PARTITION EVERYTHING: WHOLE-IMAGE SEGMENTATION BY PALETTE-CONDITIONED FLOW MATCHING

Anonymous authors
Paper under double-blind review

ABSTRACT

Decomposing an image into its constituent objects and parts provides a fundamental representation for scene understanding and image editing. Yet whole-image segmentation remains cumbersome with promptable models such as SAM, requiring dense point prompting followed by heuristic merging and suppression of overlapping, duplicate masks. We introduce Partition Everything Model (PEM), which formulates whole-image partitioning as palette-conditioned image generation. Our rectified-flow model learns to repaint each region with the mean color of a supplied continuous palette field, representing the partition as an image without prescribing semantic labels or the number of regions. To distinguish regions at different spatial scales, we condition the model on two complementary palettes and jointly predict their corresponding color maps, combining broad spatial variation with local color contrast. Together, these maps provide a six-dimensional representation from which we extract an exhaustive, non-overlapping partition. Furthermore, we introduce annotated-region conditioning to learn from incomplete annotations while predicting complete image partitions at inference. Across seven benchmarks containing 12,138 images and 117,300 instances, PEM outperforms SAM3 by 15.57 percentage points in instance-weighted mIoU (55.27% vs. 39.70%), while predicting all regions jointly in a single forward pass.

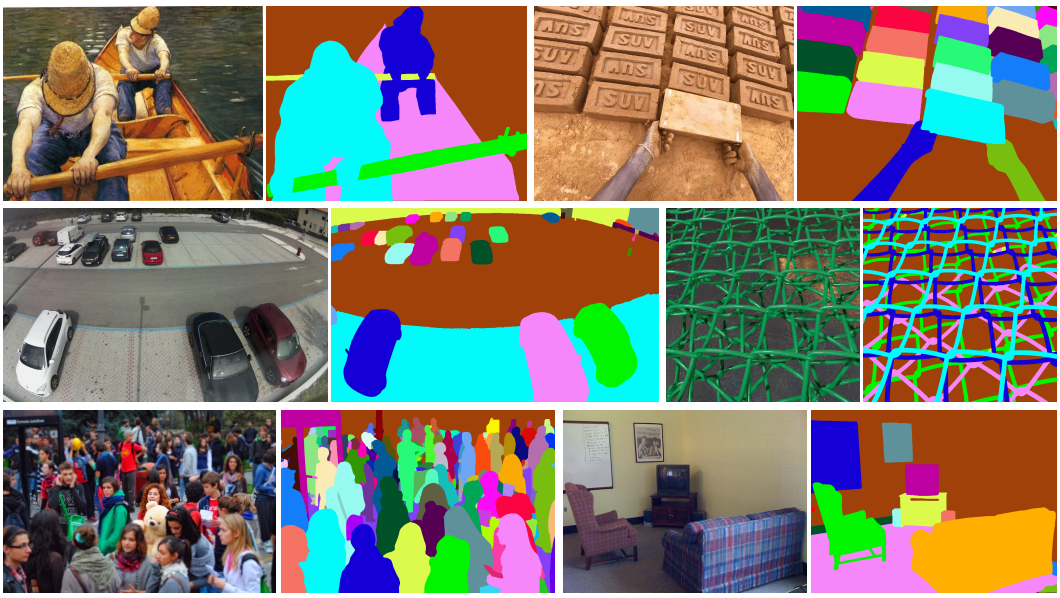

Figure 1: **Partition Everything.** Our palette-conditioned flow model predicts all regions jointly in a single forward pass, without prompting individual objects. The resulting partitions assign every pixel to exactly one region. The examples illustrate separation of visually similar objects and recovery of thin structures across artwork, crowded scenes, and indoor environments. Each input is paired with its predicted partition, with colors indicating distinct regions.

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.

1 INTRODUCTION

Image editing, robotic manipulation, autonomous navigation, and scene understanding benefit from decomposing an image into distinct objects and regions. Object masks allow edits to be confined to selected areas (Liu et al., 2024), help robots grasp unfamiliar objects (Xie et al., 2021), and support navigation toward semantic goals (Chaplot et al., 2020). A complete partition also captures the surrounding scene layout, providing a representation for holistic scene understanding (Kirillov et al., 2019). Whole-image segmentation seeks such a representation by assigning every pixel to exactly one region, without restricting those regions to a predefined class vocabulary.

The SAM family (Kirillov et al., 2023; Ravi et al., 2025; Carion et al., 2026) provides general-purpose segmentation through prompting. Applying these models to an entire image typically involves prompting a dense grid of points and collecting the resulting masks. These masks often overlap, capture the same object repeatedly, or represent both an object and its constituent parts. Confidence filtering and non-maximum suppression reduce this redundancy, but do not ensure that every pixel is assigned to exactly one region. Recovering a partition therefore requires selecting among these masks, resolving their overlaps, and assigning uncovered pixels. Our goal is to predict the regions jointly and recover their boundaries as a single image partition.

A natural way to represent a partition is to paint each region a distinct color (Khangaonkar & Pirsiavash, 2026). This represents a variable number of regions in a fixed-dimensional image. The color assignment is not unique, however, as different color maps can describe the same partition. With this in mind, we introduce Partition Everything Model (PEM), a framework for whole-image segmentation through palette-conditioned flow matching. Our approach assigns each region the mean color of a spatially varying palette over its pixels and uses the resulting color map as the training target. Conditioned on the palette, the model learns a flow in latent space from the input image to this target map.

In our formulation, the palette must provide enough color contrast to distinguish regions of different sizes and locations. A slowly varying palette may assign similar mean colors to nearby regions, while rapid variation can average out within large regions. We therefore use two complementary palettes, one with broad spatial variation and one with finer local variation (Figure 2), and train the model to predict both color maps jointly. Their combined representation helps distinguish regions that receive similar colors in either map alone. A fixed region-extraction procedure then converts the predictions into a complete, non-overlapping partition without specifying the number of regions in advance.

Training for whole-image partitioning is also limited by the availability of complete annotations. Many segmentation datasets label only selected objects within each image. To make use of these datasets, we introduce annotated-region conditioning, which specifies the area to be partitioned through a spatial mask. During training, this mask indicates the annotated area, allowing the model to learn from partial supervision. At inference, it covers the entire image. We can therefore train on predominantly partial annotations, supplemented by a small number of fully annotated images. Figure 1 shows the resulting partitions across diverse scenes and visual styles, illustrating the separation of repeated objects and the recovery of thin structures.

Contributions.

- We introduce PEM, a flow-matching model conditioned on two complementary palettes that jointly predicts their encoded partition targets in a single forward pass. The two maps help distinguish regions at different spatial scales and are used to recover a complete, non-overlapping partition through region extraction.
- We propose a palette-based partition encoding that assigns each region the mean palette color over its pixels. This represents a variable number of regions in a three-channel image and defines explicit training targets independent of region label ordering.
- PEM achieves the highest mIoU among evaluated methods across the benchmarks. Over 117,300 annotated instances, it exceeds SAM3 by 15.57 percentage points in instance-weighted mIoU (55.27% vs. 39.70%).

2 Related Work

Promptable and whole-image segmentation. The SAM family (Kirillov et al., 2023; Ravi et al., 2025; Carion et al., 2026) supports promptable segmentation of images and videos. For whole-image segmentation, spatial prompting can be applied over a dense grid, followed by filtering to reduce redundant and overlapping masks. Semantic-SAM (Li et al., 2024) explicitly predicts masks at multiple granularities, including objects and parts. Other methods learn from automatically generated masks. CutLER (Wang et al., 2023c) discovers pseudo-masks from self-supervised features, while SOHES (Cao et al., 2024) combines feature clustering with teacher–student refinement. UnSAM (Wang et al., 2024b) uses divide-and-conquer grouping to generate multi-granular pseudo-masks for training. Our method jointly predicts color maps encoding all regions, from which a complete partition is recovered without point prompting or duplicate-mask suppression.

Segmentation from diffusion representations. DiffSeg (Tian et al., 2024) merges diffusion self-attention maps into segmentation masks, while M2N2 (Karmann & Urfalioglu, 2025) uses self-attention for training-free, point-based segmentation. Diffuse2Seg (Hümmer et al., 2026) propagates a grid of point prompts through diffusion self-attention to generate multi-granular pseudo-masks, then trains a segmentation model on these labels. Rather than extracting masks from pretrained attention, we train a flow model to generate a representation of the whole-image partition.

Segmentation as image generation. Painter and SegGPT (Wang et al., 2023a;b) represent dense predictions as images and support inference from visual examples. Diffusion-based methods generate segmentation maps through image-conditioned denoising (Amit et al., 2021; Ji et al., 2023), including panoptic prediction in pixel and latent spaces (Chen et al., 2023; Van Gansbeke & De Brabandere, 2024). InstructDiffusion (Geng et al., 2024) expresses segmentation through instruction-conditioned image manipulation, while GenMask (Yang et al., 2026) directly generates binary masks within an image-generation framework.

