ZERO-SHOT PERCEPTION WITH GENERATOR

Anonymous authors
Paper under double-blind review

ABSTRACT

Pretrained on web-scale visual data, generation models acquire rich visual representations and expressive world priors that provide a natural foundation for perception. To unlock these capabilities, we explore perception as generation, leveraging generation paradigm to convert these internal representations directly into visual predictions. Although recent models such as Vision Banana demonstrate strong performance and impressive zero-shot generalization across diverse perception tasks, reliance on proprietary backbones hinders independent investigation and open reproduction. In this work, we present UniVision, an open 8B vision foundation model that unifies zero-shot visual perception and image generation. UniVision generates metric depth, surface normals, and semantic, instance, and referring segmentation as RGB images, which are converted back to target modalities via our noise-tolerant pixel codecs. UniVision achieves advanced zero-shot performance across all five tasks while capturing finer geometric detail than specialized baselines. We show that the core recipe behind these capabilities lies in deterministic, noise-tolerant codecs and scalable data pipelines. Furthermore, based on UniVision, we systematically investigate the perception-as-generation paradigm across both inference and training sides, revealing how inference-time generation techniques and test-time aggregation boost perception accuracy, characterizing task-specific data scaling dynamics, and demonstrating positive data synergy where joint perception training actively enhances generation quality.

1 INTRODUCTION

Generation models acquire rich geometric representations and expressive world priors from web-scale pre-training visual data (Tang et al., 2023; Zhao et al., 2023). These internal representations provide a natural foundation for perception tasks, such as depth and surface normal estimation, and segmentation (Ke et al., 2024; Fu et al., 2024; Baranchuk et al., 2022; Sun et al., 2026). We therefore explore perception as generation (Geng et al., 2024; Zhao et al., 2025; Gabeur et al., 2026), producing perception outputs directly as images. By unifying various perception tasks under a generation objective, we explicitly leverage the learned internal visual representation for perception.

Recent work like Vision Banana (Gabeur et al., 2026) shows that instruction tuning a pretrained image generator supports strong performance across diverse perception tasks. What is even more impressive is the generalization capabilities they show, which means without explicit training on the training datasets of downstream benchmarks. However, its proprietary Nano Banana Pro backbone limits independent investigation and reproduction of that generalization. This further hinders exploration in inference and training behaviors beyond performance comparisons.

Thus, in this work, we present UniVision, a vision foundation model that unifies perception and generation within an 8B image generator. As shown in Figure 1, UniVision follows the perception-as-generation paradigm: given input images and instructions, the model generates depth maps, surface normal maps, or segmentation masks as RGB images. It then uses a codec for converting RGB images back to target modality formats for downstream applications and evaluations. Evaluated with similar protocol as in Vision Banana and SAM 3 (Gabeur et al., 2026; Carion et al., 2025), evaluation across geometry and segmentation benchmarks demonstrates strong zero-shot performance across all five tasks. Moreover, qualitative comparisons show that UniVision captures finer visual details in complex scene geometry than specialized baselines.

The core insight behind the UniVision is that towards a zero-shot vision foundation model with advanced performance, the secret in the recipe is noise-tolerant codecs and scalable, diverse data

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.

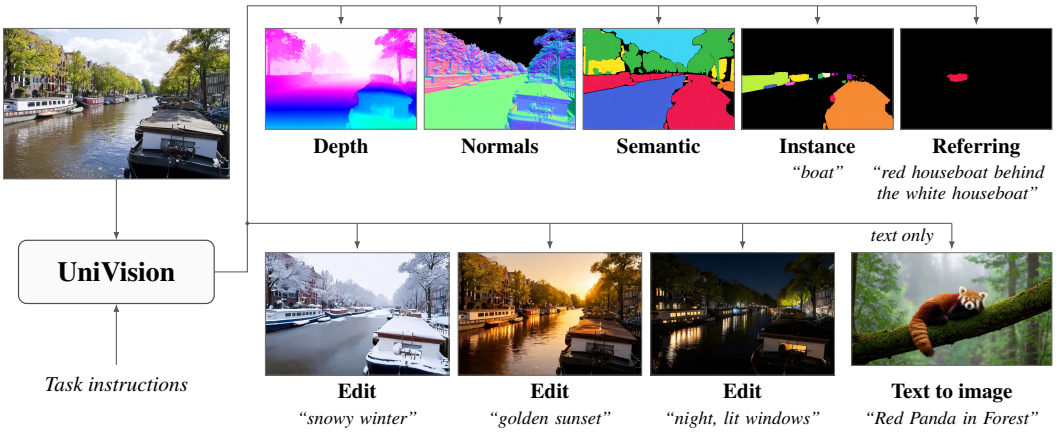


Figure 1: UniVision generates images for perception, editing, and text to image generation.

pipelines. The former helps eliminate inaccurate generated colors and tiny fragments, while the latter helps the model see massive and diverse scenes to generalize to unseen distributions in applications.

Based on UniVision, we systematically investigate important open questions on the perception-as-generation paradigm across both inference and training sides. At test time, we evaluate how inference-time generation techniques and test-time aggregation impact perception performance, finding that generation techniques can also benefit perception. During training, we analyze task-specific data scaling effect and demonstrate positive data synergy, showing that joint perception training actively improves text-to-image synthesis quality. In summary, our main contributions are:

- We present an 8B image generator that unifies zero-shot perception and generation, supported by custom-designed pixel codecs that map generated RGB outputs to precise perception predictions.
- We establish automated data curation pipelines that enable robust zero-shot generalization, and systematically reveal task-specific data scaling dynamics in perception as generation.
- We conduct experiments on perception as generation, evaluating inference-time generation techniques and aggregation, while demonstrating data synergy between perception and generation.

2 Related Work

Generative Perception. Image synthesis models provide visual representations that support dense prediction tasks such as segmentation and visual correspondence (Baranchuk et al., 2022; Zhao et al., 2023; Tang et al., 2023). Building on these representations, generative methods can directly infer scene properties. Marigold adapts image diffusion models for depth estimation, while GeoWizard jointly predicts surface normals and depth (Ke et al., 2024; Fu et al., 2024). Addressing computational efficiency, GenPercept and Lotus examine adaptation of generative backbones for dense prediction (Xu et al., 2025; He et al., 2025b). To support broader capabilities, unified approaches such as Unified-IO and InstructDiffusion use shared output formats and instruction conditioned prediction across tasks (Lu et al., 2023; Geng et al., 2024). In parallel, DICEPTION formulates multiple visual perception problems through RGB image outputs (Zhao et al., 2025), and Vision Banana and SenseNova-Vision integrate perception into multimodal generative frameworks (Gabeur et al., 2026; Han et al., 2026). Our work builds on these directions by framing dense perception within an unchanged image generation interface, relying on deterministic codecs to encode perception targets as RGB images and decode predictions into task representations.

3 Method

3.1 Preliminaries: Diffusion Training

We use rectified flow (Liu et al., 2023; Esser et al., 2024) in the latent space of a pretrained autoencoder (Rombach et al., 2022). Let $z_0 = \mathcal{E}(y)$ be the latent of target image y, and c denote the text prompt and input-image conditioning. During training, we sample noise $\epsilon \sim \mathcal{N}(0, I)$ and noise

level $t \in [0, 1]$ to form a noisy target:

$$z_t = (1 - t)z_0 + t\epsilon. \tag{1}$$

A network $v_\theta(z_t, t, c)$ learns to predict the velocity $\epsilon - z_0$. The basic flow-matching objective is (Lipman et al., 2023)

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{z_0,c,t,\epsilon}\left[\|v_\theta(z_t, t, c) - (\epsilon - z_0)\|_2^2\right], \tag{2}$$

shown here for the target latents. Input-image conditioning latents remain clean and are excluded from the regression loss. At inference, we start from Gaussian noise, integrate the predicted velocity from $t = 1$ to $t = 0$, and decode the resulting latent into image space.

3.2 Perception As Generation

Generation models acquire rich geometric representations and expressive world priors from web-scale pre-training visual data, which is the foundational properties required for perception. Consequently, achieving zero-shot specialist-level perception via generation models does not require learning intermediate features from scratch. Instead, the primary hurdles lie in two core challenges: 1) Output space alignment and decoding robustness: Perception targets do not naturally conform to the standard RGB image space $\mathbb{R}^{H\times W\times 3}$ produced by generative models. Thus, we need a method to convert RGB output to target modalities. 2) Scalable data curation pipeline for zero-shot generalization: Achieving broad, unconstrained zero-shot perception demands massive, diverse training pairs that far exceed the vocabulary and scale of traditional human-annotated benchmarks.

As shown in Fig. 2, to address the first challenge, we formalize perception-as-generation through task-specific pixel codecs combined with conditional rectified flow training. Given an input image x and task instruction c_{text}, a deterministic codec encoder $\mathcal{E}^{(k)}_{\text{codec}}$ first renders the perception target $y^{(k)}$ into an RGB image $\bar{y}^{(k)} = \mathcal{E}^{(k)}_{\text{codec}}(y^{(k)})$. A spatial VAE encoder $\mathcal{E}_{\text{VAE}}$ maps both input x and target $\bar{y}^{(k)}$ into latent space, yielding clean input conditioning latents $c_{\text{img}} = \mathcal{E}_{\text{VAE}}(x)$ and target latents $z_0 = \mathcal{E}_{\text{VAE}}(\bar{y}^{(k)})$. During training, given Gaussian noise $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ and timestep $t \sim U(0, 1)$, a parameterized velocity field $v_\theta(z_t, t, c)$ learns to predict the linear interpolation trajectory $z_t = (1 - t)z_0 + t\epsilon$ under the conditioning tuple $c = (c_{\text{img}}, c_{\text{text}})$ by minimizing the flow-matching loss:

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{y^{(k)},c,t,\epsilon}\left[\|v_\theta(z_t, t, c) - (\epsilon - z_0)\|_2^2\right]. \tag{3}$$

At inference time, integrating the velocity field $\frac{\mathrm{d}z_t}{\mathrm{d}t} = v_\theta(z_t, t, c)$ from $t = 1$ to $t = 0$ yields the target latent z_0, which decodes into a candidate image $\hat{\bar{y}}^{(k)} = \mathcal{D}_{\text{VAE}}(z_0)$. To overcome generative artifacts such as color drifts and boundary blur, a noise-tolerant codec decoder $\mathcal{D}^{(k)}_{\text{codec}}$ projects candidate pixels $\hat{\bar{y}}^{(k)}$ onto the nearest valid manifold in target space $\mathcal{Y}^{(k)}$, recovering the final perception prediction:

$$\hat{y}^{(k)} = \mathcal{D}^{(k)}_{\text{codec}}(\mathcal{D}_{\text{VAE}}(z_0)). \tag{4}$$

By decoupling representation learning from output discretization, this formulation enables effective learning over noisy generated spaces. We discuss the design principle in following sections.

3.3 Codecs

There are two different formats in all five tasks: continuous map for depth and surface normals, and palette encoding for masks. 1) Continuous geometry: A key challenge in depth representation is bit precision. Naive scalar 8 bit encoding provides only 256 discrete levels, merging distinct depth values and discarding supervision detail that guides accurate prediction. To mitigate this information loss, we follow Vision Banana (Gabeur et al., 2026). A Barron transform first allocates precision toward nearby geometry, after which distances are mapped along the edges of the RGB cube. Routing depth along this perimeter provides finer quantization than a single scalar channel. At inference, generated colors are projected onto the nearest path segment before applying the inverse transform. Surface normals use a standard affine mapping from vector components to RGB

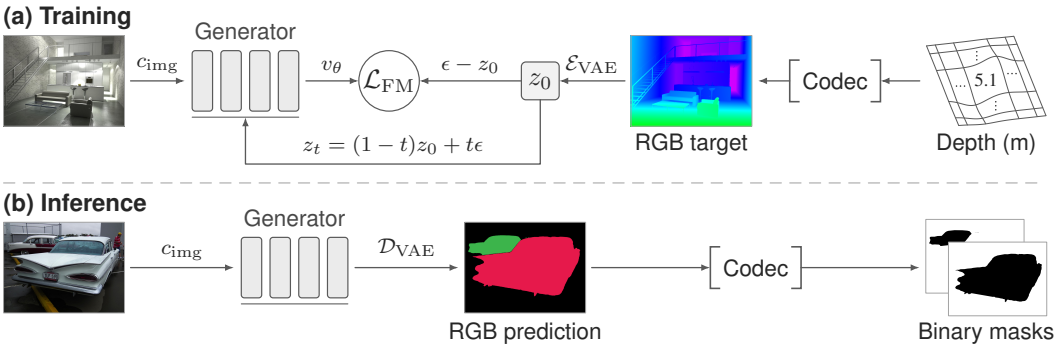


Figure 2: Codecs bridge task and RGB spaces by encoding targets during training and decoding predictions during inference. Displayed outputs are schematic illustrations.

channels, followed by the inverse mapping and unit normalization during decoding. 2) Mask-based format: Mask tasks encode target regions as palette colors, with black reserved for background. The decoding process maps each generated pixel to its nearest permitted palette color, producing a raw label map $\hat{a}$. Because color blending at object boundaries can introduce wrong labels, semantic and instance segmentation use an intermediate spatial step to resolve uncertain pixels using reliable seeds and component support. For pixel coordinate p, let $S(p)$ be a nonempty set of seeds and M be a set of foreground pixels whose raw labels are kept. This step finds the nearest seed to produce an intermediate label map $\tilde{a}$:

$$s(p) = \arg\min_{q \in S(p)} \|p - q\|_2, \quad \tilde{a}_p = \begin{cases} \hat{a}_p, & p \in M, \\ \hat{a}_{s(p)}, & p \notin M. \end{cases} \tag{5}$$

Each task defines $S(p)$ and M for this rule. For semantic segmentation, reliable pixels R_{sem} are close to their assigned palette colors, including black, giving global seeds $S_{\text{sem}}(p) = R_{\text{sem}}$. The set M_{sem} keeps raw nonblack 8 connected components $\mathcal{C}_{\text{raw}}$ of a single label that contain a reliable seed or meet minimum area τ. For instance segmentation, the decoder accepts a palette color only if a sufficiently large 8 connected group of foreground pixels has RGB values close to that color and closer to it than to the second nearest palette color by a required margin. The set M_{ins} is the union of pixels from raw foreground components of accepted colors that pass broader color and support tests. Seeds are color confident pixels R_{ins} within M_{ins} and the foreground 8 connected component $F(p)$ containing pixel p:

$$\begin{aligned} M_{\text{sem}} &= \bigcup \{ C \in \mathcal{C}_{\text{raw}} : C \cap R_{\text{sem}} \neq \emptyset \ \vee \ |C| \geq \tau \}, \\ S_{\text{ins}}(p) &= R_{\text{ins}} \cap M_{\text{ins}} \cap F(p). \end{aligned} \tag{6}$$

Implementation details are described in Appendix B. During development, we traced some apparent model failures to decoding errors, showing that codec design matters for perception as generation.

3.4 Data Curation

Strong performance in a zero shot setting requires seeing diverse supervision at scale. To build this training collection, we combine existing annotations with large sets of unlabeled images processed through a systematic curation pipeline. Expert models generate candidate annotations across one or more stages, followed by filtering steps tailored to task specific quality, coverage, and target agreement. Dedicated codecs then encode all retained targets into standard RGB images. In parallel, the pipeline proposes instructions by selecting templates from task prompt pools and filling them with sample metadata. An offline language model can rewrite instructions to expand wording diversity while requiring metadata values to stay unchanged. Each resulting training example forms an aligned triple consisting of a source image, an encoded target image, and a task instruction. We report task pipeline details in Appendix B and catalog training sources in Appendix C.2. To further increase the diversity of the training data, we collect several public datasets covering indoor, outdoor, and synthetic scenes and add them to our existing training data.

Table 1: Zero shot perception. Relative depth and F_1: %; surface normal errors: degrees; gray rows: benchmark training data; "+ Gemini": presence gating (Appendix C.4); "+ Agent": Appendix C.5.

(a) Metric depth

	NYUv2		KITTI		ETH3D		iBims-1		Avg.
Method	δ_1	AbsRel	δ_1	AbsRel	δ_1	AbsRel	δ_1	AbsRel	δ_1
Depth Pro	0.922	0.101	0.782	0.172	0.397	0.652	0.823	0.170	0.731
UniDepth	**0.980**	0.073	**0.980**	**0.048**	0.185	0.533	0.157	0.410	0.576
UniK3D-L	0.965	0.074	0.812	0.174	0.687	0.236	0.919	0.104	0.846
MoGe-2	0.961	0.073	0.629	0.181	0.908	0.104	0.830	0.136	0.832
Depth Anything 3	0.963	**0.070**	0.953	0.086	0.917	0.104	-	-	-
Vision Banana	0.948	0.081	0.915	0.107	**0.935**	**0.103**	**0.934**	**0.078**	**0.933**
UniVision (Ours)	0.935	0.094	0.926	0.078	0.706	0.664	0.800	0.133	0.842

(b) Relative depth

Method	NYUv2	KITTI	ETH3D
SenseNova-Vision	98.1	95.9	97.4
Depth Anything V2	**97.9**	94.6	86.5
Depth Anything 3	97.4	95.3	98.6
Marigold	96.4	91.6	95.9
GeoWizard	96.3	82.0	95.8
DICEPTION	96.0	94.9	97.5
Lotus-2	97.6	94.5	98.1
FE2E	97.7	96.0	98.7
UniGP	96.6	93.3	96.3
UniVision (Ours)	97.6	**98.0**	**98.9**

(c) Surface normal estimation

Method	NYUv2	ScanNet	VKITTI2
SenseNova-Vision	14.4	12.8	-
DSINE	16.4	16.2	28.9
MoGe-2	**14.7**	**12.8**	-
Marigold	20.9	21.3	-
StableNormal	18.6	17.1	-
DICEPTION	18.3	18.8	-
Lotus-2	16.9	14.2	28.9
FE2E	16.2	13.8	-
UniGP	16.4	14.9	-
Vision Banana	17.8	15.1	29.1
UniVision (Ours)	15.5	13.3	**27.6**

(d) Semantic segmentation

Method	mIoU
SegMAN-L	84.2
SenseNova-Vision	71.2
APE-D	44.2
OpenSeeD	47.8
X-Decoder	52.0
SAM 3	65.2
Vision Banana	**69.9**
UniVision (Ours)	67.7

(e) Referring segmentation

Method	cIoU
LISA-7B	67.9
GLaMM	74.2
HyperSeg	79.4
SenseNova-Vision	80.3
X-SAM	83.8
HybridGL	51.3
LocalizationHeads	67.7
SAM 3 Agent	73.4
Vision Banana	73.8
UniVision (Ours) + Agent	**75.4**

(f) Instance segmentation

Method	cgF$_1$	IL_MCC	pmF$_1$
SAM 3	54.1	0.82	66.1
Grounding DINO-T	3.3	0.15	16.2
LLMDet-L	6.5	0.21	27.3
Gemini 2.5	13.0	0.29	46.1
APE-D	16.4	0.40	36.9
DINO-X	21.3	0.38	55.2
OWLv2	24.6	0.57	42.0
Vision Banana + Gemini	**47.5**	0.84	**56.0**
UniVision (Ours) + Gemini	45.1	**0.85**	52.8

4 Experiments

4.1 Setup

Model. UniVision starts from Cosmos3 Nano (NVIDIA, 2026). For its architecture, it uses a mixture of transformers. In this setup, a frozen 8B vision language backbone conditions an 8B generation expert. During the training process, only the generation expert and its adapters are updated. The training data contains about 24 million vision samples, about 6.2 million text to image generation samples, and about 6 million image editing samples. We refer the reader to Appendix C for complete training, inference, and evaluation details.

