_I, are then concatenated and fed into a ResNetBlock to obtain the final *Query* vector, $f_Q \in \mathbb{R}^{C_Q}$, with $C_Q = 256$. Our choice to embed appearance using a 1×1 convolution is motivated by the intuition that guidance from the high-resolution image should preserve sharpness as much as possible, thereby avoiding the blurring introduced by operators with a larger spatial support.

Key Encoder. As we want our upsampler to generalize to features F_{in} provided by any encoder $\mathcal{M}$, the *Key* should not depend on $\mathcal{M}$; we therefore build it directly from the input image I rather than from any encoder-specific features, similarly to Chambon et al. (2026). Moreover, to upsample effectively, we want the *Key* to have a similar spatial resolution to F_{in}, which helps the model learn the correct upsampling operation. Based on these considerations $\mathcal{K}$ takes as input a downsampled version of I, $I_{key} \in \mathbb{R}^{3\times h\times w}$, with $h < H$, $w < W$, chosen so that h and w are $4\times$ the resolution of F_{in}. Because $\mathcal{K}$ has total stride $s_k = 4$, the resulting feature map $F_{key} = \mathcal{K}(I_{key})$ ends up with exactly the resolution of F_{in}, as required. As shown in Fig. 2 (top-right), $\mathcal{K}$ consists of two convolutions – kernel 3×3, stride 2, 128 filters, GELU activations (Hendrycks & Gimpel, 2016) – followed by a final ResNetBlock (He et al., 2016) with output dimension $C_K = 256$. A further benefit of this design is efficiency: since F_{key} is computed once per input image, the computational cost of $\mathcal{K}$ is independent of the number of queried coordinates.

Cross Attention. Given the query coordinates (x, y), we first map them to (x_V, y_V), i.e. the corresponding spatial location in the coarse feature map F_{in} produced by $\mathcal{M}$. Specifically, $(x_V, y_V) = (x/(s \cdot s_w),\ y/(s \cdot s_h))$, where s is the stride of $\mathcal{M}$, and s_w, s_h denote the horizontal and vertical scaling factors between the original image and the – possibly – downsampled input image fed into $\mathcal{M}$ (see Fig. 1) . Since F_{key} and F_{in} have the same spatial resolution, these coordinates also identify the corresponding spatial location into F_{key}: $(x_K, y_K) = (x_V, y_V)$. After projecting the query coordinates (x, y) into F_{in} and F_{key}, we extract a local neighbourhood of $N = 5 \times 5$ vectors via bilinear interpolation from both feature maps, obtaining two sequences of tokens, f_{in} and f_K. Then, CoordUp predicts $\hat{f}(x, y)$, i.e. the feature at (x, y), by a single multi-head cross-attention oper-

ation between the query vector f_Q and the set of keys and values given by f_K and f_{in}, respectively. As the query f_Q, and the keys f_K, have the same dimensionality ($C_K = C_Q = 256$), we do not apply projection layers before computing the dot product that yields the attention weights. Similarly, the values f_{in} are drawn from the feature map F_{in} produced by $\mathcal{M}$ and used in our cross-attention layer without undergoing any projection. Therefore, the architecture of CoordUp imposes no constraints on the number of channels C of the frozen VFM or TFM, other than requiring C to be divisible by the number of attention heads, $n_h = 4$. This requirement is typically satisfied by VFM and TFM architectures; otherwise simple channel-wise padding can be applied.

3.2 Training

As illustrated in Fig. 2 (left), given a training image I_{train}, the low-resolution features F_{in} provided in input to CoordUp are extracted by $\mathcal{M}$ from an image I obtained by downsampling I_{train} by a factor of 2, whereas the target features F_{train} are extracted by $\mathcal{M}$ from the full-resolution training image I_{train}:

$$F_{in} = \mathcal{M}(I)\,, \qquad F_{train} = \mathcal{M}(I_{train})\,. \tag{2}$$

Hence, training consists in a $\times 2$ upsampling of $\mathcal{M}$'s own output. Since the query coordinates are sampled from a continuous spatial domain, the upsampling factor used during training does not constrain the factors that can be used at inference time. To train CoordUp we rely on DINOv3-B as the foundation model $\mathcal{M}$. Once trained along with DINOv3-B, CoordUp can be deployed as a frozen component to upsample the features from any VFM or TFM at any desired resolution.

Coordinate sampling. During training, we sample batches of $B_T = 256$ coordinate tiles, each consisting of $n_t = (s/2) \times (s/2)$ coordinates (see Fig. 2 top-left), where s is the stride of $\mathcal{M}$. Since I is obtained by downsampling I_{train} by a factor of 2, each tile corresponds to a $s \times s$ patch in I_{train}, and, consequently, to a single target feature vector on the spatial grid of F_{train}. For each tile, the n_t vectors computed by $\mathcal{Q}$ are averaged to obtain the *Query* f_Q (Fig. 2 bottom-right) to be used in the cross-attention layer which yields the prediction $\hat{f}$ associated with the tile. The keys and values, f_K and f_{in}, are gathered from F_{key} and F_{in}, respectively, by extracting a local neighbourhood of $N = 5 \times 5$ vectors around the tile center. This coordinate sampling strategy provides uniform supervision across the target feature vectors in F_{train}, while exposing $\mathcal{Q}$ to distinct query locations within each tile. As a result, CoordUp learns a continuous function of spatial coordinates.

Loss. Raw features exhibit large per-image and per-channel magnitude variation. We therefore normalize the residuals by the target's per-channel (c) standard deviation σ_c, computed over the B_T tiles sampled from each image, and apply a Huber penalty (Girshick, 2015),

$$L_H = \frac{1}{B_T\, C} \sum_{i=1}^{B_T} \sum_{c=1}^{C} \mathcal{H}_\beta\Big(\tfrac{\hat{f}_c(x^i,y^i) - f_c(x^i,y^i)}{\sigma_c} \Big)\,, \quad \beta = 1, \tag{3}$$

where $f(x^i, y^i)$ is sampled from F_{train} at the corresponding tile center coordinate (x^i, y^i) and $f_c(x^i, y^i)$ denotes its c-th channel. The robust Huber penalty reduces the tendency to average over ambiguous feature values, encouraging the model to commit to one of the plausible values. This is particularly important near boundaries, where a coarse cell may contain features from two distinct regions and the upsampler is expected to recover the appropriate one.

Training recipe. We train on SA-1B (Kirillov et al., 2023), one image per step, 50k steps, AdamW (Loshchilov & Hutter, 2019), learning rate 1×10^{-4}, weight decay 10^{-5}, and `bf16` autocast.

3.3 Inference

At inference time, the input image I is directly processed to produce the coarse features $F_{in} = \mathcal{M}(I)$. The key feature map F_{key} is computed from I_{key} only once and then cached, after which each queried coordinate is processed independently by the cross-attention layer. The computational cost of an inference pass is therefore linear in the number of desired output coordinates, with no term that grows quadratically with the output resolution. Since CoordUp performs per-coordinate prediction (Eq. (1)), the query set can be chosen freely. Thus, CoordUp can be evaluated over the full input image or, under constrained memory or time budgets, the queried coordinates may instead correspond to a small region of interest or a sparse set of points. Moreover, peak memory usage is determined by the number of coordinates processed simultaneously in an inference batch, rather than by the spatial resolution required by the downstream task. We exploit this property by streaming coordinate

queries in batches, so that, unlike other upsamplers, a dense high-resolution feature map need never be materialized. For instance, a 64 Mpx map with $C = 1024$ would occupy 131 GB in fp16. A visual overview of the inference procedure is provided in App. Fig. 5 and App. A.1, illustrating how each queried coordinate is processed independently and how, unlike during training, no coordinate tiling or *Query* averaging is required.

4 Experimental evaluation

We evaluate CoordUp by upsampling frozen VFM features for high-resolution semantic segmentation and depth estimation (Sec. 4.2), by upsampling the final activations of frozen TFMs for zero-shot inference (Sec. 4.3), and by profiling its memory and runtime (Sec. 4.4). In our experiments, we compare CoordUp with other encoder-agnostic upsamplers: AnyUp (Wimmer et al., 2026), NAF (Chambon et al., 2026). *Due to space constraints, ablations are in App. A.3, additional analysis in App. A.4, further implementation details in App. A.5, and more qualitatives in App. A.6.*

4.1 Datasets and Metrics

For semantic segmentation, we use MaSS13K (Xie et al., 2025) (8–64 Mpx; 500 validation and 1500 test images). For monocular depth estimation, we use UnrealStereo4K (Tosi et al., 2021) ($\sim$8 Mpx; 80 validation and 200 test frames) and Synth4K (Yu et al., 2026) ($\sim$8 Mpx; 3950 test frames). For both tasks, predictions are evaluated at the native resolution of the input images, and we report both global metrics and metrics designed to capture the recovery of fine-grained image structures. The latter are essential because boundaries, where coarse features mix neighboring regions, cover only a small fraction of pixels, so localization errors barely affect global metrics. Specifically, for semantic segmentation, we use mIoU together with the boundary metrics BIoU and BF1, following the evaluation protocol proposed for MaSS13K (Xie et al., 2025). For depth estimation, we report A.Rel as the global metric, together with the accuracy term and a modified completeness term of the Pseudo Depth Boundary Error (Pham et al., 2025), denoted by $\epsilon_{\mathrm{PDBE}}^{\mathrm{acc}}$ and $\epsilon_{\mathrm{PDBE}}^{\mathrm{compl}}$, respectively. More details on the metrics are presented in App. A.2.

