# Supplementary Material: Synthetic Object Compositions for Scalable and Accurate Learning in Detection, Segmentation, and Grounding

Supplementary Material

## 1. Limitations and Future Work

**Synthetic–real domain gap.** Our relighting and blending strategies bring synthetic images closer to real photographs than directly pasting segments onto images. The remaining differences in surface texture, material response, and global illumination are subtle. These minor artifacts have a limited impact on downstream tasks compared to the performance boost we observed relative to real data, and they can be further reduced in future work by integrating state-of-the-art diffusion models.

**3D Coherence.** Although we operate on standalone 2D object segments and directly paste them into images without explicit 3D constraints, this simplification poses little difficulty for current 2D-focused benchmarks (e.g., detection, segmentation, referring expressions). For applications requiring full 3D coherence, such as depth estimation or novel-view synthesis, future extensions could incorporate 3D geometry priors or 3D assets into the generation pipeline.

**Object Interactions.** For tasks such as detection, grounding, and segmentation, inter-object relations are not a primary domain feature. Therefore, for simplicity, our pipeline currently treats objects independently and does not model inter-object relations. In practice, this has a negligible impact on object-centric recognition tasks, and future work could address this by adding fine-grained relation control—using diffusion models in our pipeline—if we wish to extend it to relation recognition.

**Panoptic segmentation.** In this work, we focus on object-level recognition. We produce high-quality object segments for discrete "things," but do not yet annotate amorphous "stuff" regions or enforce an all-pixel partitioning, as we are aiming to improve on object detection, instance segmentation, and visual grounding. This limitation does not impede most instance-level or semantic benchmarks, and can be readily addressed by integrating off-the-shelf "stuff" predictors or by extending our annotation pipeline to include full-scene panoptic labels in future releases.

## 2. Ablation of Scaling Only Object Segments or Images

In our main experiment, since we generate 20M object segments, we simultaneously scale the number of object segments and the number of images—each object segment is used only once within the composed images—and observe that larger datasets yield better performance. To disentangle the individual effects of scaling images versus object segments, we conduct two ablation studies:

**Fixing the number of images and scaling object segments.** We fix the image count at 50 K and vary the total number of object segments through four settings: 100 K, 200 K, 500 K, and 1 M. In each case, we sample 20 segments per image. Consequently, when only 100 K segments are available, each segment must be reused $20 \times 50\,\text{K} \,/\, 100\,\text{K} = 10$ times; with 200 K segments the reuse factor falls to 5 times; at 500 K it drops to 2 times; and at 1 M segments every sampled segment is unique. As shown in Figure 1 (left), AP increases monotonically from 0.3842 at 100 K segments to 0.4013 at 1 M segments, with the largest incremental gain occurring between 500 K and 1 M segments. This pattern indicates diminishing returns beyond 500 K but still underscores the benefit of richer segment diversity.

**Fixing the number of segments and scaling images.** Conversely, we fix the segment count at 100 K and scale the number of images through 50 K, 100 K, 200 K, and 400 K, mirroring the unlimited-segment setup in our main experiments. Under this fixed–100 K–segment regime (Figure 1, right, blue curve), AP climbs modestly from 0.3882 to 0.3947 as images quadruple, reflecting the limited benefit of reusing the same segments. In contrast, when we allow an unlimited pool of segments (red curve), AP grows more strongly from 0.3942 to 0.4033 over the same image range. The larger gap under the unlimited-segment condition demonstrates that adding fresh segments is crucial to fully leverage additional images.

**Analysis.** These ablation studies reveal that, while scaling images alone yields modest gains when segment diversity is capped, increasing the number of unique object segments provides larger improvements in AP. The strongest performance arises when both image count and segment diversity

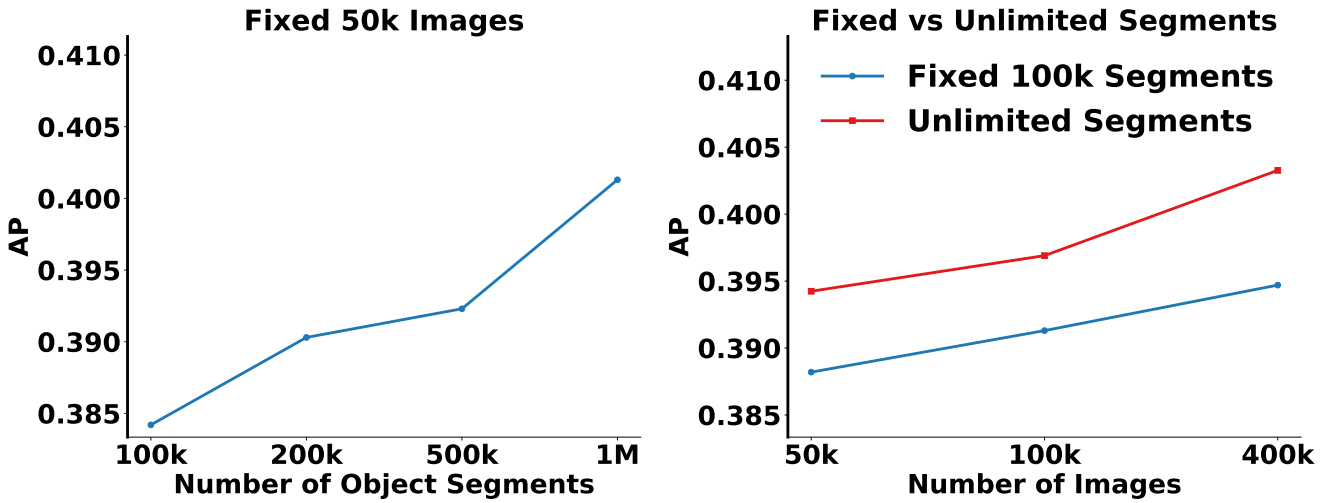



Figure 1. Impact of scaling object segments (left) and images (right) on AP. In the left plot, we fix the number of images to 50 K and vary segments from 100 K to 1 M; in the right plot, we compare fixed–100 K segments versus an unlimited-segment regime as we scale images from 50 K to 400 K. Models are evaluated on **LVIS-Mini Val** and report AP.

are scaled together, confirming the advantage of our joint scaling strategy for open-vocabulary detection and segmentation.

## 3. Compare with Other Synthetic Method

We position **SOC** within the broader synthetic landscape through qualitative comparisons in Table 1 and Table 3.

### 3.1. Comparison with Synthetic Object Segments

Subjects 200K [1] contains roughly 200K FLUX-generated segments, but they cover <1K categories and are captured from only a handful of canonical viewpoints. Because the library offers neither broad category coverage nor multi-view variation, it limits the semantic and geometric diversity attainable when composing new scenes.

By contrast, **SOC** delivers **20M** segments spanning **46K+** categories and explicitly samples multiple camera angles for every object prompt, ensuring rich viewpoint diversity. This two-orders-of-magnitude increase in scale—coupled with fine-grained control over category and viewpoint—unlocks far more varied, photorealistic composites than any prior segment library.

**Quantitative comparison with Subject200K.** To validate the quality advantage of our synthetic object segments, we conduct a controlled experiment on COCO instance segmentation using Mask2Former trained on 10K synthetic images. As shown in Table 2, when combining real segments with synthetic object segments, **SOC** achieves 12.79 AP, outperforming Subject200K (12.06 AP) by +0.73 AP (+6.1% relative improvement). This demonstrates that our object segment pipeline—with its broader category coverage (46K+ vs. <1K) and multi-view diversity—produces higher-quality segments that lead to better downstream performance.

### 3.2. Comparison with Synthetic Data Pipelines

**Simulator-rendered datasets**—SYNTHIA [2], GTA5 [3], Virtual KITTI 2 [4], Synscapes [5], and the indoor-focused Hypersim [6]—deliver pixel-accurate masks and boxes by design, yet they remain confined to pre-built domains (mostly driving or indoor scenes) and a closed object vocabulary; a noticeable photo-to-sim gap still emerges when models are applied to real photographs.

**Copy-paste families** such as Simple Copy-Paste [7], InstaBoost [8], and the diffusion-refined X-Paste [9] transplant real segments into new images. They are inexpensive and controllable, but the pasted objects often betray seam artefacts or lighting clashes. Because the pipeline augments a fixed real corpus, its scale is capped by the number of host images and its category list rarely exceeds ~1.3 K classes from LVIS/COCO.

**Diffusion-plus-pseudo-label pipelines** invert the recipe. Methods like SynGround [10] and Learning VG [11] first synthesise entire scenes with text-to-image models [12–14], then harvest boxes or phrases via detectors or VLMs. Although the images are photorealistic and open-vocabulary, the labels inherit detector noise and therefore suit only coarse grounding rather than dense segmentation.

**SOC** is best seen as an *object-centric composition pipeline*. The detailed comparisons are shown in Table 1 and Table 3.

The outcome is a dataset that matches simulators in annotation fidelity, rivals diffusion images in photorealism,

Table 1. Qualitative comparison of synthetic *object-segment* libraries. **SOC** offers orders-of-magnitude more segments, explicit category control, and multi-view diversity—key to the photorealistic, richly annotated composites described in § 4.1.

| Library | Diffusion model | Category | Scale (segments) | Viewpoint diversity |
|---|---|---|---|---|
| Subject 200K | FLUX | Less than 1000 | 200K | ✗ |
| **SOC (ours)** | FLUX | 46K+ | 20M | ✓ |

Table 2. Quantitative comparison with Subject200K on COCO instance segmentation (Mask2Former, zero-shot). All models are trained on 10K synthetic images composed from real segments + synthetic object segments.

| Object Segments | AP | $AP_S$ | $AP_M$ | $AP_L$ |
|---|---|---|---|---|
| Real segments only | 7.03 | 0.55 | 3.15 | 7.22 |
| Real + Subject200K | 12.06 | 1.43 | 6.53 | 14.57 |
| Real + **SOC** (ours) | **12.79** | **1.58** | **7.71** | **15.74** |
| **Improvement** | **+0.73** | **+0.15** | **+1.18** | **+1.17** |

and, through open-vocabulary, layout-controlled synthesis, vastly outstrips prior copy-paste methods in diversity and scale. These qualities underpin the improvements reported in Secs. 4.1–4.5 of the main paper, where **SOC** demonstrates strong performance against both synthetic-based methods and real-image training data (e.g., GRIT, V3Det)—a stark contrast to existing synthetic datasets that are commonly treated as mere complements to real data.

### 3.3. Visual Realism: FID Score Analysis

To quantify the visual realism of our synthetic images, we computed Fréchet Inception Distance (FID) [15] scores on a 1K-sample subset comparing **SOC** against Copy-Paste and X-Paste baselines. As shown in Table 4, **SOC** achieves an FID of 131.93, substantially lower than both Simple Copy-Paste (165.55) and X-Paste (166.03). This indicates that even without explicitly optimizing for photorealism, **SOC** produces images that are more distributionally aligned with natural image statistics than alternative synthetic pipelines.

We note that **SOC** optimizes for annotation accuracy (high-integrity masks), compositional diversity (46K+ categories), and training effectiveness rather than photorealism alone. The strong downstream performance against both synthetic-based and non-synthetic-based methods (Sec. 4.1 of the main paper) validates this design choice.

## 4. Details of SOC Pipeline

### 4.1. Details of Mask-Area-Weighted Blending Algorithm

To achieve realistic lighting, we adopt IC-Light [16], a foreground-conditioned diffusion model that ingests an image containing foreground objects, generates a matching background with a text prompt, and relights the composite to achieve photorealism.

However, IC-Light introduces two challenges in multi-object scenes: (1) strong relighting may distort fine details of small objects, making them unrecognizable, and (2) excessive relighting can alter object colors, breaking consistency with color-based descriptions from the original data.

To address this, we use a segment-area-aware blending process: We introduce a blending weight $\alpha_i \in [0, 1]$ for each object mask $M_i$, which controls the degree of relighting applied to that object. Smaller objects receive higher $\alpha_i$, preserving more of their original appearance; larger objects receive lower $\alpha_i$, allowing more of the relit image to show through.

