DEOR: OBJECT-CENTRIC REPRESENTATIONS FROM AND FOR MULTIMODAL LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Representing objects individually supports scene understanding and object-level manipulation. Slot-based methods decompose scenes into object representations, but many rely on a fixed number of slots despite varying object counts across images. Moreover, these representations often lack explicit object descriptions, making it difficult to identify and select individual objects for manipulation. To address these limitations, we propose DeOR (Description-guided Object Representations), a framework that uses a multimodal large language model (MLLM) to construct a variable number of object representations that can be identified through language. DeOR sequentially generates an object description and a corresponding object token for each object. Each description specifies which object its paired token should represent. Each representation combines the token's semantic information with visual features from the image. This structure allows objects to be identified through their descriptions and manipulated individually through their representations. DeOR supports compositional generation by recombining object representations across images. Its generated descriptions also allow an external language model to select objects based on user instructions. DeOR then edits the selected representations to remove objects or transfer appearance from another image. On COCO scenes with 4–6 objects, DeOR-L achieves a 26.6% relative improvement in FG-ARI over the strongest evaluated baseline. In zero-shot transfer from COCO to PASCAL VOC, it achieves a 44.2% relative improvement in instance-level mBO over the strongest VOC-trained baseline.

1 INTRODUCTION

Humans organize real-world scenes around individual objects and their relationships (Kahneman et al., 1992; Spelke & Kinzler, 2007), motivating methods that learn a separate representation for each object in an image. Such representations offer a useful basis for structured scene understanding (Schölkopf et al., 2021), visual reasoning (Yang et al., 2020), and compositional generation (Singh et al., 2022a). Learning them from images remains challenging, particularly in real-world scenes with diverse appearances and complex backgrounds (Seitzer et al., 2023; Nguyen et al., 2026).

Object-Centric Learning (OCL) addresses this challenge by decomposing scenes into object-level representations (Locatello et al., 2020; Seitzer et al., 2023). A representative approach, *Slot Attention*, uses competitive attention to group image features into latent slots, each intended to represent an individual entity (Locatello et al., 2020). Building on this formulation, subsequent works have incorporated self-supervised visual features (Caron et al., 2021; Oquab et al., 2024; Seitzer et al., 2023), autoregressive decoders (Singh et al., 2022a;b; Kakogeorgiou et al., 2024), and diffusion-based generative models (Wu et al., 2023; Jiang et al., 2023; Akan & Yemez, 2025; Nguyen et al., 2026), leading to steady progress on complex scenes and real-world data.

Representing individual objects in real-world scenes nevertheless raises two challenges. First, methods that form representations from visual features alone do not explicitly specify which object each representation should capture. The resulting groups may split an object into parts or merge distinct objects (Seitzer et al., 2023; Nguyen et al., 2026). Even when a region corresponds to an object, its representation is not necessarily associated with a semantic description that identifies that instance (Didolkar et al., 2025). For example, selecting one of two dogs for editing requires

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.

identifying which representation corresponds to the intended dog. Explicitly describing the intended object provides a semantic reference for forming and identifying its representation.

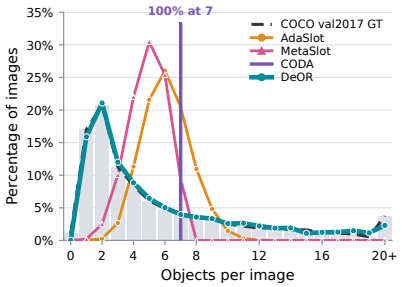


Figure 1: Histograms of predicted and ground-truth object counts on COCO.

Second, object counts vary substantially across real-world images, making a fixed representation budget difficult to choose (Lin et al., 2014). Too few slots limit the number of objects that can be represented separately in crowded scenes, while a larger budget does not itself determine how many slots an image needs. AdaSlot selects active slots from a fixed candidate pool (Fan et al., 2024), and MetaSlot removes duplicate slots through codebook matching (Liu et al., 2025). Figure 1 shows that prior methods favor narrow or fixed counts despite the broad ground-truth distribution. Beyond adapting the count, we seek to determine what each allocated representation should capture by coupling allocation with explicit object descriptions.

Recent work has incorporated language into object-centric learning and visual grounding. CTRL-O conditions object representations on supplied text queries and optional position cues (Didolkar et al., 2025), while Slot-MLLM uses discrete codes from an object-centric visual tokenizer for multimodal understanding and generation (Chi et al., 2025). LISA and GLaMM predict masks from segmentation-token states (Lai et al., 2024; Rasheed et al., 2024); GLaMM also uses these masks to guide external inpainting. These approaches motivate a closer connection between describing objects and representing them: can a model generate descriptions from an image and construct corresponding representations that serve directly as units for recombination and editing?

We propose DeOR (Description-guided Object Representations), a framework that uses a multimodal large language model (MLLM) to describe objects and construct their representations from an image. Building on the ability of MLLMs to describe and identify objects (Wang et al., 2024), DeOR learns to autoregressively generate each object description followed by a corresponding object token. Each description specifies what its paired token should represent, and the number of generated pairs determines how many object representations are constructed. The object token integrates its description with image information to form a semantic representation of the described instance.

Each object representation combines a semantic object token with visual memories aggregated from source image features, forming a unit that can be recombined or edited. During training, masks paired with the descriptions supervise competitive patch ownership among object tokens and registers, which account for regions outside the described objects. The Reader combines the semantic representations and visual memories into spatial conditions for image reconstruction. We build this reconstruction pathway on Scale-RAE (Tong et al., 2026), using a Diffusion Transformer (DiT) (Peebles & Xie, 2023) to generate target visual features and a pretrained decoder to convert them into images.

Experiments on COCO (Lin et al., 2014) and PASCAL VOC (Everingham et al., 2010) show improved object decomposition across varying object counts. On COCO images with 4–6 objects, DeOR-L improves FG-ARI by 26.6% relative to the strongest evaluated baseline. In transfer from COCO to PASCAL VOC without additional training, it improves instance-level mBO by 44.2% relative to the strongest VOC-trained baseline. For compositional generation on COCO, DeOR-L achieves an FID of 17.19, compared with the reported 31.03 for CODA (Nguyen et al., 2026). We also demonstrate language-guided object removal and appearance transfer.

Our contributions are summarized as follows.

- We formulate object representation allocation as sequential generation of object descriptions and corresponding tokens in an MLLM, jointly specifying their semantic identities and an image-dependent number of representations.
- We construct object representations that combine semantic tokens with visual memories, allowing objects to be identified through their descriptions and their representations to be recombined or edited for image generation.
- We demonstrate improved object decomposition across varying object counts, particularly in crowded real-world scenes, and cross-dataset transfer without additional training. These representations support compositional generation and language-guided object editing.

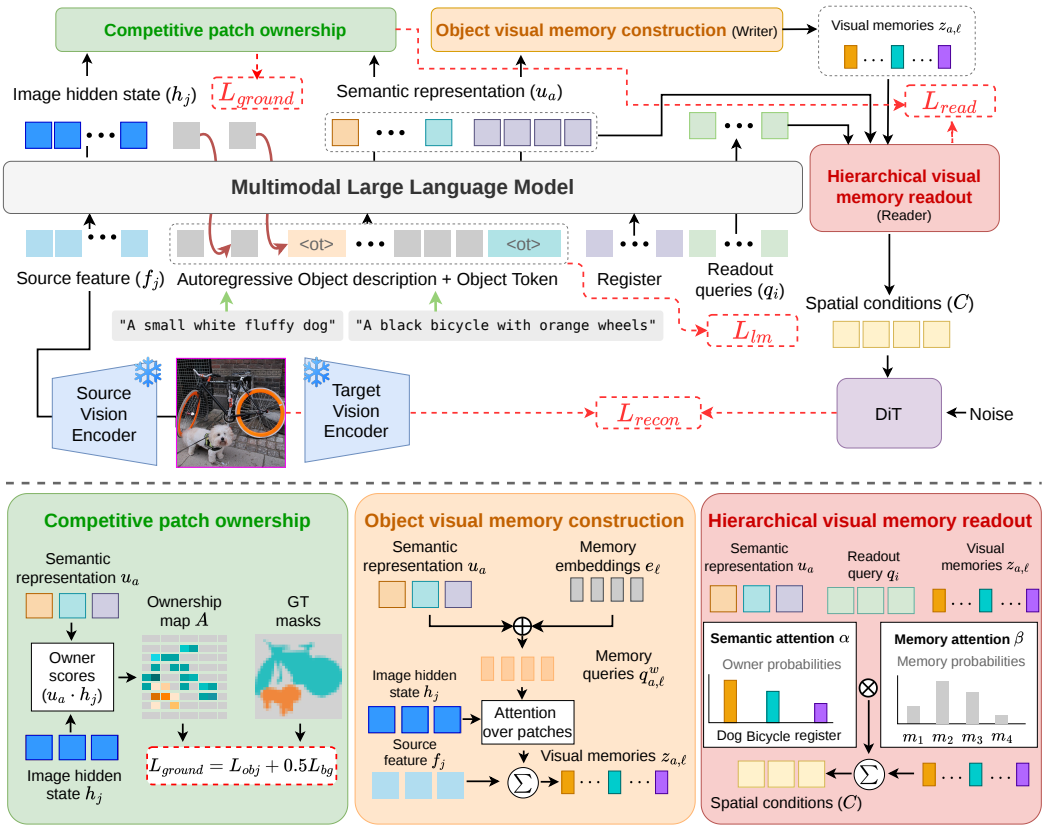


Figure 2: **Overview of DeOR.** An MLLM generates object descriptions and tokens; competitive patch ownership grounds their semantic states using paired masks. The Writer builds visual memories from source features, and the Reader converts them into spatial conditions for DiT reconstruction.

2 DeOR: Description-guided Object Representations

We propose DeOR, which uses an MLLM to construct object representations from an image, as illustrated in Figure 2. DeOR enables an MLLM to autoregressively generate a description and object token for each object, yielding an image-dependent number of representations (Section 2.1). Each representation combines a semantic state grounded in its described region with visual memories collected from source features, while registers represent residual image content (Sections 2.2 and 2.3). The Reader converts object and register representations into spatial conditions for DiT reconstruction (Sections 2.4 and 2.5). Further technical details and algorithms are provided in Appendix A.

2.1 Object Token Formation

We fine-tune an MLLM to describe objects in an image and generate one object token after each description. A frozen source visual encoder extracts P patch features $\{\mathbf{f}_j\}_{j=1}^{P}$ from image I. We project these features into the MLLM input space and provide them with a prompt p requesting object descriptions. Let d_k denote the description of object k. The MLLM autoregressively generates

$$Y = [\texttt{<thing>}, d_1, \texttt{<ot>}, \ldots, \texttt{<thing>}, d_K, \texttt{<ot>}, \texttt{<scene_end>}]. \quad (1)$$

Each description is followed by <ot>, after which the model either starts another object entry or emits <scene_end>. The number of completed entries determines K, allowing the number of object representations to vary across images.

We initialize each object token from its own description so that it starts with information about the object it should represent. Within the MLLM, an object token directly attends only to the image tokens, its own description, and itself, while description tokens follow causal attention. The final

hidden state $\mathbf{s}_k$ of object token k serves as the semantic representation of object k, and we denote the MLLM image-token hidden states by $\mathbf{h}_j$.

During training, we order the annotated object descriptions by descending visible mask area to form the reference sequence Y^*; the paired masks are used by the grounding objective in Section 2.2. With teacher forcing, we supervise description text and special tokens using

$$\mathcal{L}_{\mathrm{LM}}(\theta_{\mathrm{gen}}) = -\mathbb{E}_{(I,p,Y^*)\sim\mathcal{D}}\left[\frac{1}{|\mathcal{T}_{\mathrm{LM}}|}\sum_{g\in\mathcal{T}_{\mathrm{LM}}}\log P_{\theta_{\mathrm{gen}}}\left(y_g^* \mid I, p, Y_{<g}^*\right)\right]. \tag{2}$$

Here, θ_{gen} denotes the trainable parameters used for sequence generation, and $\mathcal{T}_{\mathrm{LM}}$ contains only the supervised output positions for description text and special tokens.

After the object sequence, we append R learnable registers $\mathbf{E}_{\mathrm{reg}}$ and N learnable readout queries $\mathbf{E}_{\mathrm{qry}}$, whose final hidden states are denoted by $\mathbf{b}_r$ and $\mathbf{q}_i$, respectively. Registers represent residual image content outside the described objects, while each $\mathbf{q}_i$ corresponds to one location of the target feature grid and serves as a Reader query in Section 2.4. These tokens are appended rather than generated autoregressively. During training, a single teacher-forced MLLM pass produces the language modeling loss and all required hidden states. At inference, we generate Y, append the registers and readout queries, and run one additional MLLM pass to obtain their states.

2.2 Semantic Object Grounding through Competitive Patch Ownership

We connect each semantic object representation to the image region specified by its description. We treat object and register states as a common set of owners and collect them as $\mathbf{U} \in \mathbb{R}^{(K+R)\times D}$. We collect the MLLM image-patch states as $\mathbf{H} \in \mathbb{R}^{P\times D}$, where D denotes the MLLM hidden dimension. For each image patch, we compute scaled dot-product scores and normalize them across owners:

$$\mathbf{A}_{:,j} = \mathrm{softmax}\left(\mathbf{g}_{:,j}^{\mathrm{own}}\right), \quad \text{where} \quad \mathbf{g}^{\mathrm{own}} = \frac{\mathrm{LN}(\mathbf{U})\,\mathrm{LN}(\mathbf{H})^\top}{\sqrt{D}\,\tau_{\mathrm{own}}} \in \mathbb{R}^{(K+R)\times P}, \tag{3}$$

where LN denotes a layer normalization, and τ_{own} is the temperature. This softmax normalization is applied across all owners at each patch, so the ownership probabilities for each patch sum to one. Objects and registers therefore compete for ownership of each image patch. During training, the instance mask paired with description d_k supervises the ownership map $\mathbf{A}_k$ of object k. The residual region outside all annotated objects supervises $\mathbf{A}_{\mathrm{bg}}$, obtained by summing the register ownership maps. Let $\bar{\mathbf{m}}_k$ and $\bar{\mathbf{m}}_{\mathrm{bg}}$ denote the spatially normalized targets. For images with at least one annotated object, we define

$$\mathcal{L}_{\mathrm{ground}}(\theta_{\mathrm{gen}}, \mathbf{E}_{\mathrm{reg}}) = \mathbb{E}_{\mathcal{D}}\left[-\frac{1}{K}\sum_{k=1}^{K}\bar{\mathbf{m}}_k^\top \log \mathbf{A}_k - \lambda_{\mathrm{bg}}\bar{\mathbf{m}}_{\mathrm{bg}}^\top \log \mathbf{A}_{\mathrm{bg}}\right]. \tag{4}$$

We set $\lambda_{\mathrm{bg}} = 0.5$. This objective encourages each object state to assign high ownership to the region identified by its description, while registers account for the remaining image content.

2.3 Object Visual Memory Construction

A semantic state identifies an object and is grounded in its image region, but reconstruction also requires its visual details. The Writer therefore uses each object or register state to aggregate source visual features into a set of visual memories.