Flow matching has also been applied to segmentation. SemFlow and SymmFlow (Wang et al., 2024a; Caetano et al., 2025) connect semantic segmentation and image synthesis through flows between their representations. FlowSDF (Bogensperger et al., 2025) generates signed distance fields for medical segmentation, while FlowDIS (Sargsyan & Navasardyan, 2026) learns an image-to-mask flow for language-guided dichotomous segmentation.

UniGS (Qi et al., 2024) generates segmentation colormaps using a fixed location-aware palette and recovers masks through recursive clustering. It assigns one of 124 foreground colors according to the grid cell containing each region's centroid, so regions whose centroids fall in the same cell receive the same target color. Gen2seg (Khangaonkar & Pirsiavash, 2026) instead learns RGB embeddings by encouraging uniform colors within instances and separation between them. Our palette-based partition encoding averages continuous fields over each region, making target colors depend on its spatial extent without a fixed color vocabulary. PEM jointly predicts the resulting maps through flow matching conditioned on two complementary palettes.

3 Method

We first introduce our palette-based partition encoding and the complementary fields used to construct its color targets (Section 3.1). We then present PEM, which learns to predict these targets jointly through palette-conditioned flow matching (Section 3.2).

3.1 Palette-Based Partition Encoding

Let $I \in \mathbb{R}^{H\times W\times 3}$ be an RGB image defined on the pixel grid $\Omega = \{1,\dots,H\} \times \{1,\dots,W\}$. Its annotation consists of K nonempty, pairwise disjoint regions $\mathcal{R} = \{R_k\}_{k=1}^{K}$, where each $R_k \subseteq \Omega$ contains the pixels of region k. Their union $S = \bigcup_{k=1}^{K} R_k$ denotes the annotated area, with $S = \Omega$ for full annotations and $S \subsetneq \Omega$ for partial annotations.

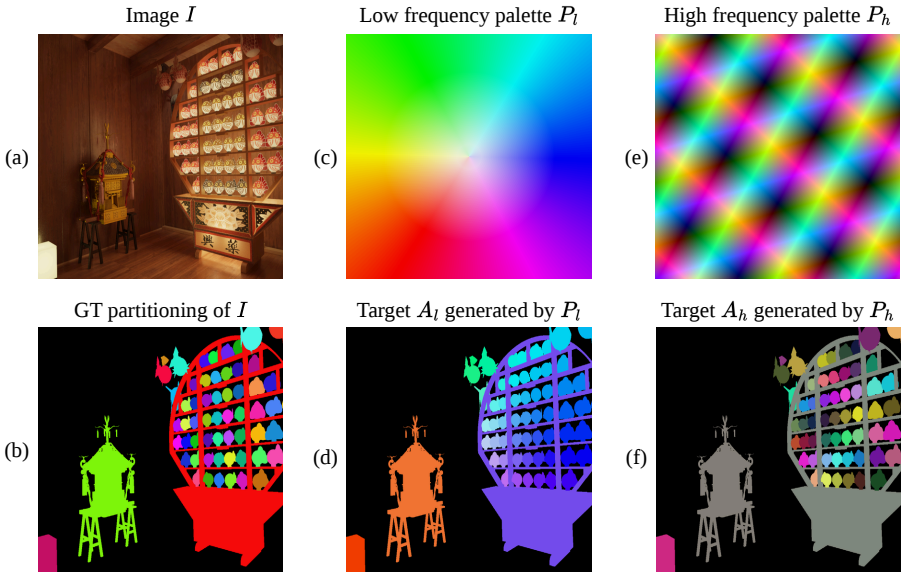

Figure 2: **Palette-based partition encoding.** (a) Input image and (b) its annotated regions, shown in arbitrary colors. The (c) low-frequency and (e) high-frequency palettes define the corresponding targets in (d) and (f). Each annotated region is assigned its mean palette color, while unannotated pixels are set to black.

Let $P : \Omega \to [0,1]^3$ be a palette field that assigns a normalized RGB color to each pixel. We encode the partition as a target map A by assigning each region its mean palette color:

$$\bar{P}_k = \frac{1}{|R_k|} \sum_{q \in R_k} P(q), \qquad A(p) = \begin{cases} \bar{P}_k, & p \in R_k, \\ \mathbf{0}, & p \notin S. \end{cases} \tag{1}$$

This encoding is invariant to region label permutations and uses a fixed number of channels regardless of the number of regions.

We use two complementary palette fields to distinguish regions at different spatial scales. Let P_l denote a low-frequency palette field with broad spatial color variation, and P_h a high-frequency field with stronger local contrast. While P_l may assign similar mean colors to nearby regions, averaging P_h over large regions can reduce color differences.

The low-frequency palette P_l forms a color wheel centered in the image (Figure 2(c)). We construct it in hue–saturation–value (HSV) space, with hue determined by angle and saturation increasing with distance from the center:

$$P_l(p) = \mathrm{HSVtoRGB}\left(\left(\frac{\vartheta_p + \varphi}{2\pi}\right) \bmod 1,\ s(r_p),\ v\right), \tag{2}$$

where (r_p, ϑ_p) are the polar coordinates of pixel p relative to the image center, $s(r_p) \in [0,1]$ specifies the radial saturation profile, and v is a fixed brightness. The function HSVtoRGB converts these HSV components into an RGB color. During training, we sample $\varphi \sim \mathcal{U}(0, 2\pi)$ to vary the hue assignment across training examples.

The high-frequency palette P_h consists of three directional triangle waves, one per RGB channel (Figure 2(e)):

$$P_{h,c}(p) = T\left(\frac{u_c^\top p}{\lambda_c} + \phi_c\right), \qquad T(a) = 2\left|a - \lfloor a \rfloor - \frac{1}{2}\right|, \tag{3}$$

where $c \in \{1,2,3\}$ indexes the color channels, u_c is a unit direction vector, λ_c is the spatial period, and ϕ_c is a random phase. The function T is a triangle wave with period one and values in $[0,1]$. We resample both palettes for each training example and construct their target maps A_l and A_h using Equation 1.

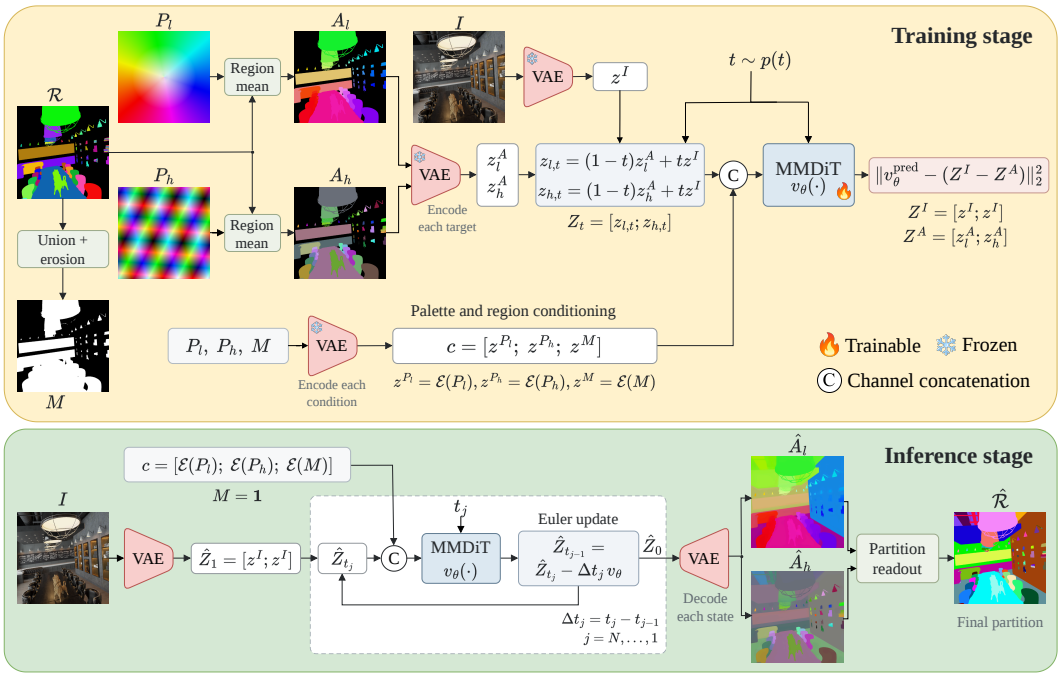

Figure 3: **Overview of PEM. Training.** We construct target maps A_l and A_h by assigning each annotated region in $\mathcal{R}$ its mean color under palettes P_l and P_h, respectively. The region-conditioning mask M is obtained by eroding the union of annotated regions. A frozen VAE encodes the image, targets, and conditioning inputs. At timestep t, we interpolate between each target latent and the image latent to form the joint flow state Z_t. The transformer receives Z_t together with the encoded palettes and region mask, and jointly predicts both velocities. Training minimizes the squared error against the concatenated target velocities. **Inference.** We set M to all ones and initialize both flow states with the image latent. Euler integration from $t = 1$ to $t = 0$ produces two latents, which the VAE decodes into color maps. We use $N = 1$ for single-step inference in our main experiments. Region extraction converts these maps into a complete, non-overlapping partition.