4.2 Linear Probing with VFMs

In Tab. 1 we consider five VFMs of comparable capacity but different pre-training objectives as model $\mathcal{M}$: DINOv3-B and DINOv3-S+ (Siméoni et al., 2025), DINOv2-B with registers (Darcet et al., 2024), Franca-B (Venkataramanan et al., 2026) and RADIO 2.5-B (Heinrich et al., 2025). Both the VFMs and the upsamplers are kept frozen, only the prediction head, comprising batch normalization followed by a linear layer, is trained on the training split of the considered datasets. Because they require the entire output feature map to be materialized, with the considered datasets AnyUp, NAF, and bilinear upsampling do not permit training the prediction heads at full output resolution on the high-end GPU used in our experiments (A100 with 64 GB of VRAM). Therefore, for each method, we carefully tune the upsampler resolution and batch size (Tab. 1) that enable prediction-head training and use the same resolution at inference time. For evaluation against the ground truth, the resulting predictions—segmentation logits or depth values—are then bilinearly upsampled to the full output resolution. In contrast, because CoordUp needs to store only the features predicted for each coordinate batch, it can always generate predictions directly at the full output resolution. Finally, it is worth noting that CoordUp has the fewest number of parameters (466k) among the learned upsamplers considered.

Semantic segmentation results. On MaSS13K, bilinear interpolation achieves an mIoU of 83.7–86.4 across the different VFMs, but only 14.5–17.6 BIoU, showing that good global accuracy does not translate into the ability to recover accurate boundaries. CoordUp roughly doubles the boundary score of bilinear interpolation for every VFM. Moreover, it consistently outperform NAF, the strongest learned competitor, across all considered VFMs and on both splits, improving the BIoU by 0.53–1.45 points on Val and by 1.31–1.87 points on Test. It also achieves the best BF1 in all ten VFM–split combinations. With bilinear interpolation boundary quality varies by more than three BIoU points between the weakest and the strongest VFMs; with CoordUp, by contrast, four of the five VFMs perform within 0.75 points of one other (32.02–32.77), and we observe no performance bias in favor of DINOv3-B, the model $\mathcal{M}$ used to train CoordUp. Boundary quality thus becomes largely

Table 1: **Results for linear probing with VFMs.** Semantic segmentation is evaluated on MaSS13K (Val/Test), depth estimation on UnrealStereo4K (Val/Test) and Synth4K (Test). Params denotes the number of parameters of the upsampler only. "↑" ("↓") indicates that higher (lower) values are better. The top-3 results are highlighted as first, second, and third.

Method	Encoder Input Res.	Upsampler Res.	Batch Size	Params (M)	Semantic Segmentation						Depth Estimation								
					MaSS13K						UnrealStereo4K						Synth4K		
					Val (500)			Test (1500)			Val (80)			Test (200)			Test (3950)		
					mIoU↑	BIoU↑	BF1↑	mIoU↑	BIoU↑	BF1↑	$\epsilon^{\text{acc}}_{\text{PDBE}}$ ↓	$\epsilon^{\text{compl}}_{\text{PDBE}}$ ↓	A.Rel↓	$\epsilon^{\text{acc}}_{\text{PDBE}}$ ↓	$\epsilon^{\text{compl}}_{\text{PDBE}}$ ↓	A.Rel↓	$\epsilon^{\text{acc}}_{\text{PDBE}}$ ↓	$\epsilon^{\text{compl}}_{\text{PDBE}}$ ↓	A.Rel↓
$\mathcal{M}$: DINOv3-B																			
+ Bilinear	2048 × 2048	1536 × 1536	8	0	85.70	16.94	0.2182	85.71	16.86	0.2215	4.832	5.193	0.1129	4.272	4.999	0.5637	4.021	4.930	0.3575
+ AnyUp	2048 × 2048	896 × 896	8	0.845	85.94	27.98	0.3277	85.76	27.37	0.3269	2.498	2.263	0.1193	1.800	2.044	0.5381	2.406	2.551	0.3652
+ NAF	2048 × 2048	1280 × 1280	8	0.663	87.06	31.32	0.3698	86.74	30.17	0.3644	2.294	2.204	0.1152	1.467	1.899	0.5545	2.029	2.421	0.3556
+ CoordUp (Ours)	2048 × 2048	*full*	16	0.466	87.02	32.77	0.3836	86.83	32.04	0.3825	2.314	2.003	0.1056	1.264	1.576	0.5704	1.962	2.134	0.3551
$\mathcal{M}$: DINOv3-S+																			
+ Bilinear	2048 × 2048	1536 × 1536	16	0	84.57	16.32	0.2111	84.41	16.24	0.2139	4.660	5.082	0.1300	3.936	4.714	0.5589	4.096	5.009	0.3761
+ AnyUp	2048 × 2048	896 × 896	16	0.845	84.81	27.07	0.3195	84.44	26.22	0.3163	2.571	2.223	0.1161	1.653	1.932	0.5644	2.432	2.515	0.3644
+ NAF	2048 × 2048	1280 × 1280	16	0.663	85.90	30.89	0.3657	85.35	29.60	0.3581	2.286	2.123	0.1176	1.651	1.962	0.6033	2.107	2.533	0.3740
+ CoordUp (Ours)	2048 × 2048	*full*	16	0.466	85.98	32.02	0.3750	85.63	31.16	0.3723	2.206	1.880	0.1339	1.211	1.448	0.5148	1.912	2.146	0.3413
$\mathcal{M}$: DINOv2-B with registers																			
+ Bilinear	2044 × 2044	2044 × 2044	4	0	83.73	14.46	0.1923	83.74	14.45	0.1958	4.346	4.791	0.1202	4.365	5.010	0.5575	3.997	4.845	0.3855
+ AnyUp	2044 × 2044	1022 × 1022	4	0.845	86.02	28.10	0.3297	85.95	27.62	0.3304	2.628	2.253	0.1174	2.002	2.098	0.5502	2.469	2.521	0.3764
+ NAF	2044 × 2044	1540 × 1540	4	0.663	86.83	31.03	0.3624	86.54	29.94	0.3603	2.492	2.267	0.1177	1.690	1.989	0.5288	2.194	2.488	0.3852
+ CoordUp (Ours)	2044 × 2044	*full*	16	0.466	86.64	32.35	0.3772	86.53	31.73	0.3777	2.185	1.957	0.1049	1.304	1.685	0.5962	2.038	2.266	0.3762
$\mathcal{M}$: Franca-B																			
+ Bilinear	2044 × 2044	2044 × 2044	4	0	84.13	15.99	0.2071	84.22	15.87	0.2095	4.181	4.589	0.1288	4.416	5.033	0.4992	4.048	4.942	0.3499
+ AnyUp	2044 × 2044	1022 × 1022	4	0.845	85.48	27.41	0.3269	85.45	26.93	0.3259	2.518	2.252	0.1177	1.858	2.164	0.5038	2.496	2.609	0.3457
+ NAF	2044 × 2044	1540 × 1540	4	0.663	86.38	31.23	0.3705	86.22	30.21	0.3632	2.389	2.142	0.1067	1.785	2.101	0.5197	2.242	2.484	0.3565
+ CoordUp (Ours)	2044 × 2044	*full*	16	0.466	86.26	32.42	0.3820	86.26	31.90	0.3822	2.133	1.949	0.1175	1.225	1.612	0.4811	1.979	2.265	0.3356
$\mathcal{M}$: RADIOv2.5-B																			
+ Bilinear	2048 × 2048	1536 × 1536	8	0	86.38	17.61	0.2222	86.26	17.28	0.2261	4.155	4.572	0.0924	4.298	4.802	0.5587	4.009	4.740	0.3710
+ AnyUp	2048 × 2048	896 × 896	8	0.845	87.13	30.29	0.3482	87.01	29.69	0.3466	2.517	2.329	0.1018	2.032	2.294	0.5512	2.500	2.641	0.3524
+ NAF	2048 × 2048	1280 × 1280	8	0.663	88.44	34.34	0.3975	87.99	32.54	0.3855	2.419	2.285	0.0904	1.815	2.163	0.4771	2.352	2.579	0.3616
+ CoordUp (Ours)	2048 × 2048	*full*	16	0.466	88.20	34.87	0.3991	87.99	33.85	0.3972	1.845	2.222	0.1489	0.879	1.749	0.4818	1.754	2.409	0.3068

a property of the upsampler rather than of the VFM, as desired for an encoder-agnostic module. RADIOv2.5-B nevertheless stands out as the strongest VFM overall for semantic segmentation by linear probing on MaSS13K, yielding a significantly higher BIoU of 34.87 and 33.85 on Val and Test, respectively. The gains in boundary accuracy provided by CoordUp do not come at the expense of the mIoU. On the validation split, NAF is marginally ahead for four VFMs, by 0.04 to 0.24 points; on the larger test split the ordering reverses, with CoordUp achieving the best or tied performance for four of the five VFMs. Thus, the two strongest upsamplers perform comparably in terms of mIoU, whereas CoordUp consistently and clearly outperforms NAF on the boundary metrics.