1. SIZE-BASED WEIGHTING.

$$r_i = \frac{\text{area}(M_i) - A_{\min}}{A_{\max} - A_{\min}}, \qquad r_i \in [0,1],$$
$$\alpha_i = \alpha_{\min} + (\alpha_{\max} - \alpha_{\min})\,\sigma\big(s\,(r_i - \tfrac{1}{2})\big)\,.$$

   Smaller objects (low $r_i$) get higher $\alpha_i$ (milder relighting). The sigmoid groups "small" segments together instead of scaling linearly by size.

2. LAB-SPACE BLEND. For $p \in M_i$, convert $I_O(p), I_R(p) \rightarrow (L_O, a_O, b_O), (L_R, a_R, b_R)$. Blend lightness and chroma separately:

$$L_{\text{out}} = \alpha_i L_O + (1 - \alpha_i) L_R,$$
$$\mathbf{c}_{\text{out}} = (1 - \beta_i)\,\mathbf{c}_O + \beta_i\,\mathbf{c}_R, \quad \mathbf{c} = (a, b), \;\; \beta_i \ll 1,$$

   then convert $(L_{\text{out}}, \mathbf{c}_{\text{out}})$ back to RGB.

3. BACKGROUND PASSTHROUGH. $I_{\text{out}}(p) = I_R(p)$ for $p \notin \bigcup_i M_i$.

This segment-area-aware blending ensures that small segments avoid over-relighting by IC-Light (preserving detail) while larger ones receive stronger adjustments, and by blending only the luminance (with a small chroma factor $\beta_i$) in CIELAB space, we maintain perfect color fidelity and small object details, but also increase the photorealism.

### 4.2. Details of 3D Geometric Layout Augmentation

**Robustness-first design: Why photorealistic layouts are not our goal.** A central design principle of **SOC** is that

Table 3. Qualitative comparison of complete *synthetic-data pipelines*. **SOC** is the only approach that simultaneously offers fine-grained control, open-vocabulary coverage, realistic integration (relighting + blending), and pixel-accurate annotations across *all* dense-vision tasks.

| Pipeline | Method | Fine-grained control | Open-vocabulary | Realistic integration | Accurate masks/boxes | Supported task(s) |
|---|---|---|---|---|---|---|
| Simulator-rendered (e.g. SYNTHIA) | Game-engine scenes | ✓ | ✗ | ✓ | ✓ | DET, SEG |
| Simple Copy-Paste | Paste onto real images | ✓ | ✗ | ✗ | ✓ | DET, SEG |
| X-Paste | Paste onto real images | ✓ | ✗ | ✗ | ✓ | DET, SEG |
| SynGround | Diffusion images | ✗ | ✓ | ✓ | ✗ | VG |
| Learning VG | Diffusion images | ✗ | ✓ | ✓ | ✗ | VG |
| **SOC (ours)** | Composing new images | ✓ | ✓ | ✓ | ✓ | DET, SEG, VG |

Table 4. FID scores (lower is better) comparing visual realism of synthetic data generation methods. Computed on 1K samples against real image distribution.

| Method | FID Score |
|---|---|
| Simple Copy-Paste | 165.55 |
| X-Paste | 166.03 |
| **SOC (Ours)** | **131.93** |

photorealistic spatial layouts are neither necessary nor desirable for training robust vision models. Real-world photographs contain strong statistical regularities (e.g., cars appear large and near image bottoms, small objects cluster on surfaces) that models can exploit as shortcuts. Our 3D geometric layout augmentation deliberately breaks these correlations by sampling object depth, size, and position independently of category, creating compositions that may appear "unnatural" but force models to learn view-invariant, category-robust representations.

Generative harmonization plays a complementary role: it eliminates low-level copy-paste artifacts (lighting inconsistencies, boundary discontinuities) that would provide trivial cues for distinguishing synthetic data, without constraining spatial layouts to match photographic distributions. This separation is intentional—harmonization ensures sufficient visual coherence to avoid shortcut learning from obvious artifacts, while our layout strategy ensures sufficient diversity to avoid shortcut learning from statistical priors.

The effectiveness of this approach is demonstrated by our results (Secs. 4.1–4.5 of the main paper), where **SOC**-trained models outperform both synthetic-based methods and models trained on real images (GRIT, V3Det). These results suggest that photorealistic layouts may actually be detrimental to robustness, as they reintroduce the pictorial biases that limit generalization.

**3D scene modeling with category-independent sampling.** Our 3D geometric layout augmentation strategy models each composite image as a 3D scene where depth and spatial position are sampled independently of object category. This ensures objects of the same category appear at diverse depths, sizes, and positions, preventing category-specific pictorial patterns. For each image, we sample 5-20 object segments (matching COCO/SA-1B distributions) using balanced category sampling to avoid bias.

Each object category $c$ has a commonsense physical size range (e.g., cars: 4-5m, cups: 10-20cm) generated by Qwen2.5-32B. The complete pipeline is:

1. **Sample camera focal length**: $f \sim \mathcal{U}(f_{\min}, f_{\max})$
2. **Define maximum depth**: $D_{\max} = \alpha \cdot f$ (where $\alpha$ is a scaling constant in meters/pixel)
3. **Define depth ranges** (in meters):
   - Close: $[0.1D_{\max}, 0.3D_{\max}]$
   - Middle: $[0.3D_{\max}, 0.6D_{\max}]$
   - Far: $[0.6D_{\max}, D_{\max}]$
4. **For each object segment $i$ of category $c_i$**:
   - Sample physical size: $S_i \sim \mathcal{N}(\mu_{c_i}, \sigma_{c_i})$
   - Sample depth $d_i$ from one of the three ranges, following COCO/SA-1B distribution (40% close, 35% middle, 25% far)
   - Sample 3D position: $(X_i, Y_i) \sim \mathcal{U}(X_{\min}, X_{\max}) \times \mathcal{U}(Y_{\min}, Y_{\max})$ (in meters)
   - Project to 2D via perspective projection:
   $$x_i = f \cdot \frac{X_i}{d_i}, \quad y_i = f \cdot \frac{Y_i}{d_i}, \quad s_i = f \cdot \frac{S_i}{d_i}$$
   where $(x_i, y_i)$ is the 2D center position and $s_i$ is the apparent size in pixels
5. **Enforce constraints**: If an object's apparent size is too small/large, or if it completely occludes another object (IoU$(M_i, M_j) \geq 0.9$), resample its 3D position and depth

This approach ensures that object scale is determined by 3D geometry (depth + physical size) rather than category, breaking spurious correlations like "cars appear large and near the bottom."

### 4.3. Details of Camera Configuration Augmentation

After composing and relighting the scene, we apply camera configuration augmentation to simulate diverse camera intrinsics and viewing conditions. Each augmentation (random zoom and depth-of-field blur) is applied independently with 30% probability.

**Random zoom (scaling and cropping).** Starting from the focal length $f$ sampled during layout generation, we apply random scaling with factor $s \sim \mathcal{U}(1.0, 4.0)$ followed by random cropping to simulate camera zoom in. For a composite image $I$ of size $H \times W$:

1. Resize to $sH \times sW$ (modifying the focal length to $f' = s \cdot f$)
2. Randomly crop back to $H \times W$

This operation ensures that object scale is not a reliable cue for category recognition.

**Depth-of-field blur.** To simulate realistic depth-of-field effects controlled by aperture size, we apply selective Gaussian blur based on object depth:

1. Generate a depth map for the composed image: We first use Depth Anything V2 to predict relative depth for the entire scene, then scale it such that the predicted depth of the farthest object matches its sampled depth $d_{\max}$ from the 3D scene modeling. Finally, we replace each object region with its exact sampled depth $d_i$. This approach ensures physical consistency with the layout while allowing natural depth variation in the background
2. Randomly sample a focal plane depth $d_{\text{focal}}$ from the scene's depth distribution
3. Sample an f-number $N \sim \mathcal{U}(1.4, 16)$ representing the aperture size (smaller f-numbers = larger apertures = shallower depth-of-field)
4. Compute blur kernel size for each pixel at depth $d$ via the circle of confusion formula:

$$\sigma(d) = \frac{f^2}{N \cdot d_{\text{focal}}} \cdot \frac{|d - d_{\text{focal}}|}{d}$$

   where $f$ is the focal length sampled during layout generation and $d$ is the absolute depth in meters from the depth map

Objects near the focal plane remain sharp ($\sigma \approx 0$), while those farther away are progressively blurred. Smaller f-numbers (e.g., f/1.4) produce strong background blur mimicking portrait photography, while larger f-numbers (e.g., f/16) keep most objects in focus, simulating landscape photography.

These camera configuration augmentations, combined with our 3D geometric layout augmentation strategy, create a rich distribution of visual configurations that force models to learn robust, view-invariant representations.

### 4.4. Prompts used in SOC

We provide our prompts used in belowing images.

## 5. Details of Experiments

In this section, we provide the training and evaluation details of the experiments conducted in the previous sections.

### 5.1. Details of Task 1: Open Vocabulary Object Detection

**Training details.** We use the training scripts provided by the official repo of MM-Grounding-DINO, with 8xH100 GPUs; We use the MM-Grounding-DINO-Tiny model, initialized with pretrained weights on Object365 and GoldG (with additional pretraining on O365 + GoldG + GRIT + V3Det as complementary data). During the **SOC** synthetic-data stage, we use a batch size of 128: for each batch, 70% of the samples are drawn from our **SOC** dataset and 30% from Object365 or GoldG to improve training stability. When training on both the FC and GC splits, we sample 35% from each split to form the 70% synthetic-data portion. Furthermore, for each synthetic bounding-box annotation, 66% are paired with the category label (e.g., *apple*) and the remaining 33% with a short phrase (e.g., *the red apple*) to diversify the training signal. This configuration is applied at the 50K, 100K, and 400K **SOC** data scales, each run lasting 10 epochs. We use an initial learning rate of $4 \times 10^{-4}$, reduced to $4 \times 10^{-5}$ at the sixth epoch, and apply a learning-rate multiplier of 0.1 to both the visual and language backbones. Due to the syn2real gap, this stage does not directly improve benchmark performance but injects **SOC** information into the model. Afterward, we fine-tune the model on the original Object365 and GoldG datasets for 3 epochs, sampling equally from Object365, GoldG, and our synthetic data (1:1:1). We use a learning rate of $2 \times 10^{-4}$, again scaling the visual and language backbone rates to 0.1 of the overall rate. After this stage, we observed strong performance gains over benchmarks. For the baseline with V3Det and GRIT, we directly use the weights provided by the MM-Grounding-DINO repo.

To validate that any performance gains stem from **SOC** rather than extended training on real data, we also test on using only Object365 and GoldG, but matched for FLOPs and learning rates of synthetic data training, which only keeps the performance on pretrained weights, demonstrating the effectiveness of **SOC** data.

### 5.2. Details of Task 2: Visual Grounding

**Training details.** We follow the same setup as Task 1 for Visual Grounding, but after the original synthetic-data stages, we further train the models for five epochs on the same images, but using referring expressions as the training signal. And then follow the same real data fine-tune stage.

We also test that training with our data in Task 1 (using category and short phrase as signal instead of referring expression) doesn't improve model performance on the visual grounding benchmark. Only training on **SOC** referring

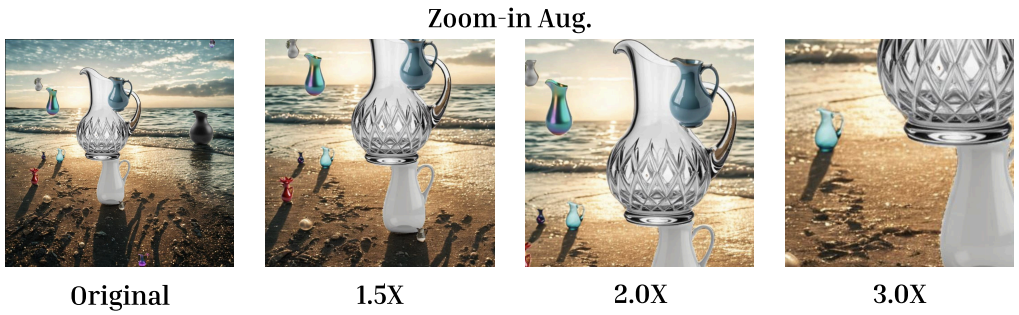



Figure 2. Examples of random zoom augmentation. Starting from the composed image (left), we apply random scaling with factor $s \sim \mathcal{U}(1.0, 4.0)$ followed by random cropping (right). This simulates camera zoom in and ensures object scale is not a reliable cue for category recognition.