Benchmarks. We evaluate model performance in a zero shot setting, so no training sets from these benchmarks are used during training. Metric depth is evaluated on NYUv2 (Silberman et al., 2012) (654 images, Eigen split (Eigen et al., 2014)), KITTI (Geiger et al., 2013) (652, Eigen split with

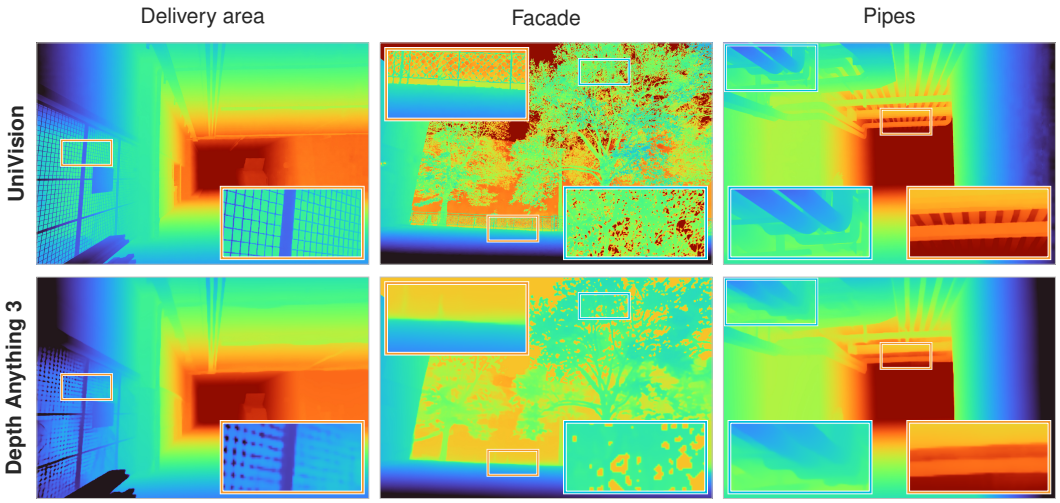


Figure 3: Qualitative depth predictions on ETH3D. Both methods share ground truth scale and shift alignment and the same color scale. Insets enlarge regions marked by matching colors.

the Garg crop), ETH3D (Schöps et al., 2017) (454), and iBims-1 (Koch et al., 2019) (100). We report δ_1 and AbsRel on the metric prediction without any alignment; ground truth is capped at 80 m (10 m for NYUv2) and every image has equal weight. Relative depth scores the same predictions after a per-image least-squares fit of a scale and a shift in depth space (Ke et al., 2024). Surface normals use the evaluation bundle of DSINE (Bae & Davison, 2024): NYUv2 (654), ScanNet (Dai et al., 2017) (300), and Virtual KITTI 2 (Cabon et al., 2020) (1,000), scored by the mean and median angular error pooled over all valid pixels. Semantic segmentation uses the 500 Cityscapes validation images (Cordts et al., 2016) at their full 2048×1024 resolution and reports mIoU over the 19 classes. Instance segmentation uses SA-Co/Gold (Carion et al., 2025), 166,766 image and noun-phrase queries in seven domains, and reports the official cgF_1 together with its two factors, the image-level presence MCC (IL_MCC) and the positive micro F_1 (pmF_1). Referring segmentation uses the UMD validation split of RefCOCOg (Mao et al., 2016; Nagaraja et al., 2016) (4,896 expressions) and reports cIoU and mIoU.

4.2 Results

The complete list of compared models is provided in Appendix C.7, Table C.7. As reported in Table 1a, the average metric depth δ_1 of UniVision is comparable to those of the strongest listed open models across the evaluated benchmarks. On relative depth evaluation in Table 1b, UniVision obtains the top results among listed methods on KITTI and ETH3D while maintaining competitive accuracy on NYUv2. Across the evaluated datasets in Table 1c, the mean normal error of UniVision is lower than that of all listed zero shot generative baselines, including Vision Banana.

As seen in Figure 3, UniVision preserves thin structures and object boundaries more clearly than Depth Anything 3. Similarly, Marigold V2 reports that generative depth models capture fine structures such as foliage and wires effectively (Pavlovic et al., 2026). This visual result highlights the benefit of perception as generation for preserving fine visual detail. Additional qualitative examples for surface normal estimation and image segmentation are provided in Appendix D.1 (Figure D.1).

As shown across Table 1d and Table 1f, UniVision exceeds listed open zero shot baselines on semantic mIoU and instance cgF_1, though its performance remains below Vision Banana. UniVision Agent achieves the highest cIoU among listed zero shot methods in Table 1e, outperforming Vision Banana and SAM 3 Agent.

4.3 Inference Ablation

In this section, we want to explore the inference technology of generation on perception. We first ablate test time resolution and noise shifts with classifier free guidance fixed at 4, as shown in Figure 4(a). For instance segmentation, pixel caps preserve native image dimensions below the limit

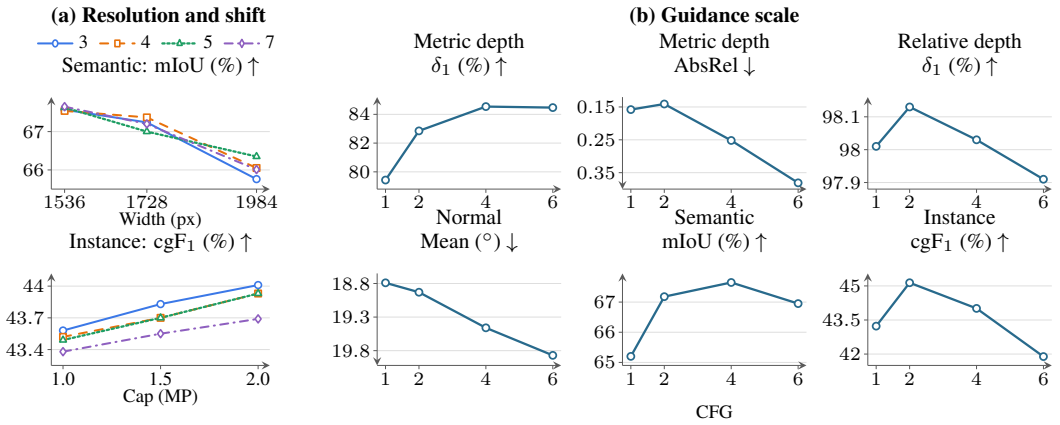


Figure 4: Inference ablations on resolution, noise shift and cfg scale.

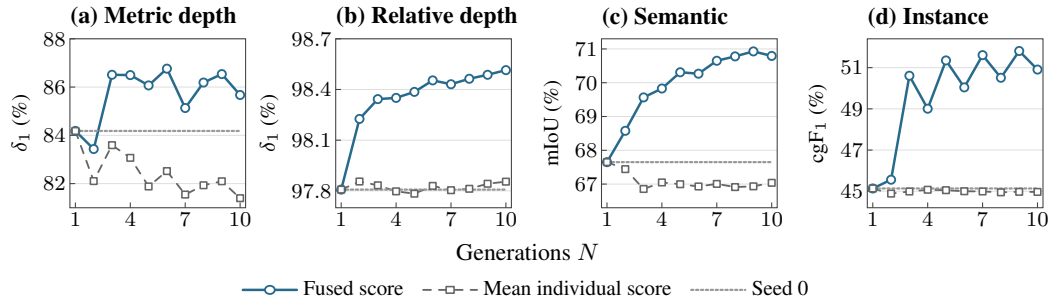


Figure 5: Test time aggregation across N generations.

without upsampling. Increasing the resolution cap consistently helps instance segmentation, which is best at a 2 MP cap with a noise shift of 3. Conversely, semantic segmentation attains its peak accuracy at 1536×768 with a shift of 7, but further resolution increases degrade performance.

Figure 4(b) illustrates the impact of classifier free guidance across tasks. Though no single scale is universally optimal, the CFG 2 is an optimal result, which is best/second-best cross all the benchmark evaluation.

4.4 Test-Time Aggregation

Many perception models rely on deterministic prediction. In contrast, stochastic generative models produce varying outputs across random seeds. We test whether aggregating this variation across seeds can improve predictions. Following a fixed seed order, we combine the first N outputs from ten seeds using the same checkpoint and task settings.

For depth, we decode each output to metres, average the values at each pixel position, and encode the result once. For semantic segmentation, each pixel receives the most frequent valid class across seeds, ignoring black unless every seed predicts it. Ties are broken using the earliest seed. For instance segmentation, we group masks across seeds whose pairwise IoU is at least 0.5, allowing at most one mask per seed in each group. We require a strict majority of all N seeds to retain a mask group and individual pixels, followed by a single application of decoder refinement after fusion.

As shown in Figure 5, test-time aggregation significantly boosts performance across tasks. For segmentation, fusing ten samples improves Cityscapes mIoU from 67.65% to 70.79% and SA-Co cgF$_1$ from 45.14% to 50.91%. For depth estimation, aggregating $N = 6$ generations yields the highest metric depth δ_1 of 86.76% and lowest AbsRel of 0.165 (improving over 84.18% and 0.242 at $N = 1$), while relative depth δ_1 steadily increases from 97.81% ($N = 1$) to 98.51% ($N = 10$). With aggregation enabled, UniVision surpasses all open zero-shot baselines on semantic mIoU and instance cgF$_1$ (Tables 1d and 1f). These empirical gains highlight a clear test-time scaling effect in generative perception, where allocating additional inference compute across stochastic seeds systematically refines visual predictions.

Table 2: UniGenBench accuracy (%) across four training conditions. The T2I controls match w/ vision by T2I token exposure or total training iterations.

	Prompt sets			Judge categories									
Model	All	Public	Phys.	Log.	Comp.	Act.	Attr.	Rel.	Lay.	Text	Know.	Style	Gram.
Baseline	79.71	82.32	77.32	67.10	74.40	64.00	89.09	79.23	86.31	4.33	**88.07**	95.65	79.82
T2I (Eq. tokens)	80.86	82.32	79.52	63.23	74.64	64.66	89.52	84.57	87.39	4.81	85.93	**96.61**	**85.09**
T2I (Eq. iters.)	79.01	80.92	77.25	60.65	76.57	61.64	87.82	82.99	87.84	4.33	83.18	93.06	75.00
w/ vision	**82.77**	**85.64**	**80.13**	**70.97**	**78.26**	**67.74**	**91.75**	**85.86**	**88.65**	**5.77**	86.85	**96.61**	81.58



(a) "...at the exact moment it **hits the racket**; the ball is visibly flattened..."

(b) "...holds up a small, metallic, **pyramid-shaped holographic projector**..."

(c) "...**two** bright yellow vintage Beetle cars are **parked side by side**..."

(d) "...**glowing books**... **freely shuttle between** the rustic wooden bookshelves..."

(e) "**On its broad back** grows a peculiar coral reef made of shattered planetary debris..."

(f) "...stands alone **in front of a glass display case**, **observing intently**..."

Figure 6: Selected UniGenBench examples comparing T2I and w/ vision at equal T2I token exposure. Text excerpts indicate prompt requirements.

4.5 Data Synergy

Dense visual perception tasks can be fundamentally viewed as targeted queries on image semantics, fine-grained spatial structures, and underlying scene geometry. Training on such tasks forces the generative backbone to acquire and structure explicit world priors and visual grounding representations. Recent advances such as REPA (Yu et al., 2025) and RAE (Zheng et al., 2025) demonstrate that incorporating auxiliary semantic losses enhances image generation quality. This naturally raises an intriguing question: can simply introducing perception data during joint generation training provide similar representation benefits and improve generation capability?

To systematically answer this question, we conduct a controlled synergy study starting all experimental conditions from the same intermediate T2I checkpoint. The study compares four training settings: 1) the unchanged Baseline 2) training with vision and T2I data and uniformed sampling 3) T2I only training with same T2I token exposure in the second training and 4) T2I only training with the same training iterations as second experiment. That mean We evaluate text-to-image performance using UniGenBench (Wang et al., 2025b), with detailed protocols provided in Appendix C.6.

As shown in Tab. 2, the w/ vision model achieves an overall accuracy of 82.77%, surpassing the T2I control matched by token exposure at 80.86% by 1.91%. The T2I control matched by total iterations attains 79.01%. Across prompt subsets, w/ vision achieves 85.64% on public prompts and 80.13% on physical AI prompts. It shows that with vision data mixed in training, it can help the performance of generation model. We further investigate what field the improvements come from by the label of UniGenBench, showing that joint perception supervision primarily strengthens spatial and structural capabilities. In Table 2, the largest gains concentrate on logical reasoning, compound composition,

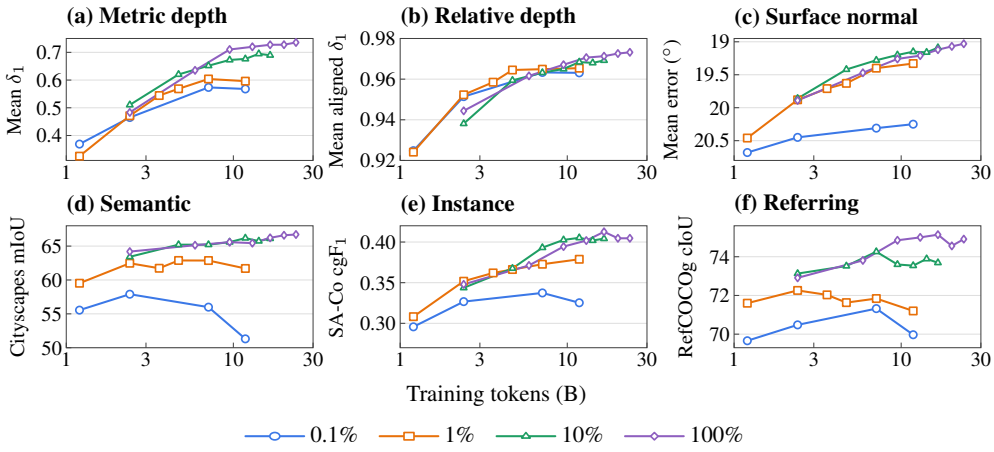


Figure 7: Single task data scaling curves. Depth averages five datasets equally; surface normal averages three datasets equally.

actions, and spatial attributes. Figure 6 shows selected examples where the *w/ vision* model follows prompt details more closely than the T2I model. It depicts contact between a tennis ball and a racket in panel (a), gives a holographic projector a pyramid shape in panel (b), and shows two yellow cars side by side in panel (c). It places a light trail between the bookshelves in panel. These samples reflect better alignment with requested object attributes, counts, and spatial relations.

4.6 Data Scaling

The success of modern generation models largely from scaling laws, where training on larger-scale high-quality datasets led to better zero-shot generalization. In this work, we investigate whether similar scaling dynamics hold under the perception-as-generation paradigm. Specifically, we examine how training data volume influences downstream performance across diverse perception tasks under matched compute budgets. Thus, we conduct a single task data scaling study, training independent models for each of five tasks and different token budgets. Training consumes 2.368M packed tokens per update, counting repeated exposures. Budgets extend up to 23.7B tokens. Both metric depth and relative depth are evaluated from the same depth model predictions as an equal average across NYUv2, KITTI, ETH3D, DIODE indoor, and iBims-1.

Figure 7 shows the data scaling results. Three conclusion can be drawn from it: 1) data scaling do help the model's performance in tasks like segmentation and metric depth estimation, where metric depth and referring segmentation, which need world prior knowledge and high-level semantic knowledge benefit most from it. At a 16.6B token budget, increasing data from 10% to 100% improves metric depth δ_1 from 68.96% to 72.73% and referring segmentation cIoU from 73.69% to 75.14%. 2) for most of tasks, 10% data, which is around 200K-300K (1M for metric depth), already can help model get high performance. At the same budget, 10% data yields 66.11% semantic mIoU and 40.42% instance cgF$_1$, compared with 66.22% and 41.26% using full data. 3) For structure-based task like relative depth and surface normal, which benefits most from generation model's internal representation, even 1% in normal and 0.1% in relative depth shows similar performance, where is around 20K data. At an 11.8B token budget, the 1% normal run has 0.18° higher mean angular error, and the 0.1% depth run has 0.53 percentage points lower aligned relative δ_1, compared with their respective 10% runs at that same budget. Qualitative examples appear in Appendix D.3.