Depth estimation results. On UnrealStereo4K, CoordUp obtains the best $\epsilon^{\text{acc}}_{\text{PDBE}}$ on Test for all five VFMs, and for four of them on Val. It also yields the best $\epsilon^{\text{compl}}_{\text{PDBE}}$ on both splits for all five VFMs. On Synth4K, which is an order of magnitude larger, CoordUp achieves the lowest values for both boundary errors with every VFM. Thus, the consistently lower $\epsilon^{\text{acc}}_{\text{PDBE}}$ and $\epsilon^{\text{compl}}_{\text{PDBE}}$ indicate that the depth boundaries predicted by CoordUp are more accurately localized and recovered. Regarding the reconstruction of the global depth structure, CoordUp also achieves the best A.Rel for all five VFMs on Synth4K, our largest evaluation dataset. On UnrealStereo4K, however, the A.Rel rankings are mixed across both the Val and Test splits, with no upsampler emerging as consistently superior. We conjecture that this greater variability may be attributable to the substantially smaller number of evaluation samples in UnrealStereo4K, which makes the average performance more susceptible to being skewed by a few frames. In summary, as observed for semantic segmentation, CoordUp delivers the strongest performance on boundary-sensitive metrics for depth estimation, whereas no upsampler exhibits a clear overall advantage in terms of the global A.Rel metric.

4.3 Zero-shot Inference with TFMs

In this section, we investigate whether CoordUp and other upsamplers can be used to upsample the internal features of a frozen Task-specific Foundation Model (TFM). We evaluate two representative models (Tab. 2): Depth Anything 3 (DA3) (Lin et al., 2026) for depth estimation, and Segment Anything Model 3 (SAM3) (Carion et al., 2026) for semantic segmentation. *Both the TFM and the upsampler are kept frozen.* In both models, we insert the upsampler at the same functional position, immediately before the final pointwise projection of the task head: we feed a downscaled image to the TFM, upsample the activations at this point with CoordUp, NAF, AnyUp, or bilinear interpolation, and apply the final projection to obtain the prediction.

DA3. The left part of Tab. 2 reports results with DA3 on UnrealStereo4K and Synth4K. We feed images to the encoder at the standard DA3 resolution of 504×504, upsample the resulting 504×504

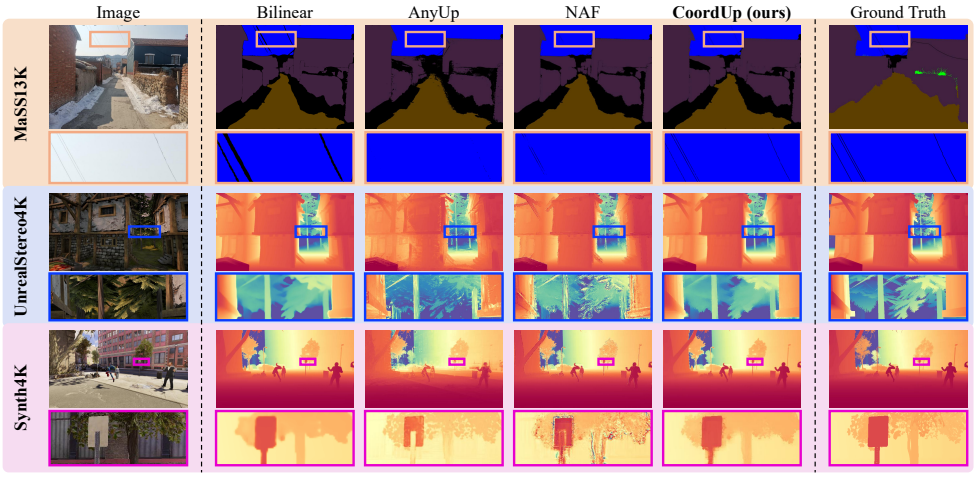


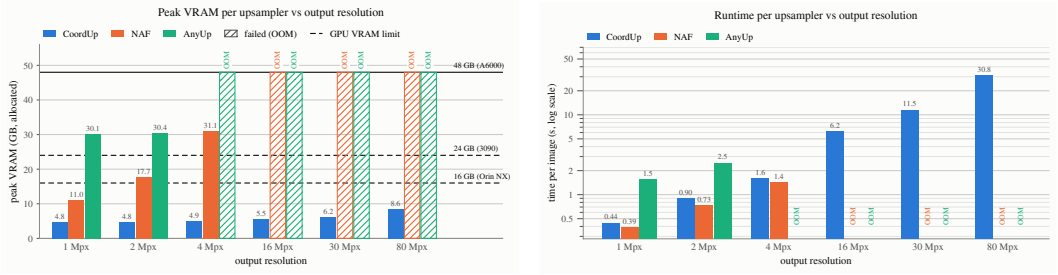
Figure 3: **Qualitative results for zero-shot inference with TFMs.** Upsamplers applied to SAM3 on MaSS13K and DA3 on UnrealStereo4K, Synth4K. Zoom-in (600%) for better analysis.

Table 2: **Results for zero-shot inference with TFMs.** *Left:* depth estimation with DA3. *Right:* semantic segmentation with SAM3. "↑" ("↓") indicates that higher (lower) values are better. The top three results are highlighted as first, second, and third.

Method	Depth estimation — frozen DA3							Semantic segmentation — frozen SAM3							
	Upsampler Res.	UnrealStereo4K-Test (200)			Synth4K-Test (3950)			Upsampler Res.	MaSS13K-Val (500)			MaSS13K-Test (1500)			
		$\epsilon^{\text{acc}}_{\text{PDBE}}\downarrow$	$\epsilon^{\text{compl}}_{\text{PDBE}}\downarrow$	A.Rel ↓	$\epsilon^{\text{acc}}_{\text{PDBE}}\downarrow$	$\epsilon^{\text{compl}}_{\text{PDBE}}\downarrow$	A.Rel ↓		mIoU ↑	BIoU ↑	BF1 ↑	mIoU ↑	BIoU ↑	BF1 ↑
Bilinear	*full*	2.053	3.016	0.311	2.111	3.103	0.205	*full*	80.14	13.11	0.1927	78.84	13.15	0.1955
AnyUp	*full*	1.677	1.511	0.350	2.039	1.879	0.252	1536 × 1536	79.17	18.78	0.2582	78.22	18.50	0.2622
NAF	*full*	2.442	3.577	0.356	2.803	3.632	0.249	2600 × 2600	81.11	19.89	0.2925	79.73	19.67	0.2945
CoordUp (Ours)	*full*	1.042	1.704	0.309	1.400	2.023	0.206	*full*	81.03	21.00	0.3288	79.62	20.66	0.3289

final activations to the full dataset resolution of 3840×2160, and apply the final projection to obtain the depth map. Thanks to the small number of feature channels (32), every upsampler can operate at full resolution. CoordUp substantially reduces the boundary error of the default bilinear pipeline, from 2.053 to 1.042 on UnrealStereo4K and from 2.111 to 1.400 on Synth4K, while preserving region-level accuracy: its A.Rel is the best of all methods on UnrealStereo4K (0.309 vs. 0.311 for bilinear) and on par with bilinear on Synth4K (0.206 vs. 0.205). The other upsamplers behave quite differently. NAF degrades both boundary and region-level metrics relative to plain interpolation, with $\epsilon^{\text{acc}}_{\text{PDBE}}$ of 2.442 and 2.803 and A.Rel of 0.356 and 0.249 on the two datasets. AnyUp achieves the best completeness scores (1.511 and 1.879, vs. 1.704 and 2.023 for CoordUp), but at the cost of noticeably worse boundary accuracy and A.Rel. Both competing methods also introduce visible high-frequency artifacts, most pronounced for NAF (Fig. 3). CoordUp is thus the only method that improves boundary localization without sacrificing depth accuracy.

SAM3. The right part of Tab. 2 reports results with SAM3 on MaSS13K. We feed images to the encoder at the standard SAM3 resolution of 1008×1008, upsample the resulting 288×288 activations, and apply the final segmentation heads to obtain the segmentation masks. We prompt SAM3 with a set of tuned textual prompts associated with the semantic classes (see App. A.5), obtaining one mask per class, and combine these masks into the final segmentation map; the *others* class is excluded, as it has no unambiguous textual prompt. Unlike with DA3, not every upsampler can reach the full output resolution: due to memory constraints, NAF is limited to a 2600×2600 feature grid and AnyUp to 1536×1536, which we then bilinearly upsample to full resolution. CoordUp, in contrast, directly queries coordinates at the target resolution. On the test set, bilinear upsampling achieves 78.84 mIoU but only 13.15 BIoU. CoordUp raises BIoU to 20.66 and BF1 from 0.1955 to 0.3289, while also slightly improving mIoU. It outperforms NAF by 0.99 BIoU points and roughly 12% in relative BF1, while matching it in mIoU (79.62 vs. 79.73). These gains are visible on thin structures such as power lines (Fig. 3): bilinear upsampling over-thickens them and AnyUp almost entirely loses them, whereas CoordUp recovers thin lines close to the ground truth. AnyUp is the only method that lowers mIoU below bilinear (78.22 vs. 78.84), mirroring its degraded A.Rel on DA3. Results on the validation set follow the same trends for all methods. Across both tasks, CoordUp is the only upsampler that consistently achieves the best boundary accuracy, while keeping region-level accuracy on par with the best alternative.