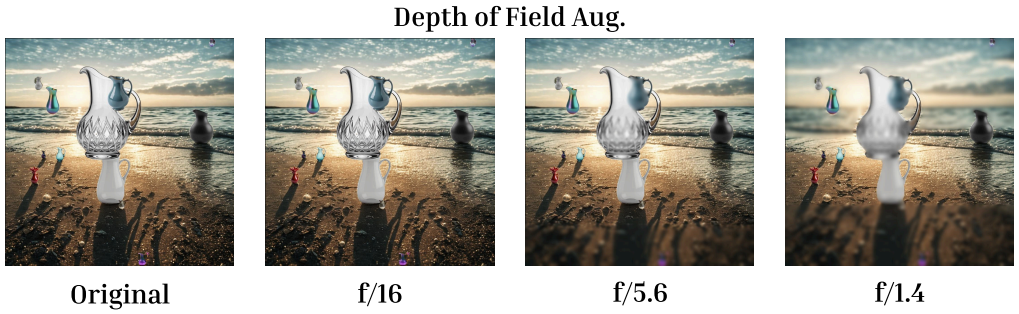



Figure 3. Examples of depth-of-field blur augmentation. Starting from the composed image (left), we apply selective Gaussian blur based on depth (right). Objects near the focal plane remain sharp while those farther away are progressively blurred, simulating realistic camera aperture effects with varying f-numbers.

expressions brings the performance boost.

### 5.3. Details of Task 3: Instance Segmentation

**Data generation.** For 50K **SOC-LVIS-Category** we used. We generated only by using the synthetic object segments that are in the LVIS categories. The other data configurations are the same as the **SOC-FC** and **SOC-GC**.

**Training details.** All experiments were conducted using the training scripts provided by the official APE repository on 4 NVIDIA H100 GPUs. We initialized this from the APE-L_A checkpoint, which was pre-trained jointly on LVIS, COCO, Object365, OpenImages, and Visual Genome. We load those weights, set a batch size of 8, and apply a halved learning-rate schedule compared to the default (the official APE training uses batch size 16). We sample LVIS and **SOC-LVIS-Category** data equally (50% each) and train for 30000 steps, then continue training solely on LVIS for an additional 10000 steps. For the baseline, we train directly on LVIS for 40000 steps.

### 5.4. Details of Comparison with Synthetic Baselines and Ablation Study

**COCO instance segmentation (Mask2Former).** For both the comparison with synthetic baselines (§4.7) and the ablation study (§4.8), we use the same experimental setup to ensure fair comparison. All models are trained on 10K synthetic images generated by each method, initialized from ImageNet-pretrained ResNet-50 weights, and trained from scratch without using any COCO images. We use the official Mask2Former training scripts on 4 A100 GPUs with batch size 32 and learning rate 0.0001. All models are evaluated zero-shot on the COCO validation set and report AP.

**LVIS-Mini open-vocabulary detection (MM-Grounding-DINO).** For the comparison with synthetic baselines (§4.7), we train MM-Grounding-DINO on 50K synthetic images generated by each method. We follow the same training setup as Task 1 (§4.1) and evaluate on the LVIS-Mini validation set, reporting AP.

```
Generate {count} creative and distinct descriptions for the subject '{subject}'.
Each description should be a detailed sentence describing only the '{subject}' itself. It can include adjectives and
descriptive visuable details (for example, 'a nice rounded apple with a rotted dot on its surface'), but it must not include
any additional subjects or environment description (phrases like 'apple over the tree' or 'apple in a starry night' 'apple
that are sweet with each bites' are NOT GOOD and TOTALLY FORBIDDEN).
The descriptions should be creative and unique from each other, and vary in length, with some being simple like 'a blue
apple' and some more detailed, but don't be too long. Diversity is the key.
The descriptions should be suitable for a caption of an image containing the subject, and should be detailed enough to
be useful for someone who can't see the image. It should emphasize on visuable feature like color, material, shape,
pattern.
Your descriptions must contain the '{subject}' word.
Return the output as a JSON object with a single key 'descriptions' mapping to an array of descriptions.
Now generate descriptions for '{subject}' with rules above:
```

Figure 4. Prompt for generating diverse text descriptions.

```
Given the following description, perform two tasks:
1. Create a "short phrase" summarizing the description in at most 7 words.
2. Extract the features or attributes mentioned in the description.
Return the result in JSON format with keys "short phrase" and "features" (a list of phrases).
Description: "{description}"
Example output:
{
  "short phrase": "an minimalist façade air conditioner",
  "features": ["minimalist façade", "circular control buttons"]
}
Only output the JSON.
```

Figure 5. Prompt for extracting features and generating shortening phrases.

**Visual grounding on gRefCOCO (MM-Grounding-DINO).** For the comparison with synthetic baselines (§4.7), we train MM-Grounding-DINO on 50K synthetic images with referring expressions. We follow the same training setup as Task 2 (§4.2) and evaluate on gRefCOCO, reporting Precision@($F_1$=1, IoU$\geq$0.5) and no-target accuracy.

### 5.5. Details of Task 4: Small-Vocabulary, Limited-Data Regimes

**Training details.** We train Mask2Former (R–50 backbone) using the official Mask2Former training scripts on 4 A100 GPUs. Because the repository does not include a four-GPU configuration, we initialize only from the ImageNet-pretrained ResNet-50 weights and train the remaining components from scratch. We apply the same batch size (32) and learning rate (0.0001) to both our method and the baseline, ensuring a fair comparison.

The baseline model is trained solely on COCO and is stopped when its validation AP has not improved for 2,000 consecutive iterations.

For our setup, we first perform 4,000 training iterations on real COCO images at each data scale (1K, 10K, 50K, and the full COCO set) to obtain an initial representation. Training then continues exclusively on our COCOMIX data until the training loss fails to decrease for 2,000 iterations, signalling a plateau. Finally, we fine-tune the model on its scale of COCO data and terminate once the performance on the test set no longer improves.

### 5.6. Details of Task 5: Intra-class referring

**Training details.** We follow the setup from Task 1, but instead of training over **SOC-FC** and **SOC-GC**, we train over **SOC-SFC** and **SOC-SGC**, which are images with same-category-different-attributes objects specifically generated to solve the intra-class problems.

{ViewPoint (Left view, Top view, Bottom view ....)} {description} with pure white background.generate the entire object instead of a part of it. Don't contain any other objects, generate with the full structure of the object, in high quality with photorealistic details, accurate textures.

Figure 6. Prompt for generating object segments.

You are a referring expression detection data generator. I will provide you with a list of objects WITHIN an IMAGE, and you will generate referring expressions similar to RefCOCO, RefCOCO+, RefCOCOg, and GrefCOCO.

We categorized referring expressions into 3 types: attribute-based, spatial-based, and reasoning-based.
**Requirements**:
- Attribute-based: ask about `features`, `category`, or `short_phrase` of exactly one object. (e.g. The white dog)
- Spatial-based: infer absolute or relative positions strictly from the `bbox` values (e.g. left/right, above/below, center). (e.g. The dog left to the people with brown shirt)
- Reasoning-based: combine features, `short_phrase`, `category` and spatial bbox relationships between objects. (e.g. The white animal left to the person with brown shirt)
- Use `short_phrase` or `features` preferentially to refer to objects; also can use `category` with some features to refer it.
- Return ONE JSON block matching the schema exactly, with **exactly** the requested counts per bucket. No extra keys.

For each referring expression, we have 3 types of returning objects:
1. **Single object**: Expression refers to exactly one object in the image. (e.g. The white dog)
2. **Multi-object**: Expression refers to 2 or more objects in the image. (e.g. All the white dogs in the image)

### Example segments and expressions
# Example annotation
[ID: 7377552] tongs | short_phrase: tongs with rough iron texture | features: [rough iron texture] | description: tongs with a rough iron texture, painted in old bronze | bbox: [318, 535, 128, 282]
[ID: 10569372] bath_towel | short_phrase: bath towel with tribal flair | features: [geometric tribal flair] | description: a bath_towel with geometric tribal flair in coppery tones | bbox: [474, 10, 493, 879]
[ID: 2187630] Canned | short_phrase: cylindrical can of recycled aluminum | features: [cylindrical, recycled aluminum] | description: A cylindrical can made of recycled aluminum. | bbox: [376, 217, 102, 247]
[ID: 10733385] shovel | short_phrase: shovel with gleaming blade | features: [gleaming blade] | description: a shovel with a blade that gleams like polished alabaster | bbox: [424, 726, 48, 184]
[ID: 519546] knitting_needle | short_phrase: knitting needle with glossy finish | features: [glossy finish] | description: a knitting_needle with a glossy, clear finish and a spiral ridge | bbox: [0, 80, 97, 125]
[ID: 4995055] strap | short_phrase: slim clear strap with blue stripe | features: [clear, blue stripe] | description: a slim, clear strap with a spray-painted blue stripe | bbox: [725, 178, 54, 60]
[ID: 9339368] teakettle | short_phrase: teakettle with glass body | features: [glass body] | description: A teakettle with a round glass body and a charming, twisted copper handle. | bbox: [324, 789, 103, 117]
[ID: 8109537] cushion | short_phrase: hunter green leather cushion | features: [hunter green leather] | description: A hunter green, sleek leather cushion. | bbox: [684, 219, 89, 79]
[ID: 4123758] raspberry | short_phrase: glossy raspberry with maroon tinge | features: [glossy, maroon tinge] | description: a glossy raspberry with a subtle maroon tinge | bbox: [183, 324, 134, 140]
[ID: 9903309] dropper | short_phrase: dropper with matte black body | features: [matte black body] | description: A dropper with a matte black body and a glossy dropper tip | bbox: [204, 868, 93, 122]
[ID: 2998739] snowmobile | short_phrase: snowmobile with white surface | features: [white surface, longitudinal red strips] | description: a snowmobile with a glossy white surface decorated with longitudinal red strips | bbox: [898, 451, 63, 38]
[ID: 13570439] box | short_phrase: box with bold stripes and smiley | features: [bold stripes, smiley] | description: a box painted in bold stripes with a quirky smiley face | bbox: [305, 837, 46, 42]
[ID: 232766] ram_animal | short_phrase: compact ram with patchwork wool | features: [patchwork wool] | description: a compact ram boasting an intricate pattern of color on its wool, resembling patchwork | bbox: [152, 704, 35, 36]
[ID: 15801147] birthday_card | short_phrase: cheerful pirate ship with map | features: [cheerful pirate shi] | description: A birthday_card featuring a cheerful pirate ship with a colorful map. | bbox: [34, 924, 45, 46]
[ID: 8631767] Egg_tart | short_phrase: marigold custard egg tart | features: [marigold-colored custard] | description: An egg tart with a marigold-colored custard that slightly spills over around the cornflower-blue border. | bbox: [442, 207, 43, 44]
[ID: 12491521] cornbread | short_phrase: crispy cornbread with golden flecks | features: [golden flecks] | description: A crispy slice of cornbread, with a myriad of shimmering, golden flecks and patches. | bbox: [191, 621, 42, 42]
[ID: 7054199] wooden_spoon | short_phrase: stout wooden spoon for dough | features: [stout] | description: a stout, thick wooden_spoon with a hefty feel, perfect for tackling hefty dough mixtures | bbox: [207, 292, 46, 40]
[ID: 2379817] motor | short_phrase: tiny pink motor with meta plates | features: [pink, with meta plates] | description: a tiny, delicate motor painted in a pastel pink with tiny meta plates | bbox: [93, 812, 40, 25]