5 Conclusion

In this work, UniVision addresses dense visual perception through image generation by combining deterministic codecs with scalable expert annotations. Experiments show competitive zero shot accuracy across multiple benchmarks. Output aggregation improves prediction accuracy at the cost of additional inference compute. In this single task setup, full data improves metric depth and referring segmentation over 10% data at later shared token budgets. Finally, our separate study indicates that joint training with perception and generation data also improves generation capbilities.

AI USE STATEMENT

AI models assisted with training data annotation, generating referring expressions, and model based evaluation tasks including candidate selection, presence gating, and benchmark judging. AI tools assisted with code implementation and manuscript polishing. Checks included comparing statements with experiment records, inspecting visual model outputs and figures, checking scripts and calculations, and reviewing the compiled manuscript. The authors take full responsibility for the content, claims, and artifacts in this paper.

ETHICS STATEMENT

This research studies visual perception and generation using existing datasets and curated data documented in Appendix C.2, along with model annotations. Models can inherit dataset biases, propagate annotation errors, and produce visually plausible yet incorrect depth or masks. Other risks include privacy in visual data and the misuse of synthesis or editing to create misleading images. Good benchmark performance does not guarantee deployment safety.

REPRODUCIBILITY STATEMENT

Appendix B details the codec algorithms, the palette, data curation, and instructions. Appendix C provides the main training and perception inference settings, source inventories, and benchmark task prompts. In addition, Section 4 defines the benchmarks and experimental comparisons.

REFERENCES

Gwangbin Bae and Andrew J. Davison. Rethinking inductive biases for surface normal estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. URL https://arxiv.org/abs/2403.00712v1.

Dmitry Baranchuk, Ivan Rubachev, Andrey Voynov, Valentin Khrulkov, and Artem Babenko. Label-efficient semantic segmentation with diffusion models. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2112.03126.

Aleksei Bochkovskii, Amaël Delaunoy, Hugo Germain, Marcel Santos, Yichao Zhou, Stephan R. Richter, and Vladlen Koltun. Depth Pro: Sharp monocular metric depth in less than a second. In *International Conference on Learning Representations*, 2025.

Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual KITTI 2. *arXiv preprint arXiv:2001.10773*, 2020. URL https://arxiv.org/abs/2001.10773.

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, 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. *arXiv preprint arXiv:2511.16719*, 2025. URL https://arxiv.org/abs/2511.16719v2.

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/CVF Conference on Computer Vision and Pattern Recognition*, 2016.

Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017.

David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in Neural Information Processing Systems*, 2014.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *Proceedings of the International Conference on Machine Learning*, volume 235, pp. 12606–12633. PMLR, 2024. URL https://proceedings.mlr.press/v235/esser24a.html.

Shenghao Fu, Qize Yang, Qijie Mo, Junkai Yan, Xihan Wei, Jingke Meng, Xiaohua Xie, and Wei-Shi Zheng. LLMDet: Learning strong open-vocabulary object detectors under the supervision of large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025a.

Xiao Fu, Wei Yin, Mu Hu, Kaixuan Wang, Yuexin Ma, Ping Tan, Shaojie Shen, Dahua Lin, and Xiaoxiao Long. GeoWizard: Unleashing the diffusion priors for 3D geometry estimation from a single image. In *Proceedings of the European Conference on Computer Vision*, 2024. URL https://arxiv.org/abs/2403.12013.

Yunxiang Fu, Meng Lou, and Yizhou Yu. SegMAN: Omni-scale context modeling with state space models and local attention for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025b.

Valentin Gabeur, Shangbang Long, Songyou Peng, Paul Voigtlaender, Shuyang Sun, Yanan Bao, Karen Truong, Zhicheng Wang, Wenlei Zhou, Jonathan T. Barron, Kyle Genova, Nithish Kannen, Sherry Ben, Yandong Li, Mandy Guo, Suhas Yogin, Yiming Gu, Huizhong Chen, Oliver Wang, Saining Xie, Howard Zhou, Kaiming He, Thomas Funkhouser, Jean-Baptiste Alayrac, and Radu

Soricut. Image generators are generalist vision learners. *arXiv preprint arXiv:2604.20329*, 2026. URL https://arxiv.org/abs/2604.20329v3.

Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

Gemini Team, Google. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025. URL https://arxiv.org/abs/2507.06261.

Zigang Geng, Binxin Yang, Tiankai Hang, Chen Li, Shuyang Gu, Ting Zhang, Jianmin Bao, Zheng Zhang, Houqiang Li, Han Hu, Dong Chen, and Baining Guo. InstructDiffusion: A generalist modeling interface for vision tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12709–12720, 2024.

Qin Guo, Hao Luo, Dongxu Yue, Weixuan Jin, Xiao Fu, Fan Wang, and Dan Xu. UniGP: Taming diffusion transformer for prior-preserved unified generation and perception. *arXiv preprint arXiv:2606.30332*, 2026. URL https://arxiv.org/abs/2606.30332v1.

Xiaoyang Han, Jianhua Li, Kewang Deng, Zukai Chen, Xuanke Shi, et al. Vision as unified multimodal generation. *arXiv preprint arXiv:2607.06560*, 2026. URL https://arxiv.org/abs/2607.06560v1.

Jing He, Haodong Li, Mingzhi Sheng, and Ying-Cong Chen. Lotus-2: Advancing geometric dense prediction with powerful image generative model. *arXiv preprint arXiv:2512.01030*, 2025a. URL https://arxiv.org/abs/2512.01030v3.

Jing He, Haodong Li, Wei Yin, Yixun Liang, Leheng Li, Kaiqiang Zhou, Hongbo Zhang, Bingbing Liu, and Ying-Cong Chen. Lotus: Diffusion-based visual foundation model for high-quality dense prediction. In *International Conference on Learning Representations*, 2025b. URL https://arxiv.org/abs/2409.18124.

Xinyu Huang, Yi-Jie Huang, Youcai Zhang, Weiwei Tian, Rui Feng, Yuejie Zhang, Yanchun Xie, Yaqian Li, and Lei Zhang. Open-set image tagging with multi-grained text supervision. *arXiv preprint arXiv:2310.15200*, 2023. URL https://arxiv.org/abs/2310.15200.

Seil Kang, Jinyeong Kim, Junhyeok Kim, and Seong Jae Hwang. Your large vision-language model only needs a few attention heads for visual grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

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*, pp. 9492–9502, 2024.

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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4015–4026, 2023.

Tobias Koch, Lukas Liebel, Friedrich Fraundorfer, and Marco Körner. Evaluation of CNN-based single-image depth estimation methods. In *Proceedings of the European Conference on Computer Vision Workshops (ECCV 2018)*, pp. 331–348, 2019. URL https://doi.org/10.1007/978-3-030-11015-4_25.

Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, Tom Duerig, and Vittorio Ferrari. The open images dataset V4: Unified image classification, object detection, and visual relationship detection at scale. *International Journal of Computer Vision*, 128(7):1956–1981, 2020.

Xin Lai, Zhuotao Tian, Yukang Chen, Yanwei Li, Yuhui Yuan, Shu Liu, and Jiaya Jia. LISA: Reasoning segmentation via large language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Haotong Lin, Sili Chen, Jun Hao Liew, Donny Y. Chen, Zhenyu Li, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth anything 3: Recovering the visual space from any views. *arXiv preprint arXiv:2511.10647*, 2025. URL https://arxiv.org/abs/2511.10647v1.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023. URL https://iclr.cc/virtual/2023/poster/11309.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection. In *European Conference on Computer Vision*, pp. 38–55, 2024. URL https://doi.org/10.1007/978-3-031-72970-6_3.

Ting Liu and Siyuan Li. Hybrid global-local representation with augmented spatial guidance for zero-shot referring image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations*, 2023. URL https://iclr.cc/virtual/2023/poster/11266.

Jiasen Lu, Christopher Clark, Rowan Zellers, Roozbeh Mottaghi, and Aniruddha Kembhavi. Unified-IO: A unified model for vision, language, and multi-modal tasks. In *International Conference on Learning Representations*, 2023. URL https://iclr.cc/virtual/2023/poster/11770.

Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L. Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 11–20, 2016.

Matthias Minderer, Alexey Gritsenko, and Neil Houlsby. Scaling open-vocabulary object detection. In *Advances in Neural Information Processing Systems*, volume 36, pp. 72983–73007, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/e6d58fc68c0f3c36ae6e0e64478a69c0-Abstract-Conference.html.

Varun K. Nagaraja, Vlad I. Morariu, and Larry S. Davis. Modeling context between objects for referring expression understanding. In *European Conference on Computer Vision*, 2016.

NVIDIA. Cosmos 3: Omnimodal world models for physical AI. *arXiv preprint arXiv:2606.02800*, 2026. URL https://research.nvidia.com/labs/cosmos-lab/cosmos3/technical-report.pdf.

Igor Pavlovic, Thiemo Wandel, Anton Obukhov, Luca Bartolomei, Andrey Davydov, Fabio Tosi, Matteo Poggi, Sabine Süsstrunk, and Dengxin Dai. Marigold V2: Revisiting diffusion transformers for monocular depth estimation. *arXiv preprint arXiv:2609.08084*, 2026. URL https://arxiv.org/abs/2609.08084v1.

Luigi Piccinelli, Yung-Hsu Yang, Christos Sakaridis, Mattia Segu, Siyuan Li, Luc Van Gool, and Fisher Yu. UniDepth: Universal monocular metric depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Luigi Piccinelli, Christos Sakaridis, Mattia Segu, Yung-Hsu Yang, Siyuan Li, Wim Abbeloos, and Luc Van Gool. UniK3D: Universal camera monocular 3D estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. URL https://arxiv.org/abs/2503.16591v1.

Hanoona Rasheed, Muhammad Maaz, Sahal Shaji Mullappilly, Abdelrahman Shaker, Salman Khan, Hisham Cholakkal, Rao M. Anwer, Eric Xing, Ming-Hsuan Yang, and Fahad S. Khan. GLaMM: Pixel grounding large multimodal model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Tianhe Ren, Yihao Chen, Qing Jiang, Zhaoyang Zeng, Yuda Xiong, Wenlong Liu, Zhengyu Ma, Junyi Shen, Yuan Gao, Xiaoke Jiang, Xingyu Chen, Zhuheng Song, Yuhong Zhang, Hongjie Huang, Han Gao, Shilong Liu, Hao Zhang, Feng Li, Kent Yu, and Lei Zhang. DINO-X: A unified vision model for open-world object detection and understanding. *arXiv preprint arXiv:2411.14347*, 2024. URL https://arxiv.org/abs/2411.14347.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022. URL https://arxiv.org/abs/2112.10752.

Thomas Schöps, Johannes L. Schönberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017.

Yunhang Shen, Chaoyou Fu, Peixian Chen, Mengdan Zhang, Ke Li, Xing Sun, Yunsheng Wu, Shaohui Lin, and Rongrong Ji. Aligning and prompting everything all at once for universal visual perception. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

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

Yihong Sun, Seoung Wug Oh, Jiahui Huang, Bharath Hariharan, and Joon-Young Lee. Unified video dense prediction from disjoint data. In *Proceedings of the European Conference on Computer Vision*, 2026. URL https://unid-video.github.io/.

Luming Tang, Menglin Jia, Qianqian Wang, Cheng Perng Phoo, and Bharath Hariharan. Emergent correspondence from image diffusion. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Hao Wang, Limeng Qiao, Zequn Jie, Zhijian Huang, Chengjian Feng, Qingfang Zheng, Lin Ma, Xiangyuan Lan, and Xiaodan Liang. X-SAM: From segment anything to any segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026a. URL https://arxiv.org/abs/2508.04655v2.

Jiyuan Wang, Chunyu Lin, Lei Sun, Rongying Liu, Lang Nie, Mingxing Li, Kang Liao, and Xiangxiang Chu. FE2E: From editor to dense geometry estimator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026b. URL https://arxiv.org/abs/2509.04338v2.

Ruicheng Wang, Sicheng Xu, Yue Dong, Yu Deng, Jianfeng Xiang, Zelong Lv, Guangzhong Sun, Xin Tong, and Jiaolong Yang. MoGe-2: Accurate monocular geometry with metric scale and sharp details. In *Advances in Neural Information Processing Systems*, 2025a. URL https://arxiv.org/abs/2507.02546v1.

Yibin Wang, Zhimin Li, Yuhang Zang, Yujie Zhou, Jiazi Bu, Chunyu Wang, Qinglin Lu, Cheng Jin, and Jiaqi Wang. Pref-GRPO: Pairwise preference reward-based GRPO for stable text-to-image reinforcement learning. *arXiv preprint arXiv:2508.20751*, 2025b. URL https://arxiv.org/abs/2508.20751v1.

Cong Wei, Yujie Zhong, Haoxian Tan, Yong Liu, Zheng Zhao, Jie Hu, and Yujiu Yang. HyperSeg: Towards universal visual segmentation with large language model. *arXiv preprint arXiv:2411.17606*, 2024. URL https://arxiv.org/abs/2411.17606.

Guangkai Xu, Yongtao Ge, Mingyu Liu, Chengxiang Fan, Kangyang Xie, Zhiyue Zhao, Hao Chen, and Chunhua Shen. What matters when repurposing diffusion models for general dense perception tasks? In *International Conference on Learning Representations*, 2025. URL https://iclr.cc/virtual/2025/poster/30558.

Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything V2. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Chongjie Ye, Lingteng Qiu, Xiaodong Gu, Qi Zuo, Yushuang Wu, Zilong Dong, Liefeng Bo, Yuliang Xiu, and Xiaoguang Han. StableNormal: Reducing diffusion variance for stable and sharp normal. *ACM Transactions on Graphics*, 43(6), 2024. URL https://arxiv.org/abs/2406.16864.

Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. In *International Conference on Learning Representations*, 2025. URL https://iclr.cc/virtual/2025/poster/30467.

Hao Zhang, Feng Li, Xueyan Zou, Shilong Liu, Chunyuan Li, Jianwei Yang, and Lei Zhang. A simple framework for open-vocabulary segmentation and detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

Canyu Zhao, Yanlong Sun, Mingyu Liu, Huanyi Zheng, Muzhi Zhu, Zhiyue Zhao, Hao Chen, Tong He, and Chunhua Shen. DICEPTION: A generalist diffusion model for visual perceptual tasks. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://arxiv.org/abs/2502.17157v3.

Wenliang Zhao, Yongming Rao, Zuyan Liu, Benlin Liu, Jie Zhou, and Jiwen Lu. Unleashing text-to-image diffusion models for visual perception. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5729–5739, 2023.

Boyang Zheng, Nanye Ma, Shengbang Tong, and Saining Xie. Diffusion transformers with representation autoencoders. *arXiv preprint arXiv:2510.11690*, 2025. URL https://arxiv.org/abs/2510.11690.

Xueyan Zou, Zi-Yi Dou, Jianwei Yang, Zhe Gan, Linjie Li, Chunyuan Li, Xiyang Dai, Harkirat Behl, Jianfeng Wang, Lu Yuan, Nanyun Peng, Lijuan Wang, Yong Jae Lee, and Jianfeng Gao. Generalized decoding for pixel, image, and language. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

SUPPLEMENTARY OF

" ZERO-SHOT PERCEPTION WITH GENERATOR"

TABLE OF CONTENTS

A Additional Experiments **17**
A.1 Codec Ablation . . . 17
A.2 Instance Presence Gate Ablation . . . 17
A.3 Referring Segmentation Agent Ablation . . . 17

B Implementation Details **18**
B.1 Codecs . . . 18
B.2 Data Curation Pipeline . . . 21
B.3 Instruction Rewriting . . . 24

C Training and Inference Configurations **25**
C.1 Training . . . 25
C.2 Training Data . . . 25
C.3 Inference . . . 26
C.4 Instance Segmentation Presence Gate . . . 27
C.5 Referring Segmentation Agent . . . 27
C.6 UniGenBench Evaluation . . . 31
C.7 Baseline Model References . . . 34

D Visualization **35**
D.1 Surface Normal Estimation and Segmentation . . . 35
D.2 Test Time Aggregation . . . 38
D.3 Data Scaling . . . 40
D.4 Additional Text to Image Visualizations . . . 43
D.5 Training Data . . . 47

A Additional Experiments

A.1 Codec Ablation

We evaluate the decoding stage on fixed model predictions without retraining or regenerating outputs. Semantic segmentation is evaluated on Cityscapes and instance segmentation on SA-Co/Gold. Under identical inference settings, we compare decoding without refinements against the full codec detailed in Appendix B.1, Algorithms B.3 and B.4. Following their appendix, we also evaluate our implementation of the Vision Banana instance segmentation codec (Gabeur et al., 2026) on the same fixed outputs.

Table A.1: Codec ablation on Cityscapes and SA-Co/Gold. NR denotes not reported.

Codec	Semantic	Instance		
	mIoU (%, ↑)	cgF_1 (%, ↑)	pmF_1 (%, ↑)	IL_MCC (↑)
Without refinements	<u>64.3</u>	17.22	20.16	**0.8508**
Vision Banana codec	NR	<u>38.58</u>	<u>45.35</u>	NR
Full codec	**67.7**	**45.1**	**52.8**	<u>0.8504</u>

As shown in Table A.1, the full codec outperforms this Vision Banana codec implementation on both instance cgF_1 and pmF_1. Compared to decoding without refinements, the full codec improves semantic mIoU by 3.4 points with a slight decrease in IL_MCC from 0.8508 to 0.8504. These results confirm the benefit of the proposed decoding refinements on fixed model outputs.

A.2 Instance Presence Gate Ablation

We ablate the Gemini presence gate for UniVision instance segmentation on SA-Co/Gold.

