# Promising or Elusive? Unsupervised Object Segmentation from Real-world Single Images

**Yafei Yang** **Bo Yang**

vLAR Group, The Hong Kong Polytechnic University

ya-fei.yang@connect.polyu.hk bo.yang@polyu.edu.hk

## Abstract

In this paper, we study the problem of unsupervised object segmentation from single images. We do not introduce a new algorithm, but systematically investigate the effectiveness of existing unsupervised models on challenging real-world images. We firstly introduce four complexity factors to quantitatively measure the distributions of object- and scene-level biases in appearance and geometry for datasets with human annotations. With the aid of these factors, we empirically find that, not surprisingly, existing unsupervised models catastrophically fail to segment generic objects in real-world images, although they can easily achieve excellent performance on numerous simple synthetic datasets, due to the vast gap in objectness biases between synthetic and real images. By conducting extensive experiments on multiple groups of ablated real-world datasets, we ultimately find that the key factors underlying the colossal failure of existing unsupervised models on real-world images are the challenging distributions of object- and scene-level biases in appearance and geometry. Because of this, the inductive biases introduced in existing unsupervised models can hardly capture the diverse object distributions. Our research results suggest that future work should exploit more explicit objectness biases in the network design.

## 1 Introduction

The capability of automatically identifying individual objects from complex visual observations is a central aspect of human intelligence [53]. It serves as the key building block for higher-level cognition tasks such as planning and reasoning [28]. In last years, a plethora of models have been proposed to segment objects from single static images in an unsupervised fashion: from the early AIR [22] and MONet [8] to the recent SPACE [39], SlotAtt [40], GENESIS-V2 [20], *etc*. They jointly learn to represent and segment multiple objects from a single image, without needing any human annotations in training. This process is often called perceptual grouping/binding or object-centric learning. These methods and their variants have achieved impressive segmentation results on numerous synthetic scene datasets such as dSprites [42] and CLEVR [33]. Such advances come with great expectations that the unsupervised techniques would likely close the gap with fully-supervised methods for real-world visual understanding. However, few work has systematically investigated the true potential of the emerging unsupervised object segmentation models on complex real-world images such as COCO dataset [38]. This naturally raises an essential question:

*Is it promising or even possible to segment generic objects from real-world single images using (existing) unsupervised methods?*

**What is an object?** To answer the above question involves another fundamental question: what is an object? Exactly 100 years ago in Gestalt psychology, Wertheimer [61] first introduced a set of Gestalt principles such as proximity, similarity and continuation to heuristically define visual

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

Figure 1: The failure of SlotAtt [40] on three real-world images (right-hand side), although it can perfectly segment simple objects on three synthetic images (left-hand side).

data as objects. However, these factors are highly subjective, whilst the real-world generic objects are far more complex with extremely diverse appearances and shapes. Therefore, it is practically impossible to quantitatively define what is an object, *i.e.*, the objectness, from visual inputs (*e.g.*, a set of image pixels). Nevertheless, to thoroughly understand whether unsupervised methods can truly learn objectness akin to the psychological process of humans, it is vital to investigate the underlying factors that potentially facilitate or otherwise hinder the ability of unsupervised models. In this regard, by drawing on Gestalt principles, we instead define a series of new factors to quantitatively measure the complexity of objects and scenes in Section 2. By taking into account both the appearance and geometry of objects and scenes, our complexity factors explicitly assess the difficulty of segmenting a specific object. For example, a chair with colorful textures tend to have higher complexity than a single-color ball for unsupervised methods. With the aid of these factors, we extensively study whether and how existing unsupervised models can discover objects in Section 4.

**What is the problem of unsupervised object segmentation from single images?** A large number of models [63] aim to tackle the problem of unsupervised object segmentation from single images. They share several key problem settings: 1) all training images do not have any human annotations; 2) every single image has multiple objects; 3) each image is treated as a static data point without any dynamic or temporal information; 4) all models are trained from scratch without requiring any pretrained networks on additional datasets. Ultimately, the goal of these models is to segment all individual objects as accurate as the ground truth human annotations. In this paper, we regard these settings as the basic and necessary part of unsupervised object segmentation from single images, and empirically evaluate how successfully the existing models can exhibit on real-world images.

**Contributions and findings.** This paper addresses the essential question regarding the potential of unsupervised segmentation of generic objects from real-world single images. Our contributions are:

- We firstly introduce 4 complexity factors to quantitatively measure the difficulty of objects and scenes. These factors are key to investigate the true potential of existing unsupervised models.
- We extensively evaluate current unsupervised approaches in a large-scale experimental study. We implement 4 representative methods and train more than 130 models on 6 curated datasets from scratch. The datasets, code and pretrained models are available at `https://github.com/vLAR-group/UnsupObjSeg`
- We analyze our experimental results and find that: 1) existing unsupervised object segmentation models cannot discover generic objects from single real-world images, although they can achieve outstanding performance on synthetic datasets, as qualitatively illustrated in Figure 1; 2) the challenging distributions of both object- and scene-level biases in appearance and geometry from real-world images are the key factors incurring the failure of existing models; 3) the inductive biases introduced in existing unsupervised models are fundamentally not matched with the objectness biases exhibited in real-world images, and therefore fail to discover the real objectness.

**Related Work.** Recently, ClevrTex [35] and the concurrent work [43] also study unsupervised object segmentation on single images. Through evaluation on (complex) synthetic datasets only, both works focus on benchmarking the effectiveness of particular network designs of baselines. By comparison, our paper aims to explore what and how the objectness distribution gaps between synthetic and real-world images incur the failure of existing models. The recent work [60] which investigates video object discovery is orthogonal to our work as the motion signals do not exist in single images.