For an owner state $\mathbf{u}$, the Writer adds its projected state to a shared bank of learnable memory embeddings $\mathbf{E}_{\mathrm{mem}}$ to form memory queries $\mathbf{Q}(\mathbf{u})$. Objects use n memory queries and registers use n_{reg}. We collect the source visual features as $\mathbf{F} = [\mathbf{f}_1, \ldots, \mathbf{f}_P]^\top$. Using the MLLM image states $\mathbf{H}$ as keys and the source visual features $\mathbf{F}$ as values, the Writer constructs the visual memories as

$$\mathbf{Q}(\mathbf{u}) = W_Q\mathbf{u} + \mathbf{E}_{\mathrm{mem}}, \qquad \mathbf{Z}(\mathbf{u}) = \mathrm{Attn}\big(\mathbf{Q}(\mathbf{u}), \mathbf{H}, \mathbf{F}\big). \tag{5}$$

Here, $\mathbf{Z}(\mathbf{u})$ denotes the visual memories of the corresponding owner. Each memory is a weighted aggregation of source visual features, with weights determined by the owner-specific queries and MLLM image states. Because these attention weights depend on $\mathbf{u}$ and $\mathbf{H}$, the grounding objective in Eq. 4 guides which source visual features are collected into each memory. Each object k is therefore represented by its semantic state $\mathbf{s}_k$ together with its visual memories $\mathbf{Z}(\mathbf{s}_k)$. The Writer is trained through the reconstruction objective.

2.4 HIERARCHICAL VISUAL MEMORY READOUT

The Reader converts the semantic states and visual memories of objects and registers into information required at each location of the target feature grid. For each readout query $\mathbf{q}_i$, it performs a two-level soft readout: owner-level attention allocates attention across objects and registers, while memory-level readout retrieves visual content from the corresponding memories. This hierarchy allows spatial supervision at the owner level while memory-level attention selects visual details for reconstruction. At target location i, we compute the owner-level attention over all objects and registers as

$$\boldsymbol{\alpha}_{i,:} = \mathrm{softmax}\left(\mathbf{g}^{\mathrm{read}}_{i,:}\right), \qquad \text{where} \quad \mathbf{g}^{\mathrm{read}}_{i,:} = \frac{\mathrm{LN}(\mathbf{q}_i)^\top \mathrm{LN}(\mathbf{U})^\top}{\sqrt{D}\,\tau_R} \in \mathbb{R}^{1\times(K+R)}. \tag{6}$$

Here, τ_R denotes the Reader temperature, and $\alpha_{i,a}$ denotes the a-th entry of $\boldsymbol{\alpha}_{i,:}$, corresponding to the owner-level attention assigned to owner a at target location i. The attention weights sum to one across all owners. Let u_a denote the a-th owner state in $\mathbf{U}$. For each owner a, $\mathrm{MemRead}(q_i, Z(u_a))$ attends over its visual memories and returns a weighted visual-memory representation. The Reader combines them using owner-level attention:

$$r_i = \sum_{a=1}^{K+R} \alpha_{i,a}\,\mathrm{MemRead}\left(q_i, Z(u_a)\right). \tag{7}$$

Thus, owner-level attention determines which object or register contributes at each location, while memory-level readout determines which visual content is retrieved from that representation. We combine the multi-head readout and map it to the Reader output $\mathbf{c}_i$, which is used as a spatial condition for the DiT in Section 2.5. During training, we supervise the owner-level attention using spatial targets derived from the ownership map $\mathbf{A}$ in Eq. 3. Let $\bar{\mathbf{t}}_k$ and $\bar{\mathbf{t}}_{\mathrm{bg}}$ denote the stop-gradient targets for object k and the combined register region, respectively. We average owner-level attention across heads to obtain the object attention map $\bar{\boldsymbol{\alpha}}_k$, and sum register attention to obtain $\bar{\boldsymbol{\alpha}}_{\mathrm{bg}}$. Let θ_{read} denote the Reader parameters used for owner-level attention.

$$\mathcal{L}_{\mathrm{read}}(\theta_{\mathrm{gen}}, \mathbf{E}_{\mathrm{reg}}, \mathbf{E}_{\mathrm{qry}}, \theta_{\mathrm{read}}) = \mathbb{E}_{\mathcal{D}}\left[-\frac{\lambda_{\mathrm{read,obj}}}{K}\sum_{k=1}^{K}\bar{\mathbf{t}}_k^\top \log\bar{\boldsymbol{\alpha}}_k - \lambda_{\mathrm{read,bg}}\bar{\mathbf{t}}_{\mathrm{bg}}^\top \log\bar{\boldsymbol{\alpha}}_{\mathrm{bg}}\right]. \tag{8}$$

We set $\lambda_{\mathrm{read,obj}} = 0.5$ and $\lambda_{\mathrm{read,bg}} = 0.25$. Reader supervision directly trains owner-level attention, while reconstruction trains the entire Reader, including memory-level readout.

2.5 DIFFUSION RECONSTRUCTION AND TRAINING OBJECTIVES

The Reader produces a spatial condition for each location of the target feature grid. A learned conditioning module maps each Reader output $\mathbf{c}_i$ to a DiT condition $\mathbf{C}_i$, and stacking them gives the conditioning tensor $\mathbf{C}$. Each $\mathbf{C}_i$ conditions the corresponding DiT token through adaptive layer normalization (AdaLN) (Peebles & Xie, 2023).

Let $\mathbf{x}_0$ denote the normalized target features from a frozen target visual encoder, and let Θ denote all trainable parameters. Given Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$ and $t \in (0, 1)$, we form $\mathbf{x}_t = (1-t)\mathbf{x}_0 + t\boldsymbol{\epsilon}$. Conditioned on $\mathbf{C}$, the DiT predicts the velocity target $\boldsymbol{\epsilon} - \mathbf{x}_0$. We minimize

$$\mathcal{L}_{\mathrm{recon}}(\Theta) = \mathbb{E}_{I,t,\boldsymbol{\epsilon}}\left[\mathrm{MSE}\left(v_{\mathrm{DiT}}(\mathbf{x}_t, t, \mathbf{C}), \boldsymbol{\epsilon} - \mathbf{x}_0\right)\right]. \tag{9}$$

The reconstruction objective jointly trains the Writer, Reader, conditioning module, and trainable DiT components, with gradients also reaching the trainable MLLM components. The full objective is

$$\mathcal{L}_{\mathrm{total}}(\Theta) = \mathcal{L}_{\mathrm{LM}} + \mathcal{L}_{\mathrm{ground}} + \mathcal{L}_{\mathrm{read}} + \lambda_{\mathrm{recon}}\mathcal{L}_{\mathrm{recon}}. \tag{10}$$

We set $\lambda_{\mathrm{recon}} = 1.5$. Together, the four objectives train object description generation, semantic grounding, object-level memory readout, and visual reconstruction. Both visual encoders and the image decoder remain frozen. At inference, the DiT generates target features from Gaussian noise; we denormalize the generated features and decode them with the frozen decoder.

Object manipulation. Because each object is represented separately by a semantic state and visual memories, individual representations can be modified before Reader readout. For object removal, we discard the selected object's representation and recompute the visual memories of the remaining objects and registers while excluding its predicted region. For appearance transfer, we retain the target object's semantic state and replace its visual memories with those of a donor object. The Reader then produces updated spatial conditions from the modified representations.

Table 1: **Object decomposition on COCO and PASCAL VOC across object-count ranges.** We report FG-ARI, mBO, mIoU, count MAE, and object-count statistics (mean ± std). Higher is better for decomposition metrics; lower for MAE. COCO-trained models are evaluated zero-shot on VOC.

Method	FG-ARI ↑				mBO ↑				mIoU ↑				Count	
	1–3	4–6	7–10	11+	1–3	4–6	7–10	11+	1–3	4–6	7–10	11+	MAE ↓	Active
COCO (in-domain)														*GT avg 5.78 ± 5.83*
AdaSlot	30.1	46.8	49.0	44.2	37.0	24.6	16.7	9.4	38.6	26.0	16.8	8.9	4.41	6.11 ± 1.54
MetaSlot	32.4	50.0	48.5	42.0	39.0	24.2	15.1	8.2	40.5	25.8	15.7	8.1	4.36	4.94 ± 1.23
CODA	35.8	58.3	57.9	51.8	51.0	33.8	22.1	12.4	49.4	34.4	21.8	11.6	4.84	7.00 ± 0.00
DeOR-B	48.9	71.5	69.4	64.0	**66.6**	49.1	**37.4**	**28.6**	**72.6**	53.3	**40.0**	**29.4**	**1.45**	5.66 ± 5.13
DeOR-L	**51.8**	**73.8**	**71.8**	**66.8**	66.3	**49.5**	37.3	27.2	72.5	**53.8**	39.9	28.0	2.27	6.77 ± 6.28
PASCAL VOC (zero-shot, COCO→VOC)														*GT avg 2.22 ± 1.90*
AdaSlot	19.2	45.4	42.9	41.5	44.9	29.1	19.5	10.7	44.2	28.7	18.5	10.5	3.91	5.91 ± 1.53
MetaSlot	23.0	46.8	42.1	38.3	46.7	27.8	16.0	6.9	46.0	28.0	15.8	6.9	3.15	4.95 ± 1.24
CODA	25.6	58.0	54.5	55.2	61.5	40.7	25.6	14.9	55.9	39.3	23.2	13.5	4.92	7.00 ± 0.00
DeOR-B	37.1	79.8	76.7	70.7	83.0	**63.5**	53.8	34.2	83.4	**64.9**	54.3	33.0	**1.33**	3.43 ± 3.55
DeOR-L	**37.8**	**80.5**	**80.0**	**86.5**	**83.8**	62.7	**55.0**	**40.9**	**84.1**	64.1	**55.5**	**37.9**	1.91	4.07 ± 4.79

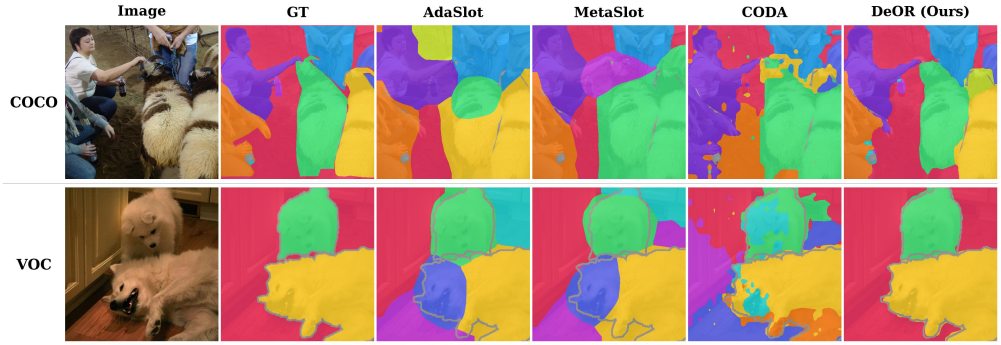


Figure 3: **Qualitative object decomposition on COCO and PASCAL VOC.** Columns show input, ground truth, and model predictions; colors denote regions. Additional examples are in Appendix E.1.

3 Experiments

We evaluate DeOR across diverse aspects of object-centric representation learning, including variable-count object decomposition, cross-dataset generalization, description–region alignment, compositional generation, and language-guided object editing. We address four research questions:

- **RQ1.** Can DeOR adapt the number of object representations to each image while accurately separating individual objects?
- **RQ2.** How well does DeOR generalize to scenes from another dataset without additional training?
- **RQ3.** Does each object representation correspond to the image region identified by its description?
- **RQ4.** Can DeOR support compositional generation and language-guided object editing?

3.1 Experimental Setup

Data and evaluation. We train DeOR on COCO train2017 (Lin et al., 2014) using object descriptions and masks from Pix2Cap-COCO (You et al., 2025). We evaluate object decomposition on COCO val2017 and PASCAL VOC (Everingham et al., 2010) using FG-ARI (Hubert & Arabie, 1985), mBO, mIoU (Kakogeorgiou et al., 2024), and count MAE; description–region alignment on RefCOCO (Kazemzadeh et al., 2014) using mIoU and TargetHit@1; and compositional generation using FID (Heusel et al., 2017) and KID (Bińkowski et al., 2018).

Model configuration. DeOR uses a Qwen2.5-based MLLM (Qwen et al., 2024), SigLIP2-based source and target visual encoders (Tschannen et al., 2025), and a Scale-RAE-based reconstruction

Table 2: **Cross-dataset object decomposition on PASCAL VOC.** Top: VOC-trained references; bottom: zero-shot COCO-trained models. Superscripts i/c denote instance/class metrics.

VOC	FG-ARI↑	mBO^i↑	mBO^c↑	mIoU^i↑	mIoU^c↑
In-domain (VOC-trained, VOC-eval)					
SPOT (w/ENS)	19.7	48.3	55.6	46.8	–
MetaSlot	43.3	43.9	–	42.1	–
CODA	32.2	55.4	61.3	50.8	56.3
Zero-shot (COCO-trained, VOC-eval)					
SPOT	21.1	56.5	64.0	52.3	59.0
MetaSlot	26.7	43.0	45.6	42.5	45.4
CODA	30.8	57.4	58.9	52.4	54.2
DeOR-B	44.0	79.3	76.7	79.8	**79.8**
DeOR-L	**44.9**	**79.9**	**76.9**	**80.3**	**79.8**

pipeline (Tong et al., 2026). By default, we use eight visual memories per object and four registers with 32 memories each. DeOR-B uses a 1.5B language backbone and a 2.4B DiT, whereas DeOR-L uses a 7B language backbone and a 9.8B DiT. Unless otherwise specified, we use DeOR-B; DeOR-L is used for compositional generation and reconstruction.

Baselines. For object decomposition, we compare with fixed-count methods (Kakogeorgiou et al., 2024; Nguyen et al., 2026) and adaptive methods (Fan et al., 2024; Liu et al., 2025), using publicly released checkpoints for Table 1 without additional training. Because DeOR is trained with paired object descriptions and masks, comparisons should be interpreted in light of differences in training supervision. Further details are provided in Appendix B.

3.2 Object Decomposition in Real-World Images with Varying Object Counts

We evaluate whether DeOR can adapt its representation count while maintaining accurate object decomposition on COCO and zero-shot COCO→VOC, grouping images by ground-truth object count into 1–3, 4–6, 7–10, and 11+. At inference, DeOR generates object descriptions and tokens without ground-truth descriptions, counts, or masks.

As shown in Table 1, DeOR-B and DeOR-L outperform the evaluated baselines across all object-count ranges. On COCO images with 4–6 objects, DeOR-L achieves 73.8 FG-ARI versus 58.3 for CODA, a 26.6% relative improvement; on VOC, it achieves 80.5 versus 58.0, showing strong foreground clustering across varying object counts. Additional object-centric baselines are compared in Appendix C.1, while Appendix D.1 evaluates fixed-query controls under matched supervision and backbones.

DeOR-B also achieves the lowest count MAE on COCO (1.45) and VOC (1.33). On COCO, its Active statistics (5.66 ± 5.13) are close to the ground-truth statistics (5.78 ± 5.83), while CODA remains fixed at 7.00 ± 0.00, further supporting image-dependent representation allocation. Figure 3 further shows examples where DeOR avoids merge and split errors observed in the baselines. These results indicate that DeOR adapts representation cardinality while maintaining strong object decomposition across varying object counts. Appendix D.2 further analyzes count errors and decoding limits.