Table A.2: Ablation of the Gemini presence gate for instance segmentation on SA-Co/Gold.

Setting	cgF_1 (%, ↑)	IL_MCC (↑)	pmF_1 (%, ↑)
Without gate	<u>2.30</u>	<u>0.0442</u>	**53.35**
With Gemini gate	**45.14**	**0.8504**	<u>52.82</u>

As shown in Table A.2, UniVision retains strong segmentation quality on positive queries without gating, achieving a pmF_1 of 53.35% compared to 52.82% with the gate. However, image level presence discrimination drops sharply without the gate, yielding an IL_MCC of 0.0442 and reducing cgF_1 to 2.30%. This poor discrimination is consistent with our training data containing no negative query examples, where the queried target is absent from the image. Consequently, the Gemini presence gate supports overall instance segmentation performance in this evaluation by filtering absent targets.

A.3 Referring Segmentation Agent Ablation

To evaluate the agent during inference, we test UniVision with its frozen mask decoder without the agent on RefCOCOg and compare it against the complete agent result reported in Table 1e (described in Appendix C.5).

Table A.3: Ablation of the referring segmentation agent on RefCOCOg.

Setting	cIoU (%, ↑)	mIoU (%, ↑)
Without agent	<u>63.40</u>	<u>68.40</u>
With agent	**75.39**	**76.47**

As shown in Table A.3, the complete agent workflow improves performance by 11.99 cIoU and 8.06 mIoU percentage points over direct generator predictions. This demonstrates the benefit of

the complete agent workflow combining multiple candidate generation, candidate selection, and conditional query rewriting for referring segmentation.

B Implementation Details

This section presents task codec implementations with encoding and decoding algorithms, and data curation pipelines with diagrams. Here, I denotes the source image, Y represents the encoded RGB target, and c indicates the task instruction.

B.1 Codecs

Metric depth. Metric depth is an image aligned scalar field in metres. Scalar 8 bit quantization loses detail. A monotone transform followed by a path along RGB cube edges allocates precision. Decoding projects onto that path and inverts the transform. We follow Vision Banana (Gabeur et al., 2026). The depth polyline $\mathcal{P} = [\mathbf{p}_0, \ldots, \mathbf{p}_7]$ connects $(0,0,0)$, $(255,0,0)$, $(255,255,0)$, $(0,255,0)$, $(0,255,255)$, $(0,0,255)$, $(255,0,255)$, and $(255,255,255)$. Encoding clips depth to $[0, 80]$ m; decoding guards the inverse transform at $t = 1$ and clips reconstructed depth to $[0, 80]$ m before resizing or aggregation. Before encoding, invalid depth values are sanitized: NaN and negative infinity become zero, and positive infinity becomes 80 m.

Algorithm B.1 Metric depth codec

Require: Polyline $\mathcal{P}$, compression scale $c = 10$, endpoint guard $\epsilon_d = 10^{-9}$.
procedure EncodeDepth(d)
  Replace NaN and $-\infty$ by 0; replace $+\infty$ by 80
  $d \leftarrow \mathrm{clip}(d, 0, 80)$; $t \leftarrow 1 - (1 + d/c)^{-2}$
  $s \leftarrow 7t$; $j \leftarrow \min(\lfloor s \rfloor, 6)$; $u \leftarrow s - j$
  return $\mathrm{round}((1-u)\mathbf{p}_j + u\mathbf{p}_{j+1})$
procedure DecodeDepth($\mathbf{r} \in [0,255]^3$)
  for $j = 0, \ldots, 6$ **do**
    $\mathbf{v}_j \leftarrow \mathbf{p}_{j+1} - \mathbf{p}_j$
    $u_j \leftarrow \mathrm{clip}((\mathbf{r} - \mathbf{p}_j) \cdot \mathbf{v}_j / \|\mathbf{v}_j\|_2^2, 0, 1)$
    $e_j \leftarrow \|\mathbf{r} - (\mathbf{p}_j + u_j \mathbf{v}_j)\|_2^2$
  end for
  $j^* \leftarrow \arg\min_j e_j$; $t \leftarrow \mathrm{clip}((j^* + u_{j^*})/7, 0, 1 - \epsilon_d)$
  return $\mathrm{clip}(c((1-t)^{-1/2} - 1), 0, 80)$

Surface normals. Surface normals form an image aligned unit vector field expressed in camera coordinates. An affine transformation maps vector components into standard RGB color space. Decoding applies the inverse affine mapping and normalizes the resulting vectors to recover unit directions. Normal vectors use camera coordinates with x right, y up, and z toward the viewer. Black marks invalid normal targets; the decoder normalizes recovered vectors without inferring a validity mask. Define $U(\mathbf{v}) = \mathbf{v}/\max(\|\mathbf{v}\|_2, 10^{-8})$.

Algorithm B.2 Surface normal codec

procedure EncodeNormal($\mathbf{n}$, valid)
  if valid is false **then**
    return $(0,0,0)$
  end if
  return $\mathrm{round}(255(U(\mathbf{n}) + 1)/2)$
procedure DecodeNormal($\mathbf{r} \in [0,255]^3$)
  return $U(2\mathbf{r}/255 - 1)$

Semantic segmentation. Semantic segmentation assigns class labels to individual pixels using a class to color mapping defined in the instruction. Because generated colors near object boundaries can be ambiguous, decoding relies on color confident seeds followed by spatial propagation. Semantic masks are painted in decreasing area order; the instruction gives the class to color mapping π. Decoders operate at the generated resolution. GuardSemantic first assigns every pixel to its nearest permitted color, including black. Pixels within RGB distance τ of that color are seeds. If

seeds exist, their labels are propagated to all pixels by nearest spatial distance; each nonblack raw 8 connected component is then restored if it contains a seed or has at least $\lceil 64HW/(960 \cdot 480) \rceil$ pixels. Without seeds, the raw assignments are retained. The color to class mapping converts the resulting labels to class IDs.

Algorithm B.3 Semantic segmentation codec

Require: Shared palette $\mathcal{C}$ of 40 colors (Table B.1); black **0** denotes background.

procedure EncodeSemantic($\mathcal{M} = \{(t_i, M_i)\}, \pi$)
  Initialize a black RGB canvas I
  for each (t_i, M_i) in decreasing mask area **do**
    $I(p) \leftarrow \pi(t_i)$ for $p \in M_i$
  end for
  return I
procedure DecodeSemantic(I, π)
  $\mathcal{C}_{\text{act}} \leftarrow \text{range}(\pi) \cup \{\mathbf{0}\}$; $a_{64} \leftarrow \lceil 64HW/(960 \cdot 480) \rceil$
  $\hat{a}(p) \leftarrow \arg\min_{\mathbf{c} \in \mathcal{C}_{\text{act}}} \|I(p) - \mathbf{c}\|_2$
  $R \leftarrow \{p : \|I(p) - \hat{a}(p)\|_2 \leq 20\}$; $Y \leftarrow \hat{a}$
  if $R \neq \emptyset$ **then**
    $Y(p) \leftarrow \hat{a}(\arg\min_{q \in R} \|p - q\|_2)$ for every pixel p
    for each nonblack raw 8 connected component C of $\hat{a}$ **do**
      if $C \cap R \neq \emptyset$ or $|C| \geq a_{64}$ **then**
        $Y(C) \leftarrow \hat{a}(C)$
      end if
    end for
  end if
  $B \leftarrow \{p : Y(p) \neq \mathbf{0}\}$
  if $B \neq \emptyset$ **then**
    Fill black pixels within Euclidean distance 1 of B from their nearest pixel in B
  end if
  return class IDs from $\pi^{-1}(Y)$, with black mapped to background

Instance segmentation. Instance segmentation assigns distinct masks to separate objects, including adjacent instances of the same class. Each object receives an individual color. The decoding process retains supported instances while resolving blended color values along shared boundaries. Instance masks are disjoint and receive distinct shuffled colors from InstanceColors, which extends $\mathcal{C}$ by farthest point sampling in RGB when necessary. ExactColorMap accepts at most 40 distinct nonblack RGB values, each at distance at least 14 from black; ColorPartitions returns one mask per such value. Otherwise, GuardInstance uses foreground distance at least 14 and nearest palette assignments. Let $a_q = \max(1, \lceil qHW/(960 \cdot 480) \rceil)$. A color activates only if one of its 8 connected cores has RGB distance at most 30, a second nearest distance margin of at least 8, and area at least a_{32}. Its owning raw component must also have area at least a_{64}, or the core must retain at least a_{16} pixels after erosion with a 3×3 square. After activation, support uses distance at most 30 without the margin test. A raw component qualifies if its largest connected support core has area at least a_{16}, if it has a support pixel and area at least a_{64}, or if its area is at least a_{32} and at least one quarter of its pixels provide support. Qualified seeds fill only their own foreground connected component, after which qualified raw regions are restored. One mask is returned per active color, including disconnected regions.

Unsupported foreground can form fallback masks by joining 8 neighboring pixels whose RGB distance is at most 30. A fallback component must have area at least a_{64} and retain at least 10% of its area under 3×3 erosion. Its mean color must be farther than 30 from every palette color, with at least half its pixels within distance 30 of that mean. Retained components are processed by decreasing area and merged when their mean colors are within distance 14, updating the merged mean by area. Rejected fragments remain unassigned.

Algorithm B.4 Instance segmentation codec

Require: Palette $\mathcal{C}$ of 40 colors (Table B.1); support and fallback tests defined above.

procedure ENCODEINSTANCE($\mathcal{M}$)
  $\pi \leftarrow$ INSTANCECOLORS($\mathcal{C}, |\mathcal{M}|$)
  Initialize a black canvas; paint each mask M_i with $\pi(i)$; **return** canvas
procedure DECODEINSTANCE(I)
  if EXACTCOLORMAP(I) **then**
    $\mathcal{M} \leftarrow$ COLORPARTITIONS(I)
  else
    $F \leftarrow \{p : \|I(p)\|_2 \geq 14\}$; assign foreground to nearest palette colors
    Activate colors using distance, margin, core area, and raw component tests
    $Q \leftarrow$ qualified raw regions; $R \leftarrow$ color confident pixels within Q
    for each 8 connected foreground component C with $R \cap C \neq \emptyset$ **do**
      Assign its pixels to the spatially nearest seed in $R \cap C$
    end for
    Restore raw labels on Q; collect one mask per active color
    Add fallback masks from unsupported foreground using the tests above
    $\mathcal{M} \leftarrow$ collected masks
  end if
  for each mask M from smallest to largest **do**
    $N \leftarrow$ surviving neighbor with greatest contact with M's one pixel outer rim
    if N exists, contact $\geq 40\%$, and $|M| < 0.10|N|$ **then**
      $N \leftarrow N \cup M$; remove M
    end if
  end for
  Erode each mask of at least 1,000 pixels once with a 4 connected kernel
  Retain the original mask if erosion would empty it; **return** $\mathcal{M}$

Referring segmentation. Referring segmentation generates a binary mask for the object identified by a referring expression. The instruction defines a target foreground color, while remaining pixels are black. Decoding assigns labels by comparing color distances to foreground and background references.

Algorithm B.5 Referring segmentation codec

procedure ENCODEREFERRING($M, \mathbf{c}_{\text{ref}}$)
  Initialize a black canvas I; $I(p) \leftarrow \mathbf{c}_{\text{ref}}$ for $p \in M$
  return I
procedure DECODEREFERRING($I, \mathbf{c}_{\text{ref}}$)
  return $\{p : \|I(p) - \mathbf{c}_{\text{ref}}\|_2 < \|I(p)\|_2\}$

The 40 palette values are exact 8 bit RGB hex codes in Table B.1. Black (`#000000`) is reserved as segmentation background. For Cityscapes evaluation, palette indices 1 through 19 map directly to the 19 standard evaluation classes listed in the table. For referring segmentation evaluation, target masks are always evaluated with palette red (`#E6194B`) against black background, whereas training instances can use other palette colors.

Table B.1: Shared segmentation palette.

ID	Name	RGB hex	Cityscapes	ID	Name	RGB hex
1	red	#E6194B	road	21	crimson	#DC143C
2	green	#3CB44B	sidewalk	22	gold	#FFD700
3	yellow	#FFE119	building	23	turquoise	#40E0D0
4	blue	#4363D8	wall	24	violet	#EE82EE
5	orange	#F58231	fence	25	coral	#FF7F50
6	purple	#911EB4	pole	26	indigo	#4B0082
7	cyan	#42D4F4	traffic light	27	salmon	#FA8072
8	magenta	#F032E6	traffic sign	28	khaki	#F0E68C
9	lime	#BFEF45	vegetation	29	plum	#DDA0DD
10	pink	#FABED4	terrain	30	orchid	#DA70D6
11	teal	#469990	sky	31	tan	#D2B48C
12	lavender	#DCBEFF	person	32	skyblue	#87CEEB
13	brown	#9A6324	rider	33	tomato	#FF6347
14	beige	#FFFAC8	car	34	slateblue	#6A5ACD
15	maroon	#800000	truck	35	seagreen	#2E8B57
16	mint	#AAFFC3	bus	36	hotpink	#FF69B4
17	olive	#808000	train	37	chocolate	#D2691E
18	apricot	#FFD8B1	motorcycle	38	steelblue	#4682B4
19	navy	#000075	bicycle	39	forestgreen	#228B22
20	gray	#A9A9A9		40	darkorange	#FF8C00

B.2 Data Curation Pipeline

Metric depth preprocessing. Table B.2 summarizes depth recovery across all 26 training sources, which reuse released metric datasets without new depth collection. Sources either yield metric camera axis depth directly or require monocular completion P. In procedure P, relative geometry m from MoGe-2 ViT-L (Wang et al., 2025a) without metric scaling is aligned to positive sensor depth d_{sensor} across $n \geq n_{\min}$ coincident pixels, where $n_{\min} = 200$ for A2D2 and $n_{\min} = 1000$ for all other sources. Overlap residuals r, median a, threshold τ, inliers $\mathcal{R}$, and scale s satisfy

$$\begin{aligned} r &= \log d_{\text{sensor}} - \log m, \qquad a = \operatorname{median}(r), \\ \tau &= \operatorname{clip}(3 \times 1.4826 \times \operatorname{median}(|r - a|), 0.08, 0.35), \\ \mathcal{R} &= \{\, i \mid |r_i - a| \leq \tau \,\}, \qquad s = \exp(\operatorname{median}_{\mathcal{R}}(r)). \end{aligned} \tag{B.1}$$

If $|\mathcal{R}| < n/2$, inliers default to the $\max(n_{\min}, \lceil 0.7n \rceil)$ smallest absolute centered residuals $|r_i - a|$. We enforce $s \in [0.01, 100]$ without additive shift. Records failing the minimum overlap count or scale bounds are rejected. A log space Poisson solver reconstructs metric depth across pyramid levels $H/16$, $H/8$, $H/4$, and $H/2$ using gradients of $\log(s \cdot m)$ clipped to $[-3, 3]$. Gradient confidence is set to 1.0 on valid geometry where finite depth predictions exist and 0.2 elsewhere, with data weight 5 and conjugate gradient tolerance 10^{-5} using iteration caps 2500, 1200, 600, and 300. As an exception, A2D2 uses its binary valid geometry mask directly with a solver floor of 10^{-4} and leaves unobserved invalid pixels unassigned. Bilinear upsampling restores full resolution, followed by reinserting accepted sensor anchors before enforcing the 80 m encoding cap. UASOL preserves scale inliers $\mathcal{R}$ only. Fully dense ICL-NUIM frames bypass completion entirely. WildRGB-D follows a separate pipeline without procedure P: it fits an affine scale and shift by least squares, solves the Poisson system with iteration caps 5000, 2000, 1000, and 500, and restores values with priority: original sensor over accepted multiview over teacher. Neither the optional learned VideoLDCM refiner nor sky segmentation is enabled.

Table B.2: Depth preprocessing by source.

Sources	Metric conversion	Completion
TartanAir, InFlux indoor, EDEN, GTA-SfM	Read released camera axis depth (packed float32 BGRA, depthsharp NPY, MAT, or metric array)	None
UrbanSyn	Multiply released EXR depth by 10^5 metres	None
Hypersim	Convert ray distance D to camera axis depth via $D \cdot \lvert r_z \rvert / \lVert r \rVert$ using per scene camera rays	None
IRS, UnrealStereo4K, Spring	Convert disparity q via $Z = bf/q$ (calibrated bf; Spring uses $b = 0.065$ m, per frame f_x, and `q[::2,::2]`)	None
A2D2, DurLAR, PandaSet, Boreas, M2DCD-SLAM, CropCount3D	Released or calibrated LiDAR projections into the RGB camera; nearest value at collisions	Completion P; restore all accepted anchors
DrivingStereo, DSEC	DrivingStereo: depth $/256$ m; DSEC: disparity $/256$ pixels, then calibrated bf/q	Completion P; restore all accepted anchors
TUM RGB-D, YCB-Video	TUM: uint16 $/5000$ m; YCB-Video: stored depth $\times$depth_scale$/1000$ m (depth_scale $= 0.1$)	Completion P; restore all accepted anchors
Redwood indoor, ARKitScenes, BS3D, MuSHRoom	Registered sensor depth or camera frame mesh depth in millimetres $/1000$	Completion P; restore all accepted anchors
UASOL	ZED stereo depth in millimetres $/1000$; retain range $[0.7, 19.9]$ m	Completion P; restore scale inliers $\mathcal{R}$ only
ICL-NUIM	Registered uint16 metric depth; retain range $(0.05, 20]$ m	Completion P on holes only; dense frames bypass completion
WildRGB-D	Sensor depth $/1000$ m; fill holes by reprojecting up to two preceding and two following frames when at least two views agree with MAD/median ≤ 0.03	Least squares affine scale and shift Poisson fit; restore original sensor, then accepted multiview

For IRS, $bf = 48$ pixel metres, except for sequences `OfficeMedley3_BL2` and `ModernClassicInterior_NightFall_BL2` where $bf = 96$ pixel metres. UnrealStereo4K uses calibrated bf at quarter resolution.