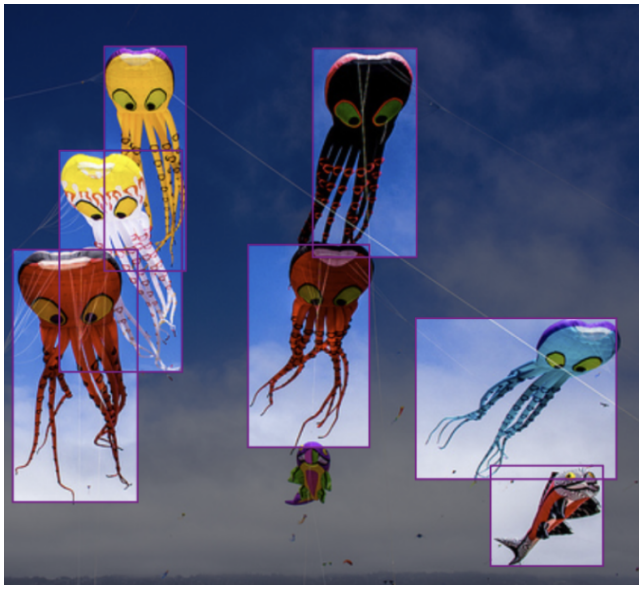
# Example generated expression
```
{
  "single_object": {
    "attribute": [
      { "q": "The glossy raspberry with maroon tinge", "ids": [4123758] },
      { "q": "The slim clear strap with blue stripe", "ids": [4995055] },
      { "q": "The cylindrical can of recycled aluminum", "ids": [2187630] },
      { "q": "The hunter green leather cushion", "ids": [8109537] }
    ],
    "spatial": [
      { "q": "The knitting needle with glossy finish on the far left", "ids": [519546] },
      { "q": "The snowmobile with white surface on the far right", "ids": [2998739] }
    ],
    "reasoning": [
      { "q": "The teakettle with glass body below the shovel with the gleaming blade", "ids": [9339368] },
      { "q": "The box with bold stripes and smiley to the left of the snowmobile with white surface", "ids": [13570439] },
      { "q": "The glossy raspberry with maroon tinge to the left of the cylindrical can of recycled aluminum", "ids": [4123758] },
      { "q": "The slim clear strap with blue stripe on the bath towel with tribal flair", "ids": [4995055] },
      { "q": "The stout wooden spoon for dough above the shovel with the gleaming blade", "ids": [7054199] }
    ]
  },
  "multi_object": {
    "attribute": [
      { "q": "All the objects with a glossy finish", "ids": [519546, 4123758] },
      { "q": "All the striped objects", "ids": [13570439, 2998739] },
      { "q": "All the containers", "ids": [2187630, 13570439] }
    ],
    "spatial": [
      {
        "q": "All the objects above the horizontal midpoint of the image",
        "ids": [10569372, 2187630, 4995055, 8109537, 4123758, 8631767, 7054199, 519546]
      },
      {
        "q": "All the objects that span the central vertical band of the image",
        "ids": [7054199, 4123758, 2187630, 7377552, 2998739]
      },
      {
        "q": "All the objects to the left of center and above the shovel with gleaming blade",
        "ids": [7377552, 2187630, 519546, 4123758, 7054199, 13570439, 232766, 2379817, 8631767]
      },
      {
        "q": "All the objects surrounding the cylindrical can of recycled aluminum",
        "ids": [7377552, 7054199, 8631767, 4123758]
      }
    ],
    "reasoning": [
      { "q": "All the metallic objects to the left of the bath towel with tribal flair", "ids": [7377552, 2187630, 2379817] },
      { "q": "All the objects with a glossy finish above the box with bold stripes and smiley", "ids": [519546, 4123758] },
      { "q": "All the containers to the right of the knitting needle with glossy finish", "ids": [2187630, 13570439] }
    ]
  }
}
```

### Here is the objects in the image we want to annotate (Bounding Box is provided in XYWH COCO format, you should compare them in COCO's way):
{User annotations}

### Numbers of expressions to generate
{counts}

Figure 7. Prompt for generating referring expressions.

(a) Multiple donuts.



(b) Multiple kites.

Figure 8. Examples for the intra-class referring benchmark.

(a) Traininig data sampling from **SOC-SFC** and **SOC-SGC**.

(b) Traininig data sampling from **SOC-SFC** and **SOC-SGC**.

Figure 9. Examples for our target-generated training data.

## 6. Real Segments Filtering Pipeline

Our pipeline provides the flexibility of using both synthetic object segments and real object segments collected from a segmentation dataset, We demonstrate that **SOC** pipeline can incorporate real segments from the COCO dataset. Therefore, we build the following real object segments collecting pipeline for future exploration:

Masks extracted directly from the above datasets are frequently occluded, truncated, or loosely bounded, making them unsuitable as composable assets. We therefore introduce a three-stage pipeline that first filters low-quality instances and then enriches the survivors with additional

metadata.

**Filtering.** For every candidate segment we predict three independent quality scores:

- **Integrity**: Is the object complete and unfragmented?
- **IsObject**: Does the mask depict a discrete "thing" (e.g., *car*) rather than amorphous "stuff" (e.g., *road*)?
- **Mask Quality**: How accurately does the mask separate foreground from background?

We annotate 4,000 samples—with GPT-4o-assisted chain-of-thought prompts—for ground truth and train three ViT-B/16 classifiers, one per dimension [17]. At inference, we average the predicted scores and retain the top 30% of segments. We use this pipeline to filter the SA-1B, COCO, VOC, ADE20K, and get a total of 10M of real object segments for future exploration.

## 7. Quality of Synthetic Segments

Rather than extracting segments from existing photographs, we generate synthetic objects one at a time, ensuring that each segment is complete and free of occlusion. To evaluate quality, we conducted a human-annotation study in which annotators reviewed 200 randomly sampled segments; 92% were judged correct.

Because annotators considered the vast majority of synthetic segments high-quality, we include all of them when composing synthetic images—and have already observed a significant performance boost in downstream models. Additionally, our codebase also includes a lightweight pipeline that uses the CLIP score to measure the semantic similarity between each segment and its caption. We left more exploration on filtering for further work.

## 8. Computational Cost Breakdown

| | FLUX | DIS | IC-Light | LLM | Overall |
|---|---|---|---|---|---|
| Time/unit | 2s/seg | 0.1s/seg | 4.8s/img | 1s/obj | – |
| Count | 20M | 20M | 2M | 20M | – |
| A100 GPU-hrs | 11.1K | 0.6K | 2.7K | 5.5K | 19.9K |
| Human filter rate | 98% | 98% | 97% | 99% | 92% |

Table 5. Compute & QC breakdown. Total: ≈19.9K A100 GPU-hrs (≈$30K). A 100K-image dataset costs ≈$1.5K, competitive with V3Det annotation.

## 9. Backbone Fairness: FLUX vs. SD1.5

Different synthetic baselines use different diffusion backbones (X-Paste: SD1.4; SegGen: SDXL; SynGround: SD2.1), raising the question of whether **SOC**'s gains stem from the pipeline design or solely from using a stronger backbone (FLUX). To disentangle this, we replace FLUX with SD1.5 for segment generation and re-run the full pipeline.

| Method | COCO Seg | LVIS Det | gRefCOCO VG |
|---|---|---|---|
| Copy-Paste | 9.32 | 35.2 | – |
| X-Paste (SD1.4) | 9.41 | 37.2 | – |
| SegGen (SDXL) | 9.73 | 36.8 | – |
| SynGround (SD2.1) | – | – | 40.1/89.2 |
| **SOC (FLUX)** | **12.79** | **38.6** | **41.2/93.9** |
| **SOC** (SD1.5) | 11.83 | 38.1 | 40.8/92.3 |

Table 6. Backbone fairness. Even with SD1.5, **SOC** outperforms all baselines, showing pipeline design drives the gains.

## 10. Harmonization and Blending Trade-off Analysis

Our mask-area-weighted blending partially undoes harmonization for small objects: blending restores original pixels while harmonization adjusts lighting. This is an intentional empirical trade-off. IC-Light's strong relighting distorts small objects—in a manual review of 100 composed images (284 small objects), 53 objects became unrecognizable without blending versus only 2 with blending. Preserving object identity outweighs lighting consistency for detection and segmentation tasks. The potential bias (small objects with slightly inconsistent lighting) does not hurt downstream generalization: adding blending improves AP from 10.58 to 12.79 on COCO instance segmentation, confirming that recognizable objects matter more than perfect lighting consistency.

## 11. Additional Ablations: Object Count and Layout Strategy

| Setting | AP |
|---|---|
| *Object count (3D layout)* | |
| 5 obj/img | 9.57 |
| 5–20 (default) | 10.03 |
| 25 obj/img | **10.12** |
| *Layout strategy (5–20 obj)* | |
| COCO-constrained co-occurrence | 8.60 |
| Random 2D | 9.07 |
| 3D geometric (default) | **10.03** |

Table 7. Ablation on object count per image and layout strategy, evaluated on COCO instance segmentation (Mask2Former, 10K images). Object count has a mild effect; we choose 5–20 as default to match the COCO/SA-1B distribution. COCO-constrained co-occurrence layout underperforms both random 2D and our 3D geometric layout, suggesting that breaking real-world spatial correlations improves model robustness.

## 12. Intra-Class Referring Benchmark: Extended Evaluation

To strengthen the ICR evaluation, we expanded the benchmark from 100 to 500 manually curated images. Each image contains multiple instances of the same category with distinct attributes, annotated with bounding boxes and fine-grained attribute labels.

| Method | 100 imgs | 200 imgs | 500 imgs |
|---|---|---|---|
| O365+GoldG | 37.5/88.0 | 36.5/87.0 | 37.5/87.0 |
| +GRIT | 36.7/92.0 | 36.0/88.0 | 36.5/87.0 |
| +V3Det | 34.6/88.7 | 35.5/89.2 | 38.0/87.0 |
| +GRIT+V3Det | 35.5/88.7 | 36.0/88.5 | 37.5/88.0 |
| **+SOC** | **40.6/90.0** | **41.0/91.0** | **42.0/91.5** |

Table 8. Intra-class referring benchmark results at different evaluation scales (AvgGap$\times$100 / Positive Gap Ratio %). **SOC** consistently achieves the best performance across all scales.

## 13. 46K Category Collection Details

Our 46K+ object categories are collected in two tiers:

- **Frequent categories (1.6K):** We pool category lists from V3Det, LVIS, Object365, COCO, and ADE20K. Duplicates and near-synonyms are merged via WordNet synsets, and categories are ranked by their frequency in the LAION corpus to prioritize commonly occurring objects.
- **General categories (40K+):** We mine noun phrases from LAION captions, GQA scene graphs, and Flickr30K entity annotations using Qwen2.5. After de-duplication via WordNet and string similarity, we retain $\approx$40K unique categories that extend coverage to long-tail and fine-grained concepts.

## 14. OdinW-35 Discussion

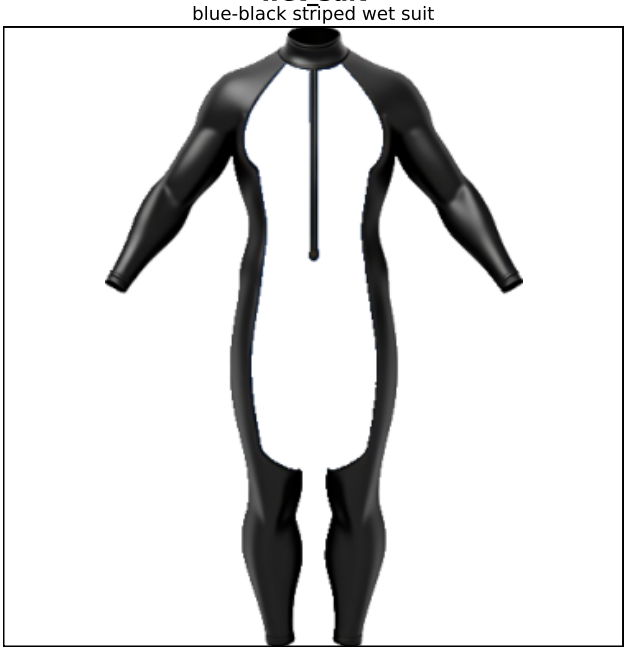
**SOC** is an object-centric pipeline without scenario-specific priors (e.g., aerial, underwater, medical), so some individual OdinW-35 datasets that are heavily scenario-dependent may not benefit as much. Nevertheless, **SOC** still improves over the base model (20.3$\rightarrow$21.2 AP at 50K), and FC-only at 400K reaches 22.8 AP, matching GRIT and exceeding V3Det. This confirms that broad object diversity from **SOC** provides useful generalization even for domain-specific detection tasks.