3.3 Generalization across Datasets without Additional Training

To assess generalization beyond the training distribution, we apply COCO-trained DeOR to PASCAL VOC without additional training or adaptation. Table 2 compares zero-shot COCO-trained models with VOC-trained methods as in-domain references.

Among the COCO-trained models, DeOR-L achieves 44.9 FG-ARI and 79.9 instance-level mBO, compared with 30.8 and 57.4 for CODA (Nguyen et al., 2026). Its instance-level mBO also exceeds VOC-trained CODA's 55.4 by 44.2%. This strong instance-level performance indicates that DeOR retains individual-object separation on real-world images from a dataset not used for training, supporting generalization beyond COCO. Full VOC comparisons are provided in Appendix C.2.

Table 3: **Language–region alignment on RefCOCO.** TH@1 denotes TargetHit@1; *All* aggregates all evaluated expressions. Results are percentages; higher is better.

Method	val		testA		testB		All	
	mIoU	TH@1	mIoU	TH@1	mIoU	TH@1	mIoU	TH@1
CTRL-O	23.26	52.18	25.01	57.82	22.12	47.16	23.47	52.54
DeOR-B	**48.40**	**65.73**	**50.70**	**69.18**	**46.24**	**61.71**	**48.52**	**65.73**



Figure 4: **Qualitative description-region alignment.** DeOR-generated descriptions align with their predicted regions, including overlapping objects. Additional examples are provided in Appendix E.2.

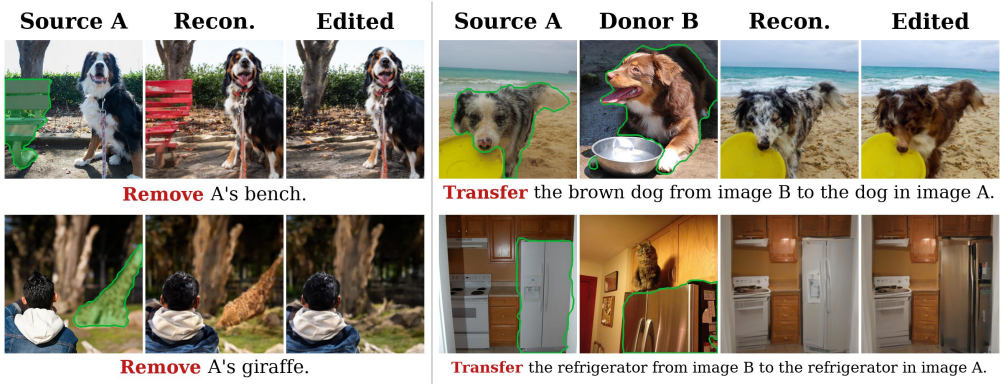


Figure 5: **Language-guided object editing.** (Left) Object removal; (Right) Appearance transfer. Green outlines mark objects selected from DeOR-generated descriptions. Additional examples are provided in Appendices E.3 and E.4.

3.4 Alignment between Object Descriptions and Grounded Regions

For each object representation to be identifiable through language, its description should correspond to the region represented by its paired object token. We evaluate this on RefCOCO (Kazemzadeh et al., 2014) by comparing DeOR with CTRL-O (Didolkar et al., 2025), after excluding images overlapping DeOR's Pix2Cap-COCO training set (You et al., 2025). Both models receive the same image and referring expression; DeOR forms an object token conditioned on the expression and predicts its corresponding region.

Table 3 reports mIoU and TargetHit@1, which measures whether the predicted region best matches the referred object. DeOR achieves 48.52 mIoU and 65.73 TargetHit@1, exceeding CTRL-O by 25.05 and 13.19 points, respectively, showing that description-conditioned object tokens ground to referred instances. Figure 4 shows that DeOR's autoregressively generated descriptions align with their predicted regions, including overlapping objects; Appendix C.4 further shows that using these descriptions for selection improves E2E Success@0.5 by 19.1 points over category-only cues.

3.5 Compositional Generation and Object Editing

For compositional generation on COCO val2017, DeOR first autoregressively generates object descriptions for each image, uses the generated descriptions as teacher-forced sequences to construct object representations, and recombines representations across images. As shown in Table 4, DeOR-L

Table 4: **Compositional generation on COCO val2017.** We report FID and KID $\times 10^3$; lower is better. Baseline results are from Nguyen et al. (2026).

Metric	LSD	SlotDiffusion	SlotAdapt	CODA	**DeOR-L**
KID$\times 10^3$	103.48	57.31	34.38	30.44	**3.53**
FID	167.23	64.21	40.57	31.03	**17.19**

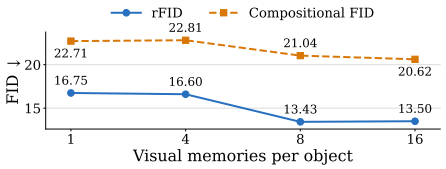


Figure 6: Effect of visual memory count.

Table 5: Ablation on training objectives.

Component			Metric			
GO	RS	LM	FG-ARI ↑	mBO^i ↑	mIoU^i ↑	Count MAE ↓
✓	✓	✓	58.87	53.53	58.02	**1.45**
×	✓	✓	23.82	17.45	23.04	1.48
✓	×	✓	**59.52**	**53.91**	**58.33**	**1.45**
✓	✓	×	56.24	50.74	54.90	5.53

Table 6: Ablation on description conditioning.

Description path		COCO		RefCOCO		Same-cat.
Init	Attn	FG-ARI ↑	mIoU^i ↑	mIoU ↑	TH@1 ↑	TH@1 ↑
✓	✓	**58.87**	**58.02**	**48.52**	**65.73**	**69.83**
×	✓	58.01	55.98	40.55	61.49	64.88
✓	×	56.52	55.02	30.05	46.00	59.55
×	×	53.63	52.15	24.12	30.42	39.80

achieves an FID of 17.19 and a KID $\times 10^3$ of 3.53, compared with 31.03 and 30.44 for CODA (Nguyen et al., 2026). These results show that DeOR representations can be reused as object-level generative units across images. Reconstruction results are provided in Appendix C.3.

For qualitative language-guided editing on COCO images, Qwen3-VL-4B-Instruct (Bai et al., 2025) selects the target and, when needed, donor objects using only a user instruction and DeOR's generated object descriptions, without access to the images or ground-truth information. DeOR then manipulates the selected representations for object removal or appearance transfer. Figure 5 shows examples of both operations, illustrating that the generated descriptions can serve as a semantic interface for selecting object representations and controlling their visual content. See Appendix B.3 for details.

3.6 Ablation Study

Training objectives. Table 5 ablates the grounding objective (GO), Reader supervision (RS), and language modeling (LM) while retaining reconstruction. Removing GO drops FG-ARI from 58.87 to 23.82, and Appendix D.3 shows that grounding also increases object-memory locality. Removing LM raises count MAE from 1.45 to 5.53. Removing RS slightly improves decomposition but reduces donor gain from 0.1156 to 0.0128 (Appendix D.4), supporting object-specific visual control.

Visual memory count. Figure 6 examines the effect of the number of visual memories per object. Increasing the number of memories from 1 to 8 improves both reconstruction and compositional generation, whereas the gains largely saturate from 8 to 16. We therefore use eight visual memories per object as the default configuration.

Description conditioning. Table 6 ablates description-based initialization (Init) and direct description attention (Attn). Removing either degrades performance, especially description–region alignment without Attn, while removing both further degrades decomposition and alignment, showing that the two paths jointly connect each representation to its described object.

4 Conclusion

We proposed DeOR, an MLLM-based framework that autoregressively generates object descriptions and corresponding tokens to construct a variable number of object representations for each image. Each representation combines semantic information with visual memories, allowing objects to be identified through their descriptions and manipulated individually. Experiments demonstrate improved object decomposition across varying object counts, cross-dataset transfer without additional training, and alignment between descriptions and image regions. We further demonstrate compositional generation and language-guided editing by recombining and manipulating these representations, connecting MLLM scene understanding with object-level image generation and control. Limitations and future directions are discussed in Appendix F.

AI USE STATEMENT

We used large language models (LLMs) to assist with research methodology and experimental feedback, interpretation of experimental results, translation, proofreading, clarity improvements, manuscript review, and the implementation and debugging of experimental code. In the language-guided object editing experiments, we used Qwen3-VL-4B-Instruct to select the objects to edit, as described in the main text. The authors designed the research and experiments, reviewed and tested all AI-assisted code and content, verified the resulting experimental outputs, and take responsibility for the final content of the paper.

ETHICS STATEMENT

This work uses existing public datasets to study object representation learning, image generation, and editing. Biases in pretrained models and training data may be reflected in object descriptions, language-based object selection, and generated images. Object removal and appearance transfer can also alter the original context of an image and be used to produce misleading content. When applying these techniques, users should clearly disclose when images have been generated or edited and consider biases and risks of misuse in the intended application.

REPRODUCIBILITY STATEMENT

Section 2 describes the model architecture and training objectives. Appendix A provides details of the input sequence, attention structure, prompts, mathematical formulations, and training and inference algorithms. Appendix B documents the public datasets and preprocessing, model configurations, training settings, and evaluation protocols. We will release the code and trained checkpoints upon acceptance.

REFERENCES

Adil Kaan Akan and Yücel Yemez. Slot-guided adaptation of pre-trained diffusion models for object-centric learning and compositional generation. In *International Conference on Learning Representations*, 2025.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, et al. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025. doi: 10.48550/arXiv.2511.21631. URL https://arxiv.org/abs/2511.21631.

Mikołaj Bińkowski, Danica J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs. In *International Conference on Learning Representations*, 2018. URL https://arxiv.org/abs/1801.01401.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9650–9660, 2021.

Donghwan Chi, Hyomin Kim, Yoonjin Oh, Yongjin Kim, Donghoon Lee, Daejin Jo, Jongmin Kim, Junyeob Baek, Sungjin Ahn, and Sungwoong Kim. Slot-MLLM: Object-centric visual tokenization for multimodal LLM. *arXiv preprint arXiv:2505.17726*, 2025.

Nicki Skafte Detlefsen, Jiri Borovec, Justus Schock, Ananya Harsh Jha, Teddy Koker, Luca Di Liello, Daniel Stancl, Changsheng Quan, Maxim Grechkin, and William Falcon. TorchMetrics - measuring reproducibility in PyTorch. *Journal of Open Source Software*, 7(70):4101, 2022. doi: 10.21105/joss.04101. URL https://doi.org/10.21105/joss.04101.

Aniket Didolkar, Andrii Zadaianchuk, Rabiul Awal, Maximilian Seitzer, Efstratios Gavves, and Aishwarya Agrawal. CTRL-O: Language-controllable object-centric visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 29523–29533, 2025.

Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The Pascal Visual Object Classes (VOC) challenge. *International Journal of Computer Vision*, 88 (2):303–338, 2010. doi: 10.1007/s11263-009-0275-4.

Ke Fan, Zechen Bai, Tianjun Xiao, Tong He, Max Horn, Yanwei Fu, Francesco Locatello, and Zheng Zhang. Adaptive slot attention: Object discovery with dynamic slot number. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23062–23071, 2024.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://arxiv.org/abs/1706.08500.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2:193–218, 1985. doi: 10.1007/BF01908075.

Jindong Jiang, Fei Deng, Gautam Singh, and Sungjin Ahn. Object-centric slot diffusion. In *Advances in Neural Information Processing Systems*, volume 36, pp. 8563–8601, 2023. doi: 10.52202/075280-0375. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b3ceb8a495a63ced4a48f8429ccdcd8-Abstract-Conference.html.

Daniel Kahneman, Anne Treisman, and Brian J. Gibbs. The reviewing of object files: Object-specific integration of information. *Cognitive Psychology*, 24(2):175–219, 1992.

Ioannis Kakogeorgiou, Spyros Gidaris, Konstantinos Karantzalos, and Nikos Komodakis. SPOT: Self-training with patch-order permutation for object-centric learning with autoregressive transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22776–22786, 2024.

Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. ReferItGame: Referring to objects in photographs of natural scenes. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 787–798. Association for Computational Linguistics, 2014. doi: 10.3115/v1/D14-1086. URL https://aclanthology.org/D14-1086/.

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*, pp. 9579–9589, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Lai_LISA_Reasoning_Segmentation_via_Large_Language_Model_CVPR_2024_paper.html.

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*, pp. 740–755. Springer, 2014.

Hongjia Liu, Rongzhen Zhao, Haohan Chen, and Joni K. Pajarinen. MetaSlot: Break through the fixed number of slots in object-centric learning. In *Advances in Neural Information Processing Systems*, volume 38, pp. 67319–67344, 2025. doi: 10.52202/085713-2258. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/614ef802d663b9cdac2e6600f8ee5764-Abstract-Conference.html.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. In *Advances in Neural Information Processing Systems*, volume 33, pp. 11525–11538, 2020.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Bac Nguyen, Yuhta Takida, Naoki Murata, Chieh-Hsin Lai, Toshimitsu Uesaka, Stefano Ermon, and Yuki Mitsufuji. Improved object-centric diffusion learning with registers and contrastive alignment. In *International Conference on Learning Representations*, pp. 51588–51616, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/54b7de0311c87ed8caa45b62cee80e15-Abstract-Conference.html.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=a68SUt6zFt.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. URL https://arxiv.org/abs/2412.15115.

Hanoona Rasheed, Muhammad Maaz, Sahal Shaji, 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. URL https://arxiv.org/abs/2311.03356.

Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Toward causal representation learning. *Proceedings of the IEEE*, 109(5):612–634, 2021.

Maximilian Seitzer, Max Horn, Andrii Zadaianchuk, Dominik Zietlow, Tianjun Xiao, Carl-Johann Simon-Gabriel, Tong He, Zheng Zhang, Bernhard Schölkopf, Thomas Brox, and Francesco Locatello. Bridging the gap to real-world object-centric learning. In *International Conference on Learning Representations*, 2023.

Gautam Singh, Fei Deng, and Sungjin Ahn. Illiterate DALL-E learns to compose. In *International Conference on Learning Representations*, 2022a.

Gautam Singh, Yi-Fu Wu, and Sungjin Ahn. Simple unsupervised object-centric learning for complex and naturalistic videos. In *Advances in Neural Information Processing Systems*, 2022b.

Elizabeth S. Spelke and Katherine D. Kinzler. Core knowledge. *Developmental Science*, 10(1):89–96, 2007.

Shengbang Tong, Boyang Zheng, Ziteng Wang, Bingda Tang, Nanye Ma, Ellis Brown, Jihan Yang, Rob Fergus, Yann LeCun, and Saining Xie. Scaling text-to-image diffusion transformers with representation autoencoders. *arXiv preprint arXiv:2601.16208*, 2026. URL https://arxiv.org/abs/2601.16208.

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, Olivier Hénaff, Jeremiah Harmsen, Andreas Steiner, and Xiaohua Zhai. SigLIP 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025. URL https://arxiv.org/abs/2502.14786.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-VL: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

Ziyi Wu, Jingyu Hu, Wuyue Lu, Igor Gilitschenski, and Animesh Garg. SlotDiffusion: Object-centric generative modeling with diffusion models. In *Advances in Neural Information Processing Systems*, 2023.