(a) **Peak VRAM vs output res.** (b) **Runtime vs output res.**

Figure 4: **Upsampling cost of the learned upsamplers at increasing output resolution.** The VRAM (a) and the runtime (b) are profiled against output resolution. The final activations of SAM3 are upsampled with AnyUp (green) , NAF (orange) and CoordUp (blue) up to 80 Mpx. A hatched bar (a) or absent one (b) denotes a case where a method exhausts the GPU memory.

4.4 Computational Analysis

We compare the computational cost of the learned upsamplers using the SAM3 setup of Sec. 4.3, upsampling features to output sizes from 1 to 80 Mpx without any resolution cap. Fig. 4 reports peak GPU memory and runtime for a single forward pass on an NVIDIA A6000 (48 GB), where runtime and peak memory include the SAM3 forward pass.

Peak allocated memory. Both NAF and AnyUp materialize intermediate tensors at the target resolution, so their memory footprint is high and grows quickly with the number of output pixels (Fig. 4a). AnyUp already requires 30.1 GB at 1 Mpx, more than the 24 GB available on high-end consumer GPUs (e.g., RTX 3090), while NAF grows from 11.0 GB at 1 Mpx to 31.1 GB at 4 Mpx; beyond these sizes, both run out of memory. This explains the intermediate resolutions of the competing methods in Tab. 2, which are dictated by memory limits. In contrast, CoordUp attends to a fixed neighborhood of N tokens per query, so its memory depends mainly on the query batch size rather than on the full output grid. Its peak memory grows from 4.8 GB at 1 Mpx to only 8.6 GB at 80 Mpx ($1.8\times$ for $80\times$ more pixels), remaining within the 16 GB typical of embedded platforms (e.g., Jetson Orin NX), a budget NAF already exceeds at 2 Mpx.

Runtime. As shown in Fig. 4b, NAF is the fastest method at low resolutions, outperforming CoordUp by 11–19% between 1 and 4 Mpx; since both run in under 2 s in this range, this amounts to at most 0.2 s per image. AnyUp, instead, is $3.4\times$ slower than CoordUp at 1 Mpx (1.5 vs. 0.44 s). However, neither competing method scales further: AnyUp runs out of memory from 4 Mpx and NAF from 16 Mpx. In contrast, CoordUp runtime grows linearly with output resolution, apart from the fixed cost of the SAM3 forward pass, as expected from its per-coordinate formulation. It increases from 0.44 s at 1 Mpx to 30.8 s at 80 Mpx, with each additional megapixel adding about 0.39 s, thus trading a predictable runtime increase for output resolutions far beyond the reach of the competing methods.

5 Conclusion

We introduced CoordUp, a coordinate-conditioned, feature-agnostic upsampler that predicts features at continuous spatial locations through cross-attention over a fixed neighborhood of low-resolution tokens. Because the full-resolution feature map is never materialized, memory stays explicitly controlled, and the continuous formulation supports full-grid upsampling and selective spatial queries. This lets CoordUp operate at full output resolution, beyond the reach of existing feature-agnostic upsamplers. Across frozen VFMs and high-resolution segmentation and depth benchmarks, CoordUp consistently achieves the best boundary accuracy while preserving global prediction quality, making boundary quality largely independent of the backbone. It also transfers zero-shot to TFMs, sharpening the boundaries predicted by DA3 and SAM3 without retraining the host models. These results position high-resolution imagery as the natural testbed for feature upsampling, and feature-agnostic upsamplers as a plug-and-play component for foundation models.

Limitations. Although memory usage remains nearly constant with output resolution, runtime scales linearly at $\sim$0.39 s/Mpx, reaching 30.8 s at 80 Mpx. A practical use at this scale therefore requires sparse querying, which CoordUp naturally supports through its coordinate formulation.

REFERENCES

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Dídac Surís, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, Jie Lei, Tengyu Ma, Baishan Guo, Arpit Kalla, Markus Marks, Joseph Greer, Meng Wang, Peize Sun, Roman Rädle, Triantafyllos Afouras, Effrosyni Mavroudi, Katherine Xu, Tsung-Han Wu, Yu Zhou, Liliane Momeni, Rishi Hazra, Shuangrui Ding, Sagar Vaze, Francois Porcher, Feng Li, Siyuan Li, Aishwarya Kamath, Ho Kei Cheng, Piotr Dollár, Nikhila Ravi, Kate Saenko, Pengchuan Zhang, and Christoph Feichtenhofer. SAM 3: Segment anything with concepts. In *International Conference on Learning Representations (ICLR)*, 2026.

Loïck Chambon, Paul Couairon, Eloi Zablocki, Alexandre Boulch, Nicolas Thome, and Matthieu Cord. NAF: Zero-shot feature upsampling via neighborhood attention filtering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8628–8638, 2021.

Bowen Cheng, Ishan Misra, Alexander G. Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1290–1299, 2022.

Paul Couairon, Loïck Chambon, Louis Serrano, Jean-Emmanuel Haugeard, Matthieu Cord, and Nicolas Thome. JAFAR: Jack up any feature at any resolution. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Iacopo Curti, Pierluigi Zama Ramirez, Alioscia Petrelli, and Luigi Di Stefano. CoordFormer: Give me any coordinates and I will give you labels, 2026. URL https://arxiv.org/abs/2609.08660.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *The Twelfth International Conference on Learning Representations*, 2024.

Mohamed El Banani, Amit Raj, Kevis-Kokitsi Maninis, Abhishek Kar, Yuanzhen Li, Michael Rubinstein, Deqing Sun, Leonidas J. Guibas, Justin Johnson, and Varun Jampani. Probing the 3D awareness of visual foundation models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Stephanie Fu, Mark Hamilton, Laura Brandt, Axel Feldmann, Zhoutong Zhang, and William T. Freeman. FeatUp: A model-agnostic framework for features at any resolution. In *International Conference on Learning Representations (ICLR)*, 2024.

Ross Girshick. Fast R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 1440–1448, 2015.

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 (CVPR)*, pp. 770–778, 2016.

Greg Heinrich, Mike Ranzinger, Hongxu Yin, Yao Lu, Jan Kautz, Andrew Tao, Bryan Catanzaro, and Pavlo Molchanov. RADIOv2.5: Improved baselines for agglomerative vision foundation models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (GELUs). *arXiv preprint arXiv:1606.08415*, 2016. doi: 10.48550/arXiv.1606.08415.

Haiwen Huang, Anpei Chen, Volodymyr Havrylov, Andreas Geiger, and Dan Zhang. LoftUp: Learning a coordinate-based feature upsampler for vision foundation models. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9913–9923, 2025.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4015–4026, 2023.

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

Haotong Lin, Sili Chen, Jun Hao Liew, Donny Y. Chen, Zhenyu Li, Yang Zhao, Sida Peng, Hengkai Guo, Xiaowei Zhou, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth Anything 3: Recovering the visual space from any views. In *International Conference on Learning Representations (ICLR)*, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision (ECCV)*, pp. 405–421, 2020.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024.

Duc-Hai Pham, Tung Do, Phong Nguyen, Binh-Son Hua, Khoi Nguyen, and Rang Nguyen. Sharpdepth: Sharpening metric depth predictions using diffusion distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17060–17069, June 2025.

Minseok Seo, Mark Hamilton, and Changick Kim. Upsample anything: A simple and hard to beat baseline for feature upsampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *European Conference on Computer Vision (ECCV)*, 2012.

Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025.

Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems*, 33:7462–7473, 2020.

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 (ECCV)*, pp. 110–128, 2024.

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in neural information processing systems*, 33:7537–7547, 2020.

Fabio Tosi, Yiyi Liao, Carolin Schmitt, and Andreas Geiger. SMD-Nets: Stereo mixture density networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Shashanka Venkataramanan, Valentinos Pariza, Mohammadreza Salehi, Lukas Knobel, Elias Ramzi, Spyros Gidaris, Andrei Bursuc, and Yuki M. Asano. Franca: Nested matryoshka clustering for scalable visual representation learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Matthew Walmer, Saksham Suri, Anirud Aggarwal, and Abhinav Shrivastava. UPLiFT: Efficient pixel-dense feature upsampling with local attenders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Thomas Wimmer, Prune Truong, Marie-Julie Rakotosaona, Michael Oechsle, Federico Tombari, Bernt Schiele, and Jan Eric Lenssen. AnyUp: Universal feature upsampling. In *International Conference on Learning Representations (ICLR)*, 2026.

Chenxi Xie, Minghan Li, Hui Zeng, Jun Luo, and Lei Zhang. MaSS13k: A matting-level semantic segmentation benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Hao Yu, Haotong Lin, Jiawei Wang, Jiaxin Li, Yida Wang, Xueyang Zhang, Yue Wang, Xiaowei Zhou, Ruizhen Hu, and Sida Peng. InfiniDepth: Arbitrary-resolution and fine-grained depth estimation with neural implicit fields. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

AI USE STATEMENT

We used generative AI tools (Claude, Anthropic, ChatGPT, OpenAI) for feedback and review of our written manuscript. We have not used generative AI tools to generate ideas, synthetic datasets, formulate or prove mathematical claims, implement methods, clean or reformat datasets, or interpret experimental results. So we have used generative AI tools only to edit parts of the manuscript. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims and artifacts produced with the aid of generative AI.