**Scope of this research.** This paper does not investigate unsupervised object discovery on saliency maps [59], static multi-views or dynamic videos [63]. Recent methods [10; 30] requiring pretrained models on monolithic object images such as ImageNet [47] are not evaluated as well.

## 2 Complexity Factors

As illustrated in the top row of Figure 2, an individual object, represented by a set of color pixels painted within a mask, can vary significantly given different types of appearance and geometric shape. A specific scene, represented by a set of objects placed within an image, can also differ vastly given different types of relative appearance and geometric layout between objects, as illustrated in the bottom row. Unarguably, such variation and complexity of appearance and geometry in both object level and scene level directly affects human's ability to precisely separate all objects. Naturally, the performance of unsupervised segmentation models are also expected to be influenced by the variation. In this regard, we carefully define the following two groups of factors to quantitatively describe the complexity of different datasets.

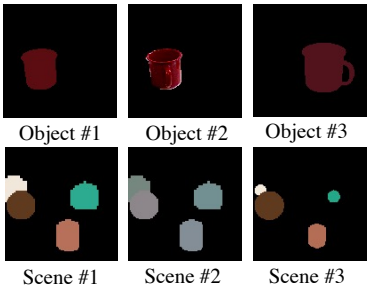


Figure 2: Complexity in appearance and geometry for objects and scenes.

### 2.1 Object-level Complexity Factors

As to a specific object, all its information can be described by appearance and geometry. Therefore we define the below two factors to measure the complexity of appearance and geometry respectively. Notably, both factors are nicely invariant to the object scale.

- **Object Color Gradient:** This factor aims to calculate how frequently the appearance changes within the object mask. In particular, given the RGB image and mask of an object, we firstly convert RGB into grayscale and then apply Sobel filter [51] to compute the gradients horizontally and vertically for each pixel within the mask. The final gradient value is obtained by averaging out all object pixels. Note that, the object boundary pixels are removed to avoid the interference of background. Numerically, the higher this factor is, the more complex texture and/or lighting effect the object has, and therefore it is likely harder to segment.
- **Object Shape Concavity:** This factor is designed to evaluate how irregular the object boundary is. Particularly, given an object (binary) mask, denoted as $\boldsymbol{M}_{obj} \in \mathbb{R}^{H \times W}$, we firstly find the smallest convex polygon mask ($\boldsymbol{M}_{cvx} \in \mathbb{R}^{H \times W}$) that surrounds the object mask using an existing algorithm [19], and then the object shape concavity value is computed as: $1 - \sum \boldsymbol{M}_{obj} / \sum \boldsymbol{M}_{cvx}$. Clearly, the higher this factor is, the more irregular object shape is, and segmentation is more tricky.

### 2.2 Scene-level Complexity Factors

As to a specific image, in addition to the object-level complexity, the spatial and appearance relationships between all objects can also incur extra difficulty for segmentation. We define the following two factors to quantify the complexity of relative appearance and geometry between objects in an image.

- **Inter-object Color Similarity:** This factor intends to assess the appearance similarity between all objects in the same image. Specifically, we firstly calculate the average color for each object, and then compute the pair-wise Euclidean distances of object colors, obtaining a $K \times K$ matrix where $K$ represents the object number. The *average color distance* is calculated by averaging the matrix excluding diagonal entries, and the final inter-object color similarity is computed as: $1 - \textit{average color distance} / (255 \times \sqrt{3})$. Intuitively, the higher this factor is, the more similar all objects appear to be, the less distinctive each object is, and it is harder to separate each object.
- **Inter-object Shape Variation:** This factor aims to measure the relative geometry diversity between all objects in the image. We firstly calculate the diagonal length of bounding box for each object, and then compute the pair-wise absolute differences for all object diagonal lengths, obtaining a $K \times K$ matrix. The final inter-object shape variation is the average of the matrix excluding diagonal entries. The higher this factor, the objects within an image have more diverse and imbalanced sizes, and therefore segmenting both gigantic and tiny objects is likely more challenging.

By capturing the appearance and geometry in both object and scene levels, the four factors are designed to quantify the complexity of objects and images. For illustration, Figure 3 shows sample images for the four factors at different values. The higher the values, the more complex the objects

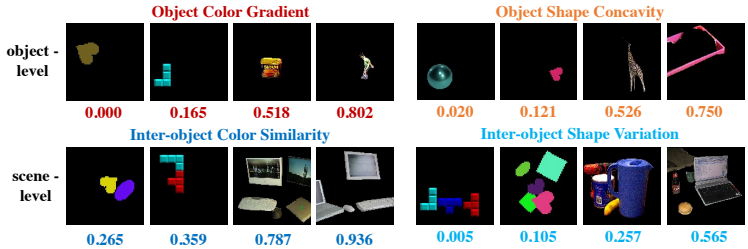


Figure 3: Sample objects and scenes for the four factors at different complexity values. All complexity values are normalized to the range of $[0, 1]$.

and scenes. In fact, these factors are carefully selected from more than 10 candidates because they are empirically more suitable to differentiate the gaps between synthetic and real-world images, and they eventually serve as key indicators to diagnose existing unsupervised models in Section 4. Calculation details of the four factors and other candidates are in appendix.