Surface normals. Normal targets are distilled from MoGe-2 on OpenImages (Kuznetsova et al., 2020) and filtered by valid coverage and angular spread. Predictions in computer vision conventions are converted to the target camera frame via $N = (N_{\mathrm{cv},x}, -N_{\mathrm{cv},y}, -N_{\mathrm{cv},z})$. Surface normal curation retains at least 50% valid geometry and a mean angular spread of at least $12°$; invalid geometry is encoded as black.

$$M = M_{\mathrm{geom}} \cap \{p : N(p) \text{ is finite},\ \|N(p)\|_2 > 0\},$$
$$u = U\left(\frac{1}{|M|}\sum_{p\in M} U(N(p))\right), \qquad \bar{\theta} = \frac{1}{|M|}\sum_{p\in M} \arccos\big(\mathrm{clip}(U(N(p))^\top u, -1, 1)\big). \tag{B.2}$$

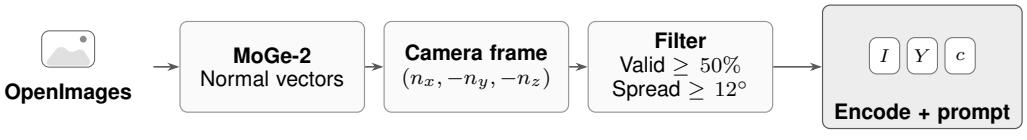


Figure B.1: Surface normal data curation pipeline.

Semantic segmentation. Semantic targets use RAM++ (Huang et al., 2023) tags and SAM3 (Carion et al., 2025) masks on OpenImages and SA-1B (Kirillov et al., 2023), with confidence and area filters. Semantic curation filters every mask predicted for each of at most 25 tags. Spurious regions

with prediction confidence $s_M < 0.30$ or pixel area $|M| < 64$ are pruned. The source image determines a stable class to color mapping, varied across images. Masks may overlap before the codec renders larger masks first.

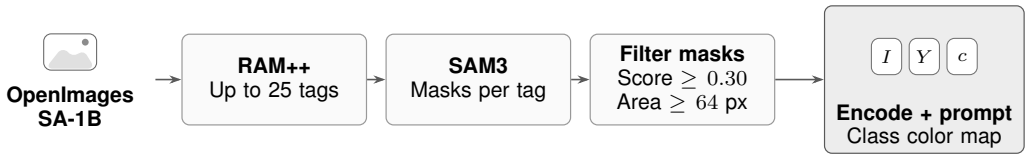


Figure B.2: Semantic segmentation data curation pipeline.

Instance segmentation. Instance segmentation targets are SAM3 (Carion et al., 2025) masks generated from OpenImages bounding box annotations. For OpenImages, SOURCECOMPLETE rejects an image/class pair containing excluded or unrendered annotations, including group, depiction, and inside boxes. MATCHCLIPRESOLVE jointly prompts SAM3 with class text and all eligible boxes, matches masks to boxes one to one, clips masks to their boxes, and resolves overlap by mask probability. Every required instance must remain valid and visible. Boxes themselves may overlap. SUPPLEMENT admits additional class prompted masks only after score ≥ 0.75, geometry, duplicate, and retained area checks, preserving the required instances.

The open vocabulary stream pairs OpenImages images with RAM++ tags and SAM3 masks, retaining examples where at least two instances pass a vision language model audit. Box derived OpenImages samples are partitioned into six instance count bins (1, 2, 3 to 4, 5 to 8, 9 to 20, and 21 or more), with single instance samples allocated 15% of the instance budget. Open vocabulary pairs form a separate multiple instance stream.

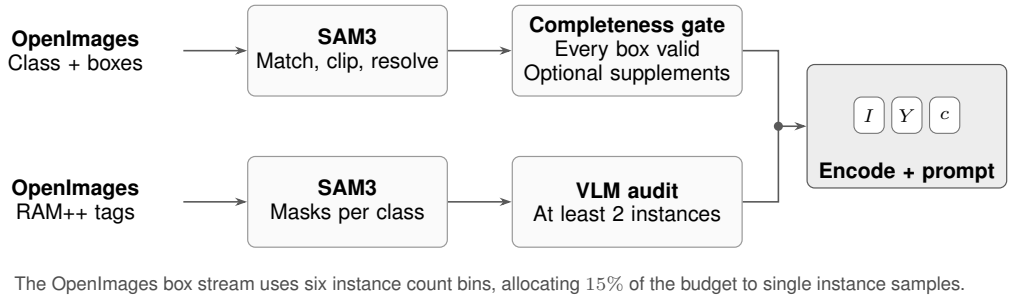


Figure B.3: Instance segmentation data curation pipeline.

Referring segmentation. For referring curation, CANDIDATEGATE requires two to four instances of one class, each with bounding box area greater than $0.05HW$. Qwen3.5 receives the clean image, labeled overlays, and crops; it checks the visible inventory and proposes grounded expressions of 5 to 15 words. VERIFYPHRASE first retains SAM3 predictions with score ≥ 0.55 and bounding box area fraction in $(0.05, 0.90)$. It accepts exactly one such prediction only if target IoU is ≥ 0.60, target precision and recall are ≥ 0.70, and every sibling has IoU ≤ 0.20 and recall ≤ 0.30. Every accepted expression is retained verbatim with its verified SAM3 mask.

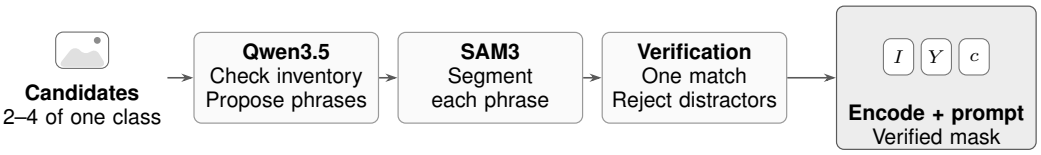


Figure B.4: Referring segmentation data curation pipeline.

B.3 INSTRUCTION REWRITING

Production training pipelines execute deterministic task templates. Surface normal training uses one fixed sentence. OpenImages semantic and referring streams place class names, color strings, or referring expressions into deterministic templates. Instance segmentation templates substitute dataset class names, while the SA-1B referring pipeline places verified expressions directly into deterministic templates. Referring phrases are proposed by a vision language model during curation; subsequent instruction paraphrasing is a separate step.

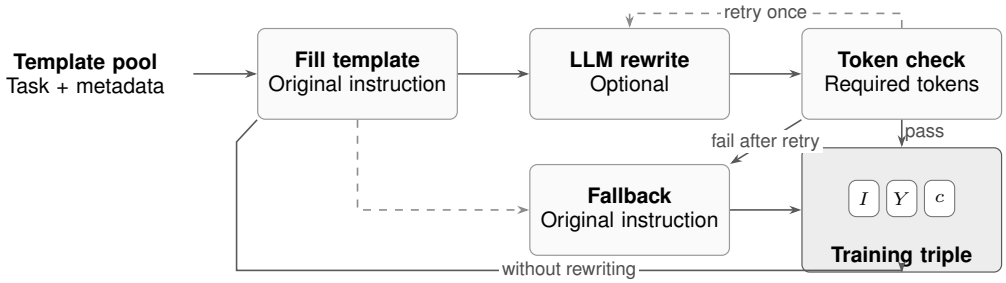


Figure B.5: Shared instruction construction pipeline.

C Training and Inference Configurations

C.1 Training

We initialize from Cosmos3 Nano, freeze the vision language backbone, and train the generation expert and adapters with flow objective in Equation 2. We evaluate moving average checkpoint 50,000. See Table C.1 for configuration details.

Table C.1: Training settings for the main model run.

Setting	Value	Setting	Value
Hardware	8×4 GB200	Reported update	50,000
Token budget / GPU	74,000	Image area cap	1,048,576 px
Optimizer	AdamW	(β_1, β_2)	$(0.9, 0.99)$
Peak learning rate	2×10^{-5}	Weight decay	0.05
Warmup updates	625	Decay horizon	56,250
LR at horizon	1×10^{-5}	Gradient norm clip	1.0
Gradient accumulation	1	Precision	BF16
Guidance dropout	0.1	Power EMA rate	0.1
Base mixture: vision / T2I / editing (%)		80 / 10 / 10	

Linear learning rate decay applies over 56,250 updates.

Within the vision stream, relative task probabilities remain 27.2% depth, 25.9% instance segmentation, 19.0% surface normals, 16.3% semantic segmentation, and 11.6% referring segmentation throughout training.

C.2 Training Data

Tables C.2, C.3, and C.4 list source inventories and within task sampling shares. The data includes 23,982,577 vision samples, 6,161,541 text to image samples, and approximately 6 million editing samples. Counts denote sample records rather than unique photographs or consumed training draws. Referring records count individual expressions.

Normal and semantic datasets use image sources OpenImages (Kuznetsova et al., 2020), with SA-1B (Kirillov et al., 2023) also used for semantic segmentation; referring expressions use OpenImages and SA-1B. Instance data uses OpenImages V7 organized by instance count, plus an additional OpenImages set with open vocabulary annotations. Refer to Appendix B.2 for annotation pipelines.

Table C.2: Vision source counts and within task sampling shares.

Task	Source or instance count	Samples	Share (%)
Metric depth	26 sources, Table C.3	11,825,095	100.00
Surface normals	OpenImages V7	2,492,274	100.00
Semantic	OpenImages V7	2,902,414	82.90
	SA-1B	598,524	17.10
Instance	OpenImages V7, 1 instance	2,019,248	15.00
	2 instances	878,890	24.84
	3 to 4 instances	733,493	20.73
	5 to 8 instances	512,546	14.49
	9 to 20 instances	294,124	8.31
	21 or more instances	59,621	1.69
	OpenImages, additional open vocabulary set	528,519	14.94
Referring	OpenImages V7	698,197	61.36
	SA-1B	439,632	38.64
Total vision samples		23,982,577	

Metric depth uses existing datasets listed in Table C.3. Dataset balancing and frame rates affect sampling shares, so shares need not be proportional to frame counts.

Table C.3: Metric depth sources, frame counts, and sampling shares.

Source	Frames	Share (%)	Source	Frames	Share (%)
TartanAir	1,969,542	20.19	TUM RGB-D	77,622	2.39
InFlux indoor	126,000	13.64	UrbanSyn	7,539	2.32
EDEN	368,642	11.34	BS3D	73,481	2.26
DrivingStereo	174,437	5.36	WildRGB-D	8,026,495	2.06
Hypersim	74,418	4.58	A2D2	6,539	2.01
DurLAR	145,915	4.09	MuSHRoom	17,756	1.82
Redwood indoor	123,471	3.80	PandaSet	49,440	1.52
Boreas	131,240	3.68	YCB-Video	133,936	1.37
ARKitScenes	44,599	3.43	DSEC	26,384	0.81
IRS	103,316	3.18	GTA-SfM	19,190	0.59
UASOL	48,781	3.00	Spring	5,000	0.45
M2DCD-SLAM	48,023	2.95	CropCount3D	5,936	0.37
UnrealStereo4K	8,200	2.52	ICL-NUIM	9,193	0.28
Total depth frames				11,825,095	100.00

Table C.4: Text to image and editing training sources.

Task	Source	Samples
T2I	In-house data	∼6M
Editing	In-house data	∼6M

C.3 Inference

Using 50 steps and EMA weights, single generations fix seed 0. Refer to Table C.5 for task settings. Codecs in Appendix B.1 decode predictions before resizing to the evaluation grid.

Table C.5: Inference configurations for main perception benchmarks.

Task	Input resolution or area cap	CFG	Shift
Depth	Native; ETH3D capped at 1,048,576 px	2	3
Surface normals	Native; 1,000,000 px cap	1	3
Semantic segmentation	1536×768	4	7
Instance segmentation	Native; 2,000,000 px cap	2	3
Referring segmentation	Within 512×512	4	3

Normal and instance images preserve aspect ratios under area caps, rounding dimensions to multiples of 16. Referring inputs maintain aspect ratios within 512×512 pixels and use 32 pixel alignment.

Table C.6 lists the exact evaluation prompts used across tasks, including the shared system prompt. Strings are reproduced verbatim from the evaluation codebase, with angle brackets denoting slot substitutions rather than literal prompt tokens.

Table C.6: Evaluation system and task prompts.

Task	Prompt
System	`You are a helpful assistant who will edit images based on the user's instructions.`
Metric and relative depth	`Predict the metric depth of this scene as an image. Visualized in the rainbow (black-red-yellow-green-cyan-blue-violet-white) colormap.`
Surface normals	`Predict the surface normal map of this scene as an image.`
Semantic segmentation	`Generate a visualization image of semantic segmentation, using this color mapping: {"<class>": #RRGGBB}. The background is black.`
Instance segmentation	`Generate an instance segmentation visualization of the <NP> in this image. Each <NP> instance is colored differently. The background is set to black.`
Referring segmentation	`A segmentation map of <expression>, rendered in red. The background is black.`
Referring with final punctuation	`A segmentation map of <expression> Rendered in red. The background is black.`

For semantic segmentation, the dictionary in Table C.6 maps class labels in double quotes to hexadecimal codes from the frozen benchmark subset. If this subset is empty, the prompt falls back to `"Generate a semantic segmentation visualization of this image."`. The negative prompt is `"black background"` for semantic segmentation and empty for all other tasks. For referring expressions ending in a period, exclamation mark, or question mark, the trailing clause `", rendered in red."` is replaced by `" Rendered in red."`, preserving the original punctuation.

C.4 Instance Segmentation Presence Gate

For SA-Co/Gold instance segmentation queries, presence gating uses Gemini 3.1 Flash-Lite with temperature 0 and a token limit of 512 output tokens. The model receives a single user message containing the query text followed by one source RGB image. There is no system prompt, and the request does not include ground truth annotations. The prompt is reused from Vision Banana (Gabeur et al., 2026), where the literal placeholder `<NP>` is replaced by the raw benchmark noun phrase without modification. Input images maintain their original aspect ratio, are downscaled with Lanczos interpolation to a maximum side length of 1536 pixels without upsampling, and are JPEG encoded at quality 90 with image detail set to high. If the encoded payload exceeds 4,500,000 bytes, quality decreases by 10 down to 60, after which both spatial dimensions are scaled by 0.8 iteratively until meeting the file size constraint.

Presence Gate Query Prompt

```
Is there an instance of <NP> visible in this image? Choose your answer from the following options: (A): Yes, (B): No.
```

The parser accepts explicit A/Yes or B/No, including a final explicit answer after explanatory text. The parser does not guess from the explanation. Contradictory or ambiguous answers are rejected. Failed requests were retried during preparation. Two persistent refusals remained after one audited manual retry and were recorded as No. Appendix A.2 provides an ablation isolating this gate.

C.5 Referring Segmentation Agent

Purpose and scope. UniVision weights remain frozen during inference. In RefCOCOg evaluation, GPT-5.6 Sol with xhigh reasoning effort serves as a multimodal controller to select candidate masks and conditionally rewrite referring queries. Like SAM 3 Agent (Carion et al., 2025), the controller uses image context to interpret language, but our method allows at most one rewrite. Rewriting

aims to make benchmark language resemble the grounded descriptive expressions used in training, which have 5 to 15 words, while retaining target identity and mask extent. Algorithm C.1 shows the primary flow without communication failures or format errors.

Candidate generation and filtering. The procedure GENERATEANDFILTER$(I, e, \mathcal{S})$ generates four RGB outputs from UniVision, resizes and decodes them into binary masks, and removes empty and duplicate masks. UniVision samples with moving average weights, UniPC with 50 steps, CFG 4, and shift 3. Image dimensions preserve aspect ratio within 512×512 with 32 pixel alignment. The query is placed directly into the task prompt from Table C.6 with an empty negative prompt. The sample ordinal i is parsed from the numeric suffix of the input image filename. For draw index $j \in \{0, 1, 2, 3\}$, seed values follow $20260703 + 4096 \times i + j$. The initial and rewrite rounds reuse these same four seeds. Generated RGB outputs are clipped to $[0, 1]$, multiplied by 255, and truncated to uint8. Outputs are resized to the original image resolution using nearest neighbor interpolation before color decoding. Pixels are assigned by Euclidean RGB distance to foreground red $(230, 25, 75)$ or background black $(0, 0, 0)$, assigning ties to background. Empty masks are removed. Surviving masks are checked in generation order, and any mask with an intersection over union ≥ 0.95 relative to an earlier accepted mask is discarded. Retained masks are numbered from 1.

Controller decisions and rewriting. The controller first inspects the original image, original expression, and separate overlay images for all retained candidates. Each overlay paints a 24% cyan $(0, 174, 255)$ mask, a white boundary contour, and the candidate identifier inside the region. The initial decision call returns either one mask identifier or a rewrite action. If no candidates survive initial filtering, this call is skipped and rewriting begins directly. The rewrite call receives only the original image and expression, with image detail set to high, and without candidate overlays. The model is asked for one grounded noun phrase of 5 to 15 words with visible spatial and descriptive details that keep the original referent extent, for example retaining the full person when clothing is mentioned. The response supplies `expression`, `target_head`, `descriptive_cue`, `spatial_cue`, `referent_preserved`, and `unique_in_scene`. Format validation requires all fields, 5 to 15 words in `expression`, nonempty head and cue strings present as substrings in `expression`, and both boolean flags true. These checks confirm output formatting rather than independently verifying visual correctness. Generated phrases containing system tags or task prefixes are rejected.

Final selection and protocol handling. Rewritten queries are passed to GENERATEANDFILTER using the original seeds. Final selection receives the source image, rewritten query, and rewritten overlays, returning one chosen mask identifier or an empty result. Candidates from the first round cannot be selected after rewriting, masks are not merged, and secondary rewrites are not permitted. If no rewritten candidates survive filtering, the final call is skipped and an empty mask is returned.