3.2 Partition Everything Model (PEM)

PEM learns to transform an input image into the two palette-encoded target maps defined above, conditioned on their palettes and the area to be partitioned. We describe training with dual-palette conditioning, whole-image inference, and partition readout in Sections 3.2.1–3.2.3. Figure 3 provides an overview of PEM.

3.2.1 Training with Dual-Palette Conditioning

We learn the joint flow in latent space, using two copies of the image latent as the starting state and the concatenated target latents as the endpoint.

To accommodate both dense and partial annotations, we provide a binary conditioning map M derived from the annotated area S. During training, we erode the indicator of S with a randomly sized square kernel. This gives the model a coarse indication of where annotations are available while reducing the boundary information exposed through conditioning. The target maps remain defined on the original regions, as in Equation 1, with unannotated pixels set to black.

Let $\mathcal{E}$ and $\mathcal{D}$ denote the encoder and decoder of a frozen autoencoder. We encode the image and targets as $z^I = \mathcal{E}(I)$ and $z_f^A = \mathcal{E}(A_f)$ for $f \in \{l, h\}$, and form the joint image and target states:

$$Z^I = [z^I; z^I], \qquad Z^A = [z_l^A; z_h^A], \tag{4}$$

where $[\cdot;\cdot]$ denotes channel concatenation. The conditioning is $c = [\mathcal{E}(P_l); \mathcal{E}(P_h); \mathcal{E}(M)]$, with M repeated across three channels before encoding.

Following the flow matching formulation (Lipman et al., 2023; Liu et al., 2023), we construct a linear path between the joint states:

$$Z_t = (1-t)Z^A + tZ^I, \qquad t \in [0,1]. \tag{5}$$

The corresponding target velocity is $Z^I - Z^A$. A shared transformer $v_\theta(Z_t, t, c)$ predicts the joint velocity and is optimized with

$$\mathcal{L}(\theta) = \mathbb{E}\left[\left\| v_\theta(Z_t, t, c) - (Z^I - Z^A)\right\|_2^2\right], \tag{6}$$

where the expectation is over training examples, palette sampling, mask erosion, and timesteps. The loss is evaluated over the full latent representation, including the encoded black areas outside the annotations. We concatenate Z_t and c along the feature dimension of each spatial token, preserving the token sequence length of a single image.

3.2.2 Whole-Image Inference

At inference, we generate both palettes using a fixed random seed and set $M = \mathbf{1}$ to request a partition of the entire image. We initialize the joint state as $\hat{Z}_1 = Z^I$ and integrate the predicted velocity from $t = 1$ to $t = 0$. For a schedule $1 = t_N > \cdots > t_0 = 0$, the Euler update is

$$\hat{Z}_{t_{j-1}} = \hat{Z}_{t_j} - \Delta t_j \, v_\theta(\hat{Z}_{t_j}, t_j, c), \qquad \Delta t_j = t_j - t_{j-1}, \tag{7}$$

for $j = N, \ldots, 1$. With a single integration step, this becomes

$$\hat{Z}_0 = Z^I - v_\theta(Z^I, 1, c). \tag{8}$$

We use single-step inference in our main experiments and evaluate additional integration steps in an ablation.

Finally, we split $\hat{Z}_0 = [\hat{z}_{l,0}; \hat{z}_{h,0}]$ and decode each component to obtain $\hat{A}_l = \mathcal{D}(\hat{z}_{l,0})$ and $\hat{A}_h = \mathcal{D}(\hat{z}_{h,0})$. The partition readout combines color and boundary information from both predicted maps to produce the final image partition.

3.2.3 Partition Readout

We recover a partition using watershed segmentation (Vincent & Soille, 1991) followed by color-based merging. The two predicted maps define a six-dimensional color representation $F(p) = [\hat{A}_l(p); \hat{A}_h(p)]$. We first smooth each predicted color map with a Gaussian filter, then detect a boundary band by retaining sharp gradient peaks. We thin this band and sample its skeleton, placing seeds on either side, outside the band. If the two seed locations have similar colors in F, we replace them with a single seed at the skeleton sample. Watershed forms initial regions using the stronger boundary response from the two maps at each pixel. We then merge regions according to their mean colors in F. Processing regions from largest to smallest, we assign each to the nearest color cluster within a distance threshold, or create a new cluster otherwise. We repeat merging until no further merges occur or an iteration limit is reached. Finally, components with little detected boundary support are merged into neighboring regions based on shared border length, yielding a complete, non-overlapping partition $\hat{\mathcal{R}}$.

4 Experiments

4.1 Implementation Details

Training dataset. We construct a new training dataset for image partitioning by combining 3,154 images from five sources: 1,856 from Hypersim (Roberts et al., 2021), 955 from DIS5K (Qin et al., 2022), 145 from iShape (Yang et al., 2021), 72 from Open Images (Kuznetsova et al., 2020), and 126 from MS COCO (Lin et al., 2014). Images from DIS5K, iShape, Open Images, and MS COCO are drawn exclusively from their respective training splits. For the Open Images and MS COCO subsets, we create high-quality region annotations from scratch rather than using the provided annotations. All annotations are represented as sets of regions without semantic class labels, retaining both fully and partially annotated images. The dataset contains between 1 and 993 annotated regions per

image, with a median of 9. Only 77 images (2.4%) are fully annotated, while the remaining 3,077 (97.6%) have partial annotations. The annotated regions cover 38.7% of pixels per image on average, with a median coverage of 30.1%. This allows us to study whole-image partitioning when training annotations cover only a subset of each image. During training, images and region annotations are resized to 1024×1024 using bilinear and nearest-neighbor interpolation, respectively.

Architecture and optimization. The velocity network is initialized from the 4B-parameter FLUX.2 Klein (Black Forest Labs, 2026) transformer. Its autoencoder remains frozen. Each spatial token contains a 2×2 patch of autoencoder latents, giving 128 features at an effective image stride of 16 pixels. We expand the input projection to 640 dimensions for the two flow states and three conditioning latents, and the output projection to 256 dimensions for the joint velocity. The two state blocks each receive half of the pretrained input weights, while conditioning weights are initialized to zero. The pretrained output weights are copied into both output blocks.

All transformer parameters are optimized with AdamW for 4,000 steps on eight NVIDIA A100-80GB GPUs, using a batch size of 16 per GPU. The learning rate is 10^{-4}, with a 25-step linear warmup and halving at steps 128, 512, 1024 and 2048. Training timesteps follow $\mathrm{Beta}(5, 1)$, with 15% of each batch assigned $t = 1$.

Conditioning parameters. For the low-frequency palette, we set $s(r) = \mathrm{clip}(r/r_0, 0.15, 1)$, $r_0 = 0.35 \min(H, W)$, and $v = 0.95$, with hue rotation $\varphi \sim \mathcal{U}[0, 2\pi)$. For the high-frequency palette, we scale a base period of 256 pixels by factors of 1, 1.13, and 1.29 for the three waves, respectively. Their directions are spaced at $2\pi/3$ before adding a shared rotation sampled uniformly from $[0, 2\pi)$ and independent angular perturbations sampled uniformly from $[-0.2, 0.2]$ radians. Phases are sampled independently from $\mathcal{U}[0, 1)$, and the three waves are randomly permuted across RGB channels. Both palettes are resampled each time a training example is loaded. After resizing the annotations to 1024×1024, we construct the region-conditioning map by eroding their union once with a square kernel whose width is sampled uniformly from $\{3, \ldots, 9\}$ pixels. The resulting binary map is repeated across three channels before encoding.

Inference and partition readout. We resize inputs while approximately preserving their aspect ratio, targeting $\sqrt{HW} = 1024$ and rounding both dimensions to multiples of 16. Inference uses a single Euler step unless otherwise specified. For region extraction, predicted colors are normalized to $[0, 1]$. Each RGB map is smoothed with a Gaussian kernel of standard deviation $\sigma = 1$. Boundary detection uses hysteresis thresholds of 0.005 and 0.012. From the thinned boundary mask, we select the strongest response in each occupied 12×12 cell. Seed locations are sought 5–15 pixels along either normal direction. A single seed is placed at the boundary sample when the two candidate colors differ by less than 0.012 in Euclidean distance. The color-merging threshold is 0.06. Readout parameters remain fixed across benchmarks.