# 3 Experimental Design

## 3.1 Considered Methods

A range of works have explored unsupervised object segmentation in recent years. They are typically formulated as (variational) autoencoders (AE/VAE) [36] or generative adversarial networks (GAN) [24]. GAN based models [13; 3; 7; 56; 5; 58; 1] are usually limited to identifying a single foreground object and can hardly discover multiple objects due to the training instabilities, therefore not considered in this paper. As shown in Table 1, the majority of existing models are based on AE/VAE and can be generally divided into two groups according to the object representation:

- **Factor-based models**: Each object is represented by explicit factors such as size, position, appearance, *etc*., and the whole image is a spatial organization of multiple objects. Basically, such representation explicitly enforces objects to be bounded within particular regions.
- **Layer-based models**: Each object is represented by an image layer, *i.e*., a binary mask, and the whole image is a spatial mixture of multiple object layers. Intuitively, this representation does not have strict spatial constrains, and instead is more flexible to cluster similar pixels as objects.

In order to decompose the input images into objects, these approaches introduce different types of network architecture, loss functions, and regularization terms as inductive biases. These biases broadly include: 1) variational encoding which encourages the disentanglement of latent variables; 2) iterative inference which likely ends up with better scene representations over occlusions; 3) object relationship regularization such as depth estimation and autoregressive prior which aims at capturing the dependency of multiple objects; and many other biases. With different combinations of these biases, many methods have shown outstanding performance in synthetic datasets. Among them, we select 4 representative models for our investigation: 1) AIR [22], 2) MONet [8], 3) IODINE [25], and 4) SlotAtt [40]. We also add the fully-supervised Mask R-CNN [29] as an additional baseline for comprehensive comparison. Implementation details are provided in appendix.

## 3.2 Considered Datasets

We consider two groups of datasets for extensive benchmarking and analysis: 1) three commonly-used synthetic datasets: dSprites [42], Tetris [34] and CLEVR [33], 2) three real-world datasets: YCB [9], ScanNet [17], and COCO [38], representing the small-scale, indoor- and outdoor-level real scenes respectively. Naturally, objects and scenes in different datasets tend to have very different types of biases. For example, the objects in dSprites tend to have the single-color bias, while COCO does not. Generally, the object-level biases can be divided as: 1) appearance biases including different textures and lighting effects, and 2) geometry biases including the object shape and occlusions. Similarly, the scene-level biases include: 1) appearance biases such as the color similarity between all objects, and 2) geometry biases such as the diversity of all object shapes. In fact, our complexity factors introduced in Section 2 are designed to well capture these biases. Table 2 qualitatively summarizes the biases of selected datasets. We may hypothesize that the large gaps of biases between synthetic and real-world datasets would have a huge impact on the effectiveness of existing models.

Table 1: Existing unsupervised models for object segmentation on single images. Each model includes different inductive biases, such as variational autoencoding (VAE), iterative inference (Iter), object relationship regularization (Rel), *etc*.

| Factor-based Models | | Inductive Biases | | | Layer-based Models | | Inductive Biases | | |
|---|---|---|---|---|---|---|---|---|---|
| | | VAE | Iter | Rel | | | VAE | Iter | Rel |
| CST-VAE [31] | ICLRW'16 | ✓ | | | Tagger [26] | NIPS'16 | | ✓ | |
| AIR [22] | NIPS'16 | ✓ | | | RC [26] | ICLRW'16 | | ✓ | |
| SPAIR [16] | AAAI'19 | ✓ | | | NEM [27] | NIPS'17 | | ✓ | |
| SuPAIR [54] | ICML'19 | ✓ | | | MONet [8] | arXiv'19 | ✓ | | |
| GMIO [64] | ICML'19 | ✓ | ✓ | | IODINE [25] | ICML'19 | ✓ | ✓ | |
| ASR [62] | NeurIPS'19 | ✓ | | | ECON [57] | ICLRW'20 | ✓ | | ✓ |
| SPACE [39] | ICLR'20 | ✓ | | | GENESIS [21] | ICLR'20 | ✓ | | ✓ |
| GNM [32] | NeurIPS'20 | ✓ | | ✓ | SlotAtt [40] | NeurIPS'20 | | ✓ | |
| SPLIT [11] | arXiv'20 | ✓ | | | GENESIS-V2 [20] | NeurIPS'21 | ✓ | | ✓ |
| OCIC [2] | arXiv'20 | ✓ | | ✓ | R-MONet [50] | arXiv'21 | ✓ | | ✓ |
| GSGN [18] | ICLR'21 | ✓ | | ✓ | CAE [41] | arXiv'22 | | | ✓ |

Table 2: The object- and scene-level biases in appearance and geometry of the considered datasets.

| | | Synthetic Datasets | | | Real-world Datasets | | |
|---|---|---|---|---|---|---|---|
| | | dSprites [42] | Tetris [34] | CLEVR [33] | YCB [9] | ScanNet [17] | COCO [38] |
| *Object-level Biases* | | | | | | | |
| Appearance | Texture: | simple | simple | simple | diverse | simple | diverse |
| | Lighting: | no | no | synthetic | real | real | real |
| Geometry | Shape: | simple | simple | simple | simple | diverse | diverse |
| | Occlusion: | minor | no | minor | severe | severe | severe |
| *Scene-level Biases* | | | | | | | |
| Appearance | Similarity: | low | low | high | high | high | high |
| Geometry | Diversity: | low | low | low | high | high | high |

To guarantee the fairness and consistency of all experiments, we carefully prepare all six datasets using the following same protocols. Preparation details for each dataset are provided in appendix.

- All images are rerendered or cropped with the same resolution of $128 \times 128$.
- Each image has about 2 to 6 solid objects with a blank background.
- Each dataset has about 10000 images for training, 2000 images for testing.