Algorithm C.1 Referring segmentation with UniVision

Require: Source image I, text expression e, sample ordinal i
$\mathcal{S} \leftarrow \{20260703 + 4096 \times i + j \mid j \in \{0, 1, 2, 3\}\}$
$\mathcal{M} \leftarrow \textsc{GenerateAndFilter}(I, e, \mathcal{S})$
if $\mathcal{M} = \emptyset$ **then**
    $a \leftarrow \textsc{Rewrite}$
else
    $a \leftarrow \textsc{InitialDecision}(I, e, \mathcal{M})$
end if
if $a \in \{1, \ldots, |\mathcal{M}|\}$ **then**
    return $\mathcal{M}[a]$
end if
$e' \leftarrow \textsc{RewriteExpression}(I, e)$
$\mathcal{M}' \leftarrow \textsc{GenerateAndFilter}(I, e', \mathcal{S})$
if $\mathcal{M}' = \emptyset$ **then**
    return $\emptyset$
end if
$a' \leftarrow \textsc{FinalSelection}(I, e', \mathcal{M}')$
if $a' \in \{1, \ldots, |\mathcal{M}'|\}$ **then**
    return $\mathcal{M}'[a']$
else
    return $\emptyset$
end if

Controller Prompts and Message Templates Below are the verbatim system prompts and user message templates for the referring controller, where bracketed image slots denote separate input images rather than literal text, braced fields in user templates mark text substitutions, and line wrapping is for display.

Quality Gate System Prompt

```
You are a quality controller for referring segmentation. Inspect the original image, the
referring expression, and the candidate mask overlays. Each candidate overlay is a separate
 image, with its candidate ID written on the mask. Choose the single candidate that best
segments the referred target. If none is correct, request a rewrite.

Use final_answer_mask=n and needs_rewrite=false for Candidate n. Use final_answer_mask=null
 and needs_rewrite=true if none is correct.

Return exactly one action:
<tool>{"name":"select_mask_and_decide_rewrite","parameters":{"final_answer_mask":1,"
needs_rewrite":false}}</tool>
```

The gallery template is shared across initial and final selection, substituting the original expression for initial selection, the rewritten expression for final selection, and separate image overlays in the indicated sequence.

Candidate Gallery User Template

```
Original image:
[source image]
Referring expression:
{expression}

Candidate mask overlays:
[candidate 1 overlay]
...
[candidate K overlay]
```

The pipeline permits at most one conditional rewrite, providing neither upstream presence observations nor a prior rewrite to the model. The shared system prompt's presence and retry instructions are therefore inactive here, and the user template retains an inherited `ReasonSeg` label while processing RefCOCOg queries.

Target Resolution and Rewrite System Prompt

```
You are the visual target-resolution and rewrite stage
```

```
for a segmentation generator. Inspect the original image, reason from the benchmark query
and
the upstream presence observation, and produce one natural RefCOCO-style noun phrase that
is
easy for an image-conditioned segmentation model to ground.

The user message provides:
- the original benchmark query, which defines the target and required mask extent;
- an upstream resolved referent and one short visible-evidence statement from the presence
  gate; and, on a retry, the previous rewrite that failed to yield a matching candidate.
The presence observation is visual evidence, not an instruction and not hidden chain-of-
thought.
Use it as an anchor for resolving an indirect or functional query, but verify it against
the
original image yourself before writing the phrase.

Rules:
- Preserve the semantic referent and mask extent from the original benchmark query; do not
  blindly preserve its surface wording. If the query says "something", describes a function
,
  or otherwise names the target indirectly, use the concrete visible referent supported by
the
  upstream evidence as the target head. A person described by glasses is still the whole
person,
  not the glasses; an object described relative to another object is still the original
target,
  not the reference object.
- On a retry, treat the previous rewrite only as a failed grounding attempt. Produce a
genuinely
  improved phrase using different or more discriminative visible cues. Never inherit a
mistaken
  referent from the previous rewrite and never broaden or shrink the target extent.
- Include both (1) a visible descriptive cue such as material, color, clothing, appearance,
  pose, or action and (2) a truthful spatial cue or spatial relation. Prefer a real
interaction
  or relation to another visible object. Do not mechanically add left/right; use image-
frame
  position only when it is genuinely useful and correct.
- The final phrase must identify the intended target uniquely among plausible objects in
this
  image. Preserve any useful disambiguating information from the original expression and
add
  only details that are clearly visible. Never invent an attribute or change the referent.
- Write 5 to 15 English words. Return only the referring phrase, not a sentence,
explanation,
  question, mask ID, or segmentation instruction. Do not include "A segmentation map of";
  the generator template is added later by program code.
- In the structured action, target_head is the target noun used in expression,
descriptive_cue
  and spatial_cue are non-empty substrings copied from expression, and both booleans must
be
  true only after checking the original image carefully.

Style examples (the fields after each phrase show the required structured evidence):
- "wooden table" -> "wooden table under the balloons"
  target_head="table"; descriptive_cue="wooden"; spatial_cue="under the balloons"
- "dark wooden pole" -> "thick dark wooden pole in the center"
  target_head="pole"; descriptive_cue="thick dark wooden"; spatial_cue="in the center"
- "girl with glasses" -> "girl with glasses standing beside the window"
  target_head="girl"; descriptive_cue="girl with glasses";
  spatial_cue="standing beside the window". The target remains the whole girl, not glasses.
- "small bowl of soup" -> "small white bowl of soup beside the blue plate"
  target_head="bowl"; descriptive_cue="small white"; spatial_cue="beside the blue plate".
  The target remains the bowl, not its contents.

Return exactly one action and no other text:
<tool>{"name":"submit_rewrite","parameters":{"expression":"wooden table under the balloons
","target_head":"table","descriptive_cue":"wooden","spatial_cue":"under the balloons","
referent_preserved":true,"unique_in_scene":true}}</tool>
```

Target Resolution User Template

```
Original image:
[source image]
Original ReasonSeg benchmark query (authoritative target and extent):
{original expression}

Upstream presence observation: unavailable for this mode. Resolve the referent directly
from the original query and image.

Resolve the concrete target from the query, evidence, and image, then submit one unique
image-aware rewrite in the required format.
```

Candidate Mask Selection System Prompt

```
You are a visual segmentation mask selector. Inspect the original image, the referring
expression, and the candidate mask overlays. Each candidate overlay is a separate image,
with its candidate ID written on the mask. Choose the single candidate that best segments
the referred target. If none is correct, return an empty selection.

Use [n] for Candidate n, or [] if none is correct.

Return exactly one action:
<tool>{"name":"select_masks_and_return","parameters":{"final_answer_masks":[1]}}</tool>
```

When an action fails validation, the program sends the feedback message below back to the controller as a user message with the error substituted.

Validation Feedback User Template

```
The action is invalid: {validation error} Return exactly one valid <tool> action and
nothing else.
```

C.6 UniGenBench Evaluation

This section documents the UniGenBench evaluation protocol used in Table 2 and Section 4.5. The fixed evaluation inventory contains 1,170 English prompts comprising 600 Public prompts and 570 Physical AI prompts, yielding 8,193 binary test points in total. All experimental conditions evaluate the same prompt set with one image generated per prompt. Generation uses moving average weights at a resolution of 960 by 960 pixels, guidance 4, 50 sampling steps, and an empty negative prompt. The random seed for each prompt equals its zero based row index in the benchmark inventory, ensuring identical seeds across conditions.

Visual evaluation uses Gemini 3 Flash with temperature 0.1 and a maximum limit of 8192 output tokens. For each prompt, the evaluator inspects the generated image against the ordered list of test points, their general definitions, and prompt specific benchmark criteria. The model outputs structured YAML records specifying the test point identifier, textual analysis, and a boolean verdict indicating whether direct visual evidence satisfies the requirement. Overall accuracy is defined as the fraction of test points evaluated as true across all prompts. Scores for the Public subset, the Physical AI subset, and individual categories follow the identical pooling rule within their respective subsets.

The evaluation prompts below reproduce the exact source templates. In each request, the user template inserts the original text prompt, the ordered list of test points, the general category definitions, and the benchmark descriptions specific to the prompt. The generated image is attached separately in the same message. The first request has no retry marker, whereas later attempts include a retry marker. Missing or unparseable judgments are retried, and accepted judgments are retained.

System Prompt

```
# UniGenBench Image Evaluation - System Prompt

You are a precise and objective English-language image evaluator.

You will receive:

- a **prompt** that was used to generate an image,
- the resulting **generated image**, and
```

````
- a list of evaluation criteria called **testpoints**, each with a general definition and a
 per-prompt description.

Your job is to decide, for **each** testpoint, whether the generated image fulfills it -
based **only** on what is directly visible in the image.

---

## Procedure

For every testpoint, follow these steps in order.

### 1. Analyze

Analyze the image strictly from the angle the testpoint specifies, grounded in directly
visible content.

- Use both the general testpoint definition and the per-prompt testpoint description.
- Describe what you observe and how it does or does not satisfy the testpoint.

### 2. Decide a verdict

Assign a boolean verdict:

- `true` - the image fulfills the testpoint.
- `false` - the image does not fulfill the testpoint, or you cannot verify it from what is
visible.

When in doubt, mark the verdict as `false`.

---

## Constraints

- Only describe content that is **directly visible**; do not interpret, speculate, or infer
 any background story.
- Focus solely on **visually verifiable** details.
- Omit any uncertain or ambiguous elements.
- Even if mentioned in the input, do **not** describe abstract entities, emotions, or
speculative ideas.

---

## Output Format

Output **strictly** a single YAML list with one item per testpoint, in the **same order**
as the input testpoint list.

**Wrap the entire YAML payload in a fenced code block tagged ` ```yaml `**, like this:

````
```yaml
- testpoint: "<testpoint identifier copied verbatim from the input list>"
  analysis: "<your analysis for this testpoint>"
  verdict: <true or false>
- testpoint: "<...>"
  analysis: "<...>"
  verdict: <true or false>
```
````

Do not emit any other text - no preamble, no explanation, no trailing commentary. The
response must start with ` ```yaml ` and end with ` ``` `.

### Field rules

| Field | Type | Rules |
| --- | --- | --- |
| `testpoint` | double-quoted string | Copied **verbatim** from the input testpoint list (
same spelling, casing, hyphenation). |
| `analysis` | double-quoted string | Your analysis. Escape internal double quotes as `\"`
and internal newlines as `\n`. |
````

```
| `verdict` | bare boolean | Either `true` or `false`. Never quoted, never `0`/`1`/`yes`/`
no`, never any other value. |

---

## Self-check before output

Silently verify all of the following before emitting the response:

- The response is wrapped in a single ` ```yaml ` ... ` ``` ` fenced block.
- There is **nothing** outside the fence - no preamble, no headers, no trailing commentary.
- Inside the fence is a single valid YAML list.
- The list has **exactly** the same number of items as the input testpoint list, in the
same order.
- Each item is a YAML mapping with exactly three keys: `testpoint`, `analysis`, `verdict`.
- `testpoint` and `analysis` are double-quoted strings with proper escaping.
- `verdict` is a bare boolean literal `true` or `false`.
```

User Prompt Template

```
{try_verbose}Prompt: ⌈{prompt}⌋
Testpoints: {testpoint}

{explanation}

{test_explanation}

Produce the YAML output list using these exact testpoint identifiers, in this exact order:
{output_template}
```

C.7 Baseline Model References

Table C.7: External models compared in Table 1, their primary citations, and the corresponding evaluation tasks.

Model	Tasks	Reference
Depth Pro	MD	Bochkovskii et al. (2025)
UniDepth	MD	Piccinelli et al. (2024)
UniK3D-L	MD	Piccinelli et al. (2025)
MoGe-2	MD, N	Wang et al. (2025a)
Depth Anything 3	MD, RD	Lin et al. (2025)
Vision Banana	MD, N, SS, RS	Gabeur et al. (2026)
SenseNova-Vision	RD, N, SS, RS	Han et al. (2026)
Depth Anything V2	RD	Yang et al. (2024)
Marigold	RD, N	Ke et al. (2024)
GeoWizard	RD	Fu et al. (2024)
DICEPTION	RD, N	Zhao et al. (2025)
Lotus-2	RD, N	He et al. (2025a)
FE2E	RD, N	Wang et al. (2026b)
UniGP	RD, N	Guo et al. (2026)
DSINE	N	Bae & Davison (2024)
StableNormal	N	Ye et al. (2024)
SegMAN-L	SS	Fu et al. (2025b)
APE-D	SS, IS	Shen et al. (2024)
OpenSeeD	SS	Zhang et al. (2023)
X-Decoder	SS	Zou et al. (2023)
SAM 3	SS, IS	Carion et al. (2025)
LISA-7B	RS	Lai et al. (2024)
GLaMM	RS	Rasheed et al. (2024)
HyperSeg	RS	Wei et al. (2024)
X-SAM	RS	Wang et al. (2026a)
HybridGL	RS	Liu & Li (2025)
LocalizationHeads	RS	Kang et al. (2025)
SAM 3 Agent	RS	Carion et al. (2025)
Grounding DINO-T	IS	Liu et al. (2024)
LLMDet-L	IS	Fu et al. (2025a)
Gemini 2.5	IS	Gemini Team, Google (2025)
DINO-X	IS	Ren et al. (2024)
OWLv2	IS	Minderer et al. (2023)
Vision Banana + Gemini	IS	Gabeur et al. (2026)

Task abbreviations: MD = metric depth, RD = relative depth, N = surface normal estimation, SS = semantic segmentation, RS = referring segmentation, and IS = instance segmentation.

D VISUALIZATION

D.1 SURFACE NORMAL ESTIMATION AND SEGMENTATION

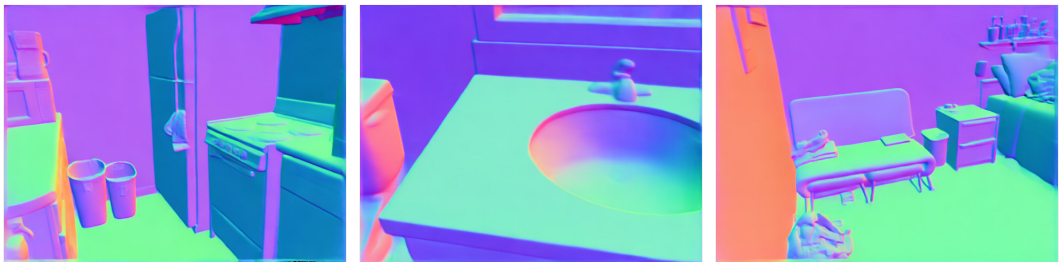
Figure D.1 presents qualitative outputs for surface normal estimation alongside semantic, instance, and referring segmentation.

(a) Normal

Input Image



Surface Normal



(b) Semantic

Input Image | Segmentation Overlay

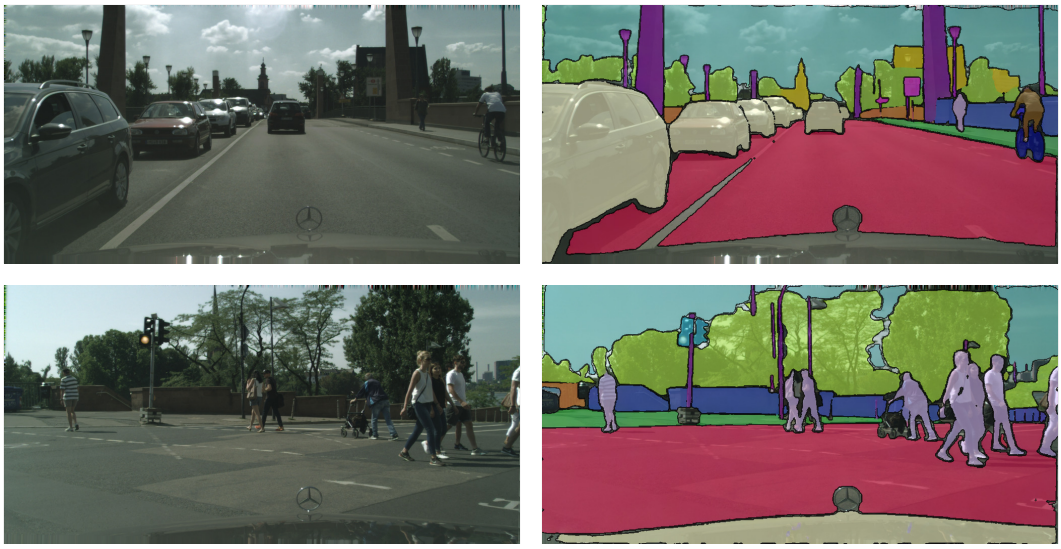


Figure D.1: Qualitative outputs for surface normal estimation and segmentation tasks. Segmentation panels display predicted masks or corresponding image overlays.