A APPENDIX

This appendix provides additional methodological details, analyses, and experiments supporting the main paper. In particular, it covers:

- In App. A.1, we provide the full design of the ResNetBlock used in the *Query and Key Encoders*, an interpretation of the cross-attention operator as a content-adaptive filtering mechanism, and a graphical overview of CoordUp at inference time.
- In App. A.2, we describe the metrics used throughout the paper, including the modified completeness term of the Pseudo Depth Boundary Error.
- In App. A.3, we investigate the effect of the number of value tokens, the training data source, the *Query* components and the resolution of I_{key}.
- In App. A.4, we complement the zero-shot depth evaluation with additional metrics from the original benchmarks and assess CoordUp on NYUv2 (Silberman et al., 2012), a standard benchmark used in previous feature upsampling methods.
- In App. A.5, we describe the implementation details for linear probing, and zero-shot evaluation experiments, as well as the layers at which feature upsamplers are inserted into the TFMs.
- In App. A.6 we provide additional visual results for both generic feature upsampling and the integration of CoordUp into task-specific foundation models. The attached ZIP archive contains the qualitative results at their original resolution, allowing the fine-grained differences between the compared methods to be inspected more clearly.

A.1 ARCHITECTURAL DETAILS

***Query and Key Encoder* breakdown.** In the *Query Encoder*, the Fourier encoding of the queried spatial coordinate (x, y) is defined as $f_{PE} = [(x, y), \{\sin(2^\ell \pi(x, y)), \cos(2^\ell \pi(x, y))\}_{\ell=0}^{9}] \in \mathbb{R}^{42}$, which contains the raw 2D coordinate together with sine and cosine features at 10 logarithmically spaced frequencies from 2^0 to 2^9. Both the *Query Encoder* and the *Key Encoder* map their inputs to 256-dimensional tokens through a ResNetBlock (He et al., 2016). Given an input token $\mathbf{u}$, the block first projects it to a 128-dimensional hidden representation using a linear layer followed by a GELU activation. The resulting features are then processed by two pre-activation residual units, each defined as

$$\mathbf{u}' = \mathbf{u} + \text{Linear}\left(\text{GELU}\left(\text{GN}\left(\text{Linear}\left(\text{GELU}\left(\text{GN}(\mathbf{u})\right)\right)\right)\right)\right), \tag{4}$$

where all linear transformations inside the residual units have dimensions 128×128 and use no bias, while GroupNorm (GN) uses 8 groups. A final linear layer maps the hidden representation from 128 to 256 dimensions. In parallel, a learned linear projection maps the original block input directly to 256 dimensions and is added as a shortcut:

$$\mathbf{f} = \text{Linear}(\mathbf{u}') + \text{Linear}_{\text{skip}}(\mathbf{u}), \qquad \mathbf{f} \in \mathbb{R}^{256}. \tag{5}$$

All operations are applied independently to each token, equivalently to 1×1 convolutions, and therefore the block does not exchange information across spatial locations. The trainable parameters breakdown of CoordUp is summarized in Tab. 3.

Interpretation of the attention. The attention in Sec. 3.1 admits a second reading, as content-adaptive filtering. For each head, the attention weights form an $n \times n$ kernel, $n = \sqrt{N} = 5$, predicted at each queried location and applied to the corresponding channel block of the frozen feature map. Per queried location, CoordUp therefore produces only $n_h \times N$ scalars, independently of the feature dimensionality C; all remaining information in the output is carried by the sampled values themselves. Fixed interpolation schemes such as bilinear and bicubic belong to the same family of normalized linear filters, but their kernels depend only on the sub-pixel offset, whereas ours additionally depends on the appearance at the query location and at each sampled neighbour.

Inference overview. Fig. 5 provides a visual overview of CoordUp at inference time. Given the coarse feature map F_{in} produced by the frozen model $\mathcal{M}$, CoordUp predicts the feature $\hat{f}(x, y)$ at an arbitrary query location using the corresponding RGB value $I(x, y)$ and the key image I_{key}. The right part of the figure details the internal computation, where the *Query and Key Encoders* produce f_Q

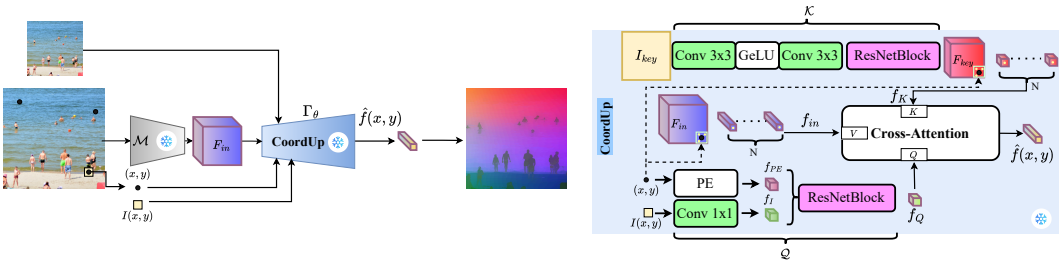


Figure 5: **CoordUp inference (left) and architecture (right).** Left: given an input image I, the frozen model $\mathcal{M}$ produces the coarse feature map F_{in}. For each queried spatial coordinate (x, y), CoordUp predicts the corresponding feature $\hat{f}(x, y)$ using the high-resolution RGB value $I(x, y)$ and the key image I_{key}. Right: detailed architecture of CoordUp.

Table 3: **Trainable parameters breakdown of CoordUp.**

Component	Output channels	Params
Query Encoder $\mathcal{Q}$	256	165,760
Conv 1×1	128	512
ResNetBlock	256	165,248
Key Encoder $\mathcal{K}$	256	300,288
Conv 3×3 + GELU + Conv 3×3	128	151,168
ResNetBlock	256	149,120
Attention operator	C	0
Total		466,048

and F_{key}, respectively, while local neighbourhoods sampled from F_{key} and F_{in} provide f_K and f_{in} for the cross-attention operator. Importantly, at inference time the training-specific tiling procedure is no longer required: each coordinate is queried directly, and the corresponding *Query* f_Q is used without the averaging step employed during training.

A.2 Metrics

For semantic segmentation, we report mean intersection-over-union (mIoU) together with two boundary-sensitive metrics, boundary IoU (BIoU) and boundary F1 (BF1), computed within a narrow band around the ground truth contours. These boundary metrics are particularly informative for evaluating feature upsampling: mIoU is dominated by large homogeneous regions and, in our experiments, changes only marginally across upsampling methods (Tab. 1), whereas BIoU and BF1 directly capture the recovery of thin structures and object boundaries that are typically lost in coarse feature grids. For monocular depth estimation, we report the *Pseudo Depth Boundary Error* (PDBE) from Pham et al. (2025). Since our benchmarks do not provide manually annotated occlusion boundaries, edge sets are extracted from both the predicted and ground-truth depth maps using a Canny detector. The distance between the two edge sets is then measured using truncated nearest-neighbour distances, yielding an accuracy term, $\epsilon^{\text{acc}}_{\text{PDBE}}$, and a completeness term, $\epsilon^{\text{compl}}_{\text{PDBE}}$. The accuracy term averages, over predicted depth edges, the distance to the nearest ground truth edge and therefore penalizes spurious or spatially displaced boundaries. For the completeness term, we modify the original formulation to make its distance criterion symmetric with that of the accuracy term. In the original implementation, accuracy considers only correspondences within a threshold of $\theta = 10$ px, whereas completeness accumulates distances over all ground-truth edges and is therefore unbounded. This asymmetry places the two terms on different scales and can cause the completeness error to be dominated by ground truth boundaries for which no nearby predicted edge exists. We instead compute the reverse distance only for ground truth edge pixels that have a predicted edge within $\theta = 10$ px. Consequently, both $\epsilon^{\text{acc}}_{\text{PDBE}}$ and $\epsilon^{\text{compl}}_{\text{PDBE}}$ are bounded by 10 px and use the same matching criterion, while measuring the correspondence in opposite directions. This modification focuses the completeness term specifically on the localization quality of matched boundaries. This is useful in our evaluation because global depth fidelity is measured separately by the absolute relative error (A.Rel), allowing PDBE to more directly characterize boundary localization. The modification, however, does not penalize ground truth boundaries for which no predicted edge lies within the matching radius. Therefore, the PDBE terms characterize the localization of recovered boundaries

Table 4: **Ablation on neighbouring tokens number to gather for f_K and f_{in} in CoordUp .** N denotes the number of tokens in f_K and f_{in} used by CoordUp. "↑" indicates that higher values are better. The top-3 results are highlighted as first , second , and third .

Method	N	MaSS13K-Val (500)		
		mIoU↑	BIoU↑	BF1↑
$\mathcal{M}$: DINOv3-B				
+ CoordUp	9	86.95	30.86	0.3819
+ CoordUp	49	86.91	32.78	0.3765
+ CoordUp	81	86.77	32.42	0.3674
+ CoordUp	25	87.02	32.77	0.3836

Table 5: **Ablation on the training source of CoordUp.** *Training source* represents the input features CoordUp is trained on: DINOv3-B Feat. denotes DINOv3-B features (canonical CoordUp), RGB Fourier denotes Fourier features with 128 channels of the RGB image, and RGB denotes the raw image, Franca-B Feat. denotes Franca-B features, RADIOv2.5-B Feat. denotes RADIOv2.5-B features. "↑" indicates that higher values are better. The top-3 results are highlighted as first , second , and third .