4.2 Experimental setup

Datasets. We evaluate on the held-out evaluation splits of seven benchmarks, comprising 12,138 images and 117,300 annotated instances: ADE20K (Zhou et al., 2017), COCO (Lin et al., 2014), Cityscapes (Cordts et al., 2016), DRAM (Cohen et al., 2022), EgoHOS (Zhang et al., 2022), NDISPark (Ciampi et al., 2022), and iShape (Yang et al., 2021). Together, these benchmarks cover natural scenes, urban environments, artwork, egocentric interactions, parking areas, and objects with complex shapes. The evaluation includes 2,000 images from ADE20K, 5,000 from COCO, 500 from Cityscapes, 718 from DRAM, 1,124 from EgoHOS, 111 from NDISPark, and 2,685 from iShape.

Evaluation. Since the benchmarks are not fully annotated, we adapt the single-point mIoU evaluation of Khangaonkar & Pirsiavash (2026) to score annotated instances without penalizing separate predictions of unlabeled regions. For each instance, we select the pixel within its mask nearest to its centroid and retrieve the predicted region containing that point. Unassigned points receive zero IoU. For gen2seg, we use its official mask extraction at the same point. Selected masks are resized to the annotation resolution using nearest-neighbor interpolation before computing IoU against the ground truth. We report per-benchmark mIoU over annotated instances and an overall instance-weighted mean.

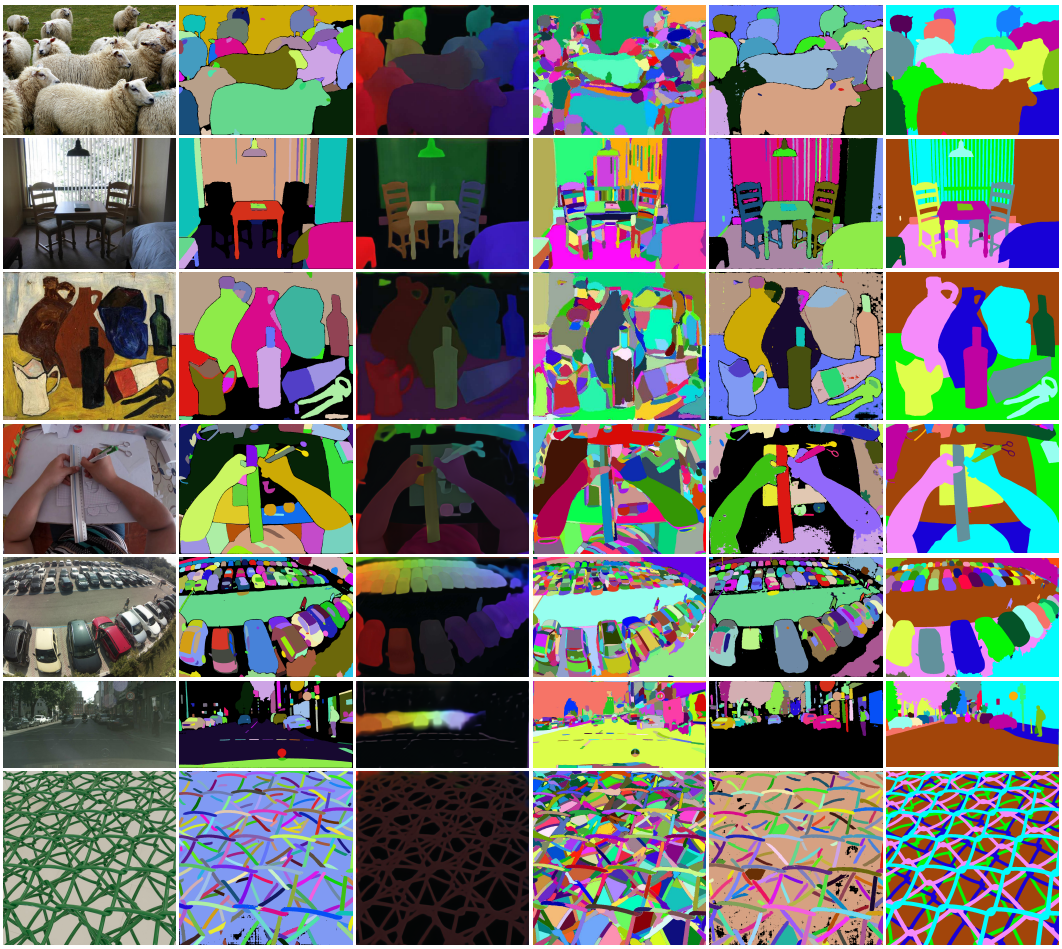

Figure 4: **Qualitative comparison.** Examples from COCO, ADE20K, DRAM, EgoHOS, NDISPark, Cityscapes, and iShape, from top to bottom. Black denotes unassigned pixels. Our method produces complete, non-overlapping partitions. Zoom in for details.

4.3 Quantitative and qualitative analysis

We compare PEM with SAM (Kirillov et al., 2023), UnSAM (Wang et al., 2024b), gen2seg (Khangaonkar & Pirsiavash, 2026) and SAM 3 (Carion et al., 2026). For SAM and SAM 3, we use their automatic mask generation pipelines with dense grids of point prompts to obtain whole-image predictions. All baselines use the authors' released implementations and pretrained checkpoints with default inference settings.

Quantitative comparison. Table 1 shows that PEM achieves the highest mIoU on all seven benchmarks. Its overall instance-weighted mIoU of 55.27% exceeds SAM3, the strongest baseline overall, by 15.57 percentage points. The corresponding gains over SAM, UnSAM, and gen2seg are 17.56, 29.11, and 34.10 points, respectively. These improvements extend to domains outside its fine-tuning distribution, including artwork (DRAM), egocentric interactions (EgoHOS), street scenes (Cityscapes), and parking scenes (NDISPark). We also evaluate zero-shot edge detection on BSDS500 (Arbelaez et al., 2011) in Appendix D.

Qualitative comparison. Figure 4 compares predictions across the evaluation benchmarks. PEM partitions both foreground objects and the surrounding scene, including areas left unassigned by SAM and SAM3, shown in black (see chairs on the second row and road on the sixth row). PEM also faithfully separates individual vehicles in the parking scene while preserving a coherent road region, and avoids several object splits produced by UnSAM in the painting (third row). In iShape

Table 1: **Quantitative comparison across seven benchmarks.** Scores are mIoU (%), with higher values indicating better performance. Overall scores are weighted by the number of annotated instances. Bold and underline indicate the best and second-best results in each column, respectively.

Method	ADE20K	COCO	Cityscapes	DRAM	EgoHOS	NDISPark	iShape	Overall
SAM	44.40	44.49	23.27	29.68	<u>57.98</u>	56.95	29.09	37.71
UnSAM	32.33	28.39	19.38	9.25	36.00	29.44	21.97	26.15
gen2seg	27.03	30.07	14.89	<u>43.91</u>	41.40	26.26	7.52	21.17
SAM3	<u>46.55</u>	<u>45.97</u>	<u>26.71</u>	41.78	49.87	<u>60.89</u>	<u>31.52</u>	<u>39.70</u>
Ours	**48.82**	**55.87**	**46.34**	**54.82**	**58.39**	**75.48**	**60.64**	**55.27**

Table 2: **Ablation of partition representation and conditioning.** All variants use single-step inference. Scores are instance-weighted mIoU (%) across seven benchmarks. Region conditioning denotes conditioning on the annotated area during training.

Target representation	Palette conditioning	Region conditioning	mIoU
Random colors	None	No	6.13
Palette-based	Low-frequency	No	41.28
Palette-based	High-frequency	No	36.77
Palette-based	Low- and high-frequency	No	49.04
Palette-based	Low- and high-frequency	Yes	**55.27**

(seventh row), PEM distinguishes many thin structures and the spaces between them, while gen2seg leaves several structures merged. Further comparisons are provided in Section E.

4.4 Ablation Studies

We examine the contributions of palette conditioning and annotated-region conditioning in Table 2. All variants use single-step inference and are evaluated across the same seven benchmarks. We report instance-weighted mIoU over all annotated instances.

Partition representation. As a baseline, we train the model with random-color targets, without palette conditioning or annotated-region conditioning. This variant achieves 6.13% mIoU and produces largely gray predictions, failing to distinguish regions. Replacing the random-color targets with our palette-based encoding and supplying the corresponding palette as conditioning raises mIoU to 41.28% with the low-frequency palette and 36.77% with the high-frequency palette. These results support defining target colors through a palette supplied to the model. Further analysis on palette-based partition encoding is provided in Sections A and B.

Complementary palettes. Combining both palettes increases mIoU to 49.04% without annotated-region conditioning, improving over the low-frequency and high-frequency variants by 7.76 and 12.27 percentage points, respectively. The improvement over either palette alone supports their complementary role in distinguishing regions.