## 15. Gallery of Object Segments

**snowmobile**
sleek white snowmobile, matte finish

**bear**
giant polar bear with white fur

**wet_suit**
blue-black striped wet suit



**toaster_oven**
green toaster oven with simple knobs

**Slippers**
slippers with floral patterns

**Fishing_Rod**
sleek silver fishing rod

**houseboat**
glass-paneled houseboat with rainbows

**potato**
elongated potato, smooth like marble

**visor**
sleek black visor with shine

**calf**
calf with bright silver eyes

**toaster_oven**
sleek toaster oven with chrome

**orange**
small spherical orange with green

**bonnet**
bonnet with tiny flowers

**Speed_Limit_Sign**
speed limit sign, distressed edges


**soup_bowl**
cracked aged soup_bowl rustic charm

**Cosmetics_Brush_and_Eyeliner_Pencil**
silver brush with dark eyeliner

**tag**
small shiny silver circular tag

**Scallop**
irregular scallop with sea specks

**minibike_motorbike**
minibike with bronze hue

**koala**
koala with mosaic fur

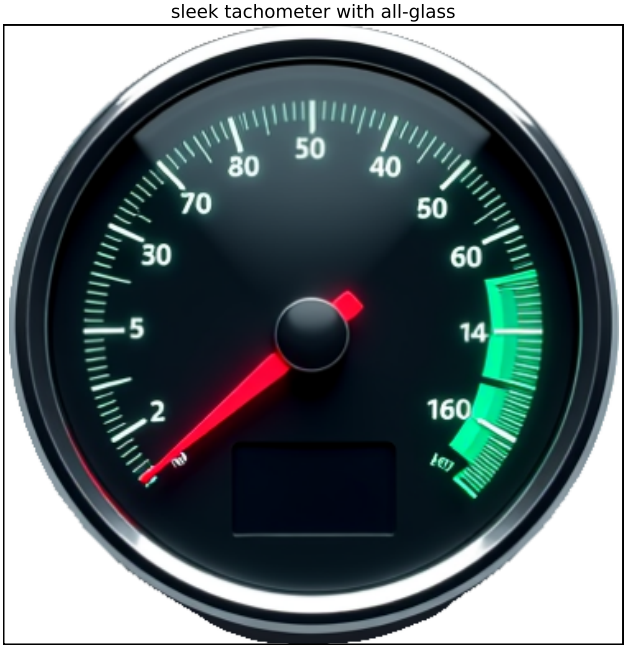
**table**
huge wooden table with carvings

**walrus**
large walrus with thick skin

**string_cheese**
creamy white string_cheese coiled

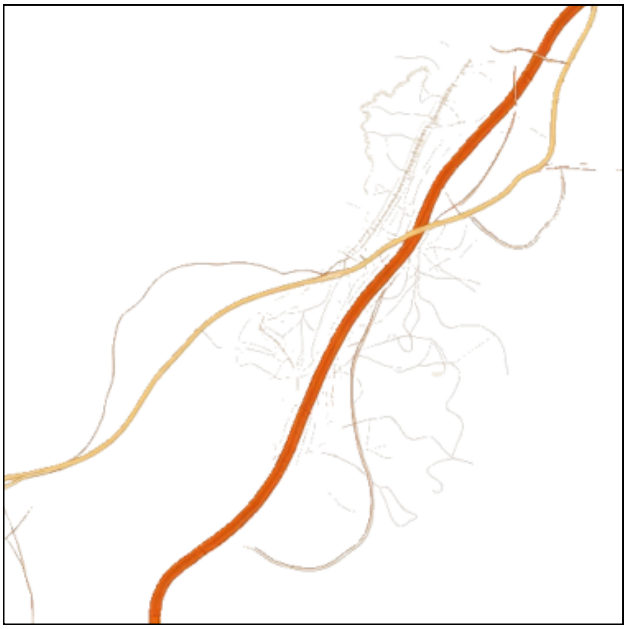
**spice_rack**
rustic spice rack with patterns

**tachometer**
sleek tachometer with all-glass



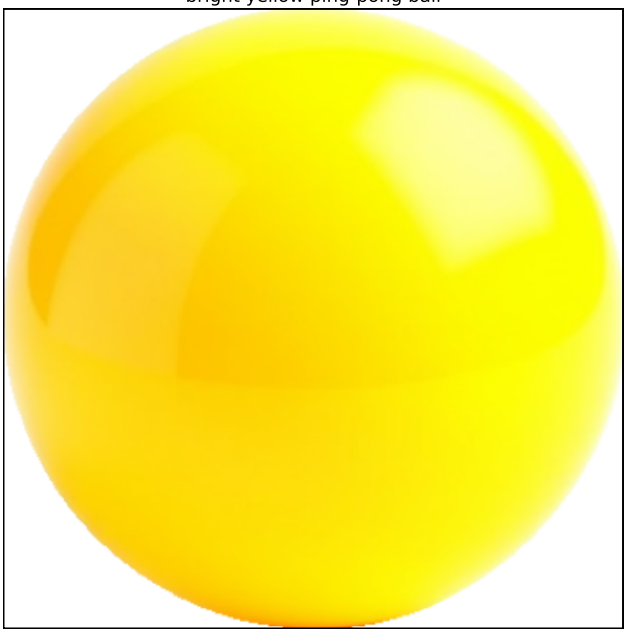

**vacuum_cleaner**
upright vacuum with chevron pattern

**road_map**
dotted auburn lines, rural routes



**cappuccino**
balanced cappuccino with harmonious elements

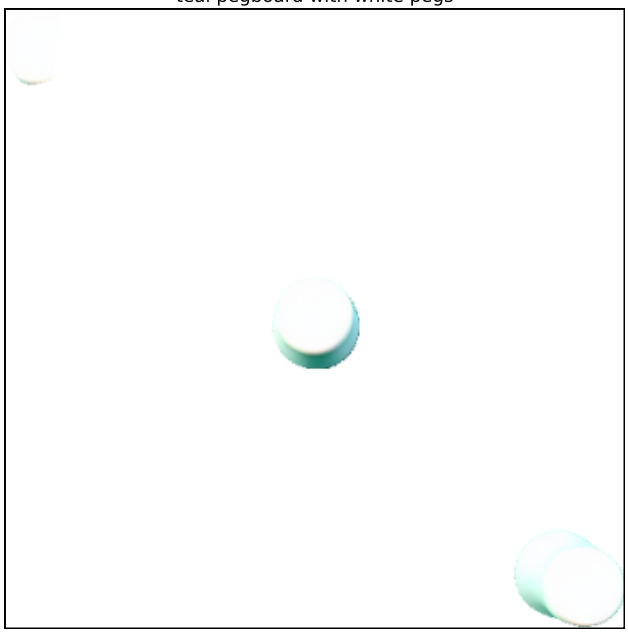
**pingpong_ball**
bright yellow ping-pong ball



**food_processor**
colorful food processor with gradient

**ashtray**
ashtray shaped like spaceship

**pegboard**
teal pegboard with white pegs



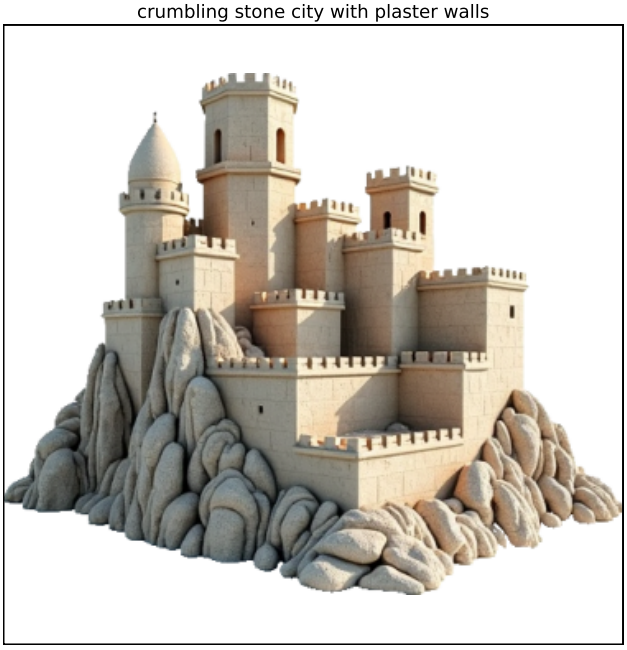
**string_cheese**
golden thin string cheese swirling

**kilt**
vibrant tartan kilt with clasps

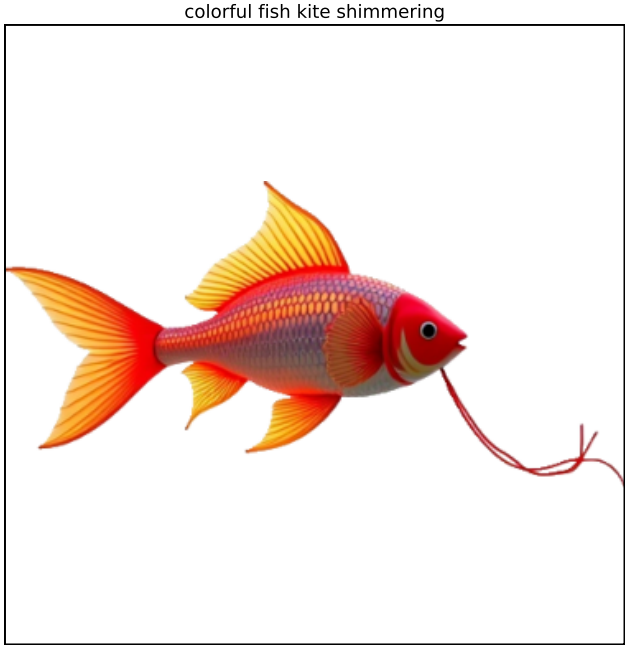
**cistern**
shimmering copper cistern mirror finish

**water_bottle**
rustic bamboo-like water bottle

**Herculanum**
crumbling stone city with plaster walls



**kite**
colorful fish kite shimmering



**palm_palm_tree**
towering palm with smooth trunk

**Soyabean_leaf**
vibrant green soyabean leaf

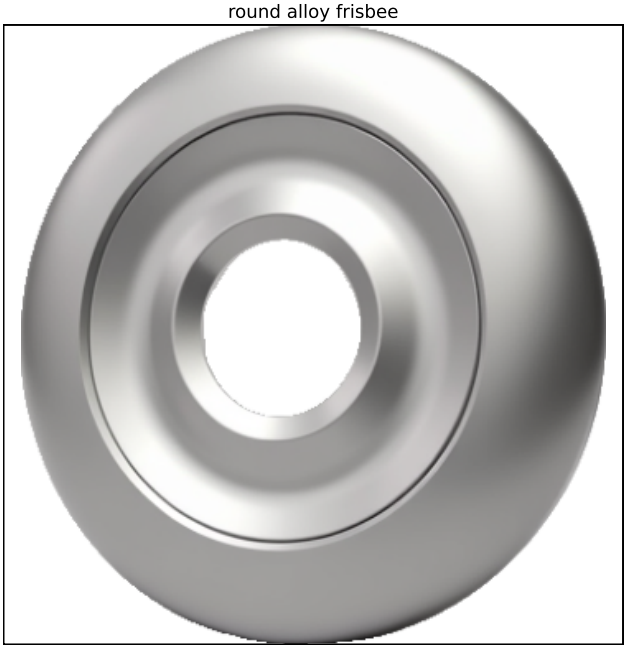
**parchment**
weathered parchment with old ink

**beanie**
knitted beanie in rainbow colors

**Noteworthy**
bold statement in sienna hues

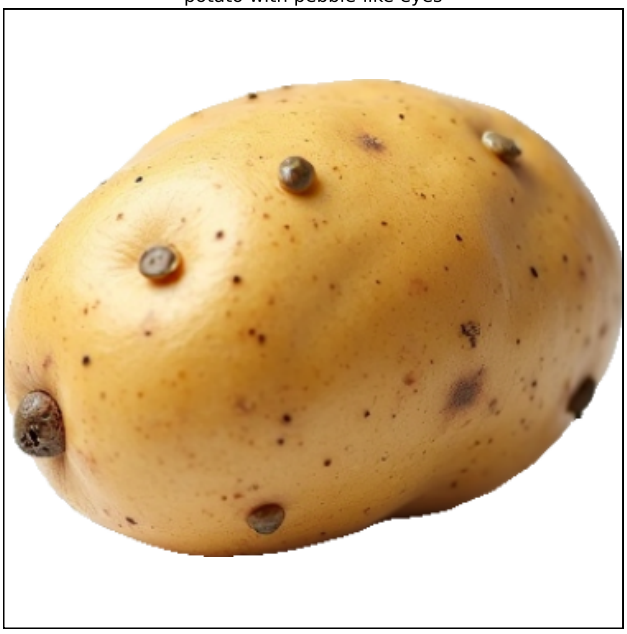
**drumstick**
sleek carbon fiber drumstick