Method	Training source	MaSS13K-Val (500)		
		mIoU↑	BIoU↑	BF1↑
$\mathcal{M}$: DINOv3-B				
+ CoordUp	RGB	84.55	23.60	0.3083
+ CoordUp	RGB Fourier	84.81	26.04	0.3356
+ CoordUp	Franca-B Feat.	86.98	32.44	0.3798
+ CoordUp	RADIOv2.5-B Feat.	87.05	32.66	0.3843
+ CoordUp	DINOv3-B Feat.	87.02	32.77	0.3836

rather than boundary coverage itself. We complement these boundary-localization metrics with A.Rel as a region-level measure of global depth accuracy.

A.3 Ablation experiments

We ablate the main design choices of CoordUp on linear probing on MaSS13K (Xie et al., 2025). All the ablation experiments share the setting of the main paper: a frozen DINOv3-B encoder processes 2048×2048 inputs, CoordUp upsamples its features to native image resolution, and the downstream head is trained with batch size 16. We report mIoU, boundary IoU (BIoU) and boundary F1 (BF1) on the 500-image validation split. In every table, the configuration adopted in the main paper is shown in bold in the last row.

Number of neighbouring tokens in f_K and f_{in}. Tab. 4 varies the number N of neighbours attended by each query, i.e., the size of the $\sqrt{N} \times \sqrt{N}$ window of low-resolution tokens that provides f_K and f_{in}. A 3×3 window (N=9) already attains competitive region-level accuracy (86.95 mIoU), but trails the larger windows by about 1.6–1.9 BIoU points. We attribute this gap to the fact that low-resolution tokens adjacent to a boundary are often mixtures of both regions: a wider window increases the likelihood that a query retrieves an uncontaminated token of the correct region. Enlarging the window beyond 5×5, however, brings no further benefit. With $N = 49$ BIoU is on par with $N = 25$ (32.78 vs. 32.77), whereas mIoU and BF1 decrease, and with N=81 all three metrics fall below the default configuration. Since each output is a convex combination over the window, larger neighbourhoods admit more distant and semantically unrelated candidates, which dilute the attention distribution while increasing the computational cost of the operator linearly in N. We therefore adopt $N = 25$.

Training source. In Tab. 5 the experiment obtained by varying the supervision signal used to train CoordUp is reported. The evaluation is always performed on DINOv3-B features on the MaSS13K dataset. Thus, all configurations except the DINOv3-B-trained one evaluate transfer from the training source to an unseen feature space. When CoordUp is trained on features extracted by a different vision foundation model, performance remains very close to that obtained when training directly on DINOv3-B features: the differences are within 0.04 mIoU, 0.33 BIoU, and 0.004 BF1. The variant trained on RADIOv2.5-B features even marginally exceeds the DINOv3-B-trained model in mIoU and BF1. These results show that CoordUp can be trained on one encoder and transferred to another without retraining, directly supporting its encoder-agnostic formulation.
Training on image-space targets, by contrast, leads to a marked degradation. Using either raw

Table 6: **Ablation on the *Query* components of CoordUp.** f_{PE} indicates whether CoordUp receives the pixel coordinates explicitly, and f_I indicates whether it receives the RGB pixel values. "↑" indicates that higher values are better. The top-3 results are highlighted as first, second, and third.

Method	Input		MaSS13K-Val (500)		
	f_{PE}	f_I	mIoU↑	BIoU↑	BF1↑
$\mathcal{M}$: DINOv3-B					
+ CoordUp	✓	✗	82.55	9.77	0.1677
+ CoordUp	✗	✓	87.01	32.66	0.3841
+ CoordUp	✓	✓	87.02	32.77	0.3836

Table 7: **Ablation on downsampling factor of I_{key}.** S_{key} denotes the factor by which the I_{key} is downsampled before being fed to CoordUp; the depth of the *Key Encoder* follows, so that F_{key} always matches the resolution of F_{in}. "↑" indicates that higher values are better. The top-3 results are highlighted as first, second, and third.

Method	S_{key}	MaSS13K-Val (500)			Params
		mIoU↑	BIoU↑	BF1↑	(M)
$\mathcal{M}$: DINOv3-B					
+ CoordUp	2	87.07	32.84	0.3854	0.614
+ CoordUp	8	86.92	31.73	0.3791	0.318
+ CoordUp	4	87.02	32.77	0.3836	0.466

RGB values or their 128-channel Fourier lifting reduces mIoU by 2.2–2.5 points relative to the default CoordUp version, while BIoU decreases by 6.7–9.2 points. When the supervision target is photometric, the self-distillation objective encourages affinities that follow image appearance, such as colour and texture similarity. These affinities do not necessarily match those required to upsample semantic feature representations, for example within textured objects or across different objects with similar appearance. The Fourier lifting partially mitigates this mismatch, improving over raw RGB by 0.26 mIoU and 2.44 BIoU, but still remains clearly below all feature-supervised variants. Overall, the results indicate that the crucial factor is supervision in a semantic feature space, rather than the specific encoder from which that feature space is obtained.

***Query* components of CoordUp.** Tab. 6 isolates the contribution of the two components of the *Query*: the positional embedding of the queried spatial coordinates f_{PE} and the RGB appearance features f_I. Removing f_I leads to a substantial performance drop, to 82.55 mIoU and 9.77 BIoU. Without image appearance, queries falling within the same low-resolution cell can be distinguished only by their spatial position; the attention weights therefore cannot adapt to local image content, limiting the ability of the model to localize boundaries that cross the cell. By contrast, removing f_{PE} has only a minor effect, yielding 87.01 mIoU and 32.66 BIoU. This indicates that the RGB signal provides the dominant cue for feature upsampling. Nevertheless, we retain the coordinate encoding in the default formulation. Although its quantitative contribution is small, it provides an explicit notion of spatial location and preserves the continuous, coordinate-conditioned nature of CoordUp. In particular, concatenating f_{PE} with f_I yields small gains in mIoU and BIoU ($+0.01$ and $+0.11$, respectively), with only a negligible change in BF1 (-0.0005). We therefore consider the coordinate information complementary to the appearance signal and beneficial for maintaining the intended formulation of the model, even when its effect on aggregate metrics is marginal.

Downsampling factor of I_{key}. In Tab. 7, we ablate the downsampling factor S_{key} applied to the key image I_{key}, before it is processed by the *Key Encoder*. The depth of the encoder is adjusted accordingly so that F_{key} always matches the spatial resolution of F_{in}. A smaller S_{key} preserves finer image details, but requires additional downsampling stages within the *Key Encoder*. Reducing the factor from 4 to 2 improves all metrics, but only marginally: $+0.05$ mIoU, $+0.07$ BIoU, and $+0.0018$ BF1. These gains come at the cost of approximately 32% more parameters (0.614M vs. 0.466M) and an input to the *Key Encoder* containing four times as many pixels. Conversely, increasing the factor to 8 reduces the parameter count by approximately 32% (0.318M), but decreases BIoU by 1.04 points, indicating that an excessively downsampled key image loses fine-grained spatial information that is useful for accurate boundary localization. We therefore adopt $S_{key} = 4$, which retains nearly all the accuracy of the $\times 2$ configuration while substantially reducing the parameter count and keeping CoordUp the most parameter-efficient learned upsampler among those considered.

Table 8: **Zero-shot depth estimation on Synth4K.** "↑" indicates that higher values are better. The top-3 results are highlighted as first, second, and third.

Type	Method	Synth4K-1			Synth4K-2			Synth4K-3			Synth4K-4			Synth4K-5		
		$\delta_{0.5}\uparrow$	$\delta_1\uparrow$	$\delta_2\uparrow$	$\delta_{0.5}\uparrow$	$\delta_1\uparrow$	$\delta_2\uparrow$	$\delta_{0.5}\uparrow$	$\delta_1\uparrow$	$\delta_2\uparrow$	$\delta_{0.5}\uparrow$	$\delta_1\uparrow$	$\delta_2\uparrow$	$\delta_{0.5}\uparrow$	$\delta_1\uparrow$	$\delta_2\uparrow$
Full Image	DA3 + Bilinear	65.0	80.2	90.3	80.9	89.0	93.6	64.7	81.0	92.2	63.4	80.1	91.2	77.9	91.0	96.8
	DA3 + AnyUp	56.5	74.4	87.2	73.2	86.3	93.0	56.5	75.6	89.9	55.2	74.1	88.4	65.7	84.3	94.7
	DA3 + NAF	62.2	78.1	89.2	77.7	87.1	92.8	62.6	78.9	91.0	61.8	78.9	90.6	76.0	89.6	96.2
	DA3 + CoordUp	64.7	80.1	90.3	81.0	89.2	93.8	64.7	81.0	92.3	63.2	79.9	91.2	77.8	91.0	96.9
High-Freq Details	DA3 + Bilinear	44.8	61.6	77.7	55.6	68.4	79.0	40.8	59.9	77.1	42.6	60.1	78.6	58.4	74.6	87.5
	DA3 + AnyUp	38.8	56.8	75.5	49.8	66.7	79.8	36.7	57.0	77.5	36.6	55.1	75.7	49.6	70.0	87.1
	DA3 + NAF	37.7	54.2	72.1	47.1	62.2	75.9	33.0	50.9	70.5	36.8	53.9	73.6	50.8	67.8	83.3
	DA3 + CoordUp	45.4	62.8	79.2	57.1	70.0	80.4	41.9	61.4	78.8	42.7	60.8	79.6	59.3	76.3	89.1

Table 9: **Zero-shot depth estimation on UnrealStereo4K.** "↓" indicates that lower values are better. The top-3 results are highlighted as first, second, and third.