Annotated-region conditioning. Adding annotated-region conditioning to the dual-palette model further increases mIoU from 49.04% to 55.27%, a gain of 6.22 points. Given that 97.6% of our training images are partially annotated, this result highlights the importance of specifying the annotated area during training, while requesting whole-image partitioning at inference.

5 Conclusion

We introduced PEM, a method for whole-image partitioning through flow matching conditioned on two complementary palettes. Its palette-based partition encoding assigns each region the mean palette color over its pixels, providing explicit color maps as training targets. PEM jointly predicts both maps in a single forward pass, from which a complete, non-overlapping partition is recovered. Annotated-region conditioning enables learning from predominantly partial annotations. PEM outperforms the evaluated baselines across the benchmarks, supporting palette-conditioned generation as an effective approach to whole-image segmentation.

AI USE STATEMENT

We used generative AI tools to assist with drafting and revising the manuscript, identifying related literature, clarifying mathematical notation and method descriptions, preparing figures and LaTeX code, and accelerating code implementation. The authors reviewed and verified the AI-assisted code and checked the manuscript for technical accuracy and consistency with the implementation and experiments. We take responsibility for the final content of the paper, including all claims, references, code, and artifacts.

REPRODUCIBILITY STATEMENT

The method section describes the partition representation, training objective, and inference procedure. Section 4.1 provides the dataset composition, model initialization, optimization settings, and parameters used for region extraction. To support reproducibility and further research, we will release our training and inference code, together with the curated training dataset and newly created annotations.

REFERENCES

Tomer Amit, Tal Shaharbany, Eliya Nachmani, and Lior Wolf. SegDiff: Image segmentation with diffusion probabilistic models. *arXiv preprint arXiv:2112.00390*, 2021. URL https://arxiv.org/abs/2112.00390.

Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):898–916, May 2011. ISSN 0162-8828.

Black Forest Labs. FLUX.2 [klein]: Towards interactive visual intelligence. https://bfl.ai/blog/flux2-klein-towards-interactive-visual-intelligence, 2026.

Lea Bogensperger, Dominik Narnhofer, Alexander Falk, Konrad Schindler, and Thomas Pock. Flowsdf: Flow matching for medical image segmentation using distance transforms. *International Journal of Computer Vision (IJCV)*, 133:4864 – 4876, 2025.

Francisco Caetano, Christiaan Viviers, Peter HN De With, and Fons van der Sommen. Symmetrical flow matching: Unified image generation, segmentation, and classification with score-based generative models. *arXiv preprint arXiv:2506.10634*, 2025.

Shengcao Cao, Jiuxiang Gu, Jason Kuen, Hao Tan, Ruiyi Zhang, Handong Zhao, Ani Nenkova, Liangyan Gui, Tong Sun, and Yu-Xiong Wang. SOHES: Self-supervised open-world hierarchical entity segmentation. In *The Twelfth International Conference on Learning Representations*, 2024.

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris Coll-Vinent, 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 Dollar, Nikhila Ravi, Kate Saenko, Pengchuan Zhang, and Christoph Feichtenhofer. SAM 3: Segment anything with concepts. In *The Fourteenth International Conference on Learning Representations (ICLR)*, 2026.

Devendra Singh Chaplot, Dhiraj Prakashchand Gandhi, Abhinav Gupta, and Russ R Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 4247–4258, 2020.

Ting Chen, La-La Li, Saurabh Saxena, Geoffrey E. Hinton, and David J. Fleet. A generalist framework for panoptic segmentation of images and videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 909–919, 2023.

Luca Ciampi, Carlos Santiago, João Paulo Costeira, Claudio Gennaro, and Giuseppe Amato. Night and day instance segmented park (ndispark) dataset: a collection of images taken by day and by

night for vehicle detection, segmentation and counting in parking areas (version 1.0). https://doi.org/10.5281/zenodo.6560823, May 2022. URL https://doi.org/10.5281/zenodo.6560823.

Nadav Z. Cohen, Yael Newman, and Ariel Shamir. Semantic segmentation in art paintings. *Computer Graphics Forum*, 41, 2022.

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 (CVPR)*, pp. 3213–3223, 2016.

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 (CVPR)*, pp. 12709–12720, 2024.

Christoph Hümmer, Joachim Sicking, Fabian Hüger, and Hanno Gottschalk. Diffuse2seg: Diffusion models can segment anything without supervision. *arXiv preprint arXiv:2609.06491*, 2026.

Yuanfeng Ji, Zhe Chen, Enze Xie, Lanqing Hong, Xihui Liu, Zhaoqiang Liu, Tong Lu, Zhenguo Li, and Ping Luo. DDP: Diffusion model for dense visual prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 21684–21695, 2023.

Markus Karmann and Onay Urfalioglu. Repurposing stable diffusion attention for training-free unsupervised interactive segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24518–24528, 2025.

Om Khangaonkar and Hamed Pirsiavash. gen2seg: Generative models enable generalizable instance segmentation. In *The Fourteenth International Conference on Learning Representations (ICLR)*, 2026.

Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9396–9405, June 2019.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloé Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross B. Girshick. Segment anything. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3992–4003, 2023.

Alina Kuznetsova, Hassan Rom, Neil Gordon Alldrin, Jasper R. R. Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, Tom Duerig, and Vittorio Ferrari. The Open Images Dataset v4. *International Journal of Computer Vision*, 128: 1956 – 1981, 2020.

Feng Li, Hao Zhang, Peize Sun, Xueyan Zou, Shilong Liu, Chunyuan Li, Jianwei Yang, Lei Zhang, and Jianfeng Gao. Segment and recognize anything at any granularity. In *Proceedings of the 18th European Conference on Computer Vision (ECCV)*, pp. 467–484, 2024.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, pp. 740–755, 2014.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023.

Chang Liu, Xiangtai Li, and Henghui Ding. Referring image editing: Object-level image editing via referring expressions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13128–13138, June 2024.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023.

D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pp. 416–423 vol.2, 2001.

Lu Qi, Lehan Yang, Weidong Guo, Yu Xu, Bo Du, Varun Jampani, and Ming-Hsuan Yang. UniGS: Unified representation for image generation and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6305–6315, 2024.

Xuebin Qin, Hang Dai, Xiaobin Hu, Deng-Ping Fan, Ling Shao, and Luc Van Gool. Highly accurate dichotomous image segmentation. In *European Conference on Computer Vision (ECCV)*, 2022.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya K. Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloé Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao Wu, Ross B. Girshick, Piotr Dollár, and Christoph Feichtenhofer. SAM 2: Segment anything in images and videos. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025.

Mike Roberts, Jason Ramapuram, Anurag Ranjan, Atulit Kumar, Miguel Angel Bautista, Nathan Paczan, Russ Webb, and Joshua M. Susskind. Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10892–10902, 2021.

Andranik Sargsyan and Shant Navasardyan. Flowdis: Language-guided dichotomous image segmentation with flow matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 42039–42048, 2026.

Junjiao Tian, Lavisha Aggarwal, Andrea Colaco, Zsolt Kira, and Mar González-Franco. Diffuse, attend, and segment: Unsupervised zero-shot segmentation using stable diffusion. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3554–3563, 2024.

Wouter Van Gansbeke and Bert De Brabandere. A simple latent diffusion approach for panoptic segmentation and mask inpainting. In *European Conference on Computer Vision (ECCV)*, 2024.

Luc Vincent and Pierre Soille. Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6): 583–598, 1991. doi: 10.1109/34.87344.

Chaoyang Wang, Xiangtai Li, Lu Qi, Henghui Ding, Yunhai Tong, and Ming-Hsuan Yang. Semflow: Binding semantic segmentation and image synthesis via rectified flow. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024a.

Xinlong Wang, Wen Wang, Yue Cao, Chunhua Shen, and Tiejun Huang. Images speak in images: A generalist painter for in-context visual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6830–6839, 2023a.

Xinlong Wang, Xiaosong Zhang, Yue Cao, Wen Wang, Chunhua Shen, and Tiejun Huang. SegGPT: Towards segmenting everything in context. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1130–1140, 2023b.

Xudong Wang, Rohit Girdhar, Stella X. Yu, and Ishan Misra. Cut and learn for unsupervised object detection and instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3124–3134, 2023c.

XuDong Wang, Jingfeng Yang, and Trevor Darrell. Segment anything without supervision. In *Proceedings of the 38th International Conference on Neural Information Processing Systems (NeurIPS)*, 2024b.

Christopher Xie, Yu Xiang, Arsalan Mousavian, and Dieter Fox. Unseen object instance segmentation for robotic environments. *IEEE Transactions on Robotics*, 37, 2021.