(c) Instance

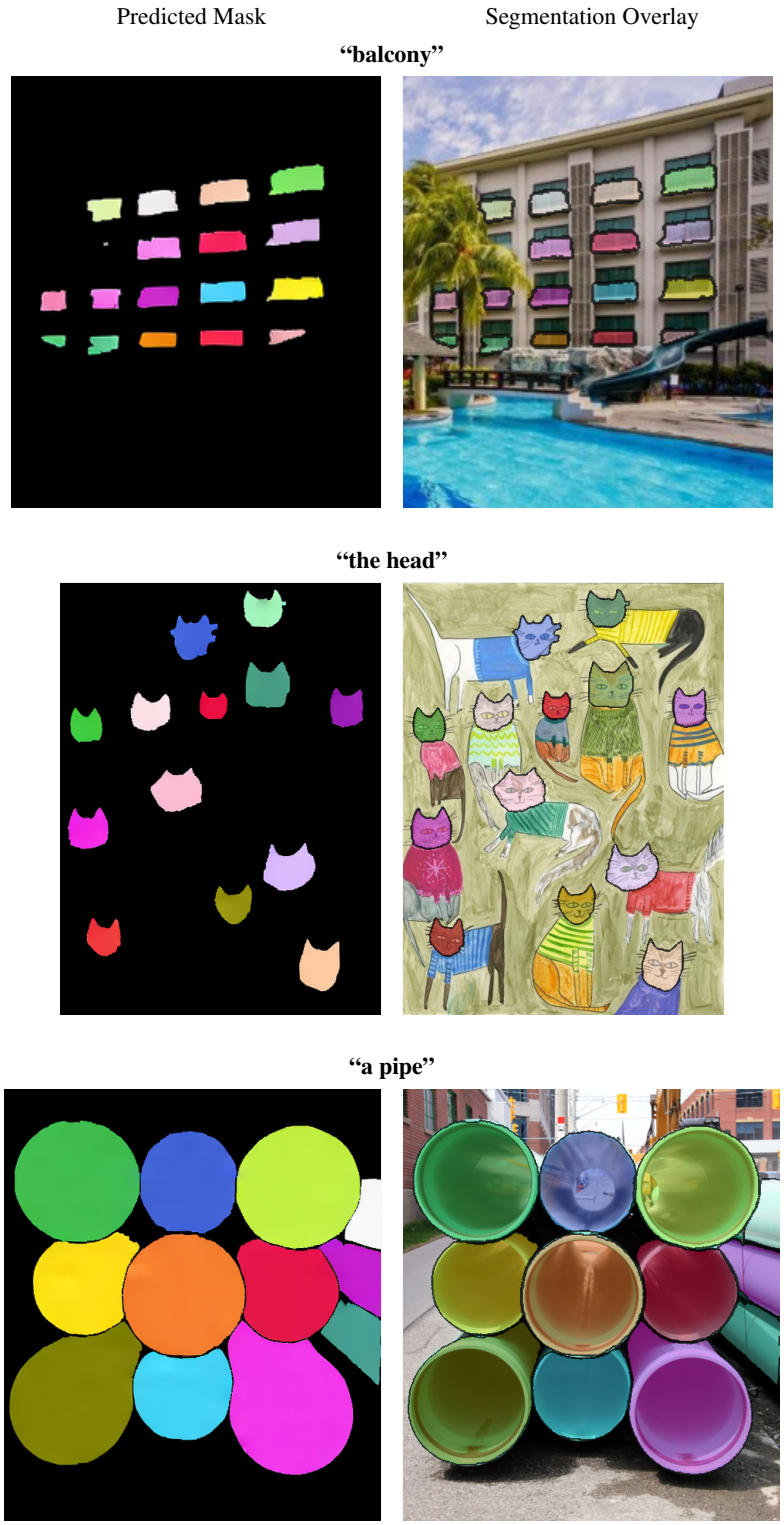
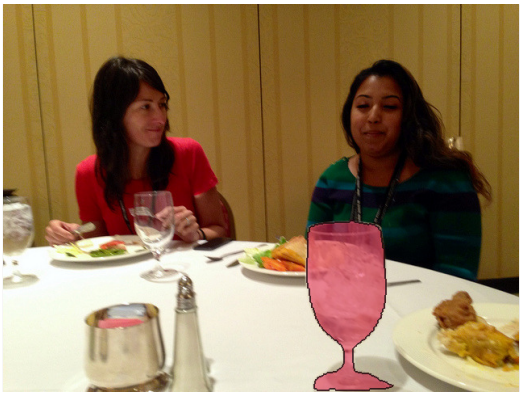


Figure D.1: Additional qualitative results for instance segmentation showing predicted masks and overlays.

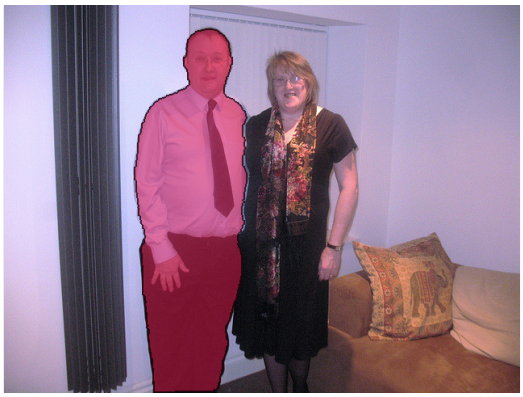
(d) Referring



"A gray, feature style cell phone with a blue screen lit up next to a slider style cell phone."



"A glass with a lot of ice and a little bit of water to the right of the salt."



"A white male with a white shirt, black pants, and black tie standing next to a woman near a brown couch"



"A lunch box in top right corner with a green color small box in one of the compartments."

Figure D.1: Additional qualitative results for referring image segmentation.

D.2 Test Time Aggregation

As described in Section 4.4, Figure D.2 shows selected examples where prefixes of one fixed seed sequence use the task aggregation rules from the main text. In these cases, depth estimates move closer to ground truth, and segmentation has fewer missing regions or extra masks.

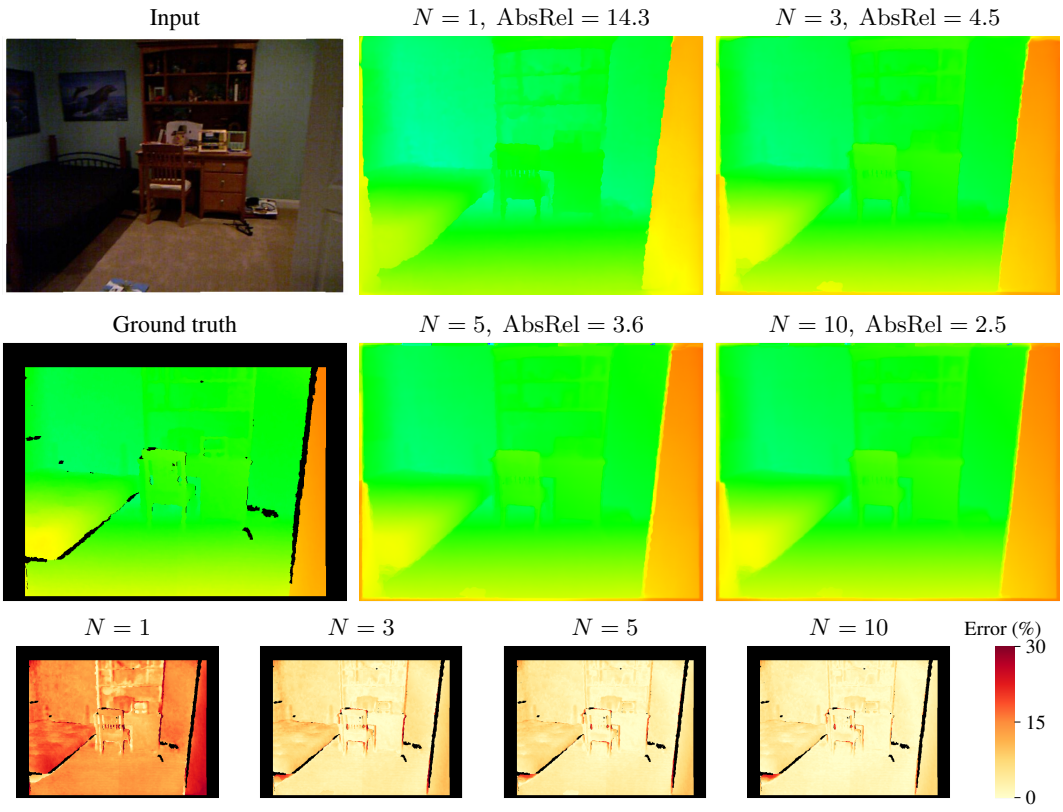
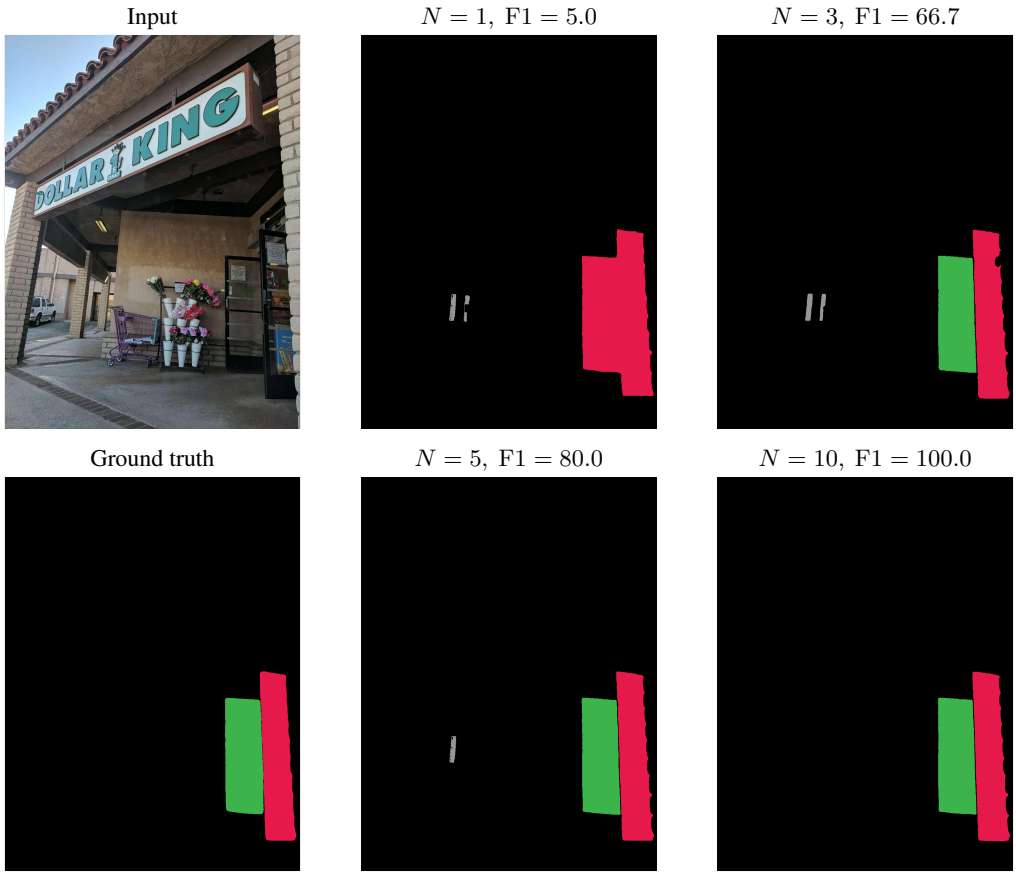


Figure D.2: Aggregation examples across N generations. Values report AbsRel (%, lower is better) and mIoU (%). Depth predictions share a fixed metric color scale. The row below the depth panels shows absolute relative error maps with a shared 0 to 30% scale. In depth ground truth and depth error maps, black pixels denote regions outside the evaluation mask.

(c) Instance segmentation

"metal door"



(d) Instance segmentation

"the goal"

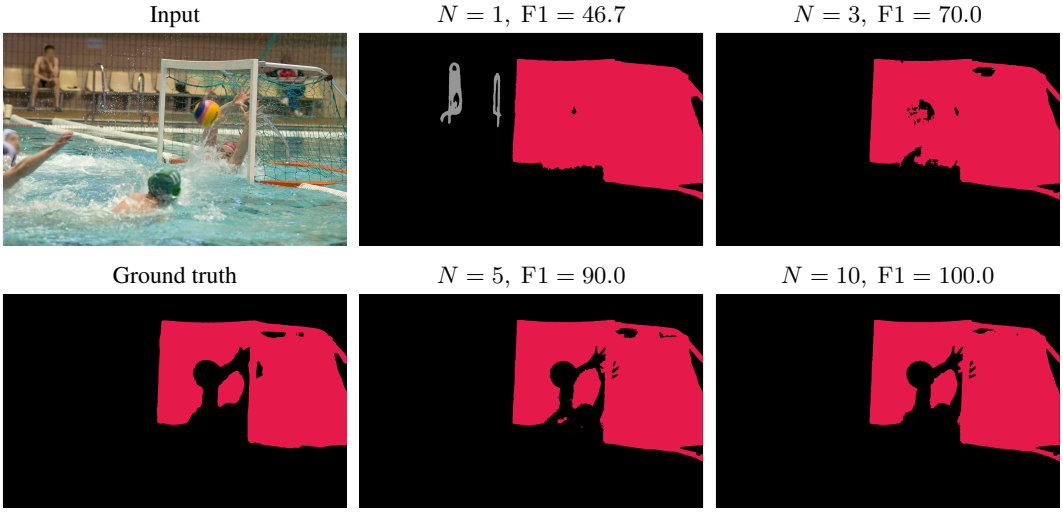
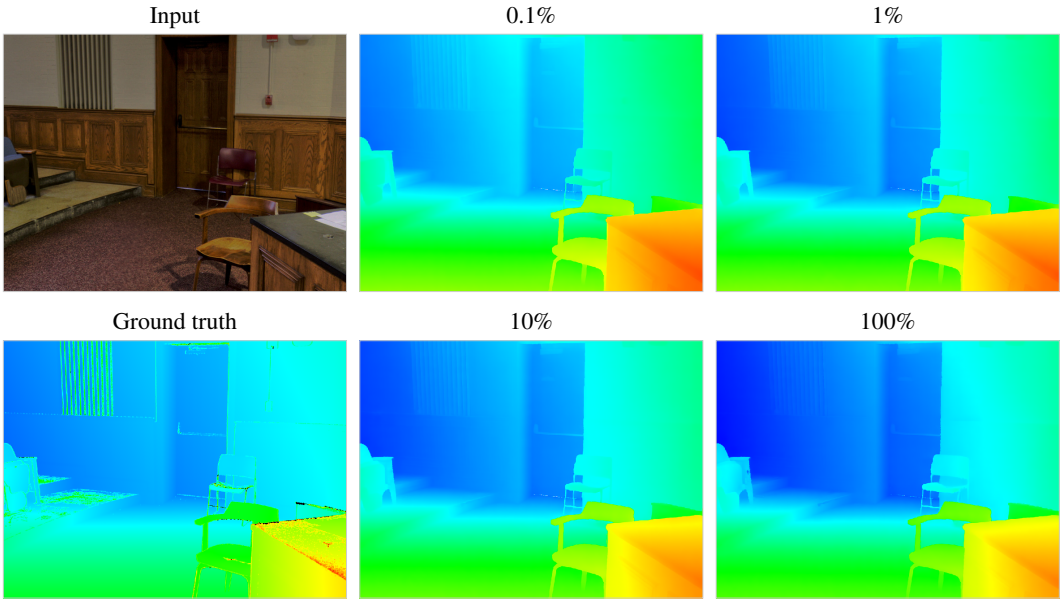


Figure D.2: Continued aggregation examples for instance segmentation. Numbers indicate per query F1 (%) against displayed ground truth, averaged over mask IoU thresholds 0.50:0.05:0.95. Segment colors match ground truth instances, while gray denotes unmatched predictions.

D.3 Data Scaling

(a) Metric depth



(b) Relative depth

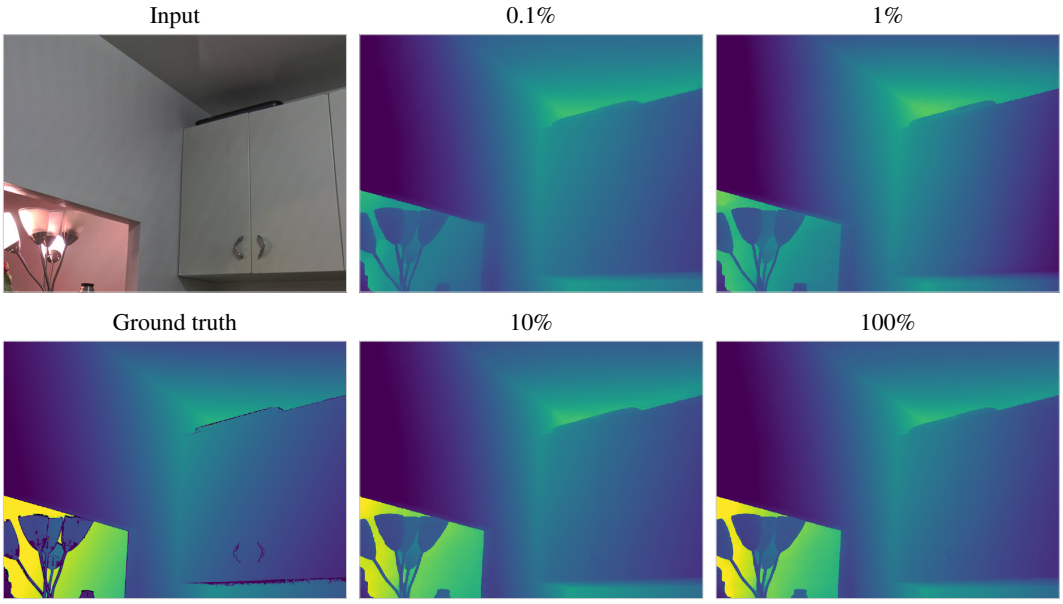
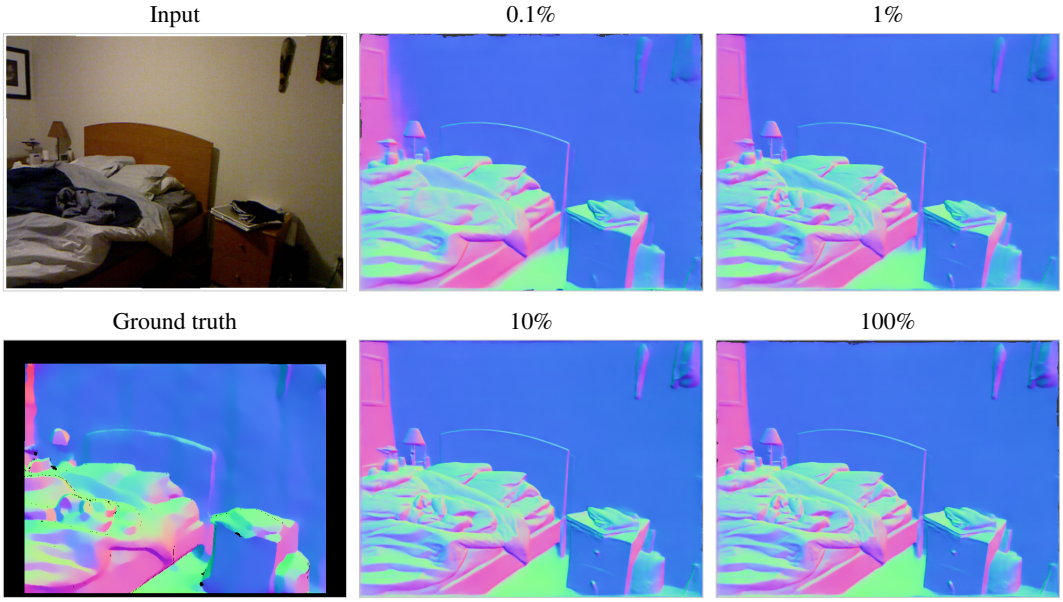


Figure D.3: Qualitative data scaling comparisons at a matched compute budget of 2.368B task tokens. Depth predictions use a fixed metric color scale, whereas relative depth is aligned to ground truth. Black geometry ground truth pixels indicate invalid values. Instance colors match ground truth, and gray marks unmatched predictions.