Method	**UnrealStereo4K-Test Mono**
	$\text{SEE}_3\downarrow$
DA3 + Bilinear	7.66
DA3 + AnyUp	7.75
DA3 + NAF	8.78
DA3 + CoordUp	7.51

A.4 Additional Experiments

In the main paper upsampled depth features are assessed in the zero-shot evaluations with our proposed protocol across all upsamplers; here we instead report each benchmark in the metrics originally defined for it. For Synth4K we follow the benchmark protocol of InfiniDepth (Yu et al., 2026), For UnrealStereo4K the one expressed in Tosi et al. (2021). Moreover, we test CoordUp on NYUv2 (Silberman et al., 2012) in the Probe3D (El Banani et al., 2024) protocol adopted by Couairon et al. (2025) and Chambon et al. (2026) with DINOv3-B backbone.

Zero-shot depth estimation on Synth4K. Tab. 8 reports $\delta_{0.5}$, δ_1, and δ_2 on the five Synth4K scenes, evaluated both over the full image and within high-frequency regions provided by the dataset. The two evaluation regimes lead to markedly different conclusions. Over the full image, the performance of all methods is compressed within a narrow range, and no learned upsampler clearly separates itself from bilinear interpolation. On average, CoordUp remains within 0.02 points of bilinear interpolation in δ_1, outperforming it on one scene, matching it on two, and underperforming it on two, with a maximum difference of 0.2 points. The full-image evaluation does, however, reveal that the competing upsamplers degrade the prediction in this regime: relative to bilinear interpolation, NAF and AnyUp decrease δ_1 by 1.74 and 5.32 points on average, respectively. The high-frequency evaluation better isolates the regions in which feature upsampling is expected to have the greatest impact. In this setting, CoordUp ranks first on all five scenes and across all three thresholds, improving over bilinear interpolation by 0.84, 1.34, and 1.44 points for $\delta_{0.5}$, δ_1, and δ_2, respectively. Moreover, the improvement increases with the threshold, suggesting that the gain does not merely arise from redistributing already accurate predictions, but also from recovering pixels that were previously farther from the ground truth. In contrast, AnyUp and NAF remain below bilinear interpolation for nearly all entries, with NAF ranking last in thirteen of the fifteen comparisons. These findings are consistent with the zero-shot depth estimation results reported in Tab. 2.

Zero-shot depth estimation on UnrealStereo4K. Tab. 9 reports SEE_3 on the monocular left split of UnrealStereo4K, at the native 3840×2160 ground-truth resolution. SEE_3 measures, for every evaluated pixel, the distance to the closest ground-truth depth within a 3×3 neighbourhood, and therefore tolerates a misplacement of up to one pixel while heavily penalizing predictions that bleed across a discontinuity. CoordUp attains the lowest error (7.51), improving over bilinear interpolation by 0.15 (2.0%). AnyUp is marginally worse than bilinear (7.75), and NAF degrades it by 1.12 (14.6%). This ranking is consistent with the high-frequency split of Synth4K and confirms that the advantage of CoordUp is concentrated in the edges of the image, where depth is discontinuous.

Low-resolution depth estimation on NYUv2. Our main benchmarks for feature upsampling focus on high-resolution data, typically at 8 Mpx or above, where fine-grained ground truth is available for both depth estimation and semantic segmentation. This is the regime for which CoordUp is

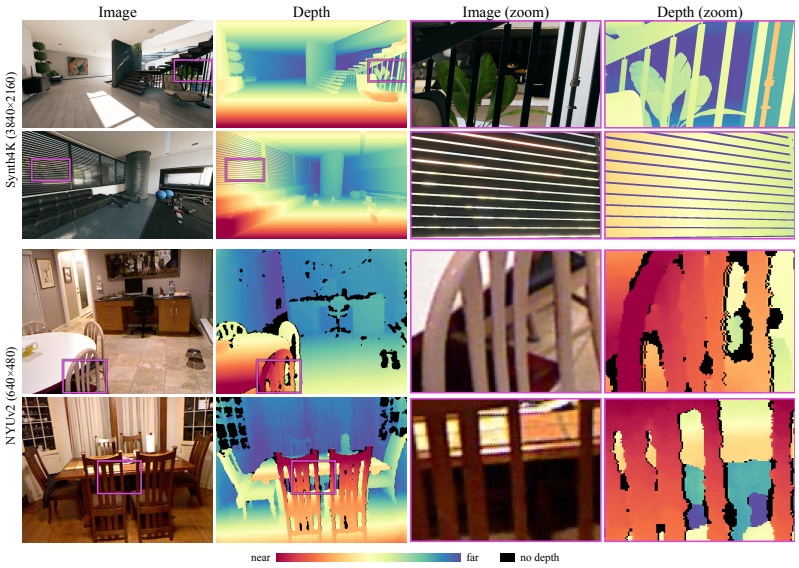


Figure 6: **Ground truth depth Synth4K vs. NYUv2.** Synth4K provides depth maps rendered at 3840×2160, preserving fine structures such as stair balusters, leaves, and individual window-blind slats. NYUv2's raw Kinect depth produces noisy and poorly localized object boundaries: chair spindles appear as jagged, fragmented bands, while chair slats break into blocky regions, some assigned incorrect depth values.

Table 10: **Performance of CoordUp on NYUv2.** *Bilinear, AnyUp and NAF results are taken from Chambon et al. (2026).

Method	NYUv2 (654) $\delta_1 \uparrow$
$\mathcal{M}$: DINOv3-B	
+ Bilinear	86.10*
+ AnyUp	86.36*
+ NAF	86.73*
+ CoordUp	90.11

primarily designed. As qualitatively illustrated in Fig. 6, the lower-resolution NYUv2 dataset provides substantially coarser and noisier depth annotations, with object boundaries often poorly localized. Since our main experiments show that the largest gains from learned feature upsampling occur around object boundaries and fine structures, NYUv2 does not provide sufficiently accurate ground truth to support a reliable boundary-level evaluation. In contrast, Synth4K is synthetically rendered and provides sharp, well-localized depth discontinuities. Nonetheless, we evaluate CoordUp on NYUv2 (Silberman et al., 2012) in Tab. 10, following the linear probing protocol adopted by prior feature-upsampling work (Chambon et al., 2026). Despite operating in a substantially lower-resolution regime, CoordUp achieves 90.11 δ_1, outperforming NAF (86.73), AnyUp (86.36), and bilinear interpolation (86.10). Although the quality of the NYUv2 ground truth prevents a meaningful assessment of boundary reconstruction, these results show that the benefits of CoordUp are not restricted to very-high-resolution imagery or large upsampling factors. Instead, the improvement in the global depth metric suggests that CoordUp can effectively combine neighboring feature tokens to recover more informative representations even in a low-resolution setting.

A.5 Implementation Details

In this section, we describe the implementation details for linear probing with VFMs and zero-shot inference with TFMs. All experiments, including the training of CoordUp, were conducted using PyTorch 2.9.1 with CUDA 12.8.

Linear probing with VFMs. The input image is resized to 2048×2048 or 2044×2044, depending on whether the VFM uses a patch size of 16 or 14, respectively. The resulting VFM features constitute F_{in} for CoordUp and the competing upsamplers. The upsampled features are then required to be

Algorithm 1: Zero-shot inference with CoordUp.

Require: High-resolution image $I \in \mathbb{R}^{3\times H\times W}$, frozen task-specific model $\mathcal{M}$, frozen CoordUp Γ_θ
Ensure: Dense prediction $\hat{Y}$ at resolution $H \times W$
1: $I_{\mathcal{M}} \leftarrow \mathrm{Resize}(I,$ training-compatible resolution of $\mathcal{M})$
2: $F_{\mathrm{in}} \leftarrow \mathrm{ForwardToInsertionPoint}(\mathcal{M}, I_{\mathcal{M}})$
3: $I_{\mathrm{key}} \leftarrow \mathrm{Resize}(I, h \times w)$
4: **for** each batch $\mathcal{B}$ of target coordinates (x, y) **do**
5: $\quad \hat{f}_{\mathcal{B}} \leftarrow \Gamma_\theta\left(F_{\mathrm{in}}, \mathcal{B}, I(\mathcal{B}), I_{\mathrm{key}}\right)$
6: $\quad \hat{Y}_{\mathcal{B}} \leftarrow \mathrm{PointwiseHead}(\mathcal{M}, \hat{f}_{\mathcal{B}})$
7: $\quad$ Write $\hat{Y}_{\mathcal{B}}$ at coordinates $\mathcal{B}$
8: **end for**
9: **return** $\hat{Y}$

Table 11: **SAM3 text prompts per MaSS13K class.** τ represents the confidence threshold used to build the mask, π is the precedence rank.