### 3.3 Considered Metrics

Having the six representative datasets and four existing unsupervised methods at hand, we choose the following metrics to evaluate the object segmentation performance: 1) AP score which is widely used for object detection and segmentation [23], 2) PQ score which is used to measure non-overlap panoptic segmentation [37], and 3) Precision and Recall scores. A predicted mask is considered correct if its IoU against a ground truth mask is above 0.5. All objects are treated as a single class. The blank background is not taken into account for fair comparison. To compute AP, we simply treat the mean value of the soft object mask as the object confidence score. Note that, the alternative metrics ARI [45] and segmentation covering (SC) [4] are not considered as they can be easily saturated.

## 4 Key Experimental Results

### 4.1 Can current unsupervised models succeed on real-world datasets?

First of all, we evaluate all baselines on our six datasets separately. In particular, we train each model from scratch on each dataset separately. For fair evaluation, we carefully tune the hyperparameters of each model on every dataset and fully optimize the networks until convergence. Figure 4 compares the quantitative results. It can be seen that all methods demonstrate satisfactory segmentation results on synthetic datasets, especially the recent strong baselines IODINE and SlotAtt. However, not surprisingly, all unsupervised methods fail catastrophically on the three real-world datasets.

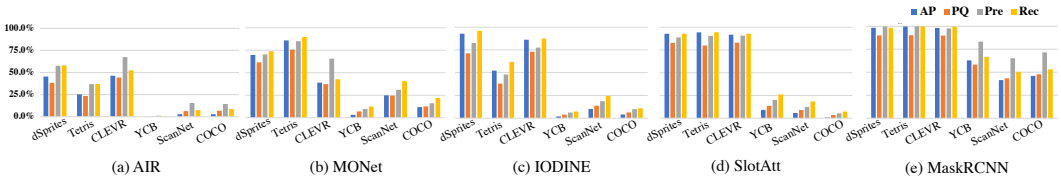


Figure 4: Quantitative results of object segmentation from the five methods on six datasets.

**Preliminary Diagnosis:** In order to diagnose the colossal failure, we hypothesize that it is because of the huge gaps in objectness biases between two types of datasets. In this regard, we quantitatively compute the distributions of our four complexity factors on the six datasets. In particular, the two object-level factors, *i.e.*, Object Color Gradient and Object Shape Concavity, are computed for each object of the six training splits. The two scene-level factors, *i.e.*, Inter-object Color Similarity and Inter-object Shape Variation, are computed for each image of the six training splits.

From the distributions shown in the top row of Figure 5, we can see that, 1) for the two object-level factors (Subfigs 1-a and 1-c), the three synthetic datasets tend to have extremely lower scores than the real-world datasets, which means that the synthetic objects are more likely have uniform colors and convex shapes; 2) for the two scene-level factors (Subfigs 1-b and 1-d), the images in synthetic datasets tend to include less similar objects in terms of color, which means that multiple objects in real-world scenes are less distinctive in appearance. In addition, the multiple objects in synthetic scenes tend to have similar sizes, whereas real-world scenes usually have diverse object scales in single images. To validate whether these distribution biases are the true reasons incurring the failure, we conduct extensive ablative experiments in Sections 4.2, 4.3 and 4.4.

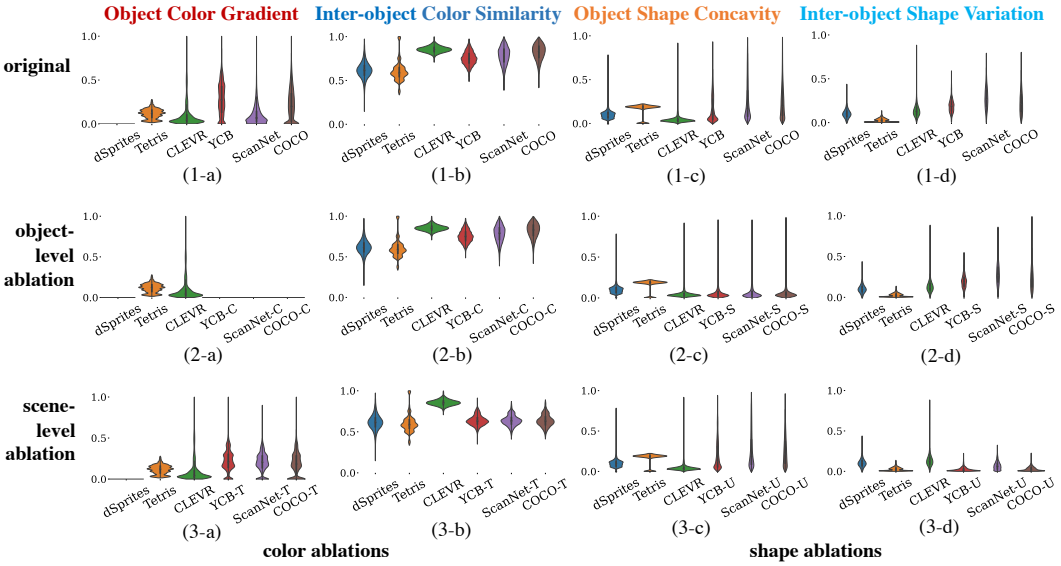


Figure 5: Distributions of the four complexity factors. The *top row* shows the distributions of original synthetic/real-world datasets in Sec 4.1. The *2nd row* shows the distributions of object-level ablated datasets in Sec 4.2. The *3rd row* shows distributions of scene-level ablated datasets in Sec 4.3.

### 4.2 How do object-level factors affect current models?

In this section, we aim to verify to what extent the distributions of object-level factors affect the segmentation performance. In particular, we conduct the following three ablative experiments.