Jianwei Yang, Jiayuan Mao, Jiajun Wu, Devi Parikh, David D. Cox, Joshua B. Tenenbaum, and Chuang Gan. Object-centric diagnosis of visual reasoning. *arXiv preprint arXiv:2012.11587*, 2020.

Zuyao You, Junke Wang, Lingyu Kong, Bo He, and Zuxuan Wu. Pix2Cap-COCO: Advancing visual comprehension via pixel-level captioning. *arXiv preprint arXiv:2501.13893*, 2025. URL https://arxiv.org/abs/2501.13893.

A METHOD DETAILS

A.1 INPUT SEQUENCE AND ATTENTION STRUCTURE

Query \ Key	Image	Description 1	Object Token 1	Description 2	Object Token 2	Registers	Readout queries
Image	Attend	Block	Block	Block	Block	Block	Block
Description 1	Attend	Causal	Block	Block	Block	Block	Block
Object Token 1	Attend	Attend	Self only	Block	Block	Block	Block
Description 2	Attend	Attend	Attend	Causal	Block	Block	Block
Object Token 2	Attend	Block	Block	Attend	Self only	Block	Block
Registers	Attend	Block	Block	Block	Block	Attend	Block
Readout queries	Block	Block	Block	Block	Block	Block	Self only

Figure 7: **Attention structure in DeOR.** Rows indicate query tokens, and columns indicate the tokens they can attend to. Attend permits attention to all tokens in a block; Block prohibits attention; Causal permits attention to preceding tokens and the current token; Self permits each token to attend only to itself. Prompt and formatting tokens are omitted for clarity.

The MLLM input consists of a system message, image tokens, a user instruction, and an assistant sequence containing object descriptions and their corresponding object tokens. Register tokens and spatial query tokens are appended to this sequence. Figure 7 illustrates the attention structure for two objects. Object descriptions follow causal attention, while each object token attends only to the image tokens, its own description, and itself. Register tokens capture residual image content, and spatial query tokens receive object and register information through the Reader. Table 7 supplements Figure 7 with the attention rules for prompt and formatting tokens. The same rules apply to the reference sequence during teacher forcing and to the available sequence prefix during autoregressive generation.

Table 7: **Attention rules for prompt and formatting tokens.** Each row specifies the positions accessible to the corresponding query tokens. Padding positions are excluded.

Query tokens	Accessible positions
Prompt and chat-template tokens	Preceding tokens and the current token, following causal attention.
Image tokens	All image tokens, together with the system message and user-prefix tokens preceding the image.
Description text, `<thing>`, and `<scene_end>`	Preceding tokens and the current token, including the image, prompt, and earlier object entries.
`<ot>`	Image tokens, the text and formatting tokens in its own object entry, and itself.
Register and readout query tokens	No attention to prompt or formatting tokens. Their other connections are shown in Figure 7.

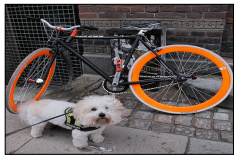
A.2 PROMPTS AND OUTPUT FORMATS

Figure 8 presents the prompts and an illustrative output sequence. The system message defines the task of decomposing an image into countable object instances. The user instruction asks the

(a) System message

You are a vision assistant that decomposes an image into countable object instances and grounded object tokens.

(b) User input



Illustrative input

Describe every visible countable object instance one at a time, in descending order of visible area. After each object description, output exactly one <ot>. Do not describe background, stuff, or uncountable regions. After the last object, output <scene_end>.

(c) Illustrative output

<thing> Dog 1: A small white fluffy dog <ot>
<thing> Bicycle 1: A black bicycle with orange wheels <ot>
<scene_end>

Figure 8: **Prompts and output format for autoregressive object representation extraction.** (a) System message. (b) User input containing an image and task instructions. (c) Illustrative output with one description and one <ot> token per object, followed by <scene_end>.

model to describe visible instances in descending order of visible area and produce exactly one <ot> token after each description. Background, stuff, and uncountable regions are excluded from the description sequence. Each object entry begins with <thing>, followed by a category name, an instance index, a description, and an <ot> token. The index distinguishes instances within the same category. Category names, indices, and descriptions are ordinary text, whereas <thing>, <ot>, and <scene_end> are special tokens. The sequence ends with <scene_end>.

A.3 Object Token Initialization and Hidden-State Extraction

For object k, let $d_k = (d_{k,1}, \ldots, d_{k,L_k})$ contain the L_k text tokens between <thing> and the corresponding <ot>, including the category name, instance index, and description. The <thing> and <ot> markers are not included in this average. To initialize the object token with information from its own description, we use

$$\mathbf{e}_k^{\mathrm{ot}} = \mathrm{Emb}(\texttt{<ot>}) + \mathrm{Proj}_{\mathrm{init}}\left(\mathrm{LN}\left(\frac{1}{L_k}\sum_{g=1}^{L_k}\mathrm{Emb}(d_{k,g})\right)\right). \tag{11}$$

Here, g indexes the text tokens in d_k, Emb is the token embedding lookup, LN is layer normalization, and $\mathrm{Proj}_{\mathrm{init}}$ is a learned linear projection. During training, we compute this embedding from the reference description. During inference, the MLLM first predicts <ot>; before processing it as an input, we replace its shared embedding with $\mathbf{e}_k^{\mathrm{ot}}$ computed from the generated description.

For object token k, let o_k denote its input position and let $\mathrm{pos}(I)$ denote the image token positions. The set $\mathcal{J}_k$ contains the positions after the preceding object token and before o_k, including formatting tokens; for the first object, this span begins at the start of Y. With g indexing input positions, the additive attention mask is

$$B_{o_k,g} = \begin{cases} 0, & g \in \mathrm{pos}(I) \cup \mathcal{J}_k \cup \{o_k\}, \\ -\infty, & \text{otherwise.} \end{cases} \tag{12}$$

The mask is added to the attention logits before softmax. Object token k attends directly only to image tokens, its own description span, and itself. Description tokens use causal attention, so their states can carry context from earlier objects. Figure 7 and Table 7 specify the remaining attention rules.

We append R register embeddings $\mathbf{E}_{\text{reg}}$ and N readout query embeddings $\mathbf{E}_{\text{qry}}$ after the object sequence. These are not autoregressively generated. Each query corresponds to a fixed location in the target feature grid. Registers attend to image tokens and all registers, while each readout query attends only to itself within the MLLM. Using the sequence-generation parameters θ_{gen} defined in Section 2.1, we denote input construction, the MLLM forward pass, and extraction of the final hidden states by

$$\big(\{\mathbf{h}_j\},\{\mathbf{s}_k\},\{\mathbf{b}_r\},\{\mathbf{q}_i\}\big) = F_{\theta_{\text{gen}}}\left(I,p,Y;\mathbf{E}_{\text{reg}},\mathbf{E}_{\text{qry}}\right). \tag{13}$$

The outputs are the image patch, object token, register, and readout query states, respectively. During training, one teacher-forced pass on Y^* computes these states and the language modeling loss. At inference, an additional pass follows generation of Y, using the same description-conditioned object embeddings and the appended registers and queries.

A.4 Attention and Projection Details

For readability, Section 2 omits some normalization, projection, and multi-head details in the Writer, Reader, and conditioning computations. Here we provide the corresponding implementation details. Let u_a denote the a-th owner state in $\mathbf{U}$, and let $z_{a,\ell}$ denote its ℓ-th visual memory. For matrix inputs, layer normalization and linear projections are applied independently to each vector.

Writer. For owner a, its ℓ-th memory query combines the owner state with a shared learnable memory embedding $\mathbf{e}_\ell$. The MLLM image states provide the keys, while the source visual features provide the values:

$$\begin{aligned}\mathbf{q}^W_{a,\ell} &= W_Q\,\mathrm{LN}(\mathbf{u}_a)+\mathbf{e}_\ell, \qquad w_{a,\ell,j} = \mathrm{softmax}_j\left(\frac{(\mathbf{q}^W_{a,\ell})^\top W_K\,\mathrm{LN}(\mathbf{h}_j)}{\sqrt{D}\,\tau_W}\right),\\ \mathbf{z}_{a,\ell} &= \sum_{j=1}^{P} w_{a,\ell,j} W_V\,\mathrm{LN}(\mathbf{f}_j).\end{aligned} \tag{14}$$

Thus, $\mathrm{Attn}(\mathbf{Q}(\mathbf{u}),\mathbf{H},\mathbf{F})$ in the main text denotes this weighted aggregation of projected source visual features over image patches.

Reader. The Reader uses multi-head attention. Below, we give the computation for one attention head and omit the head index. The readout query is first projected, after which owner-level and memory-level attention are computed as

$$\begin{aligned}\hat{\mathbf{q}}_i &= W^R_Q\,\mathrm{LN}(\mathbf{q}_i),\\ \alpha_{i,a} = \mathrm{softmax}_a\left(\frac{\hat{\mathbf{q}}_i^\top W^R_{\text{sem}}\,\mathrm{LN}(\mathbf{u}_a)}{\sqrt{d_R}\,\tau_R}\right), &\qquad \beta_{i,a,\ell} = \mathrm{softmax}_\ell\left(\frac{\hat{\mathbf{q}}_i^\top W^R_{\text{mem}}\,\mathrm{LN}(\mathbf{z}_{a,\ell})}{\sqrt{d_R}\,\tau_R}\right).\end{aligned} \tag{15}$$

Here, α is normalized over all object and register owners, whereas β is normalized over the visual memories of each owner, and d_R denotes the dimensionality of each Reader attention head. Equation 6 presents the owner-level score vector $\mathbf{g}^{\text{read}}_{i,:}$ in compact form, while the computation above gives its exact implementation with learned projections. Objects use n memories and registers use n_{reg}.

$$\begin{aligned}\mathrm{MemRead}\big(\mathbf{q}_i,\mathbf{Z}(\mathbf{u}_a)\big) &= \sum_\ell \beta_{i,a,\ell} W^R_V\,\mathrm{LN}(\mathbf{z}_{a,\ell}),\\ \mathbf{r}_i &= \sum_a \alpha_{i,a}\,\mathrm{MemRead}\big(\mathbf{q}_i,\mathbf{Z}(\mathbf{u}_a)\big).\end{aligned} \tag{16}$$

The outputs of all attention heads are concatenated and passed through an output projection followed by layer normalization to obtain the Reader output $\mathbf{c}_i$. Thus, the owner-level attention in the main text corresponds to α, while the memory-level readout corresponds to the β-weighted aggregation above.

DiT conditioning. Finally, each Reader output is mapped to the DiT conditioning space by a learned conditioning projector followed by layer normalization:

$$\mathbf{C}_i = \mathrm{LN}\left(\mathrm{Proj}_{\text{cond}}(\mathbf{c}_i)\right). \tag{17}$$

Each $\mathbf{C}_i$ is applied to the DiT token at the corresponding target-grid location through AdaLN.

A.5 Supervision Targets and Loss Details

Language modeling. For the language modeling loss in Eq. (2), $\mathcal{T}_{\mathrm{LM}}$ contains reference output positions for description text and special tokens. Image, prompt, register, readout query, and padding positions are excluded from the prediction targets. Register and readout query embeddings receive no gradients from this objective.

Grounding targets. Let $\mathbf{m}_k \in [0,1]^P$ be the mask paired with description d_k on the source patch grid. Registers are supervised jointly using a residual mask. For an object or residual mask $\mathbf{m}$, the target construction and normalization are

$$m_{\mathrm{bg},j} = 1 - \max_{1 \le k \le K} m_{k,j}, \qquad \bar{\mathbf{m}} = \frac{\mathbf{m}}{\max\left(1, \sum_{j=1}^{P} m_j\right)}. \tag{18}$$

Spatial normalization accounts for differences in region size. The normalized object and residual masks are used in the grounding loss in Eq. (4), where register ownership probabilities are summed before applying the logarithm.

Reader supervision targets. We resize the ownership map $\mathbf{A}$ to the N readout-query positions and stop gradients through the resized map before using it as a supervision target. Object ownership maps remain separate, while the register ownership maps are summed to form a single residual target. We spatially normalize each target using Eq. (18), with P replaced by N, yielding $\bar{\mathbf{t}}_k, \bar{\mathbf{t}}_{\mathrm{bg}} \in \mathbb{R}^N$.

For the Reader loss in Eq. (8), $\bar{\boldsymbol{\alpha}}_k$ is obtained by averaging object-level attention across heads, while $\bar{\boldsymbol{\alpha}}_{\mathrm{bg}}$ sums the register attention. These terms supervise the object-level attention α. The memory-level attention β is trained through reconstruction without direct spatial supervision.

A.6 Memory Configuration and Trainable Parameters

The Writer memory embeddings form a shared bank of $\max(n, n_{\mathrm{reg}})$ vectors. Objects use its first n entries, and registers use its first n_{reg} entries. Projected Writer queries, keys, values, and memory vectors all have dimension D. Memory counts, attention temperatures, and optimization settings are provided in Appendix B.2.

The parameters θ_{gen} comprise the MLLM LoRA adapters (Hu et al., 2022), image projector, description-conditioned initialization module, and the input embedding and output head rows for added special tokens. Language modeling trains these parameters. Grounding updates the register embeddings $\mathbf{E}_{\mathrm{reg}}$ and the parameters in θ_{gen} other than the language output head.

Reader supervision updates $\mathbf{E}_{\mathrm{reg}}$, $\mathbf{E}_{\mathrm{qry}}$, and the parameters in θ_{gen} other than the language output head. Within the Reader, it updates the query and semantic-key projections and their normalization layers.

Reconstruction trains the Writer, the entire Reader, the conditioning projector, register and query embeddings, and the parameters in θ_{gen} that produce their input states, excluding the language output head. Within the DiT, we train the final 16 Transformer blocks and all AdaLN modulation parameters. Both visual encoders and the image decoder remain frozen.

A.7 Feature Normalization and Sampling

Target visual features are normalized with fixed channel-wise means and standard deviations. During training, we sample Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and draw time $t \in (0,1)$ from a shifted logit-normal distribution, then form $\mathbf{x}_t = (1-t)\mathbf{x}_0 + t\boldsymbol{\epsilon}$. The reconstruction objective in Eq. (9) predicts velocity $\boldsymbol{\epsilon} - \mathbf{x}_0$.

Inference starts from Gaussian noise and uses Euler sampling with decreasing timesteps. Before decoding, generated features are multiplied by the fixed channel-wise standard deviations and the means are added back. Sampling settings and the object-editing procedure are described in Appendix B.3.

A.8 Training and Inference Algorithms

Algorithms 1 and 2 summarize the training and inference procedures, respectively. In both algorithms, F denotes features from the source visual encoder, and H_{img} denotes the MLLM hidden states at image token positions. U collects object and register states, while Q collects readout query states. The Writer constructs visual memories Z, and the Reader output c is mapped to DiT conditions C. During training, the MLLM forward pass also computes the language modeling loss. The operator sg denotes stop-gradient. Here, d_{rec} denotes the dimensionality of the reconstruction target. Decoding limits and the handling of incomplete object entries are described in Appendix B.3.