Lei Yang, Yan Zi Wei, Yisheng He, Wei Sun, Zhenhang Huang, Haibin Huang, and Haoqiang Fan. iShape: A first step towards irregular shape instance segmentation. *arXiv preprint arXiv:2109.15068*, 2021.

Yuhuan Yang, Xianwei Zhuang, Yuxuan Cai, Chaofan Ma, Shuai Bai, Jiangchao Yao, Ya Zhang, Junyang Lin, and Yanfeng Wang. GenMask: Adapting DiT for segmentation via direct mask generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20455–20467, 2026.

Lingzhi Zhang, Shenghao Zhou, Simon Stent, and Jianbo Shi. Fine-grained egocentric hand-object segmentation: Dataset, model, and applications. In *European Conference on Computer Vision (ECCV)*, 2022.

Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5122–5130, 2017.

A Color Collisions in Palette-Based Partition Encoding

We measure how often different regions receive similar target colors under our palette-based partition encoding. This analysis uses ground-truth regions before model prediction, separating the properties of the encoding from errors introduced by the model and partition readout.

Protocol. We evaluate the seven benchmarks used in our main experiments. Annotations are resized with nearest-neighbor interpolation to the inference resolution, approximately preserving aspect ratio with $\sqrt{HW} = 1024$ and dimensions divisible by 16. Unannotated pixels are excluded.

We construct the low- and high-frequency target maps from the ground-truth regions using our palette-based partition encoding, with the same palettes used at inference. We compare these individual encodings and their six-dimensional concatenation with a reproduction of the UniGS location-aware encoding (Qi et al., 2024). UniGS assigns discrete colors according to the grid cells containing region centroids. We use an 11×11 grid, the largest square grid supported by its 124 foreground colors, assigning the same color to regions whose centroids fall in the same cell.

A region counts as colliding if its target color is within Euclidean distance $\tau = 8$ of another region's target color, with channel values on the $[0, 255]$ scale. Distances are measured in three dimensions for individual palettes and UniGS, and six dimensions for the joint encoding. We report collisions with adjacent regions, defined by four-neighbor pixel contact, and with any other annotated region in the same image. Each region is counted at most once in each measure.

Table 3: Target-color collision rates across 117,300 annotated regions at $\tau = 8$. Ours (low) and Ours (high) use individual palette encodings, while Ours (high and low) concatenates both into a six-dimensional representation. UniGS uses our reproduction with an 11×11 centroid grid. Lower is better.

Collision with	UniGS	Ours (low)	Ours (high)	Ours (high and low)
An adjacent region	27.92%	11.27%	4.01%	**0.47%**
Any other region	36.06%	17.39%	6.85%	**0.48%**

Table 4: Adjacent-region collision rates grouped by the number of annotated regions per image at $\tau = 8$. Ours (low) and Ours (high) use individual palette encodings, while Ours (high and low) concatenates both. UniGS uses our reproduction with an 11×11 centroid grid. Lower is better.

Regions/image	Images	Regions	UniGS	Ours (low)	Ours (high)	Ours (high and low)
1–10	8,115	34,388	13.35%	4.09%	4.79%	**0.61%**
11–25	2,756	45,944	21.62%	9.16%	2.35%	**0.10%**
26–50	828	28,383	47.25%	19.47%	5.65%	**0.81%**
51+	134	8,582	56.13%	24.23%	4.38%	**0.79%**

Results. The joint encoding produces adjacent-region collisions for only 0.47% of regions, compared with 11.27% for the low-frequency palette, 4.01% for the high-frequency palette, and 27.92% for the reproduced UniGS encoding (Table 3). Thus, 99.53% of regions have joint target colors separated from all annotated neighbors at this threshold. Including nonadjacent regions increases the joint collision rate only slightly, to 0.48%.

The joint encoding has the lowest collision rate in every region-count group (Table 4). For the reproduced UniGS encoding, adjacent collisions increase from approximately 13.35% in images with 1–10 annotated regions to 56.13% in images with more than 50. The joint rate remains below 1% across all groups, supporting the use of complementary palettes to distinguish regions even in images with many annotations.

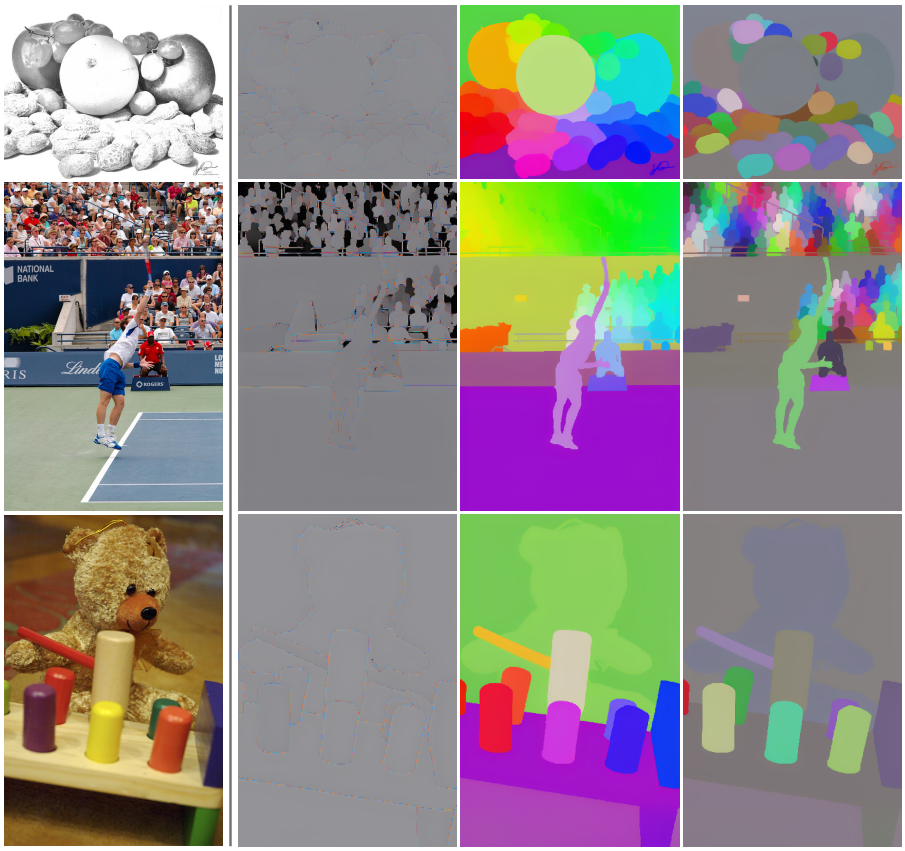

Figure 5: **Effect of palette encoding.** From left to right: input image, predictions from a model trained with random-color targets without palette conditioning, and predictions from models trained with low- or high-frequency palette encoding and conditioning. All outputs are shown before partition readout.

B Qualitative Ablations

We compare models trained with different partition encodings and conditioning inputs. All figures show predicted color maps before partition readout.

Palette encoding. Figure 5 compares three training variants. When trained with random-color targets and no palette conditioning, the model produces nearly uniform gray maps, with little color separation between regions. When trained with low-frequency palette encoding and conditioning, it produces broad color differences across the scene, but nearby regions can receive similar colors. This is visible among the spectators in the tennis example. Training with high-frequency palette encoding and conditioning improves color separation among these spectators, but produces similar colors for some large regions, such as the fruits in the first row. These complementary patterns motivate predicting both palette maps jointly.

Annotated-region conditioning. Figure 6 compares two models trained with both palettes, differing in whether annotated-region conditioning is used. When trained without this conditioning, the model predicts some objects and surrounding areas as black in both maps, including the hairbrush in the first row and much of the background in the second row. When trained with annotated-region conditioning and given an all-ones region mask at inference, the model assigns palette colors to these areas and captures more of the surrounding scene. These examples illustrate how annotated-