Label	Class	Text prompts	τ	π
1	person	`human`	0.5	1
2	building	`facade`, `wall`, `architecture`, `building wall`	0.3	5
3	tree	`foliage`, `tree`, `plant`, `vegetation`	0.3	3
4	ground	`paved ground`, `floor`, `ground`, `pavement`	0.3	4
5	sky	`sky`	0.3	6
6	water	`water`	0.2	2

at the native image resolution so that the lightweight prediction head can produce the final dense prediction. Thanks to its coordinate-based formulation, CoordUp can directly produce features at this resolution. In contrast, the competing upsamplers can only generate intermediate feature maps up to a resolution determined by their memory requirements. The prediction heads are trained for up to 200 epochs on MaSS13K (Xie et al., 2025) and 100 epochs on UnrealStereo4K (Tosi et al., 2021), using an early-stopping policy. Training is performed on four A100 GPUs. When using CoordUp, 8192 coordinates are randomly sampled at each training iteration. For MaSS13K, we optimize with AdamW (β_1=0.9, β_2=0.999, ϵ=10^{-8}), using a base learning rate of 1×10^{-4} and a weight decay of 1×10^{-4}. The learning rate is updated at every iteration according to a linear warmup followed by polynomial decay. During the first 10 epochs, it increases linearly from 1×10^{-5} ($0.1\times$ the base learning rate) to 1×10^{-4}, after which it decays with power 0.9 for the remainder of the training schedule. On MaSS13K, we use the same loss for all upsamplers, combining the standard multi-class cross-entropy loss, a class-specific binary dice loss, and a binary cross-entropy loss, with weights of 2, 5, and 5, respectively (Cheng et al., 2022). For UnrealStereo4K, we use the same learning rate, scheduling strategy and loss adopted by Probe3D (El Banani et al., 2024). It is worth noting that, after training the prediction head on UnrealStereo4K, Synth4K is used as an additional test dataset.

Zero-shot inference with TFMs. In the zero-shot experiments, we resize the input image to a resolution compatible with the operating regime of the task-specific model $\mathcal{M}$, i.e., a resolution consistent with that used during its training. The upsamplers are then used to recover features at the native image resolution. We provide the corresponding zero-shot inference procedure for TFMs in Algorithm 1. The insertion point of the upsampler depends on the architecture of $\mathcal{M}$. For DA3, F_{in} corresponds to the 32-channel activation obtained after the final 3×3 convolution and ReLU, while the remaining pointwise head consists of the final 1×1 convolution followed by the output activation. For SAM3, F_{in} corresponds to the 256-channel pixel embedding produced by the pixel decoder, and the remaining segmentation heads are evaluated pointwise on the features returned by the upsampler.

In SAM3 experiments, we perform multiple forward passes with the upsampler inserted at the location described above to generate the semantic segmentation maps and compute the metrics on MaSS13K (Xie et al., 2025). Each forward pass is associated with a textual prompt corresponding to one semantic class. We first define the set of prompts reported in Tab. 11, and then combine the resulting mask predictions to obtain the final segmentation map. The *others* class is excluded from the evaluation because it does not admit a sufficiently unambiguous textual prompt. Predictions are combined hierarchically. For each textual prompt, all instance queries whose confidence score exceeds the class-specific threshold τ contribute their binary masks, which are merged by union. If

multiple prompts are associated with the same semantic class, their masks are further combined by union to obtain a single class-level prediction. Finally, overlaps between different classes are resolved using a fixed precedence order π, rather than confidence scores, since scores obtained from different textual concepts are not directly comparable. Each pixel is therefore assigned to the highest-priority class among those predicting it. Prompts were selected independently for each class using greedy forward selection over its 4–8 candidate wordings (37 in total), repeated for each confidence threshold in $\{0.5, 0.3, 0.2, 0.1\}$. Starting from the best-performing individual wording, we iteratively added the candidate that produced the largest improvement in mean IoU, stopping when no further gain was observed or when four prompts had been selected. The selection procedure was evaluated on 200 images. For each class, only images in which that class occupied at least 1% of the labelled area were considered, resulting in 22–111 evaluation images depending on the class. We then retained the confidence threshold yielding the best performance and used the multi-prompt union only when it clearly outperformed the best single wording. For *person*, *sky*, and *water*, the gain was below 0.01 IoU, so a single prompt was used.

The fact that all zero-shot experiments produce coherent task predictions provides empirical support that the selected insertion points preserve the expected input structure of the remaining task heads.

A.6 Additional qualitative results

We close the appendix with enlarged qualitative comparisons for the two settings of the main paper: feature upsampling with a frozen VFM (Fig. 7) and zero-shot upsampling of the final activations of a frozen TFM (Fig. 8). The latter extends the crops of Fig. 3 with additional scenes and is best viewed zoomed in, since the differences between methods are concentrated in bands a few pixels wide.

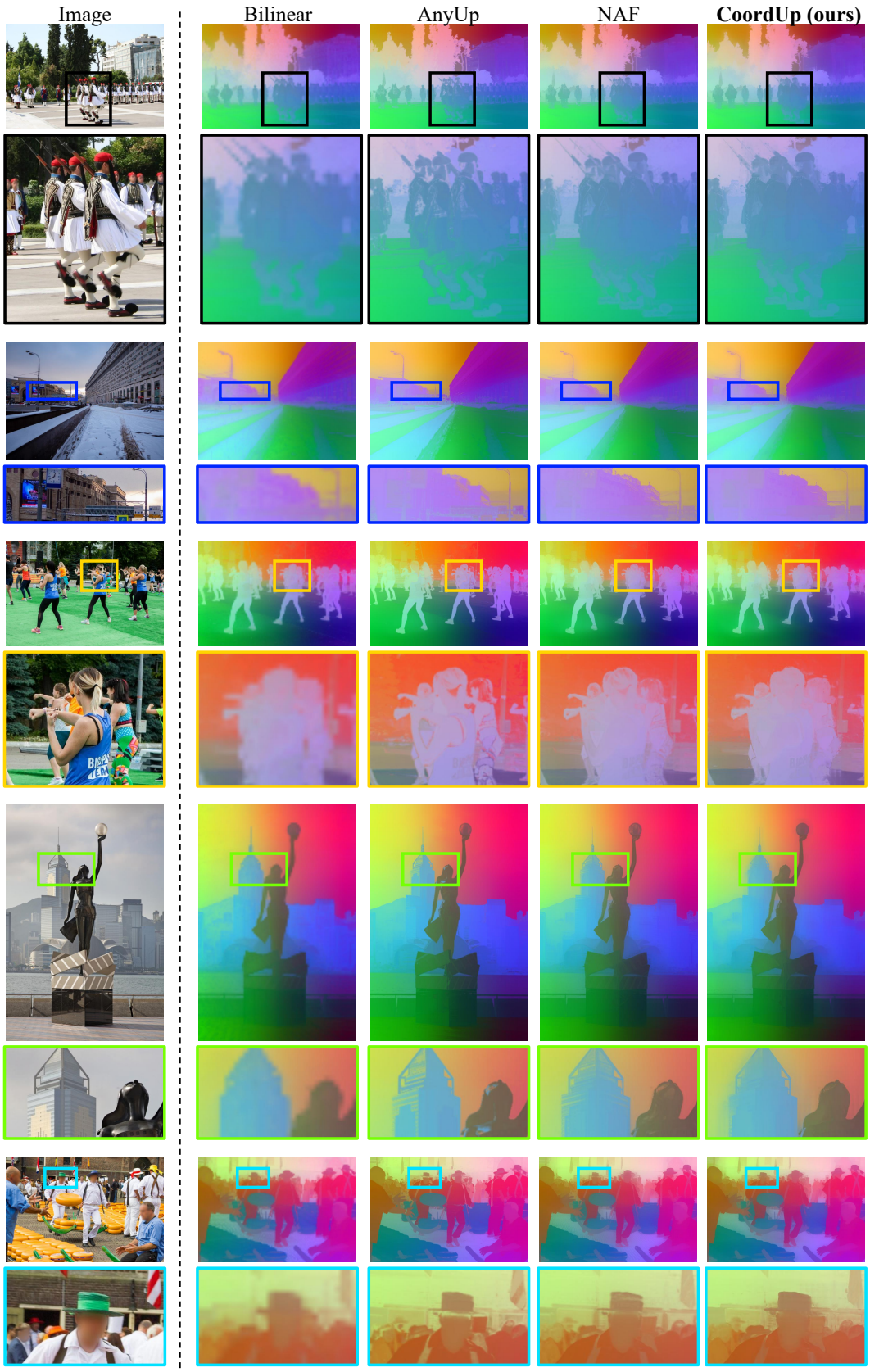


Figure 7: **Feature upsampling qualitative results.** 3-channel PCA visualizations of the upsampled features of a frozen DINOv3-B backbone. Zoom-in (600%) for better analysis.



Figure 8: **Zero-shot qualitative results.** Qualitative comparison of feature upsamplers applied to the final activations of frozen SAM3 and DA3 on MaSS13K, UnrealStereo4K and Synth4K. No host model is retrained. Zoom-in (600%) for better analysis.