Algorithm 1 Training DeOR

Require: Training dataset $\mathcal{D}$ of images I, object descriptions $\boldsymbol{d}$, and paired instance masks $\mathcal{M}$; prompt p; trainable parameters Θ
Ensure: Trained parameters Θ
Operations are evaluated per image in the minibatch.
while training is not finished **do**
  Sample a minibatch $\mathcal{B} = \{(I, \boldsymbol{d}, \mathcal{M})\}$ from $\mathcal{D}$
  $(Y^*, \mathcal{M}) \leftarrow \textsc{PrepareTargets}(\boldsymbol{d}, \mathcal{M})$ ▷ Descending mask area
  $F \leftarrow E_{\text{src}}(I)$; $x_0 \leftarrow \text{Normalize}(E_{\text{tgt}}(I))$
  $(U, H_{\text{img}}, Q, \mathcal{L}_{\text{LM}}) \leftarrow \textsc{MLLMForward}(F, p, Y^*)$ ▷ Teacher forcing
  $A \leftarrow \textsc{Ownership}(U, H_{\text{img}})$; $\mathcal{L}_{\text{ground}} \leftarrow \textsc{GroundingLoss}(A, \mathcal{M})$
  $Z \leftarrow \textsc{Writer}(U, H_{\text{img}}, F)$
  $(c, \alpha) \leftarrow \textsc{Reader}(Q, U, Z)$; $C \leftarrow \textsc{ProjectAndNormalize}(c)$
  $T \leftarrow \textsc{ReaderTargets}(\text{sg}(A))$; $\mathcal{L}_{\text{read}} \leftarrow \textsc{ReaderLoss}(\alpha, T)$
  $t \leftarrow \textsc{SampleTimestep}$; $\epsilon \sim \mathcal{N}(0, \mathbf{I})$; $x_t \leftarrow (1-t)x_0 + t\epsilon$
  $\mathcal{L}_{\text{recon}} \leftarrow \|v_{\theta_{\text{DiT}}}(x_t, t, C) - (\epsilon - x_0)\|_2^2 / d_{\text{rec}}$
  $\mathcal{L}_{\mathcal{B}} \leftarrow \text{Mean}_{\mathcal{B}}[\mathcal{L}_{\text{LM}} + \mathcal{L}_{\text{ground}} + \mathcal{L}_{\text{read}} + 1.5\,\mathcal{L}_{\text{recon}}]$
  $\Theta \leftarrow \textsc{OptimizerStep}(\Theta, \nabla_\Theta \mathcal{L}_{\mathcal{B}})$
end while
return Θ

Algorithm 2 Object Representation Extraction and Image Generation

Require: Image I; prompt p; token budget $T_{\max}$; object limit $K_{\max}$
Ensure: Object descriptions and representations $\mathcal{O}$; ownership map A; generated image $\widehat{I}$
$F \leftarrow E_{\text{src}}(I)$
$Y \leftarrow \textsc{GreedyGenerate}(F, p, T_{\max})$ ▷ Stop at `<scene_end>`
$Y \leftarrow \textsc{PackCompleteEntries}(Y, K_{\max})$; $\boldsymbol{d} \leftarrow \textsc{ExtractDescriptions}(Y)$
$(U, H_{\text{img}}, Q) \leftarrow \textsc{MLLMForward}(F, p, Y)$ ▷ Additional MLLM forward pass
$A \leftarrow \textsc{Ownership}(U, H_{\text{img}})$
$Z \leftarrow \textsc{Writer}(U, H_{\text{img}}, F)$
$\mathcal{O} \leftarrow \{(d_k, s_k, \{z_{k,\ell}\}_{\ell=1}^{n})\}_{k=1}^{|\boldsymbol{d}|}$
$(c, \alpha) \leftarrow \textsc{Reader}(Q, U, Z)$; $C \leftarrow \textsc{ProjectAndNormalize}(c)$
$\epsilon \sim \mathcal{N}(0, \mathbf{I})$; $\widehat{x}_0 \leftarrow \textsc{SampleDiT}(\epsilon, C)$
$\widehat{I} \leftarrow D_{\text{img}}(\text{Denormalize}(\widehat{x}_0))$
return $\mathcal{O}, A, \widehat{I}$

B Experimental Details

B.1 Datasets and Preprocessing

We train DeOR on COCO train2017 images (Lin et al., 2014) using object descriptions and masks from Pix2Cap-COCO (You et al., 2025). The training set contains 16,477 unique images. We use countable object instances as training targets and exclude background and stuff regions from the description sequence. Each description is paired with its corresponding mask.

We evaluate object decomposition on all 5,000 images in COCO val2017 and the 1,449 images in the PASCAL VOC 2012 validation split (Everingham et al., 2010). No additional training or adaptation is performed on VOC. For description–region alignment, we use the val, testA, and testB splits of

RefCOCO (Kazemzadeh et al., 2014). After excluding images that appear in DeOR's Pix2Cap-COCO training set, the evaluation set contains 2,600 images and 18,906 referring expressions. Overlap is checked against this DeOR training set.

For object decomposition on COCO and VOC, we resize the shorter side of each image to 512 pixels and take a central crop of 512×512 pixels. We apply the same transformation to the ground-truth masks, using nearest-neighbor interpolation to preserve discrete labels. Ground-truth object counts are determined from the instances remaining in the evaluation region after preprocessing. For COCO, pixels covered by multiple ground-truth instances are assigned to background in the ground-truth label map. Regions marked with ignore labels in VOC are excluded from evaluation. The source encoder receives images at 512×512 resolution, while the target encoder receives the same image region at 224×224 resolution for reconstruction training.

B.2 Model Configuration and Training Settings

DeOR-B and DeOR-L use the same object representation structure and training objectives, with different language backbone and DiT sizes. Unless otherwise specified, experiments use DeOR-B. We use DeOR-L for compositional generation and reconstruction. Table 8 summarizes both configurations. We fix the ownership, Writer, and Reader temperatures to $\tau_{\text{own}} = \tau_W = \tau_R = 1.0$.

Table 8: **Model configurations of DeOR-B and DeOR-L.**

Setting	DeOR-B	DeOR-L
Language backbone (Qwen et al., 2024)	Qwen2.5-1.5B	Qwen2.5-7B
DiT size (Tong et al., 2026)	2.4B	9.8B
Source encoder input resolution	512×512	512×512
Target encoder input resolution	224×224	224×224
Image patch tokens	1,024	1,024
Spatial query tokens	256	256
Visual memory vectors per object	8	8
Register tokens	4	4
Visual memory vectors per register	32	32
Reader layers	1	1
Reader attention heads	8	8
LoRA rank	16	16
LoRA scaling parameter α	32	32
LoRA dropout	0.05	0.05

We use `google/siglip2-so400m-patch16-512` as the source encoder. The target encoder is `google/siglip2-so400m-patch14-224` (Tschannen et al., 2025). The language backbones and DiTs are initialized from the corresponding pretrained Scale-RAE models (Tong et al., 2026). Both variants use the pretrained SigLIP2 image decoder released with Scale-RAE. We apply LoRA (Hu et al., 2022) to the query, key, value, and output projections in the MLLM. We also train the embeddings and projections used to construct object representations, the memory Writer, and the hierarchical Reader. For the DiT, we train the final 16 Transformer blocks and all AdaLN modulation parameters (Peebles & Xie, 2023). We use AdamW (Loshchilov & Hutter, 2019) with the optimization settings and learning rates listed in Table 9.

Training combines language modeling, grounding, Reader supervision, and reconstruction objectives. Their definitions and weights follow Section 2.

B.3 Evaluation Protocols

Object decomposition and counting. For object decomposition, DeOR autoregressively generates object descriptions and object tokens from an image and a fixed prompt. Ground-truth descriptions, object counts, and masks are not provided as inputs. We use greedy decoding with a budget of 512 generated tokens and retain at most 50 objects for subsequent processing. When generation is truncated, only object entries that include their object token are retained as object representations.

Table 9: **Training settings.** Unless otherwise indicated, settings are shared by DeOR-B and DeOR-L.

Setting	Value
Optimizer	AdamW
Default learning rate	5×10^{-5}
MLLM learning rate	1×10^{-4}
Diffusion head learning rate	3×10^{-5}
DiT body learning rate	1×10^{-5}
Image projector learning rate	1×10^{-5}
Register and spatial query learning rate	5×10^{-5}
Weight decay	0.01
Learning-rate schedule	Constant after warmup
Warmup ratio	0.03
Gradient clipping	1.0
Effective batch size	DeOR-B: 24 / DeOR-L: 32
Training steps	5,000

Incomplete entries do not contribute to the predicted object count. If no complete object entry is generated, the predicted count is zero.

We group images into four ranges based on the number of ground-truth instances after preprocessing: 1–3, 4–6, 7–10, and 11+. Images with no ground-truth objects remain in the evaluation set and are included when computing object-count MAE. For DeOR, the predicted count is the number of valid generated object tokens. For AdaSlot (Fan et al., 2024), we count the representations selected by the model's keep decisions. For MetaSlot (Liu et al., 2025), we count the representations remaining after duplicate removal. For CODA (Nguyen et al., 2026), we use the fixed number of representations specified by the released checkpoint. Active summarizes these per-image counts as the mean and standard deviation over the same evaluation images used for count MAE.

Predicted regions and background. For DeOR, the final segmentation map is constructed from the K object ownership maps and R register ownership maps. We first sum the register probabilities at each spatial location to form a single background probability, yielding $K + 1$ competing channels. The resulting 32×32 probability maps are bilinearly upsampled to the evaluation resolution, after which a pixel-wise argmax produces the final label map. Pixels assigned to the aggregated register channel receive the background label. On COCO, stuff regions are also treated as background.

Decomposition metrics. We evaluate decomposition using FG-ARI (Hubert & Arabie, 1985), mBO, and mIoU (Kakogeorgiou et al., 2024). FG-ARI is computed only over ground-truth foreground pixels. Pixels predicted as background remain in the predicted background cluster and therefore still affect the clustering score. For mBO, ground-truth background is excluded, while the predicted background region remains a candidate region; each ground-truth foreground object is independently matched to the predicted region with the highest IoU. For mIoU, both ground-truth and predicted background are included, and one-to-one matching is performed between all ground-truth and predicted regions. Unmatched ground-truth regions contribute zero.

COCO val2017 contains 54 images with no foreground instance after preprocessing. Our FG-ARI implementation assigns a score of 1.0 to these degenerate cases, and FG-ARI is therefore averaged over all 5,000 images. mBO is undefined for these images and is averaged over the remaining 4,946 images, whereas mIoU is computed over all 5,000 images because background is included. The same conventions are applied to all reevaluated methods.

For all baseline checkpoints that we reevaluate, we use the same image preprocessing, ground-truth preprocessing, region construction, background handling, matching procedure, and metric implementation as for DeOR. Baseline soft masks are bilinearly resized to the evaluation resolution before applying the same argmax-based conversion to predicted regions.

Transfer across datasets. We evaluate COCO-trained models on VOC without additional training. Instance-level evaluation uses separate ground-truth masks for individual objects, whereas class-level evaluation merges instances of the same semantic class; both use the same predicted regions and the common evaluator described above.

Alignment between object descriptions and predicted regions. On RefCOCO, DeOR and CTRL-O (Didolkar et al., 2025) receive the same image and referring expression. DeOR forms an object token conditioned on the expression and predicts the region associated with that token. Neither model undergoes additional training on RefCOCO. Ground-truth masks are used only to compute evaluation metrics. The models differ in training supervision: DeOR is trained with paired object descriptions and masks, whereas CTRL-O is trained without direct object-mask supervision.

For DeOR, we use the supplied referring expression as a single object description and construct one corresponding object token. The resulting object ownership map competes with a single background map obtained by summing all register ownership probabilities. We bilinearly upsample the object and background maps to the original image resolution and apply a pixel-wise argmax between the two channels. Pixels assigned to the object channel form the predicted binary mask.

This evaluation does not use a mask threshold, ground-truth category label, box, point, or mask prompt, Hungarian matching, or ground-truth-based selection of an object representation. The supplied expression directly conditions the single evaluated object representation, and the resulting predicted mask is compared with the referred ground-truth mask.

We compute mIoU by averaging the IoU between the predicted mask and the target mask over all referring expressions. TargetHit@1 is the percentage of expressions for which the predicted mask has a uniquely highest IoU with the referred object among the ground-truth objects in the image. Empty predictions, predictions with no overlap with the target, and ties for the highest IoU are counted as failures. Overall results are averaged over all evaluation expressions, rather than taking an unweighted average of the split-level scores. TargetHit@1 evaluates the single predicted mask associated with each expression.

For same-category evaluation, Same-cat. TargetHit@1 compares the predicted mask only against ground-truth instances with the same COCO category as the referred object. An expression is counted as correct when the predicted mask has positive IoU with the referred object and a strictly higher IoU with it than with every other ground-truth instance of the same category. Ties and zero overlap with the referred object are counted as failures. We evaluate this metric over the same 18,906 referring expressions in the non-overlapping RefCOCO evaluation set. Category labels are used only to define the candidate instances and are not provided to the model. After excluding the target instance itself, every expression contains at least one other ground-truth instance of the same COCO category. The number of same-category distractors per expression has mean 4.05, median 3, and range 1–13.

Generated-description object selection. We additionally evaluate whether object descriptions generated by DeOR can be used to identify the intended object representation. For each image, DeOR first performs its standard autoregressive inference to generate object descriptions and object tokens, together with the predicted region associated with each object representation. Qwen3-VL-4B-Instruct (Bai et al., 2025) then receives a RefCOCO referring expression and a textual list of the generated object IDs and candidate descriptions. It receives neither the image, the predicted regions, nor ground-truth information. The selector returns one object ID, and we evaluate the predicted region associated with that ID against the RefCOCO target mask.

To isolate the information provided by the generated descriptions, we compare two selector inputs while keeping the generated object representations and predicted regions fixed. In *Category only*, each candidate description is replaced by its category name, while in *Generated description*, the selector receives the full description generated by DeOR. Coverage@0.5 is the percentage of expressions for which at least one candidate region has IoU at least 0.5 with the target. Success@0.5 | Covered is the percentage of these covered expressions for which the selected candidate also reaches IoU 0.5. E2E Success@0.5 measures the same success criterion over all expressions, and E2E mIoU averages the IoU of the selected region with the target over all expressions. We evaluate the non-overlapping RefCOCO val, testA, and testB splits described in Appendix B.1; together they contain 2,600 images and 18,906 referring expressions. Results are reported in Appendix C.4.

Image reconstruction and compositional generation. For reconstruction, DeOR autoregressively extracts object descriptions and representations from the image and uses them to reconstruct the original image. For compositional generation, DeOR first autoregressively generates object descriptions for each evaluation image. We then use these generated descriptions as the teacher-forced sequence to construct object representations before recombining them across images.

Compositional generation is performed within batches of four images. For each target image, we sample the same number of object representations from the other images in the batch. Each object's semantic representation and visual memories are transferred together. Donor objects are sampled without replacement unless the donor pool is smaller than the required number of objects, in which case sampling is performed with replacement. All register representations and their associated visual memories are taken together from one randomly selected image other than the target.

We generate 5,000 compositional images, one for each image in COCO val2017, and use the same 5,000 real images as the reference distribution for FID (Heusel et al., 2017) and KID (Bińkowski et al., 2018). Generation uses Euler sampling with 10 scheduled timesteps and a guidance scale of 1.0. The random seed for compositional generation is fixed at 1234. Generated images have a resolution of 224×224. Both generated and real reference images are resized to 256×256 before being passed to the metric implementation. We compute FID and KID using the Inception-based implementations in TorchMetrics (Detlefsen et al., 2022). KID is averaged over 100 subsets of 1,000 samples and multiplied by 10^3 when reported. We report rFID and KID for reconstruction, and FID and KID for compositional generation. Baseline generation scores are taken from published results (Nguyen et al., 2026), as indicated in the corresponding tables.