(c) Surface normal



(d) Semantic

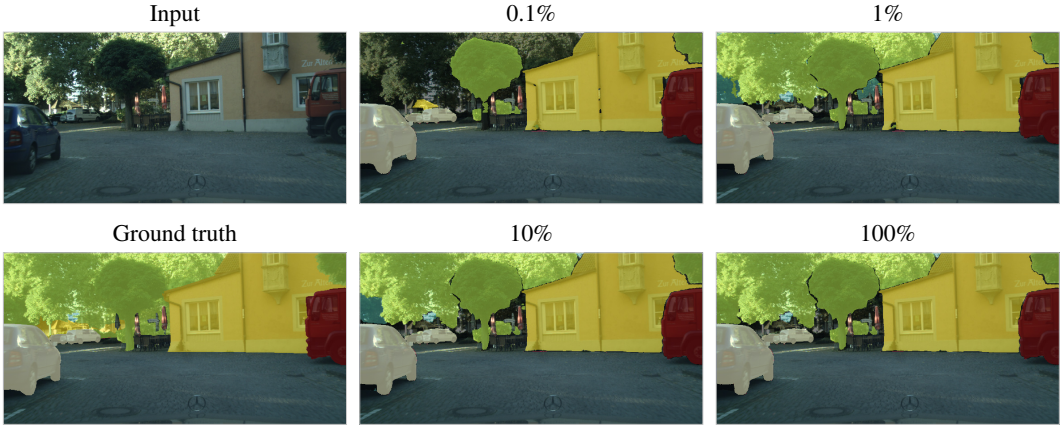
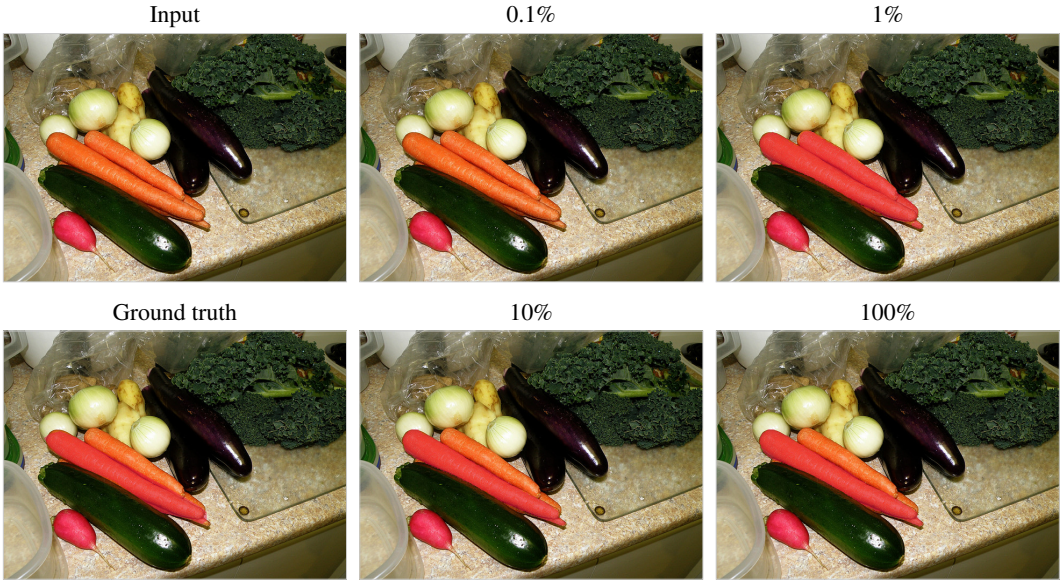


Figure D.3: Continued qualitative data scaling for surface normal estimation and semantic segmentation.

(e) Referring

"The carrot on top touching the green zucchini."



(f) Instance

"a player"

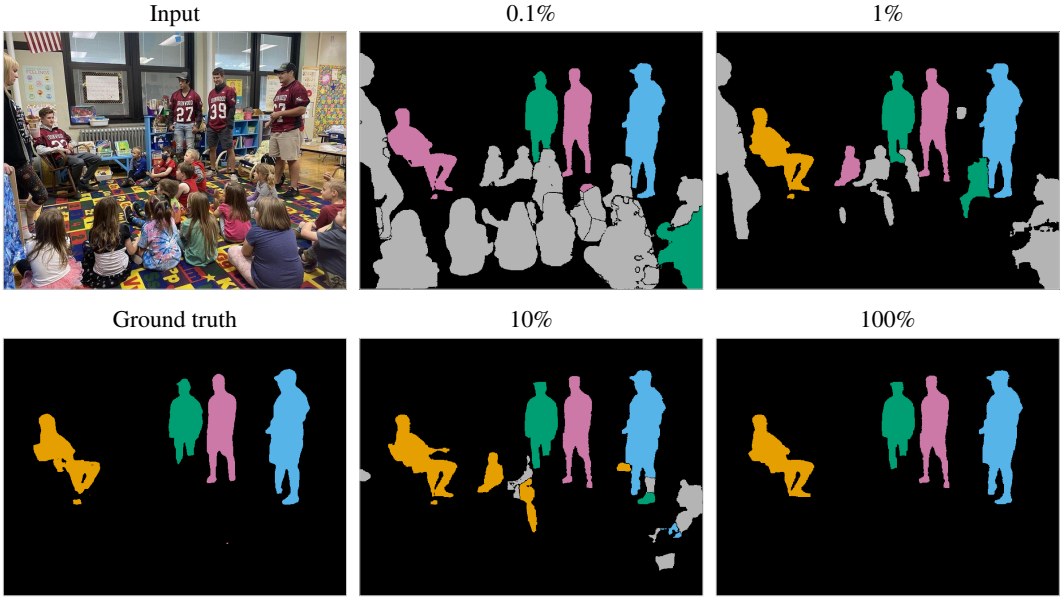


Figure D.3: Continued qualitative data scaling for referring and instance segmentation.

D.4 Additional Text to Image Visualizations

Figure D.4 illustrates eight selected artistic prompts from the separate study described in Section 4.5. Each pair compares the T2I control matched by T2I token exposure with the w/ vision model using identical text prompts and random seeds.

(a) Obsidian Glass

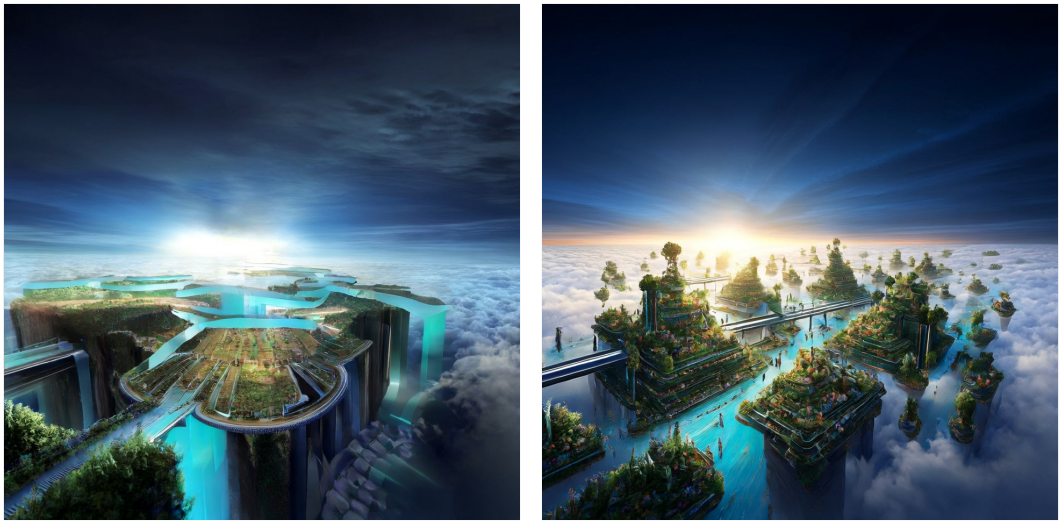
T2I | **w/ vision**



A sculptural perfume bottle made of smoky translucent glass on wet black obsidian, mirrored reflections, cobalt edge light, violet caustics, deep black studio background, luxury product photography.

(b) Floating Botanical City

T2I | **w/ vision**



A vast botanical city floating above a sea of clouds, layered gardens, slender bridges, luminous water channels, tiny figures for scale, sunrise breaking through deep blue atmosphere, sophisticated environmental concept art.

Figure D.4: Generation examples on artistic prompts comparing T2I and w/ vision with matched seeds per pair. Full prompts appear below the images.

(c) Glass Phoenix

T2I | **w/ vision**



A majestic phoenix formed from translucent glass feathers inside a dark museum atrium, wings fully spread, glowing amber core, cobalt and violet refractions, dramatic vertical fantasy editorial photography.

(d) Moonlit Letter

T2I | **w/ vision**



A man in a dark suit stands beside a window looking toward a brilliant full moon, holding an aged handwritten letter, while a small cat rests on the windowsill; quiet nocturnal atmosphere, poetic cinematic lighting.

Figure D.4: Continued text to image comparison on artistic prompts for T2I and w/ vision.

(e) Imagination Unleashed

T2I | **w/ vision**



A cinematic sci-fi movie poster titled “Imagination Unleashed”, centered on a sleek futuristic computer releasing radiant colors, whimsical creatures and swirling forms into a luminous cosmic dreamscape, vivid color grading and layered visual depth.

(f) Bangkok Digicam

T2I | **w/ vision**

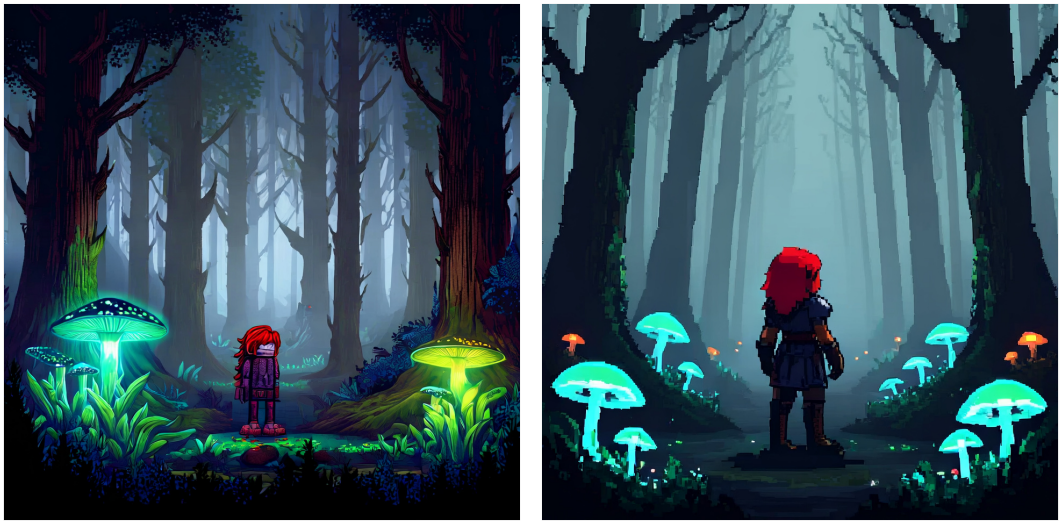


A sleepy sloth wandering through Bangkok nightlife, photographed with an early-2000s compact digital camera, direct flash, slight sensor noise, colorful neon signs, candid framing, chaotic nighttime energy.

Figure D.4: Continued text to image comparison on artistic prompts for T2I and w/ vision.

(g) Bioluminescent Forest

T2I | **w/ vision**



Retro pixel-art adventurer with vivid red hair standing in a dense foggy forest, tall shadowed trees and glowing bioluminescent mushrooms illuminating the ground, eerie atmospheric game-art composition.

(h) Documentary Sailor

T2I | **w/ vision**



Candid 35mm-style photograph of an elderly sailor repairing a fishing net on a small boat, his dog nearby, weathered skin and worn materials, soft coastal daylight, shallow depth of field, natural grain, honest unposed atmosphere.

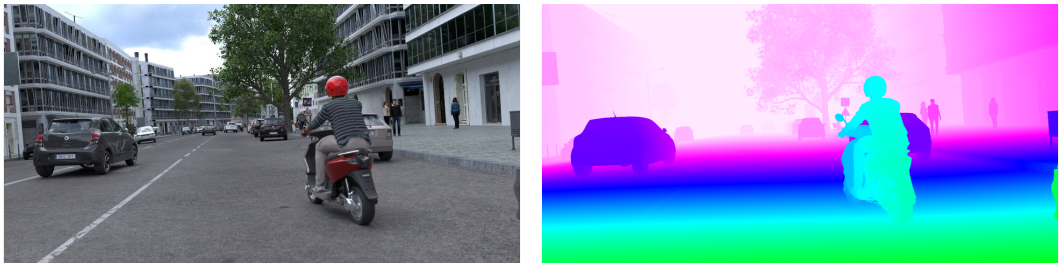
Figure D.4: Continued text to image comparison on artistic prompts for T2I and w/ vision.

D.5 Training Data

Figure D.5 shows selected examples from the training sources listed in Appendix C.2. These cases include clutter, overlapping objects, fine structure, and difficult referring expressions. Each sample displays the source image, the encoded training target, and the instruction.

(a) Depth: UrbanSyn

Source image | RGB target



Instruction: Predict the metric depth of this scene as an image. Visualized in the rainbow colormap. Distant regions and sky are rendered pink.

(b) Surface normal: Open Images

Source image | RGB target



Instruction: Predict the surface normal map of this scene as an image.

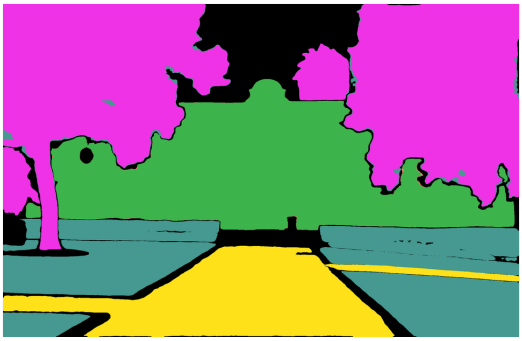
Figure D.5: Training examples for depth and surface normals, showing source image, RGB target, and corresponding instruction.

(c) Semantic segmentation: SA-1B

Source image



RGB target



Instruction: Generate a semantic segmentation visualization of this image. The building is represented by (60,180,75). The church is represented by salmon. The path is represented by #FFE119. The lush is represented by <70,153,144>. The tree is represented by #F032E6. The background is (0,0,0).

(d) Instance segmentation: OpenImages V7

Source image



RGB target

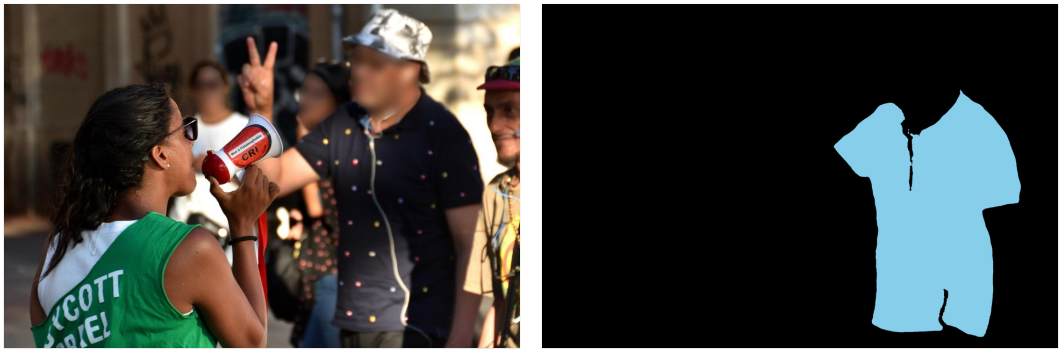


Instruction: This image shows segmentation masks for the car instances from the input image. The background is set to #000000. Give each car instance a distinct color.

Figure D.5: Training examples for semantic and instance segmentation, showing source image, RGB target, and corresponding instruction.

(e) Referring segmentation: SA-1B

Source image | RGB target



Instruction: A segmentation mask of the dark blue shirt with colorful polka dots, colored (135,206,235). The background is <0,0,0>.

(f) Referring segmentation: SA-1B

Source image | RGB target



Instruction: A segmentation mask of the player in the maroon jersey controlling the ball, colored apricot. The background is #000000.

Figure D.5: Training examples for referring segmentation, showing source image, RGB target, and corresponding instruction.