**frisbee**
round alloy frisbee



**booth**
gold booth with ornate engravings

**potato**
potato with pebble-like eyes



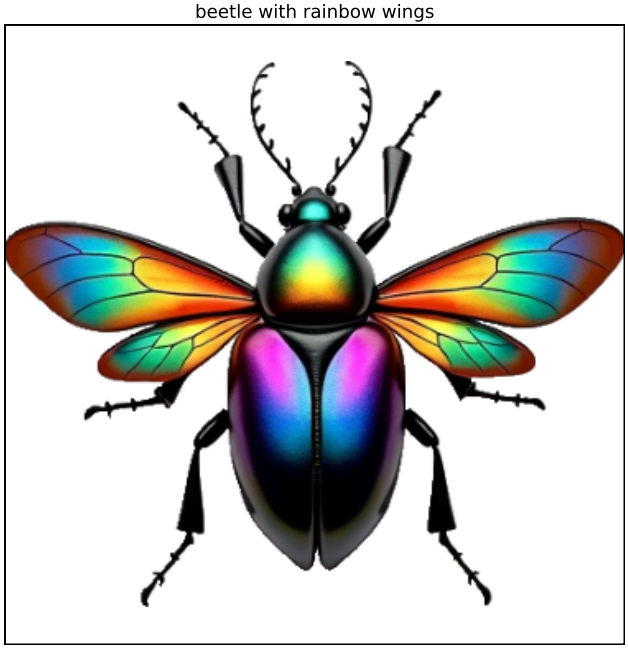
**manger**
rustic wooden manger with cross

**strawberry**
petite strawberry with glossy surface

**bean_curd**
bean curd with translucent sheen

**eyepatch**
eyepatch embroidered with silver threads

**beetle**
beetle with rainbow wings



**bead**
vibrant turquoise teardrop bead

**Rice_Cooker**
compact rice cooker with curves

**dogenglish_setter**
graceful white markings on brown

**clock**
clock with visible ticking gears

**dogenglish_cocker_spaniel**
fluffy dark-brown spaniel with wavy fur

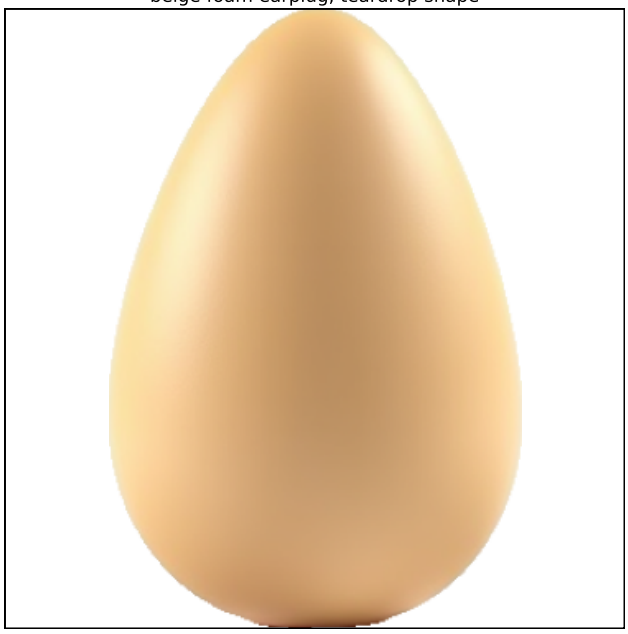
**saucer**
wooden saucer with carvings

**hog**
fluffy white hog like cloud

**earring**
vintage pearl earring with floral

**catBritish_Shorthair**
silver grey British Shorthair cat

**earplug**
beige foam earplug, teardrop shape



**doggreat_pyrenees**
large white dog with spots

**chopping_board**
rugged chopping board with marbling

**rodent**
tiny rodent with sleek coat

**potholder**
red potholder with white dots

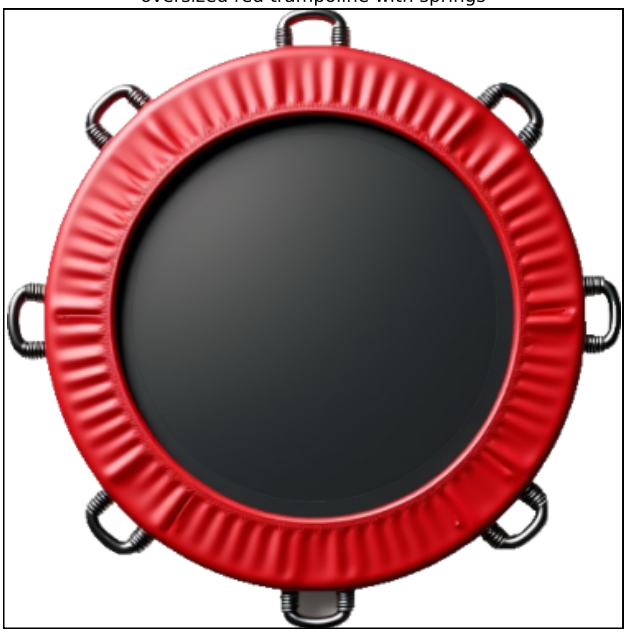
**dogpug**
adorable dog-pug with square eyes

**prune**
small wrinkled prune

**crowbar**
weathered crowbar with rust stains

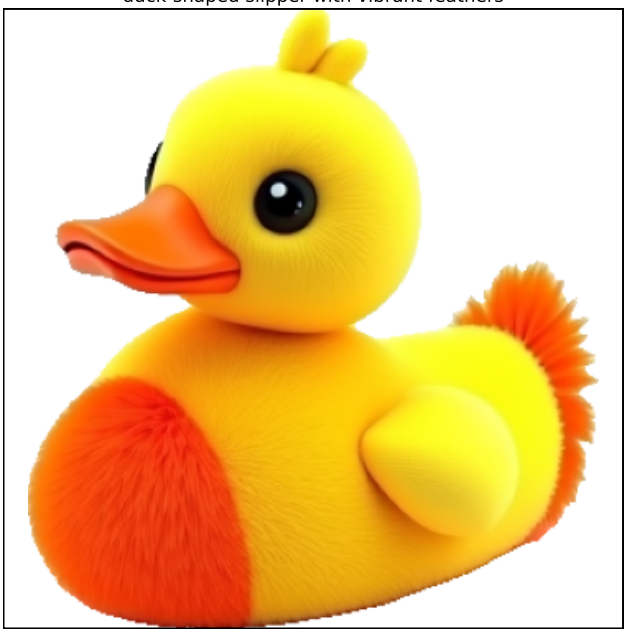
**key**
sparkly silver key with lock

**trampoline**
oversized red trampoline with springs

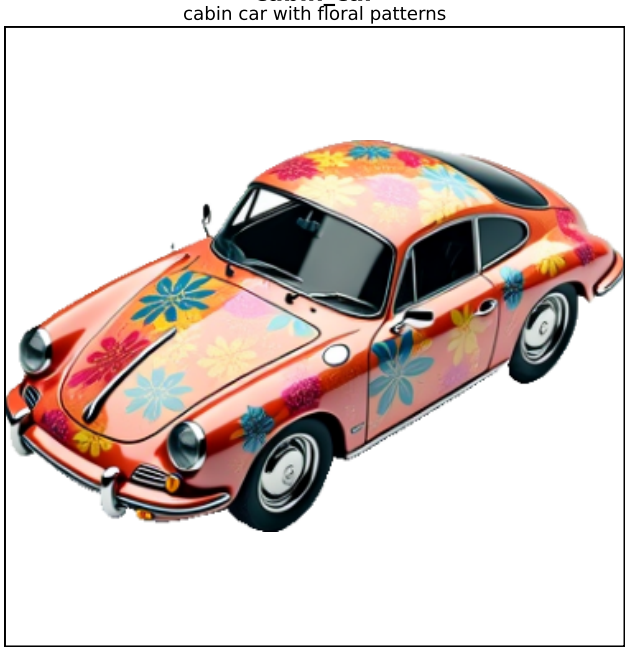


**stylus**
sleek black stylus with silver tip

**slipper_footwear**
duck-shaped slipper with vibrant feathers



**cabin_car**
cabin car with floral patterns

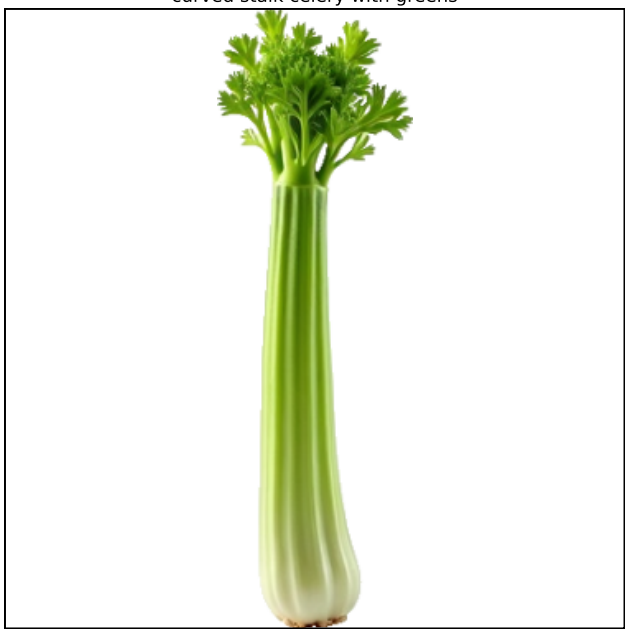


**igniter**
red igniter with burning coals

**Game_board**
mahogany game board with sheen

**bookmark**
velvet bookmark with floral embroidery

**celery**
curved stalk celery with greens



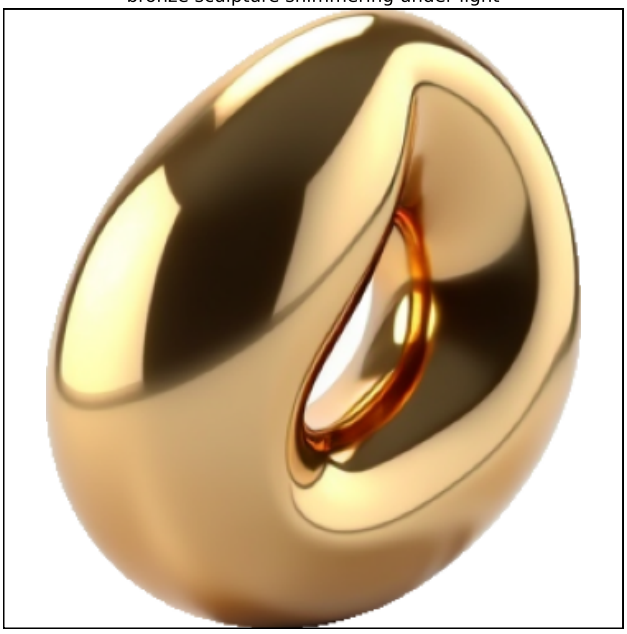
**wineglass**
sleek transparent wineglass with etchings