- *Ablation of Object Color Gradient*: For each object of the three real-world datasets, we only replace all pixel colors by its average color $rgb$ value within each object mask, without touching the object shapes. In this way, the **c**olor gradients of each object are totally erased, thus removing the potential impact of Object Color Gradient. The three ablated datasets are: YCB-C / ScanNet-C / COCO-C.
- *Ablation of Object Shape Concavity*: For each object in the real-world datasets, we find the smallest convex hull [19] for its object mask and then fill the empty pixels by shifting the original object

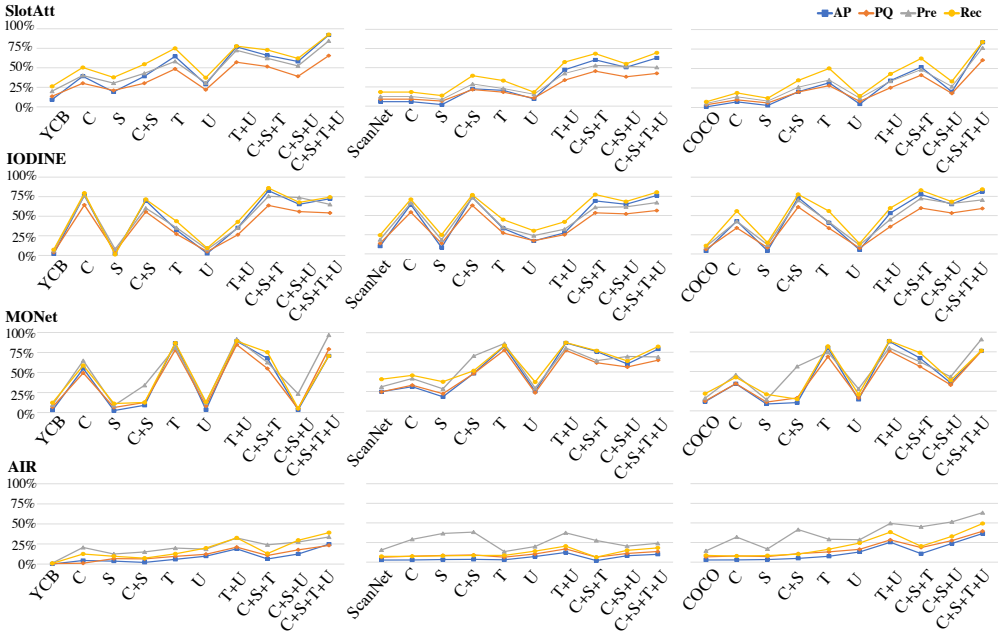



Figure 7: Quantitative results of baselines on the three real-world datasets and their variants. The letters C/S/C+S represent the three ablated datasets in Sec 4.2; T/U/T+U represent the three ablated datasets in Sec 4.3; C+S+T/C+S+U/C+S+T+U represent the three ablated datasets in Sec 4.4.

pixels. Basically, this ablation aims to only reduce the irregularity of object shapes, yet retaining the distributions of color gradients. The ablated datasets are: YCB-S / ScanNet-S / COCO-S.

- *Ablation of both Object Color Gradient and Shape Concavity*: We simply combine the above two ablation for each real-world object, getting datasets: YCB-C+S / ScanNet-C+S / COCO-C+S.

For illustration, the 2nd row of Figure 6 shows example images of three ablated datasets: YCB-C / YCB-S / YCB-C+S.

In the 2nd row of Figure 5 (Subfigs 2-a/2-b), we calculate new distributions of both Object Color Gradient and Inter-object Color Similarity on the datasets YCB-C / ScanNet-C / COCO-C. We can see that the object-level gradients become all zeros in Subfig 2-a, even simpler than three synthetic datasets. Yet, the distributions of Inter-object Color Similarity are almost the same as original YCB / ScanNet / COCO, *i.e.*, Subfig 2-b is similar to 1-b.

Similarly, the 2nd row of of Figure 5 (Subfigs 2-c/2-d) shows that the distributions of Object Shape Concavity of ablated datasets now become similar to the synthetic datasets, while the distributions of Inter-object Shape Variation keep the same, *i.e.*, Subfig 2-d is similar to Subfig 1-d. Note that, for the ablation of both Object Color Gradient and Shape Concavity, the distributions will be the same as shown in the 2nd row of Figure 5. Having the three groups of object-level ablated real-world datasets, we then evaluate segmentation performance of the baselines separately.

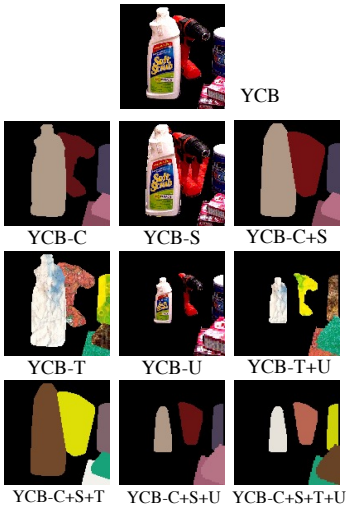



Figure 6: Sample images with different ablations. The *top row* shows an original image from YCB dataset. The *2nd row* shows examples from three ablated datasets in Sec 4.2. The *3rd row* presents examples from three ablated datasets in Sec 4.3. The last row shows examples from three ablated datasets in Sec 4.4

**Brief Analysis:** As shown in Figures 7 & 9, we can see that: 1) Once the pixels of real-world objects are replaced by its mean color, *i.e.*, without any color gradients, the object segmentation performance has been significantly improved for almost all methods. 2) Reducing the irregularity of real-world objects can also improve the object segmentation, although not significantly. 3) Overall, these results show that existing methods are more likely to learn the objectness represented by uniform colors and/or regular objects. However, comparing with Figure 4, the results of

current ablated datasets in Figure 7 still lag behind the synthetic datasets. This means that there must be some other factors that also potentially affect the object segmentation of existing models. More results are in appendix.