Object editing and ablations. For language-guided editing, Qwen3-VL-4B-Instruct (Bai et al., 2025) receives the user instruction and a list of object IDs and descriptions generated by DeOR. The external model receives neither images nor ground-truth information. DeOR edits the object representations according to the returned IDs and generates the result through the Reader, DiT, and image decoder.

Table 10 summarizes the representation components modified by each manipulation operation, where A and B denote the source and donor images, respectively. Edited and unedited outputs use the same initial noise and sampling settings.

Table 10: Definition of object manipulation operations. Each object representation consists of a semantic state s and visual memories Z. Source and donor images are denoted by A and B, respectively.

Operation	Object representation	Register / background
Removal	Remove (s_k^A, Z_k^A); recompute the visual memories of the remaining source objects	Keep source register states and recompute their memories
Appearance transfer	$(s_k^A, Z_k^A) \rightarrow (s_k^A, Z_j^B)$	Keep source-A registers
Replacement	$(s_k^A, Z_k^A) \rightarrow (s_j^B, Z_j^B)$	Keep source-A registers
Addition	Append donor (s_j^B, Z_j^B) to the source owner set	Keep source-A registers
Background transfer	Keep source-A object representations	Use donor-B register states and cleaned donor register memories

Manipulation details. For removal, the pre-edit ownership map is used only to define the target region, which is dilated by one source patch and excluded when recomputing the visual memories of the remaining objects and registers; ownership itself is not recomputed. Appearance transfer and replacement do not rerun the Writer for the remaining source representations. Replacement and addition use no explicit spatial placement constraint, and an added donor object may therefore retain a tendency toward its donor spatial configuration. For background transfer, donor register memories are constructed after excluding donor foreground regions, while the source readout queries are retained.

Quantitative visual-memory transfer. To quantify the role of Reader supervision in object-level visual control, we compare the full model and a model trained without Reader supervision on the same set of 100 randomly sampled target–donor pairs. The sampled pairs are fixed and shared by both models. The target and donor object IDs are fixed for this evaluation so that language-selection errors are not mixed with the effect of Reader supervision, and the edited and unedited outputs use the same initial diffusion noise. We compute target-region features in the frozen SigLIP2 reconstruction-target space using `google/siglip2-so400m-patch14-224` (Tschannen et al., 2025). The encoder produces a 16×16 grid of 1,152-dimensional patch features. We resize the ground-truth object mask

to this grid using area interpolation and compute a mask-weighted pooled feature

$$f(\mathbf{X}, \mathbf{M}) = \frac{\sum_p M_p \mathbf{X}_p}{\sum_p M_p}. \tag{19}$$

The pooled features are ℓ_2-normalized before cosine similarity. Ground-truth masks are used only for evaluation. Let f_{donor} denote the pooled donor-object feature, and let f_{unedited} and f_{edited} denote target-region features from the unedited and edited generations produced from the same diffusion noise. We define the donor gain as

$$\Delta S_{\text{donor}} = \cos(f_{\text{edited}}, f_{\text{donor}}) - \cos(f_{\text{unedited}}, f_{\text{donor}}), \tag{20}$$

and target change as

$$D_{\text{target}} = 1 - \cos(f_{\text{unedited}}, f_{\text{edited}}). \tag{21}$$

Higher donor gain indicates stronger transfer of donor visual information to the target, while higher target change indicates a larger change within the target region. We additionally report Non-target LPIPS between the unedited and edited outputs outside the target mask; lower values indicate smaller changes to the remaining image. Results are reported in Appendix D.4.

All ablations use DeOR-B. In the training-objective ablation, we remove the objective under examination while retaining the remaining objectives. In the description-conditioning ablation, we separately control description-based object token initialization and attention to the token's own description, while retaining description generation and the language modeling loss. In the visual memory ablation, we vary the number of memory vectors per object and evaluate generation quality.

C Additional Quantitative Results

C.1 Object Decomposition on COCO

Table 11: **Comparison of object decomposition performance with prior object-centric learning methods on COCO.** We report FG-ARI and instance- and class-level mBO and mIoU where available. Superscripts i and c denote instance-level and class-level evaluation, respectively. † denotes results reported by Wu et al. (2023), except for SlotDiffusion's mIoU scores, which are taken from Nguyen et al. (2026). Higher values indicate better performance.

COCO	FG-ARI↑	mBO^i↑	mBO^c↑	mIoU^i↑	mIoU^c↑
SA†	21.4	17.2	19.2	–	–
SLATE†	32.5	29.1	33.6	–	–
DINOSAUR	34.1	31.6	39.7	–	–
SPOT	37.8	35.0	44.7	33.0	–
SlotDiffusion†	37.2	31.0	35.0	31.2	36.5
Stable-LSD	35.0	30.4	–	–	–
SlotAdapt	41.4	35.1	39.2	36.1	41.4
MetaSlot	41.2	29.5	–	27.9	–
CTRL-O	47.5	27.2	–	–	–
CODA	47.5	36.6	41.4	36.4	42.6
DeOR-B	58.9	**53.5**	**55.0**	**58.0**	**63.7**
DeOR-L	**61.5**	53.3	54.0	57.9	62.9

To complement the analysis across object counts, we compare overall decomposition performance with a broader set of object-centric methods on COCO (Lin et al., 2014). DeOR autoregressively generates object descriptions and object tokens from each image, without ground-truth descriptions, object counts, or masks as inputs. We evaluate all 5,000 images in COCO val2017. Table 11 reports FG-ARI (Hubert & Arabie, 1985) and instance- and class-level mBO and mIoU (Kakogeorgiou et al., 2024). Baseline results are taken from Nguyen et al. (2026), with additional results for MetaSlot (Liu et al., 2025) and CTRL-O (Didolkar et al., 2025). We interpret these comparisons in light of differences in training supervision.

DeOR-B and DeOR-L outperform the evaluated baselines on all five reported metrics. Instance-level results measure how well the predictions separate individual objects, while class-level results measure how well they recover regions belonging to each semantic category.

C.2 Cross-Dataset Object Decomposition on PASCAL VOC

Table 12: **Full cross-dataset object decomposition comparison on PASCAL VOC.** The upper block reports VOC-trained in-domain references, and the lower block reports COCO-trained models evaluated on VOC without additional adaptation. Superscripts i and c denote instance- and class-level metrics. † denotes results reported by Wu et al. (2023), except SlotDiffusion's mIoU scores, which are taken from Nguyen et al. (2026); other VOC-trained results are from the original papers.

VOC	FG-ARI↑	mBO^i↑	mBO^c↑	mIoU^i↑	mIoU^c↑
In-domain (VOC-trained, VOC-eval)					
SA†	12.3	24.6	24.9	–	–
SLATE†	15.6	35.9	41.5	–	–
DINOSAUR	24.8	44.0	51.2	–	–
SPOT (w/ENS)	19.7	48.3	55.6	46.8	–
SlotDiffusion†	17.8	50.4	55.3	44.9	49.3
SlotAdapt	29.6	51.5	51.9	–	–
MetaSlot	43.3	43.9	–	42.1	–
CODA	32.2	55.4	61.3	50.8	56.3
Zero-shot (COCO-trained, VOC-eval)					
SPOT	21.1	56.5	64.0	52.3	59.0
MetaSlot	26.7	43.0	45.6	42.5	45.4
CODA	30.8	57.4	58.9	52.4	54.2
DeOR-B	44.0	79.3	76.7	79.8	**79.8**
DeOR-L	**44.9**	**79.9**	**76.9**	**80.3**	**79.8**

To complement the compact comparison in Table 2, Table 12 reports the full set of VOC-trained references together with the COCO-trained models evaluated without additional training.

C.3 Image Reconstruction

Table 13: **Image reconstruction on COCO.** We report rFID and KID, with KID values multiplied by 10^3. Baseline results are taken from Nguyen et al. (2026). Lower values indicate better performance.

Metric	LSD	SlotDiffusion	SlotAdapt	CODA	**DeOR-L**
KID$\times 10^3$	19.09	5.85	0.39	**0.35**	1.60
rFID	35.54	19.45	10.86	**10.65**	12.93

We evaluate image reconstruction using the learned object representations. DeOR-L autoregressively extracts object descriptions and representations from the input image, then reconstructs the image using the object and register representations. Table 13 reports rFID (Heusel et al., 2017) and KID (Bińkowski et al., 2018) on COCO, with KID values multiplied by 10^3. Lower values indicate better performance for both metrics. Baseline results are taken from Nguyen et al. (2026).

DeOR-L achieves an rFID of 12.93 and a KID $\times 10^3$ of 1.60. These values are lower than the reported results for LSD (Jiang et al., 2023) and SlotDiffusion (Wu et al., 2023), but higher than those for SlotAdapt (Akan & Yemez, 2025) and CODA. CODA reports an rFID of 10.65 and a KID $\times 10^3$ of 0.35. Although CODA achieves better reconstruction scores, DeOR-L achieves lower FID and KID in compositional generation.

C.4 End-to-End Object Selection with Generated Descriptions

Section 3.4 evaluates whether a supplied RefCOCO expression can be grounded to its corresponding image region. Here, we ask a different question: whether descriptions generated by DeOR itself help identify the intended object representation among the objects that DeOR generates from an image. The evaluation protocol and metric definitions are provided in Appendix B.3.

The Category-only and Generated-description conditions use exactly the same object representations and predicted regions. Across the 18,906 non-overlapping RefCOCO expressions, their shared Coverage@0.5 is 75.75%. The two conditions therefore differ only in the textual information available to Qwen3-VL-4B-Instruct (Bai et al., 2025) when selecting an object ID.

Table 14 summarizes the overall results. Success@0.5 | Covered measures selection only on expressions for which the generated object pool already contains at least one region with IoU at least 0.5 with the target. Providing only category names yields 27.20%, whereas providing the full descriptions generated by DeOR increases this value to 52.46%. E2E Success@0.5 includes both candidate coverage and selection, and increases from 20.61% to 39.74%. E2E mIoU, which measures the overlap of the selected region without thresholding at 0.5, similarly increases from 18.27 to 34.33. These results show that the generated descriptions provide instance-specific information that helps the selector distinguish the intended representation beyond category information alone.

Table 14: **End-to-end object selection using DeOR-generated descriptions.** Both conditions use the same generated object candidates, whose shared Coverage@0.5 is 75.75%. All values are percentages.

Method	Success@0.5 | Covered ↑	E2E Success@0.5 ↑	E2E mIoU ↑
Category only	27.20	20.61	18.27
Generated description	**52.46**	**39.74**	**34.33**

The absolute E2E success rate also indicates remaining room for improvement: some referred objects are not covered by any generated candidate at IoU 0.5, and selection errors remain even when a suitable candidate exists. We therefore interpret this experiment as evidence that generated descriptions are substantially more useful instance identifiers than category labels, rather than as evidence of perfect end-to-end selection reliability.

Table 15 reports the same E2E Success@0.5 comparison separately on val, testA, and testB. The generated-description condition improves over Category only on all three splits. Absolute E2E performance is highest on testA, where Coverage@0.5 is also highest, and lowest on testB, where candidate coverage is lower. The consistent improvement across all splits shows that the selection benefit of generated descriptions is not confined to a single RefCOCO split.

Table 15: **End-to-end object selection across RefCOCO splits.** Coverage@0.5 measures whether the generated object pool contains a candidate region with IoU at least 0.5 with the target. Category only and Generated description report E2E Success@0.5, and Δ E2E is their difference in percentage points.

Split	Coverage@0.5 ↑	Category only	Generated description	Δ E2E
val	75.98	19.76	**38.98**	**+19.22**
testA	82.41	22.05	**45.09**	**+23.04**
testB	67.49	20.79	**35.17**	**+14.37**

D Additional Analyses

D.1 Matched Fixed Query Baseline

To examine whether DeOR's decomposition performance can be explained by paired supervision alone, we construct matched FixedQuery baselines by replacing DeOR's description-conditioned

object states with a fixed set of learnable object queries. We evaluate $K = 7$ and $K = 16$, while retaining the same training data, language-modeling and paired-mask supervision, MLLM and visual backbones, registers, Writer, Reader, DiT reconstruction pathway, optimization settings, and training schedule as DeOR-B. The autoregressive description-generation branch and its language-modeling objective are also retained, but the generated object count is independent of the fixed number K of object owners used for decomposition.

During training, the fixed queries are matched one-to-one to the ground-truth object masks using Hungarian matching, and the grounding objective is applied only to the matched queries. Unmatched queries are excluded from the grounding loss. Consequently, when an image contains fewer than K annotated objects, some fixed queries receive no direct grounding supervision, whereas when the number of annotated objects exceeds K, at most K objects can receive distinct fixed-query assignments.

Table 16: Matched comparison between FixedQuery baselines and DeOR-B on COCO and PASCAL VOC. FixedQuery uses a fixed set of K learnable object queries, whereas DeOR dynamically constructs object representations from generated descriptions and object tokens. All values are percentages.

Dataset	Method	Allocation	FG-ARI ↑	mBO^i ↑	mBO^c ↑	mIoU^i ↑	mIoU^c ↑
COCO	FixedQuery	Fixed $K = 16$	52.0	32.1	35.3	39.2	46.6
	FixedQuery	Fixed $K = 7$	**62.3**	36.0	38.4	41.7	48.4
	DeOR-B	Dynamic	58.9	**53.5**	**55.0**	**58.0**	**63.7**
VOC	FixedQuery	Fixed $K = 16$	58.3	42.3	45.0	49.1	53.2
	FixedQuery	Fixed $K = 7$	**59.7**	45.6	44.8	50.8	52.3
	DeOR-B	Dynamic	44.0	**79.3**	**76.7**	**79.8**	**79.8**

Table 16 shows that FixedQuery remains competitive in foreground clustering: the $K = 7$ baseline achieves higher FG-ARI than DeOR-B on both COCO and VOC. In contrast, DeOR-B substantially improves region-overlap metrics. Relative to FixedQuery with $K = 7$, DeOR-B improves instance-level mBO and mIoU by 17.5 and 16.3 points on COCO, and by 33.7 and 29.0 points on VOC, respectively. The same trend holds for class-level mBO and mIoU.

These results indicate that a fixed set of learned queries can form effective foreground clusters, but is substantially less accurate at recovering the spatial extent of individual objects. DeOR's description-conditioned dynamic representations instead achieve consistently stronger instance- and class-level region recovery under the matched training setting.

We further examine how the fixed representation capacity interacts with scene cardinality. Table 17 reports instance-level mBO and mIoU on COCO across the same ground-truth object-count ranges used in Table 1.

Table 17: Instance-level region overlap on COCO across ground-truth object-count ranges. Each entry reports mBO^i / mIoU^i. FixedQuery uses the same number of object queries for every image, whereas DeOR uses an image-dependent number of object representations.