**dispenser**
colorful animated explosion dispenser

**bag**
sturdy green messenger bag

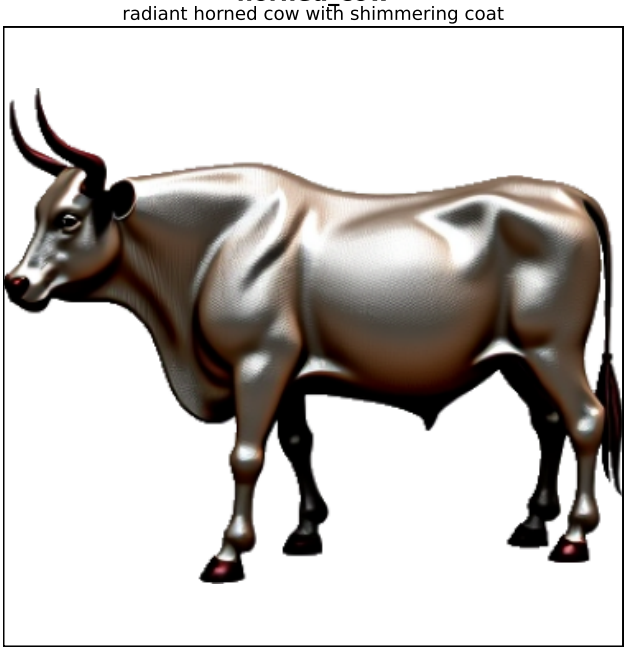
**pony**
pony with flowing sunset mane

**sculpture**
bronze sculpture shimmering under light



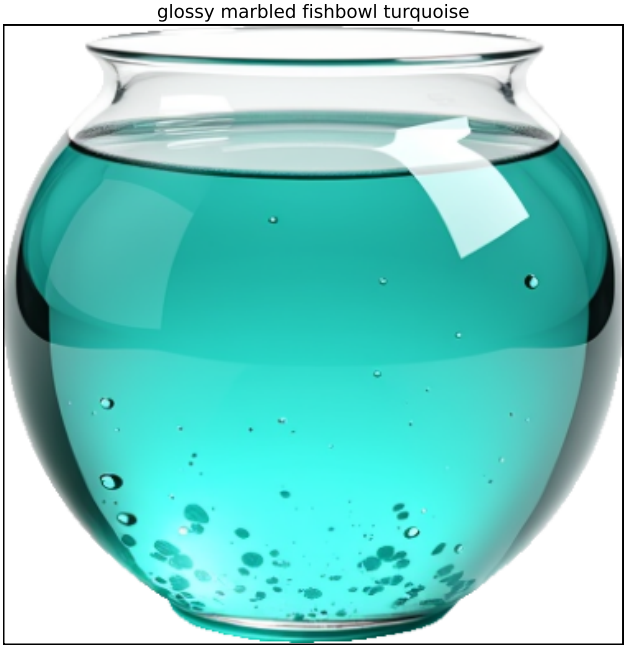
**underwear**
black lacy underwear with stretch

**horned_cow**
radiant horned cow with shimmering coat



**passport**
passport in soft leather

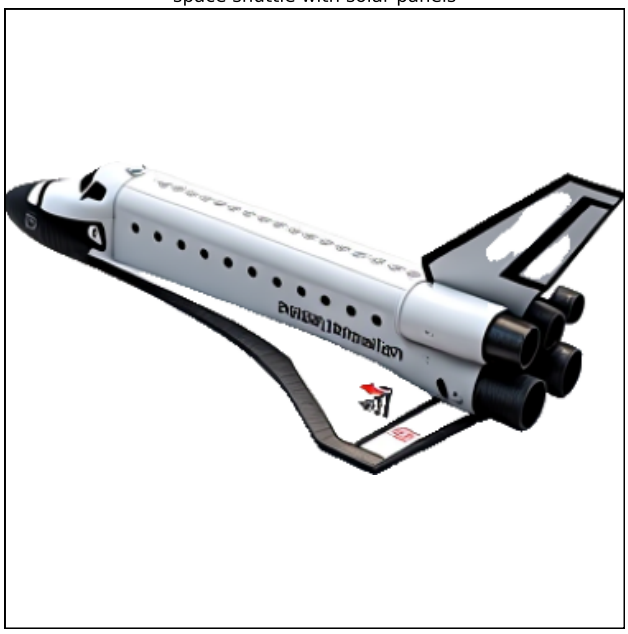

**fishbowl**
glossy marbled fishbowl turquoise



**Chicken**
chicken with white feathers and red comb

**bouquet**
dahlias in vibrant hues

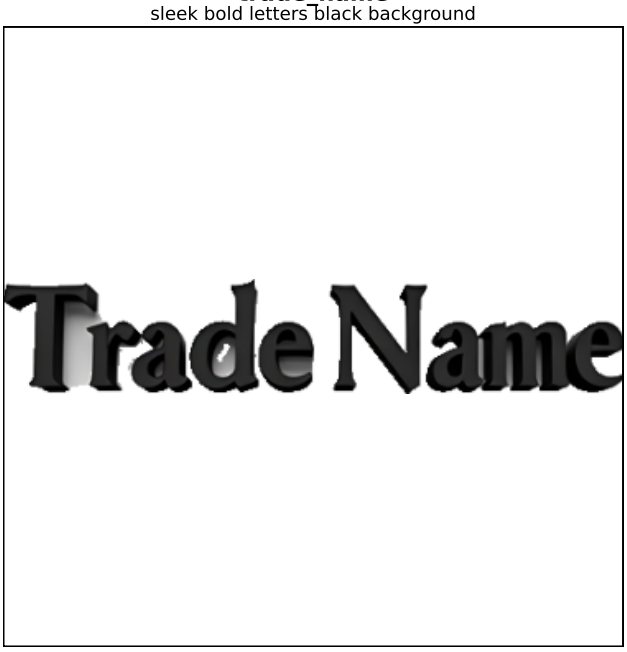
**space_shuttle**
space shuttle with solar panels



**motorcycle**
sleek black motorcycle with chrome

**gemstone**
clear gemstone reflecting like galaxy

**trade_name**
sleek bold letters black background




**recliner**
vintage brown recliner with detailing

**Folder**
sleek metallic folder

**soccer_ball**
soccer ball with floral pattern

**Red_Cabbage**
red cabbage with purple tones

**baseball**
baseball with minor scratches

**forklift**
sleek modern forklift with chrome

**gazelle**
tawny spotted gazelle under sunlight

## 16. Gallery of Training Data (Before Camera Configuration Augmentation)

*Note: Camera configuration augmentation (random zoom and depth-of-field blur) is applied on-the-fly during training. The images shown here are before this augmentation step.*

**Image**

**Segmentation Mask**

## Detections:

{"tongs": [[318, 535, 446, 817]], "bath_towel": [[474, 10, 967, 889]], "Canned": [[376, 217, 478, 464]], "shovel": [[424, 726, 472, 910]], "strap": [[725, 178, 779, 238]], "teakettle": [[324, 789, 427, 906]], "cushion": [[684, 219, 773, 298]], "raspberry": [[183, 324, 317, 464]], "dropper": [[204, 868, 297, 990]], "snowmobile": [[898, 451, 961, 489]], "box": [[305, 837, 351, 879]], "ram_animal": [[152, 704, 187, 740]], "birthday_card": [[34, 924, 79, 970]], "Egg_tart": [[442, 207, 485, 251]], "cornbread": [[191, 621, 233, 663]], "wooden_spoon": [[207, 292, 253, 332]], "motor": [[95, 812, 135, 837]]}

## Expression:

The birthday card with cheerful pirate ship at the bottom: [34, 924, 45, 46]; All glossy objects above the shovel: [[183, 324, 134, 140], [324, 789, 103, 117]]; All striped containers to the right of the wooden spoon: [[898, 451, 63, 38], [305, 837, 46, 42]]; The snowmobile with white surface on the far right: [898, 451, 63, 38]; The wooden spoon for dough near the cornbread with golden flecks: [207, 292, 46, 40]; The tongs with rough iron texture: [318, 535, 128, 282]; All containers: [[376, 217, 102, 247], [324, 789, 103, 117], [305, 837, 46, 42]]; The egg tart with marigold custard on the left side: [442, 207, 43, 44]; The hunter green leather cushion: [684, 219, 89, 79]

**Image**

**Segmentation Mask**

## Detections:

{"remote_control": [[106, 491, 378, 718]], "Board_Eraser": [[648, 31, 856, 242]], "earring": [[380, 96, 441, 156]], "hookah": [[796, 110, 825, 201]], "Dixie_cup": [[900, 376, 942, 432]], "ottoman": [[312, 260, 376, 307]], "turtle": [[738, 407, 796, 437]], "detergent": [[123, 444, 145, 517]], "root_beer": [[245, 545, 262, 587]], "Swan": [[838, 330, 864, 365]]}

## Expression:

All the containers below the turtle with stardust shell: [[106, 491, 272, 227], [123, 444, 22, 73], [245, 545, 17, 42]]; The Dixie cup shaped like bell: [900, 376, 42, 56]; The hookah on the far right: [796, 110, 29, 91]; All the objects to the left of center: [[106, 491, 272, 227], [123, 444, 22, 73], [245, 545, 17, 42]]; The industrial-styled hookah with grays above the Dixie cup shaped like bell: [796, 110, 29, 91]; The detergent bottle on the far left: [123, 444, 22, 73]; All the objects with metallic finishes: [[796, 110, 29, 91], [312, 260, 64, 47], [648, 31, 208, 211]]; All the metallic objects to the right of the detergent bottle: [[796, 110, 29, 91], [312, 260, 64, 47]]; All the containers: [[106, 491, 272, 227], [123, 444, 22, 73], [245, 545, 17, 42], [900, 376, 42, 56]]

**Image**

**Segmentation Mask**

## Detections:

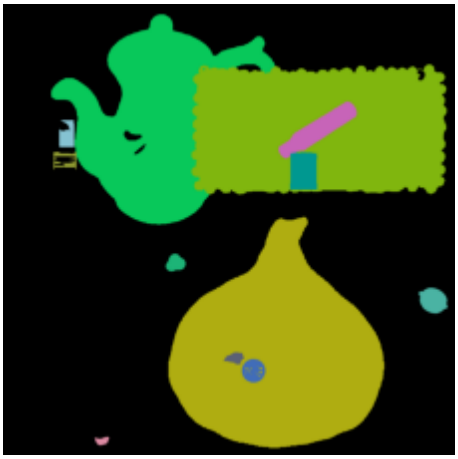
{"army_tank": [[475, 48, 966, 280]], "Seal": [[1, 119, 267, 388]], "boom_microphone": [[612, 86, 724, 383]], "deer": [[138, 631, 514, 1004]], "choker": [[702, 573, 792, 632]], "fireplace": [[780, 898, 894, 983]], "tank_top_clothing": [[400, 273, 431, 331]], "bottle": [[522, 44, 553, 107]], "papaya": [[618, 343, 714, 487]], "Papyrus": [[174, 586, 215, 635]], "raccoons": [[530, 629, 586, 677]], "Gas_stove": [[500, 391, 538, 434]], "Apple_rust_leaf": [[244, 223, 268, 265]], "soap": [[190, 780, 231, 820]], "basket": [[952, 297, 990, 337]], "machine_gun": [[415, 936, 465, 950]]}

## Expression:

The graceful and agile deer beside the raccoons with detailed fur masks: [138, 631, 376, 373]; The ringed seal with round rings on the far left: [1, 119, 266, 269]; The vibrant red boom microphone: [612, 86, 112, 297]; The vibrant red boom microphone on the right side: [612, 86, 112, 297]; All items with intricate patterns or carvings: [[475, 48, 491, 232], [612, 86, 112, 297], [415, 936, 50, 14], [400, 273, 31, 58]]; All objects to the left of the vibrant paisley print tank top: [[1, 119, 266, 269], [522, 44, 31, 63], [174, 586, 41, 49], [530, 629, 56, 48]]; The army tank suggesting speed above the ringed seal with round rings: [475, 48, 491, 232]; All objects in the top half of the image: [[475, 48, 491, 232], [1, 119, 266, 269], [612, 86, 112, 297], [522, 44, 31, 63], [618, 343, 96, 144]]; All objects with smooth finishes: [[702, 573, 90, 59], [190, 780, 41, 40], [618, 343, 96, 144]]

**Image**

**Segmentation Mask**



## Detections:

{"coffeepot": [[109, 28, 615, 503]], "label": [[426, 141, 1003, 435]], "Dumpling": [[374, 486, 862, 1008]], "Board_Eraser": [[623, 225, 800, 354]], "lemon": [[940, 646, 1005, 704]], "Notepaper": [[649, 339, 710, 424]], "hand_towel": [[127, 267, 161, 328]], "bathtub": [[207, 983, 240, 1002]], "bunk_bed": [[114, 339, 166, 379]], "dental_floss": [[368, 569, 413, 609]], "bedspread": [[499, 792, 545, 819]], "dartboard": [[540, 807, 593, 860]]}