### 4.3 How do scene-level factors affect current models?

In this section, we turn to investigate to what extent the distributions of scene-level factors affect the segmentation performance. Particularly, we conduct the following three ablative experiments.

- *Ablation of Inter-object Color Similarity*: In each image of the three real-world datasets, we replace all object textures by a set of new distinctive textures from the existing DTD database [15], as shown in Figure 8. In this way, the multiple objects look more distinctive in appearance, while the per-object texture gradients are roughly preserved. The ablated datasets are denoted: YCB-T / ScanNet-T / COCO-T.
- *Ablation of Inter-object Shape Variation*: In each image of the real-world datasets, we normalize the scales of multiple objects by shrinking or expanding the diagonal length of their bounding boxes, such that the new object sizes tend to be uniform. For each object, its shape and texture are linearly scaled up or down. Basically, this aims to remove the diversity of object sizes within single images. The ablated datasets are denoted as: YCB-U / ScanNet-U / COCO-U.
- *Ablation of both Inter-object Color Similarity and Shape Variation*: We simply combine the above two ablation strategies for each real-world image. Ablation details are in appendix. The ablated datasets are denoted as: YCB-T+U / ScanNet-T+U / COCO-T+U.

For illustration, the 3rd row of Figure 6 shows example images of three ablated datasets: YCB-T / YCB-U / YCB-T+U.

In the 3rd row of Figure 5 (Subfigs 3-a/3-b), we calculate new distributions of both Object Color Gradient and Inter-object Color Similarity on the ablated datasets YCB-T / ScanNet-T / COCO-T. We can see that the distributions of scene-level color similarity become more similar to synthetic datasets in Subfig 3-b, while the distributions of object color gradients are still similar as original real-world datasets YCB / ScanNet / COCO, *i.e.*, Subfig 3-a is similar to 1-a.

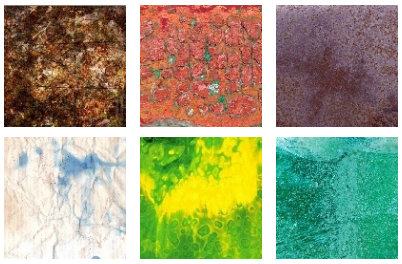


Figure 8: Six selected texture images from DTD [15].

Similarly, the 3rd row of Figure 5 shows that the distributions of Inter-object Shape Variation of ablated datasets now become similar to the synthetic datasets in Subfig 3-d, whereas the distributions of Object Shape Concavity are still the same as original real-world datasets YCB / ScanNet / COCO, *i.e.*, Subfig 3-c is similar to Subfig 1-c. Note that, for the ablation of both Inter-object Color Similarity and Shape Variation, the distributions will be the same as the 3rd row of Figure 5. Having these three groups of scene-level ablated real-world datasets, we then separately evaluate each baseline.

**Brief Analysis:** As shown in Figures 7 & 9, we can see that: 1) Once the textures of real-world images are replaced by more distinctive textures, *i.e.*, with lower similarity between object appearance, the segmentation performance has been surprisingly boosted remarkably for almost all methods. 2) Normalizing object sizes over images can also reasonably improve the segmentation performance. 3) Overall, these results clearly show that existing unsupervised models significantly favor objectness with distinctive appearance in single images. However, compared with Figure 4, the results on current scene-level ablated datasets are still inferior to synthetic datasets, meaning that the scene-level factors alone are not enough to explain the performance gap. More results are in appendix.

### 4.4 How do object- and scene-level factors jointly affect current models?

In this section, we aim to study how the object- and scene-level factors jointly affect the segmentation performance. In particular, we conduct the following three ablative experiments.

- *Ablation of Object Color Gradient, Object Shape Concavity and Inter-object Color Similarity*: In each image of the three real-world datasets, we replace the object color by averaging all pixels of the distinctive texture, and also replace the object shape with a simple convex hull. The ablated datasets are denoted: YCB-C+S+T / ScanNet-C+S+T / COCO-C+S+T.

- *Ablation of Object Color Gradient, Object Shape Concavity and Inter-object Shape Variation*: In each image of the three real-world datasets, we replace the object color by averaging its own texture, and modify the object shape as convex hull following by size normalization. The ablated datasets are denoted as: YCB-C+S+U / ScanNet-C+S+U / COCO-C+S+U.
- *Ablation of all four factors*: We aggressively combine all four ablation strategies and we get datasets: YCB-C+S+T+U / ScanNet-C+S+T+U / COCO-C+S+T+U.

For illustration, the 4th row of Figure 6 shows example images of three ablated datasets: YCB-C+S+T / YCB-C+S+U / YCB-C+S+T+U.

Since these ablations are conducted independently, the new distributions of four complexity factors on current jointly ablated datasets are exactly the same as the second and third rows of Figure 5.

**Brief Analysis:** As shown in Figure 7, we can see that: 1) Combining the two object-level factors and either of scene-level factors to ablate the real-world datasets, the segmentation performance can be improved as we expect, especially for IODINE and SlotAtt. 2) If the challenging real-world objects and images are ablated in both object- and scene-level, the segmentation performance of all unsupervised models achieves the same level with three synthetic datasets as shown in Figure 4. 3) Overall, these three groups of experiments demonstrate that the colossal failure of unsupervised models on real-world images involves both object- and scene-level dataset biases. More experiments and results are in appendix.