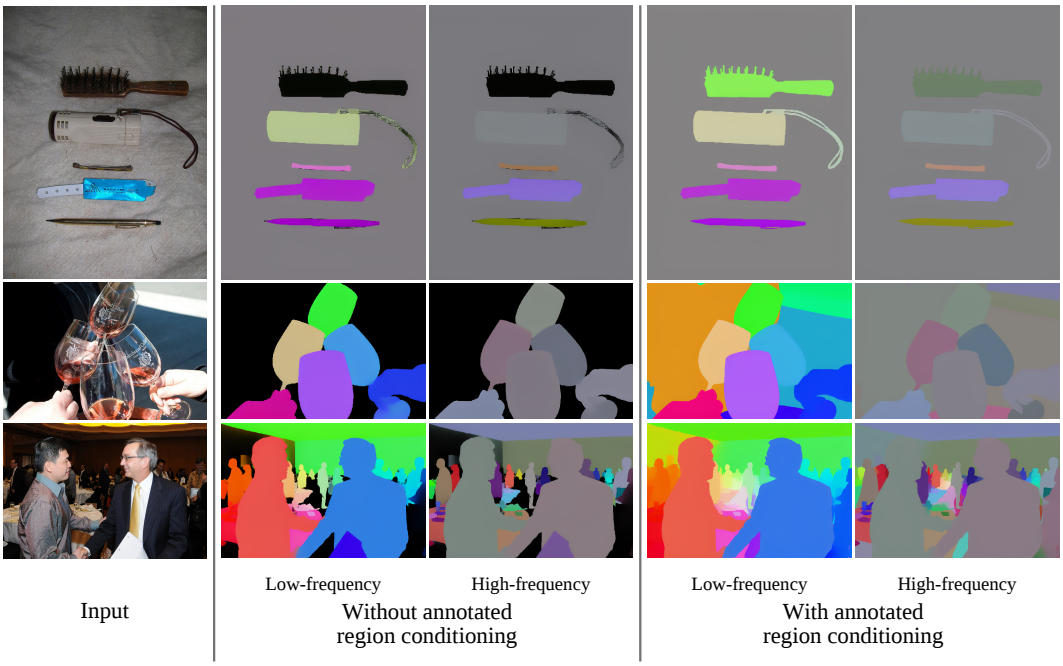

Figure 6: **Effect of annotated-region conditioning.** Each input is followed by low- and high-frequency predictions from a model trained without annotated-region conditioning, then from a model trained with it. The latter uses an all-ones region mask at inference. All outputs are shown before partition readout.

region conditioning helps the model learn from partial annotations while predicting whole-image color maps at inference.

C Effect of Inference Step Count

We vary the number of Euler steps while keeping the trained model and partition-readout settings fixed. Table 5 reports overall instance-weighted mIoU across the seven benchmarks. Performance remains similar across step counts, with a single Euler step achieving the highest mIoU, tied with two steps at the reported precision. These results support our use of single-step inference in the main experiments, requiring only one velocity-network evaluation per image.

Table 5: Effect of inference step count on overall instance-weighted mIoU (%) across seven benchmarks.

Euler steps	1	2	4	8	16
mIoU (%)	55.27	55.27	55.22	55.09	54.95

Table 6: Zero-shot edge detection on BSDS500. All methods use the same boundary extraction procedure. Higher scores are better. Best results are bold, and second-best results are underlined.

Method	ODS	OIS	AP	$AP_{R<0.2}$
SAM	<u>0.738</u>	<u>0.763</u>	0.645	0.813
SAM 3	0.715	0.728	0.562	0.734
UnSAM	0.678	0.693	0.537	0.631
gen2seg	0.642	0.647	<u>0.693</u>	<u>0.906</u>
PEM (Ours)	**0.760**	**0.776**	**0.762**	**0.935**



Figure 7: **Boundary predictions on BSDS500.** Edges are extracted from each method's output using the same Sobel and non-maximum suppression procedure, without training on BSDS500. Zoom in to view the details.

D Zero-Shot Edge Detection

Following Khangaonkar & Pirsiavash (2026), we evaluate zero-shot boundary detection on the 200-image test split of BSDS500 (Martin et al., 2001), without training or tuning on this benchmark. We extract boundaries from the predicted segmentation maps for SAM, SAM 3, and UnSAM, the color features for gen2seg, and the concatenated low- and high-frequency predictions for PEM, before partition readout. For each method, we combine Sobel derivatives across channels to estimate boundary strength and direction, then apply non-maximum suppression to obtain thin edges. Following Arbelaez et al. (2011), we report the F-measure at the optimal dataset-wide threshold (ODS), the F-measure at the optimal threshold per image (OIS), and average precision (AP). We also report $\mathrm{AP}_{R<0.2}$ for the low-recall regime, following Khangaonkar & Pirsiavash (2026). PEM achieves the highest score on all four metrics (Table 6), exceeding SAM by 0.022 in ODS and 0.013 in OIS. It also improves AP over gen2seg from 0.693 to 0.762. Qualitative comparisons are shown in Figure 7.

E Additional Qualitative Comparisons

Figures 8–14 provide additional qualitative comparisons on the test sets of all seven benchmarks. Each figure compares our PEM with SAM, gen2seg, UnSAM, and SAM3. Colors distinguish predicted regions and are not matched across methods. Gen2seg predictions are displayed as continuous color embeddings.



Figure 8: Additional qualitative comparisons on COCO. Zoom in to compare the details.



Figure 9: Additional qualitative comparisons on ADE20K. Zoom in to compare the details.

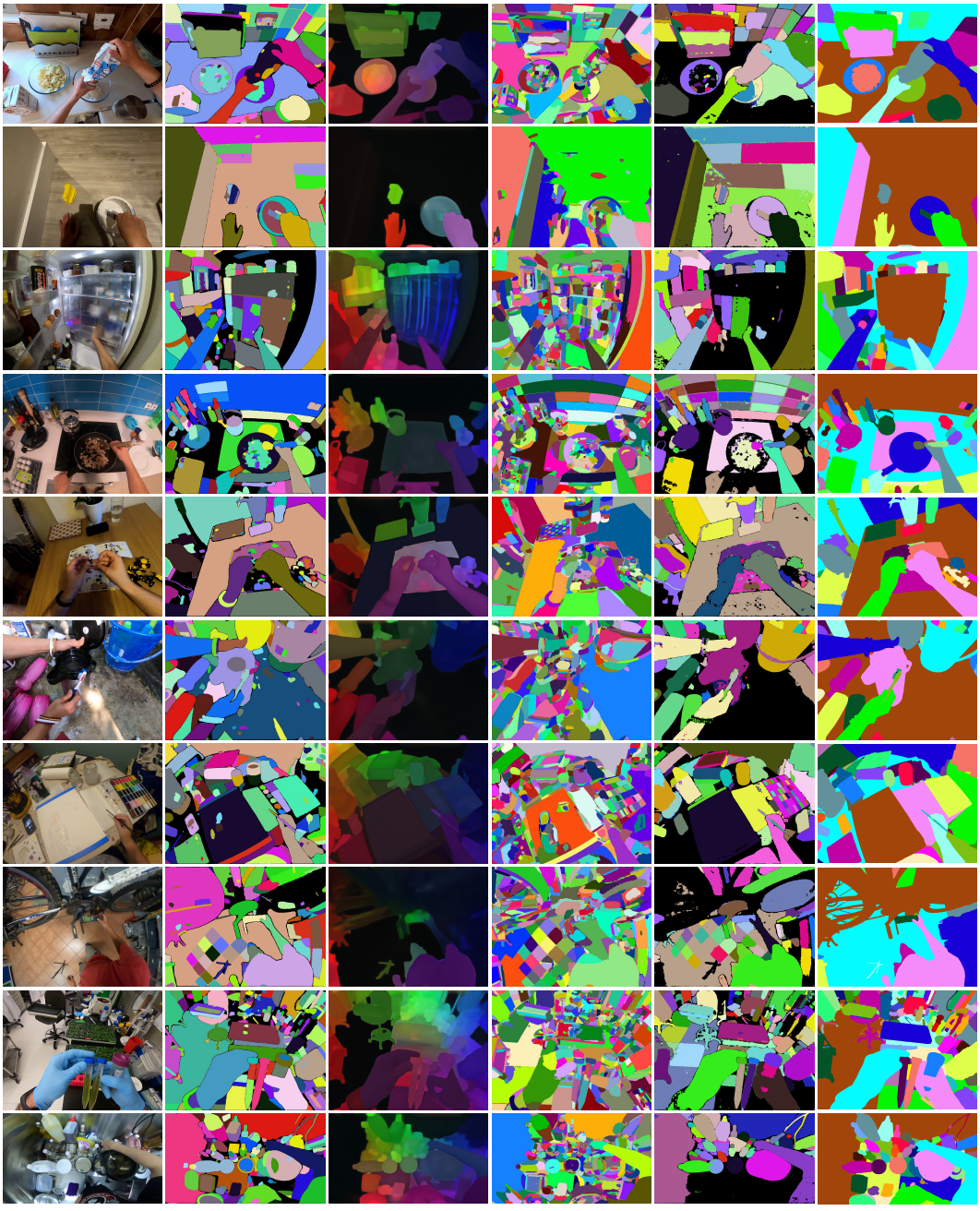

Figure 10: Additional qualitative comparisons on EgoHOS. Zoom in to compare the details.



Figure 11: Additional qualitative comparisons on DRAM. Zoom in to compare the details.