Method	1–3		4–6		7–10		11+	
	mBO^i	mIoU^i	mBO^i	mIoU^i	mBO^i	mIoU^i	mBO^i	mIoU^i
FixedQuery $K = 16$	35.9	46.4	28.5	33.3	30.8	33.0	26.2	26.4
FixedQuery $K = 7$	39.6	47.6	40.5	44.3	35.0	37.3	21.3	21.4
DeOR-B	**66.6**	**72.6**	**49.1**	**53.3**	**37.4**	**40.0**	**28.6**	**29.4**

The preferred fixed allocation changes with scene cardinality. FixedQuery with $K = 7$ outperforms $K = 16$ in the 1–3, 4–6, and 7–10 ranges, whereas $K = 16$ performs better in the 11+ range. This reflects a structural capacity trade-off: a small fixed K limits the number of objects that can receive distinct query assignments in crowded scenes, while a larger K leaves more unmatched queries in images containing fewer objects.

DeOR-B avoids selecting a single fixed capacity and achieves the highest mBO and mIoU in every COCO object-count range. The advantage is largest in the 1–3 range, where DeOR-B exceeds the stronger fixed baseline by 27.0 points in mBO and 25.0 points in mIoU, while retaining the highest overlap scores as the number of objects increases.

Overall, the matched FixedQuery comparison shows that DeOR's region-level decomposition performance cannot be attributed to paired mask supervision alone. Fixed learned queries can achieve strong foreground clustering, but their region-overlap performance is substantially lower and their preferred capacity varies with scene cardinality. In contrast, DeOR jointly constructs the semantic identity and number of object representations from generated descriptions, supporting the benefit of description-conditioned dynamic representation formation for accurate object-region recovery.

D.2 Count Behavior and Token-Budget Limits

Table 18 examines the count behavior underlying the larger count MAE of DeOR-L despite its higher FG-ARI than DeOR-B. We define the mean count error as the predicted object count minus the ground-truth count, so a positive value indicates an overall tendency to over-count. We evaluate all 5,000 images in COCO val2017 and all 1,449 images in the PASCAL VOC validation split.

Table 18: Count prediction behavior of DeOR-B and DeOR-L. Mean count error is defined as predicted count minus ground-truth count; positive values indicate over-counting. Results are reported on all 5,000 COCO val2017 images and all 1,449 PASCAL VOC validation images.

Dataset	Model	Count MAE ↓	Mean count error	Over-count (%)
COCO	DeOR-B	**1.45**	-0.12	31.55
	DeOR-L	2.27	$+0.99$	41.99
VOC	DeOR-B	**1.33**	$+1.22$	40.58
	DeOR-L	1.91	$+1.86$	43.55

Table 18 shows that the larger count MAE of DeOR-L is associated with stronger over-counting. On COCO, its mean count error is $+0.99$, compared with -0.12 for DeOR-B, and the fraction of images with more predicted than ground-truth objects increases from 31.55% to 41.99%. The same tendency appears on VOC, where the mean count error increases from $+1.22$ to $+1.86$. Thus, the higher count error of DeOR-L is primarily associated with a stronger tendency to predict counts above the annotated ground-truth count.

We next examine whether the autoregressive decoding budget limits variable-count inference. Table 19 reports the fraction of COCO images for which generation reaches the 512-token budget before emitting `<scene_end>`, grouped by ground-truth object count.

Table 19: Rate of 512-token truncation across COCO object-count ranges. Values are the percentage of images for which autoregressive generation reaches the 512-token decoding budget before emitting `<scene_end>`. Results are computed over all 5,000 COCO val2017 images. Lower is better.

GT object count	DeOR-B	DeOR-L
All	7.90	19.72
1–3	0.21	1.56
4–6	1.82	10.63
7–10	8.86	38.96
11+	48.47	85.76

The token budget is rarely reached for images containing few objects, but becomes a clear constraint as scene complexity increases. In contrast, for 7–10 objects the rates increase to 8.86% and 38.96%, and for 11+ objects they reach 48.47% and 85.76%, respectively. DeOR-L is affected more strongly, consistent with its tendency to generate more object representations and therefore longer autoregressive sequences. Accordingly, the 11+ FG-ARI result should be interpreted as foreground clustering quality under the fixed decoding budget, rather than complete recovery of all objects.

Neither model reaches the 50-object limit on COCO or VOC, indicating that the object-count cap is not an active constraint in the current evaluation. Overall, DeOR-L exhibits a stronger over-generation tendency, which is associated with higher count error and longer sequences, while the fixed 512-token budget becomes an important decoding constraint in crowded scenes. These results motivate more concise object descriptions and more efficient stopping or allocation mechanisms for variable-count inference in scenes containing many objects.

D.3 Spatial Locality of Object Visual Memories

The grounding loss directly supervises the ownership correspondence between semantic object states and image regions, but the Writer does not directly mask its attention with the ownership map. We therefore test whether grounding also changes where object visual memories collect their source information. We compare the Writer attention weights $w_{a,\ell,j}$ in Eq. 14 from the full model and the model trained without grounding on the same 512 images.

We use three quantities to characterize locality. Object inside-mask mass is the fraction of an object's Writer attention assigned to its corresponding ground-truth object mask; higher values indicate that the visual memories collect more information from their own object region. Because this quantity naturally depends on object size, we also report area-normalized enrichment, defined as the inside-mask attention mass divided by the fraction of image area occupied by the object. An enrichment of one corresponds to the concentration expected from spatially uniform attention, while larger values indicate stronger concentration on the object relative to its area. Finally, Register annotated-object mass measures the fraction of register-memory attention assigned to annotated object regions; lower values indicate a stronger separation between object memories and residual register memories.

Table 20: **Spatial locality of Writer visual memories.** Object inside-mask mass measures the fraction of object-memory attention inside the corresponding ground-truth region. Area-normalized enrichment corrects this quantity for object area. Register annotated-object mass measures register attention assigned to annotated object regions.

Metric	Full DeOR	w/o Grounding
Object inside-mask mass ↑	**0.3886**	0.1120
Area-normalized enrichment ↑	**22.92**	2.82
Register annotated-object mass ↓	**0.2590**	0.3688

Table 20 shows that grounding substantially increases Writer locality. Object inside-mask mass rises from 0.1120 without grounding to 0.3886 in the full model. After normalizing for object area, enrichment increases from 2.82 to 22.92. Register annotated-object mass decreases from 0.3688 to 0.2590. Thus, although the grounding objective does not directly mask Writer attention, it leads object memories to concentrate substantially more strongly on their corresponding regions and reduces the amount of register attention assigned to annotated objects. We interpret this as increased object-local concentration rather than strict spatial separation, since both object and register memories may still attend outside their nominal regions. We follow the COCO area convention and group objects by mask area: small objects have area below 32^2 pixels, medium objects have area between 32^2 and 96^2 pixels, and large objects have area at least 96^2 pixels.

Table 21: **Writer locality across object sizes.** Inside-mask mass is reported together with area-normalized enrichment for small, medium, and large objects. Grounding increases locality across all size groups.

Object size	Inside-mask mass ↑		Enrichment ↑	
	Full	w/o Grounding	Full	w/o Grounding
Small	**0.0837**	0.0068	**46.45**	3.68
Medium	**0.2797**	0.0412	**23.05**	2.94
Large	**0.7328**	0.2747	**7.34**	2.10

Table 21 further breaks the analysis down by object size. Inside-mask mass increases with object size, as larger objects occupy more of the image. Grounding nevertheless improves both inside-mask mass and area-normalized enrichment for small, medium, and large objects. The relative concentration after area normalization is strongest for small objects: although their absolute inside-mask mass is lower, their much smaller spatial extent yields an enrichment of 46.45 in the full model compared with 3.68 without grounding. The consistent gap across all three size groups shows that the locality effect is not driven by a single object-size regime.

D.4 Effect of Reader Supervision on Object Editing

In Table 5, the model trained without Reader supervision achieves slightly higher decomposition scores than the model trained with all objectives. The qualitative editing results in Figures 5 and 9, however, suggest a different role for Reader supervision: without it, object removal remains effective, but transferring donor visual information through the target object's memory has little visible effect. We therefore quantify this behavior using the 100-pair visual-memory transfer evaluation defined in Appendix B.3.

Table 22: **Quantitative visual-memory transfer with and without Reader supervision.** Donor gain measures increased similarity of the edited target to the donor, Target change measures the change within the target region, and Non-target LPIPS measures changes outside the target object.

Model	Donor gain ↑	Target change ↑	Non-target LPIPS ↓
Full DeOR	**0.1156**	**0.1157**	0.0734
w/o Reader supervision	0.0128	0.0050	**0.0106**

Table 22 reports three complementary quantities. Donor gain measures how much the edited target-region feature moves toward the donor object in the frozen SigLIP2 reconstruction-target space. Target change measures how much the target-region feature changes from the unedited generation. Non-target LPIPS measures perceptual changes outside the target object, where lower values indicate smaller changes to the remaining image.

The full model achieves a donor gain of 0.1156, compared with 0.0128 without Reader supervision. Target change similarly drops from 0.1157 to 0.0050 when Reader supervision is removed, showing that the target representation itself changes only minimally in the ablated model. The lower Non-target LPIPS of the model without Reader supervision should therefore not be interpreted as stronger localized control: the target itself is also barely changed. In contrast, the full model produces substantially stronger donor transfer, while also causing larger changes outside the target region. These results support the role of Reader supervision in making object visual memories effective control variables for generation, while also revealing a remaining trade-off between transfer strength and preservation of non-target content.

Grounding loss and Reader supervision supervise different correspondences. Grounding loss supervises ownership, which specifies the object associated with each image patch. Reader supervision uses this ownership to train the Reader's object-level attention to access the corresponding object's visual memory at each target location. Accurately predicting an object's region and controlling its generated appearance by modifying its memory are therefore distinct properties. The quantitative results above and the qualitative examples in Figure 9 show that the benefit of Reader supervision is evident in the effect of replacing an object's visual memory, rather than in higher decomposition scores.

Object removal and visual-memory transfer also differ in the scope of the intervention. For removal, we discard the selected object's representation and recompute the visual memories of the remaining objects and registers while excluding source patches in the removal region. For visual-memory transfer, we retain the target object's semantic representation and replace its visual memories with those of the donor object. The latter operation tests whether changing an individual object's memory can control its generated visual content. The preserved removal performance and weakened memory-transfer effect without Reader supervision support retaining Reader supervision in the default model.



Figure 9: **Object editing without Reader supervision.** We evaluate DeOR-B trained without Reader supervision using the same images and editing instructions as in Figure 5. Object removal remains effective in these examples (left), while donor visual information is not clearly transferred to the target (right). Unedited reconstructions are shown for comparison.

E Additional Qualitative Results

E.1 Object Decomposition

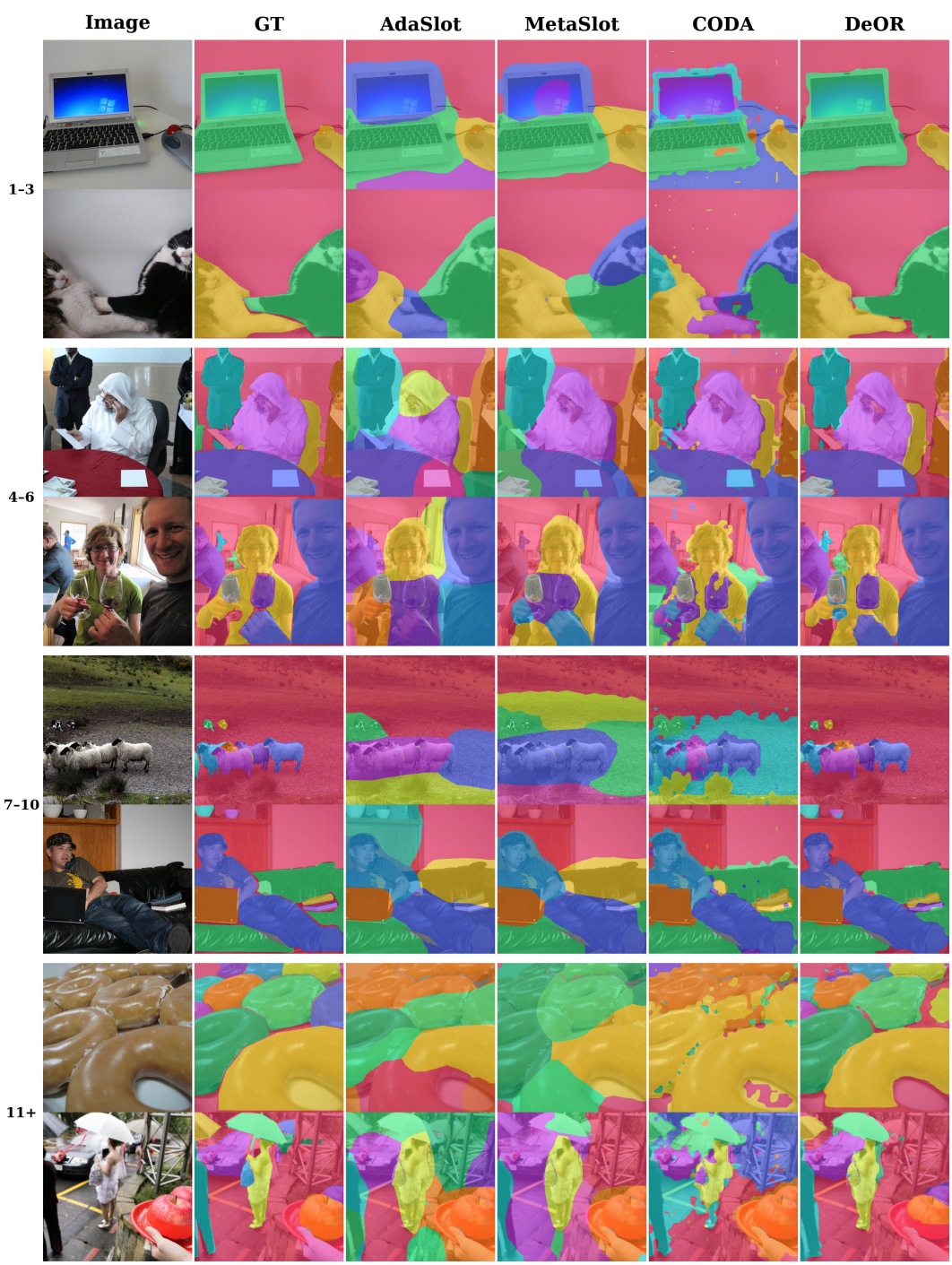


Figure 10: **Additional object decomposition results on COCO.** Examples are grouped by ground-truth instance count: 1–3, 4–6, 7–10, and 11+. Columns show the input image, ground-truth masks, and predictions from AdaSlot, MetaSlot, CODA, and DeOR. Different colors indicate different regions.