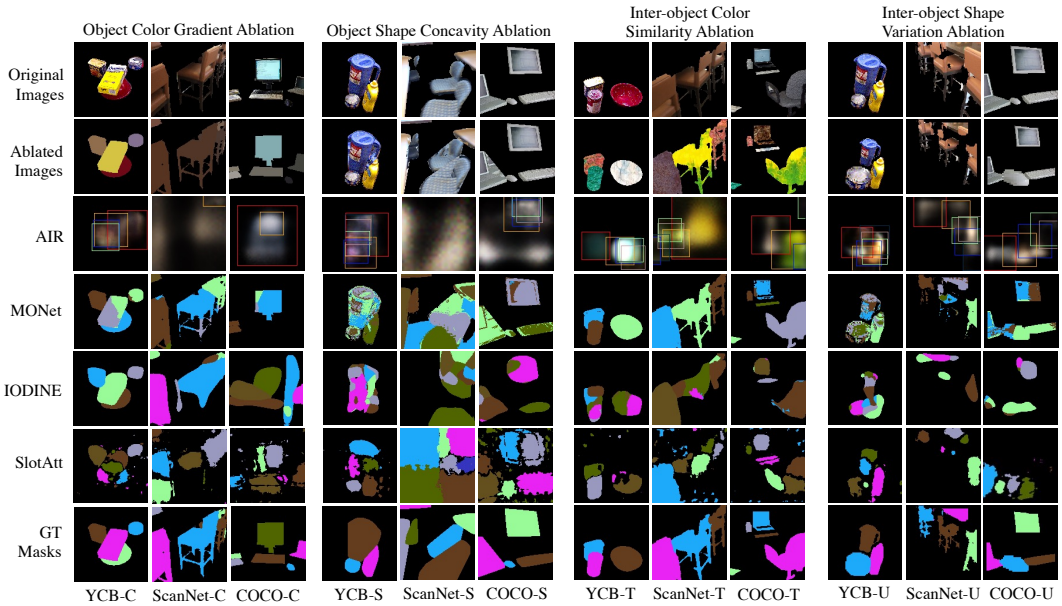



Figure 9: Qualitative results of four representative methods on multiple ablation datasets in Sec 4.2 and Sec 4.3.

### 4.5 Why do current unsupervised models fail on real-world datasets?

As demonstrated in Sections 4.2/4.3/4.4, once the complexity factors are removed from the challenging real-world datasets, existing unsupervised models can perform as excellent as on the synthetic datasets, as qualitatively illustrated in Figure 10. From this, we can safely conclude that the inductive biases designed in existing unsupervised models are far from able to match with and fully capture the true and complex objectness biases exhibited in real-world images. Nevertheless, from Figure 7, each baseline tends to favor different objectness biases. In particular,

- AIR [22]: As a factor based model, AIR has a strong spatial-locality bias. Despite its poor segmentation performance across all datasets, there is a notable improvement when inter-object shape variation is ablated from real-world datasets (U / T+U / C+S+U / C+S+T+U). More convincingly, even when all other three factors are ablated (C+S+T), it can be hardly improved. These observations show that object shape variation is a significant factor for AIR.

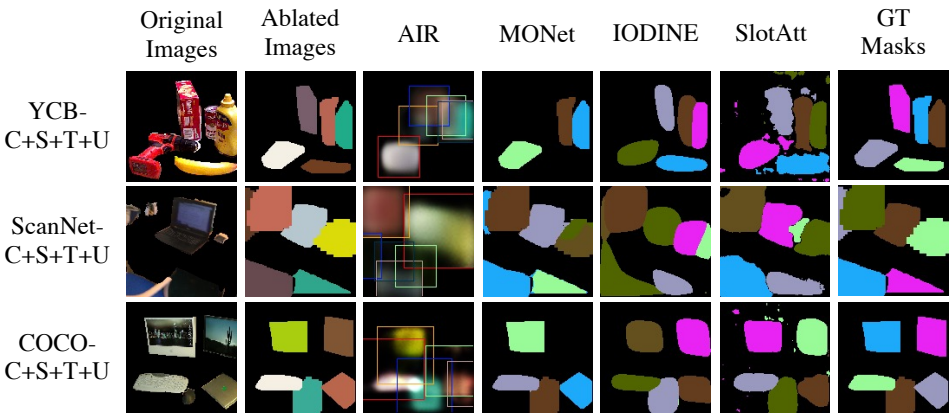



Figure 10: Qualitative results on fully ablated real-world datasets (YCB-C+S+T+U / ScanNet-C+S+T+U / COCO-C+S+T+U).

- MONet [8]: MONet is more sensitive to color-related factors than shape-related factors. The ablations of object color gradient and inter-object color similarity significantly improve its performance, while ablations of object shape concavity and inter-object shape variation make little differences. For the two color-related factors, the scene-level one is more important than the object-level factor. From this, we can see that MONet has a strong dependency on color. Similar colors tend to be grouped together while different colors are separated apart. Furthermore, the ablation on object color gradient alleviates over-segmentation whereas the ablation on inter-object color similarity alleviates under-segmentation. We conjecture that under-segmentation can be a more severe issue for MONet on real-world datasets, leading to a larger sensitivity on the scene-level color factor.
- IODINE [25]: IODINE also has a heavy dependency on both object- and scene-level color-related factors. However, different from MONet, the ablation on object color gradient brings better performance than inter-object color similarity. We speculate it is because the regularization on shape latent alleviates under-segmentation by biasing towards more regular shapes. In this way, over-segmentation is the key issue, making the object color gradient a dominant factor.
- SlotAtt [40]: The ablations on all four factors increase the performance of SlotAtt at different levels, among which object- and scene-level color-related factors are more significant. We conjecture that it is because the feature embeddings used by Slot Attention module are learnt from both pixel colors and coordinates, which contributes to its sensitivity to both shape and color factors.