## Expression:

The plump dumpling steaming with anticipation below the label of glittering sequins: [374, 486, 488, 522]; The metallic silver eraser with swirls to the right of the notepaper: [623, 225, 177, 129]; The reed dartboard with patterns at the bottom center: [540, 807, 53, 53]; All blue items with patterns: [[109, 28, 506, 475], [374, 486, 488, 522], [499, 792, 46, 27]]; The marble print hand towel elegance near the top left corner: [127, 267, 34, 61]; The squashed lemon with green hint beside the dental floss in black: [940, 646, 65, 58]; The blue coffeepot with swirling pattern next to the label of glittering sequins: [109, 28, 506, 475]; The marble print hand towel elegance above the bathtub made of recycled paper: [127, 267, 34, 61]; All items made from recycled materials: [[109, 28, 506, 475], [207, 983, 33, 19]]

**Image**

**Segmentation Mask**

## Detections:

{"coconut": [[615, 240, 796, 409]], "swivel_chair": [[810, 633, 955, 767]], "Optima": [[546, 605, 646, 710]], "truck": [[442, 214, 492, 236]], "eagle": [[887, 284, 924, 329]]}

## Expression:

The slate gray swivel chair with armrest at the bottom, near the truck: [810, 633, 145, 134]; The eagle with dark glossy feathers: [887, 284, 37, 45]; The eagle with dark glossy feathers above the truck: [887, 284, 37, 45]; All objects above the truck: [[615, 240, 181, 169], [546, 605, 100, 105], [887, 284, 37, 45]]; All objects with dark or black color: [[442, 214, 50, 22], [887, 284, 37, 45]]; All objects near the bottom: [[810, 633, 145, 134], [442, 214, 50, 22]]; The glossy black truck with flames: [442, 214, 50, 22]; The coconut with faint rings next to the eagle with dark glossy feathers: [615, 240, 181, 169]

## Image

## Segmentation Mask


# Detections:

{"arctic_type_of_shoe": [[303, 696, 632, 1014]], "Dessert": [[664, 223, 845, 407]], "Other_Shoes": [[310, 298, 707, 588]], "coffee_maker": [[113, 191, 522, 500]], "bait": [[672, 861, 807, 917]], "shopping_cart": [[177, 576, 235, 634]], "receipt": [[701, 51, 805, 183]], "crib": [[121, 634, 231, 735]], "tiger": [[71, 213, 178, 270]], "ladle": [[199, 242, 335, 388]], "crawfish": [[771, 333, 808, 360]], "aquarium": [[901, 805, 947, 860]], "loveseat": [[451, 677, 507, 707]], "eyepatch": [[406, 453, 446, 474]], "Gas_stove": [[847, 506, 916, 543]], "cart": [[103, 936, 162, 978]], "sugarcane_plant": [[396, 255, 428, 292]], "sparkler_fireworks": [[589, 907, 607, 933]]}

# Expression:

The rustic burlap shoes with textures: [310, 298, 397, 290]; All items with patterns: [[664, 223, 181, 184], [71, 213, 107, 57], [103, 936, 59, 42]]; The bait with color-changing scales below the pastel pink cart with soft glow: [672, 861, 135, 56]; The arctic shoe with leather upper: [303, 696, 329, 318]; The thin sparkler with glittering tail at the bottom edge: [589, 907, 18, 26]; The loveseat with classic elegance near the center: [451, 677, 56, 30]; All metallic objects above the gas stove with spiral etchings: [[303, 696, 329, 318], [113, 191, 409, 309]]; The receipt with jagged edges at the top center: [701, 51, 104, 132]; All items along the right edge: [[771, 333, 37, 27], [901, 805, 46, 55], [589, 907, 18, 26]]

**Image**

**Segmentation Mask**

## Detections:

{"ashtray": [[101, 468, 506, 853]], "Candy": [[674, 352, 922, 569]], "gondola_boat": [[557, 141, 956, 341]], "sharpener": [[82, 830, 190, 937]], "turtle": [[666, 644, 782, 782]], "Sailboat": [[646, 382, 685, 493]], "catRussian_Blue": [[520, 370, 565, 434]], "dog": [[135, 665, 244, 798]], "brussels_sprouts": [[248, 449, 304, 491]], "lambchop": [[264, 320, 299, 364]], "dognewfoundland": [[313, 620, 360, 666]], "Green_Onion": [[360, 645, 390, 687]], "myright": [[438, 175, 472, 233]], "saddle_blanket": [[238, 720, 287, 759]]}

## Expression:

The gondola boat in the upper middle: [557, 141, 399, 200]; The geometric ashtray with sharp angles: [101, 468, 405, 385]; The sturdy gondola in emerald green: [557, 141, 399, 200]; All items to the right of the ashtray: [[674, 352, 248, 217], [557, 141, 399, 200], [666, 644, 116, 138], [520, 370, 45, 64], [135, 665, 109, 133], [360, 645, 30, 42]]; All objects with a glossy finish: [[101, 468, 405, 385], [674, 352, 248, 217], [557, 141, 399, 200], [666, 644, 116, 138], [135, 665, 109, 133]]; The shimmering pink candy above the sailboat with black hull: [674, 352, 248, 217]; The large tan dog with lion appearance: [135, 665, 109, 133]; The shimmering pink jelly candy: [674, 352, 248, 217]; All food items to the left of the turtle: [[248, 449, 56, 42], [264, 320, 35, 44], [360, 645, 30, 42]]

**Image**

**Segmentation Mask**

## Detections:

{"padlock": [[714, 681, 875, 927]], "shoulder_bag": [[233, 261, 765, 787]], "fighter_jet": [[643, 266, 1016, 631]], "mint_candy": [[860, 62, 976, 179]], "street_sign": [[560, 268, 608, 375]], "gravestone": [[277, 313, 328, 387]], "desk": [[601, 238, 680, 281]], "perfume": [[210, 850, 243, 899]], "pool_table_billiard_table_snooker_table": [[536, 418, 573, 432]], "ram_animal": [[297, 133, 344, 188]], "packet": [[272, 602, 316, 654]]}

## Expression:

The ram with rugged flesh at the top left: [297, 133, 47, 55]; Vibrant items to the right of the jet: [[860, 62, 116, 117], [272, 602, 44, 52]]; The desk with red surface near the center: [601, 238, 79, 43]; The packet with holographic shimmer: [272, 602, 44, 52]; All items to the left of the center: [[233, 261, 532, 526], [277, 313, 51, 74], [297, 133, 47, 55]]; The matte black padlock with elegance: [714, 681, 161, 246]; Containers near the center with red surfaces: [[601, 238, 79, 43], [210, 850, 33, 49]]; The quirky yellow cartoon character bag: [233, 261, 532, 526]; The pool table with black walnut on the right side: [536, 418, 37, 14]

## Image

## Segmentation Mask

# Detections:

{"Rice_Cooker": [[387, 288, 847, 718]], "necklace": [[93, 214, 398, 607]], "broach": [[120, 514, 230, 625]], "wedding_cake": [[738, 37, 807, 137]], "bow_decorative_ribbons": [[335, 191, 487, 329]], "tablecloth": [[623, 223, 685, 261]], "machine_gun": [[275, 60, 333, 85]], "ski_pole": [[864, 864, 868, 900]]}

# Expression:

The necklace shimmering opalescent beads on the far left: [93, 214, 305, 393]; The aubergine bow with floral embroidery between the rice cooker and the tablecloth: [335, 191, 152, 138]; The angular, geometric silver-black machine_gun near the top left corner: [623, 223, 62, 38]; All the red items near the center of the image: [[387, 288, 460, 430], [120, 514, 110, 111]]; The red finish rice cooker with silver: [387, 288, 460, 430]; The red broach with Tudor rose below the necklace and to the left of the machine_gun: [120, 514, 110, 111]; All the items with red accents: [[387, 288, 460, 430], [93, 214, 305, 393], [120, 514, 110, 111]]; All the decorative items with floral patterns: [[335, 191, 152, 138], [120, 514, 110, 111]]; All the objects near the bottom right corner: [[864, 864, 4, 36], [738, 37, 69, 100]]

## Image

## Segmentation Mask

## Detections:

{"elk": [[478, 224, 921, 844]], "label": [[31, 330, 547, 725]], "bracelet": [[510, 200, 739, 403]], "breechcloth": [[838, 719, 969, 876]], "Soyabean_leaf": [[447, 224, 592, 380]], "birdhouse": [[396, 662, 458, 768]], "saltshaker": [[336, 197, 409, 343]], "blinder_for_horses": [[899, 464, 941, 501]], "awning": [[947, 892, 998, 924]], "cabinet": [[169, 118, 197, 156]], "dogenglish_setter": [[564, 315, 586, 352]], "Kohlrabi": [[35, 768, 79, 814]], "hotair_balloon": [[106, 394, 139, 443]]}

## Expression:

The fashionable leather bracelet with studs: [510, 200, 229, 203]; The dense label with barcode: [31, 330, 516, 395]; The recycled tin birdhouse in red, blue: [396, 662, 62, 106]; The crisp thin soyabean leaf gradients near the recycled tin birdhouse: [447, 224, 145, 156]; All vibrant-colored objects surrounding the majestic setter with layered coat: [[478, 224, 443, 620], [31, 330, 516, 395], [510, 200, 229, 203]]; The cabinet with glass panels below the marine blue awning: [169, 118, 28, 38]; The silver fabric hot-air balloon above the center: [106, 394, 33, 49]; All objects along the left edge of the image: [[31, 330, 516, 395], [447, 224, 145, 156], [35, 768, 44, 46]]; The saltshaker at the bottom left corner: [336, 197, 73, 146]

## References


[1] Zhenxiong Tan, Songhua Liu, Xingyi Yang, Qiaochu Xue, and Xinchao Wang. Ominicontrol: Minimal and universal control for diffusion transformer. *arXiv preprint arXiv:2411.15098*, 2024. 2

[2] Germán Ros, Laura Sellart, Joanna Materzynska, David Vázquez, and Antonio M. López. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3234–3243, 2016. 2

[3] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *European Conference on Computer Vision (ECCV)*, pages 102–118, 2016. 2

[4] Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual KITTI 2. *arXiv preprint arXiv:2001.10773*, 2020. 2

[5] Magnus Wrenninge and Jonas Unger. Synscapes: A photorealistic synthetic dataset for street scene parsing. *arXiv preprint arXiv:1810.08705*, 2018. 2

[6] Mike Roberts, Jason Ramapuram, Anurag Ranjan, Atulit Kumar, Miguel A. Bautista, Nathan Paczan, Russ Webb, and Joshua M. Susskind. Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10912–10922, 2021. 2

[7] Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D. Cubuk, Quoc V. Le, and Barret Zoph. Simple copy-paste is a strong data augmentation method for instance segmentation, 2021. 2

[8] Haoshu Fang, Jianhua Sun, Runzhong Wang, Minghao Gou, Yong-Lu Li, and Cewu Lu. Instaboost: Boosting instance segmentation via probability map guided copy-pasting. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 682–691, 2019. 2

[9] Hanqing Zhao, Dianmo Sheng, Jianmin Bao, Dongdong Chen, Dong Chen, Fang Wen, Lu Yuan, Ce Liu, Wenbo Zhou, Qi Chu, Weiming Zhang, and Nenghai Yu. X-paste: Revisiting scalable copy-paste for instance segmentation using clip and stablediffusion, 2023. 2

[10] Ruozhen He, Ziyan Yang, Paola Cascante-Bonilla, Alexander C. Berg, and Vicente Ordonez. Learning from synthetic data for visual grounding, 2024. 2

[11] Shijie Wang, Dahun Kim, Ali Taalimi, Chen Sun, and Weicheng Kuo. Learning visual grounding from generative vision and language model, 2024. 2

[12] Robin Rombach, A. Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10674–10685, 2021. 2

[13] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. In *International Conference on Machine Learning*, 2021.

[14] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *ArXiv*, abs/2204.06125, 2022. 2

[15] 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, 2018. 3

[16] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Scaling in-the-wild training for diffusion-based illumination harmonization and editing by imposing consistent light transport. In *The Thirteenth International Conference on Learning Representations*, 2025. 3

[17] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. 10