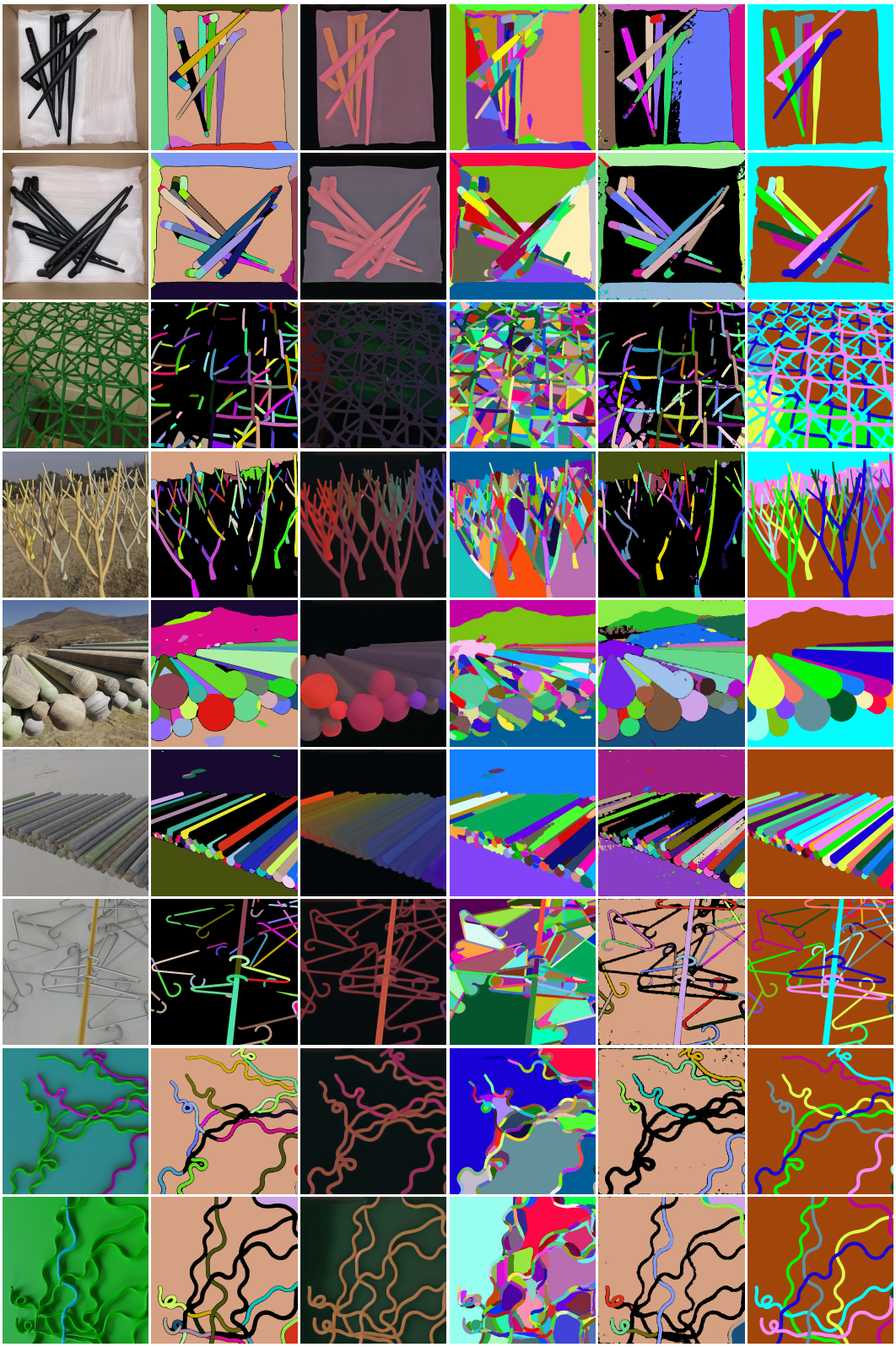

Figure 12: Additional qualitative comparisons on iShape. Zoom in to compare the details.

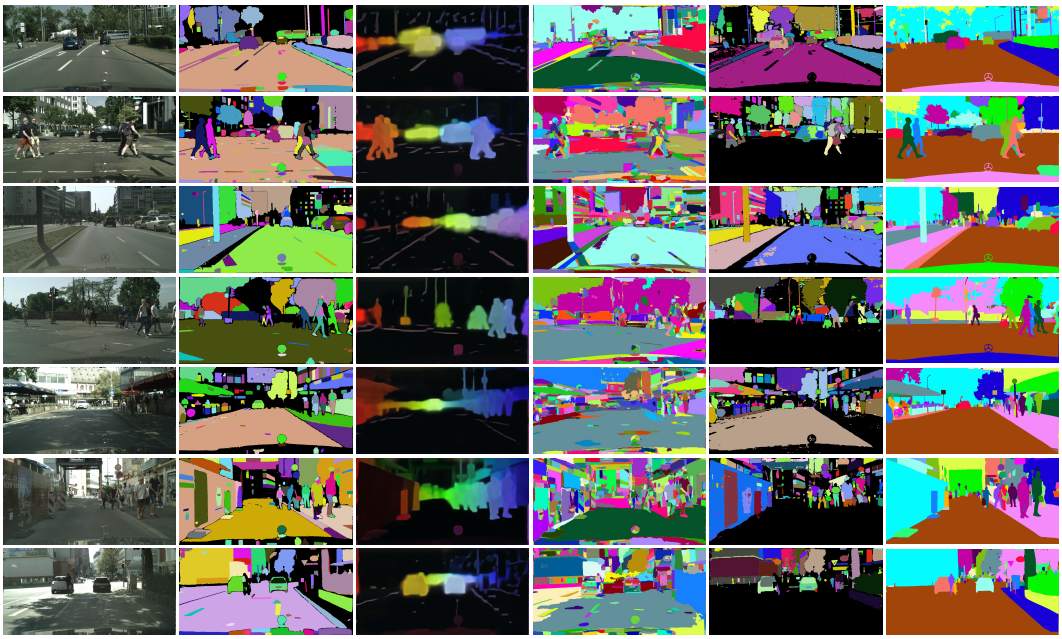

Figure 13: Additional qualitative comparisons on Cityscapes. Zoom in to compare the details.

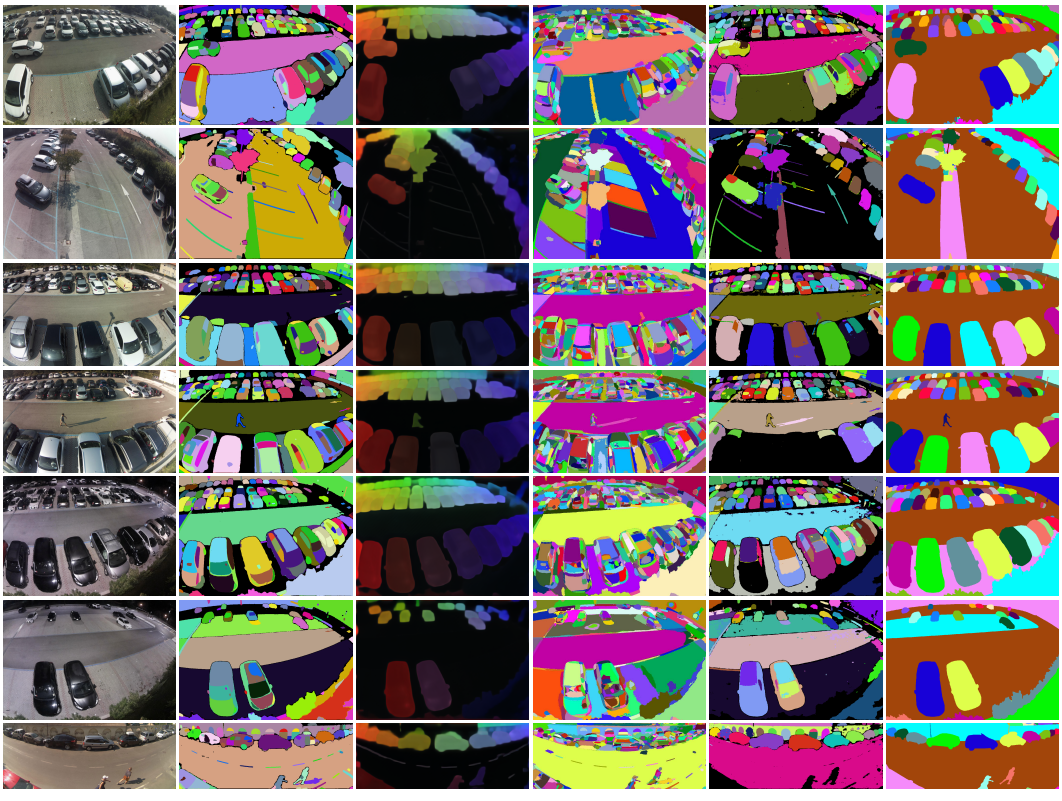

Figure 14: Additional qualitative comparisons on NDISPark. Zoom in to compare the details.