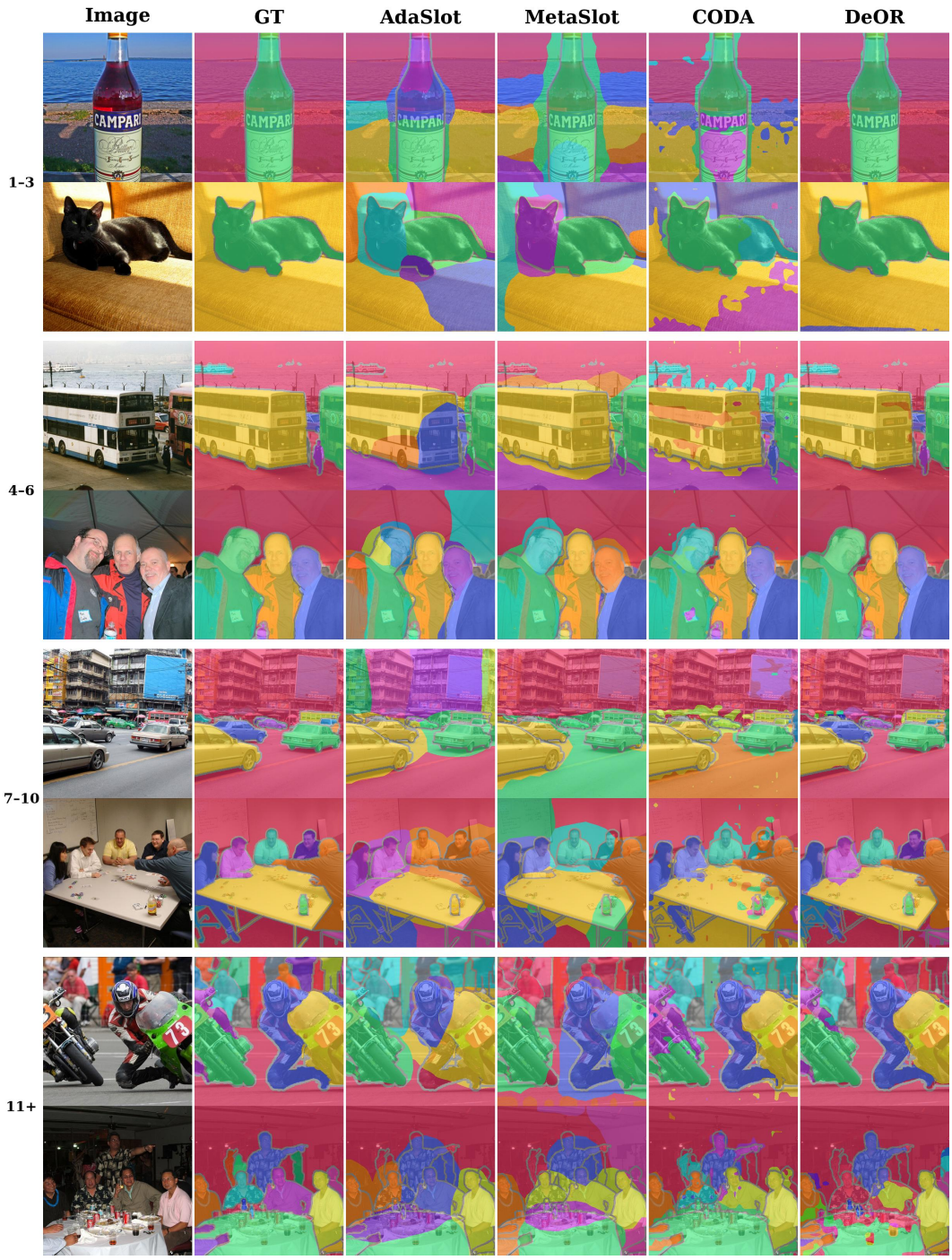


Figure 11: **Additional object decomposition results on PASCAL VOC.** Models trained on COCO are evaluated without additional training on VOC. Examples are grouped by ground-truth instance count: 1–3, 4–6, 7–10, and 11+. Columns show the input image, ground-truth masks, and predictions from AdaSlot, MetaSlot, CODA, and DeOR. Different colors indicate different regions.

E.2 DESCRIPTION–REGION ALIGNMENT

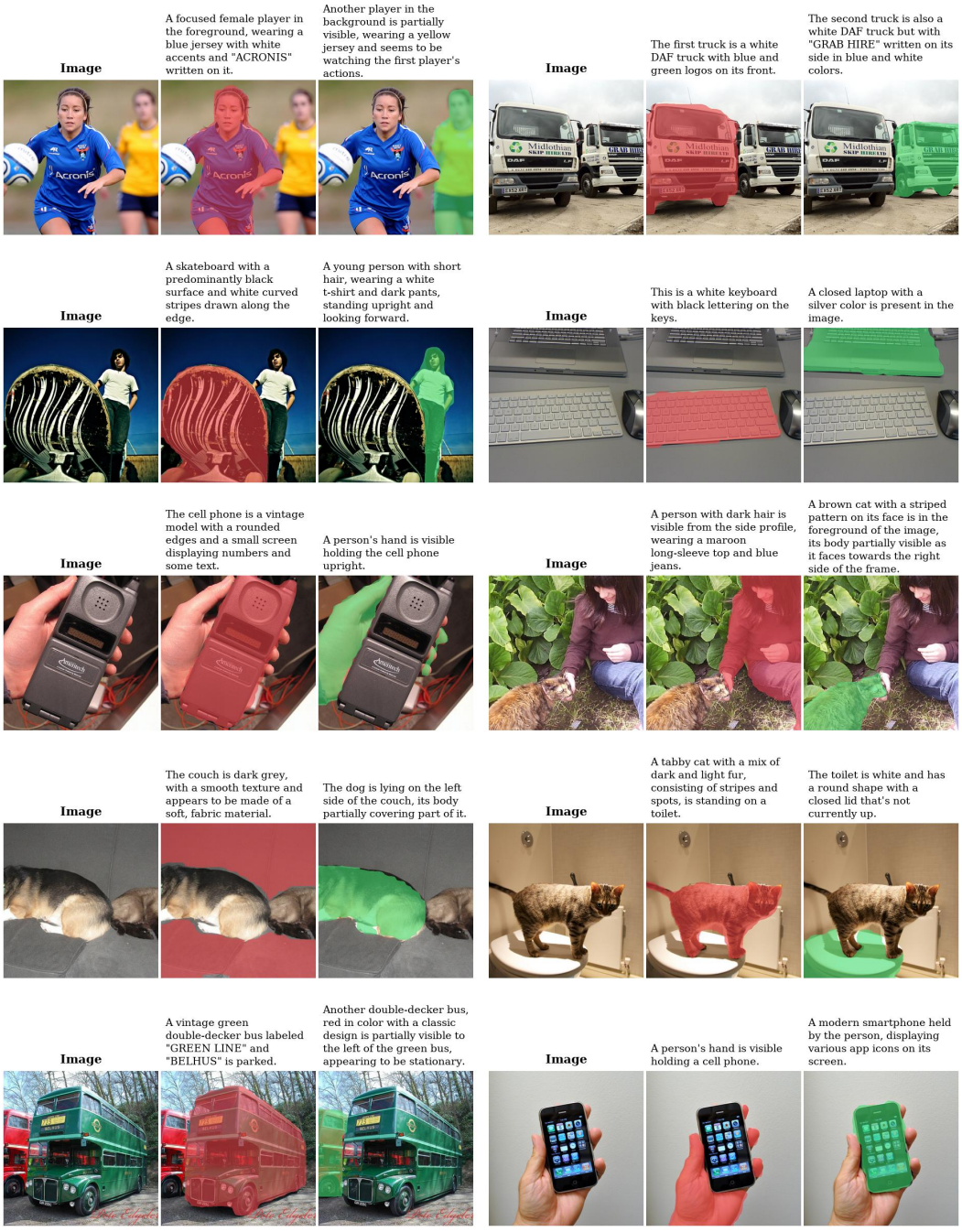


Figure 12: **Additional examples of alignment between generated object descriptions and predicted regions.** Each example shows an input image and two descriptions generated autoregressively by DeOR, together with the predicted regions associated with their object tokens. Red and green overlays distinguish the two predicted regions within each example.

E.3 Language-Guided Object Editing

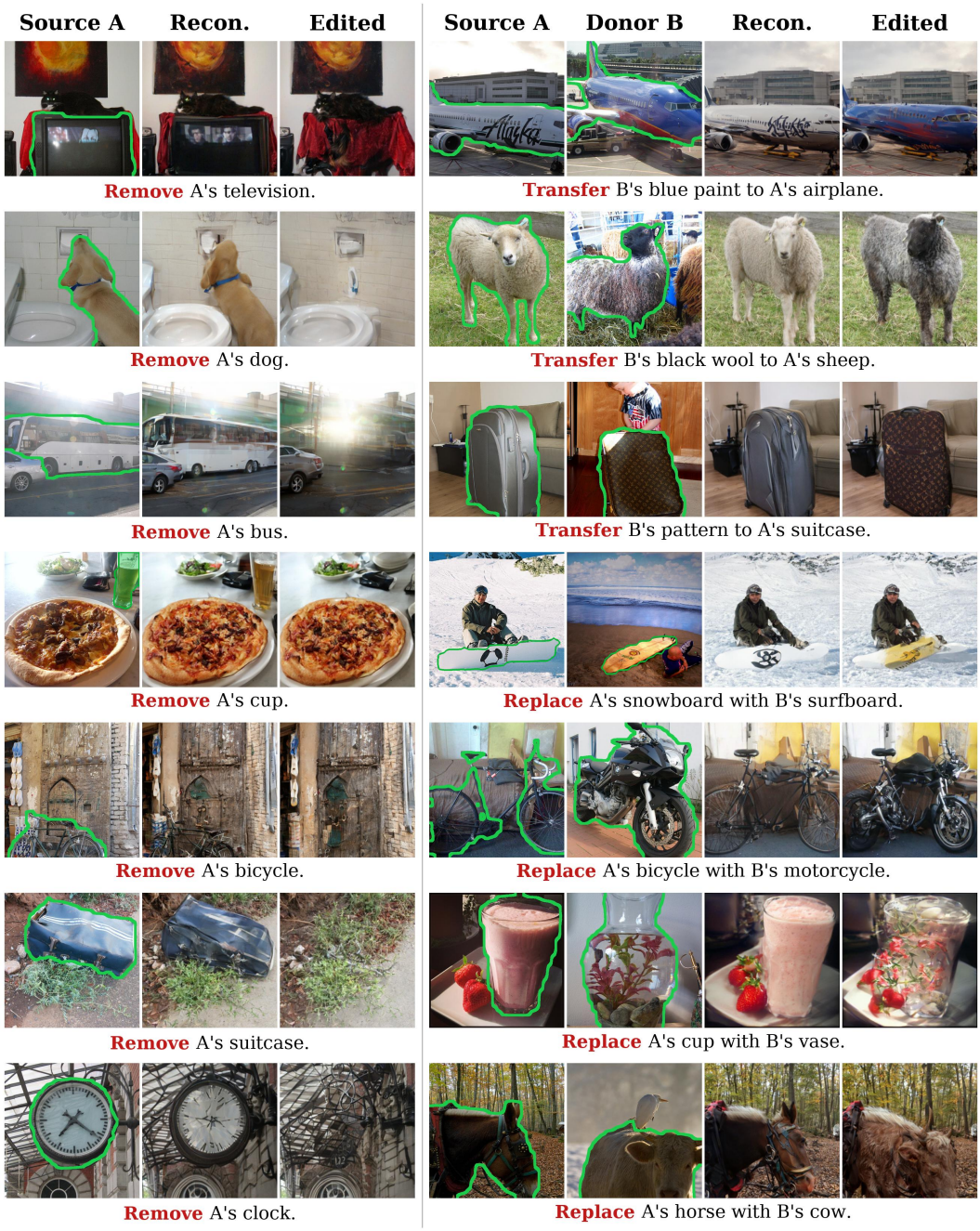


Figure 13: **Additional examples of language-guided object editing.** Left: object removal from source image A. Right: appearance transfer in the top three rows and object replacement in the bottom four rows, using donors from image B. Qwen3-VL-4B-Instruct selects objects using the editing instruction and DeOR's generated object descriptions, without access to the images. Green outlines mark the selected target and donor regions. Unedited reconstructions are shown for comparison.

E.4 Direct Manipulation of Object Representations

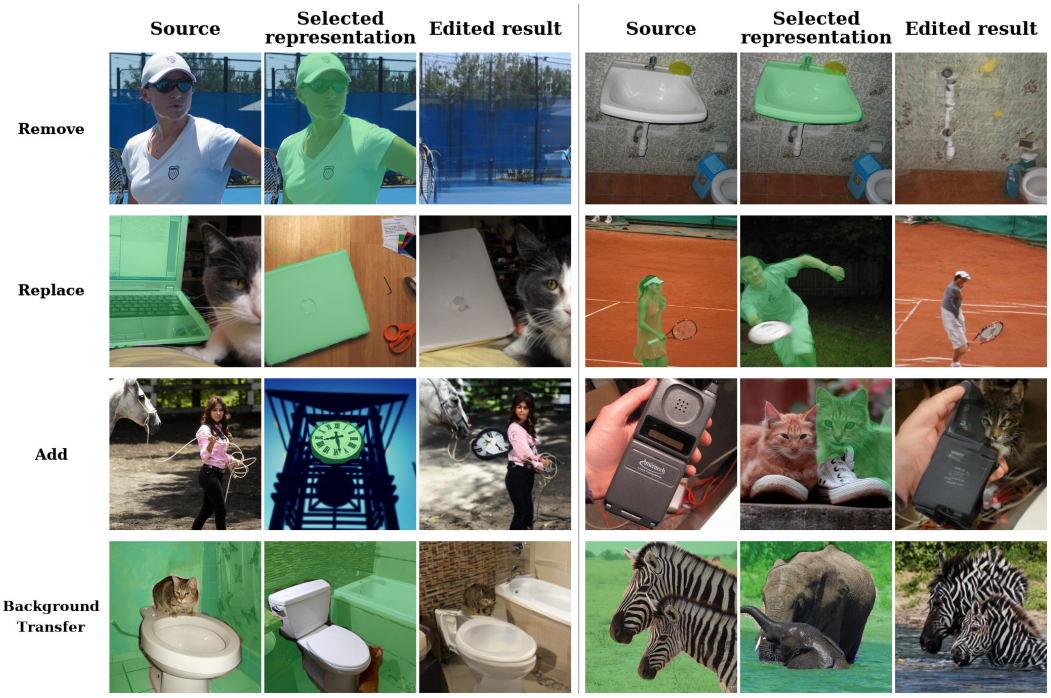


Figure 14: **Direct manipulation of object representations.** Rows show object removal, replacement, addition, and background transfer. Each example presents the source image, the region associated with the selected representation, and the edited result. The selected region comes from the source image for removal and from a donor image for the other operations. Green overlays indicate the selected object or background regions.

F Limitations and Future Work

DeOR learns explicit correspondences between object descriptions and image regions using paired descriptions and masks as supervision. This supervision helps establish what each object representation should capture, but requires corresponding annotations in the training data. Expanding training to more diverse images and objects therefore requires considering the cost of obtaining these correspondences. Future work could use descriptions and region predictions from pretrained models to reduce annotation requirements while maintaining correspondence at the object level.

DeOR also generates object descriptions and object tokens sequentially, so the resulting representations depend on the accuracy and completeness of the generated descriptions. If an object is omitted or its description lacks the details needed to distinguish it from other objects, forming its representation or selecting it through language can become difficult. As the number of objects and the length of their descriptions increase, generation requires more tokens, making it important to describe objects adequately within a limited inference budget. Future work will explore how to generate concise descriptions that cover and distinguish the objects in an image and construct their corresponding representations more efficiently.