## 5 Conclusions

We systematically show that existing unsupervised methods are practically impossible to segment generic objects from single real-world images, and investigate the underlying factors that incur the catastrophic failure. With the aid of our carefully designed four object- and scene-level complexity factors, we conduct extensive experiments on multiple groups of ablated real-world objects and images, and safely conclude that the distributions of both object- and scene-level biases in appearance and geometry of real-world datasets are particularly diverse and indiscriminative, such that current unsupervised models cannot segment real objects. Based on this finding, we suggest two main directions for future study: 1) To exploit more discriminative objectness biases such as object motions which expressively describe the ownership of visual pixels as recently explored in [55; 12; 6] for 2D images and in [52] for 3D point clouds. 2) To leverage pretrained features from single-object-dominant datasets which explicitly regard each image as an object as recently studied in [10; 30], although such settings are no longer purely unsupervised.

**Broader Impact:** One limitation of this work is the lack of study about foreground-background segmentation, but our work can still have a positive impact for researchers to design more practically useful models. A potential negative impact could be using our findings to attack existing models.

**Acknowledgements:** This work was partially supported by Shenzhen Science and Technology Innovation Commission (JCYJ20210324120603011).

## References


[1] R. Abdal, P. Zhu, N. Mitra, and P. Wonka. Labels4Free: Unsupervised Segmentation using StyleGAN. *ICCV*, 2021. 4

[2] T. Anciukevicius, C. H. Lampert, and P. Henderson. Object-Centric Image Generation with Factored Depths, Locations, and Appearances. *arXiv:2004.00642*, 2020. 5

[3] R. Arandjelovic and A. Zisserman. Object Discovery with a Copy-Pasting GAN. *arXiv:1905.11369*, 2019. 4

[4] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik. Contour Detection and Hierarchical Image Segmentation. *TPAMI*, 33(5):898–916, 2011. 5

[5] S. Azadi, D. Pathak, S. Ebrahimi, and T. Darrell. Compositional GAN: Learning Image-Conditional Binary Composition. *IJCV*, 2020. 4

[6] D. M. Bear, C. Fan, D. Mrowca, Y. Li, S. Alter, A. Nayebi, J. Schwartz, L. Fei-Fei, J. Wu, J. B. Tenenbaum, and D. L. Yamins. Learning Physical Graph Representations from Visual Scenes. *NeurIPS*, 2020. 10

[7] A. Bielski and P. Favaro. Emergence of object segmentation in perturbed generative models. *NeurIPS*, 2019. 4

[8] C. P. Burgess, L. Matthey, N. Watters, R. Kabra, I. Higgins, M. Botvinick, and A. Lerchner. MONet: Unsupervised Scene Decomposition and Representation. *arXiv:1901.11390*, 2019. 1, 4, 5, 10, 18, 20, 21, 22, 23, 24, 26, 27

[9] B. Calli, A. Singh, J. Bruce, A. Walsman, K. Konolige, S. Srinivasa, P. Abbeel, and A. M. Dollar. Yale-CMU-Berkeley dataset for robotic manipulation research. *IJRR*, 36(3):261 – 268, 2017. 4, 5, 19

[10] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging Properties in Self-Supervised Vision Transformers. *ICCV*, 2021. 2, 10

[11] R. Charakorn, Y. Thawornwattana, S. Itthipuripat, N. Pawlowski, and N. Dilokthanakul. An Explicit Local and Global Representation Disentanglement Framework with Applications in Deep Clustering and Unsupervised Object Detection. *arXiv:2001.08957*, 2020. 5

[12] H. Chen, R. Venkatesh, Y. Friedman, J. Wu, J. B. Tenenbaum, D. L. K. Yamins, and D. M. Bear. Unsupervised Segmentation in Real-World Images via Spelke Object Inference. *arXiv:2205.08515*, 2022. 10

[13] M. Chen, T. Artières, and L. Denoyer. Unsupervised object segmentation by redrawing. *NeurIPS*, 2019. 4

[14] B. Cheng, R. Girshick, P. Dollár, A. C. Berg, and A. Kirillov. Boundary IoU: Improving Object-Centric Image Segmentation Evaluation. *CVPR*, 2021. 15, 17

[15] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi. Describing Textures in the Wild. *CVPR*, 2014. 8, 23

[16] E. Crawford and J. Pineau. Spatially Invariant Unsupervised Object Detection with Convolutional Neural Networks. *AAAI*, 2019. 5

[17] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes. *CVPR*, 2017. 4, 5, 19

[18] F. Deng, Z. Zhi, D. Lee, and S. Ahn. Generative Scene Graph Networks. *ICLR*, 2021. 5

[19] S. Eddins. Binary image convex hull. *https://blogs.mathworks.com/steve/2011/10/04/binary-image-convex-hull-algorithm-notes/*, 2011. 3, 6, 15, 24

[20] M. Engelcke, O. P. Jones, and I. Posner. GENESIS-V2: Inferring Unordered Object Representations without Iterative Refinement. *NeurIPS*, 2021. 1, 5

[21] M. Engelcke, A. R. Kosiorek, O. P. Jones, and I. Posner. GENESIS: Generative Scene Inference and Sampling with Object-Centric Latent Representations. *ICLR*, 2020. 5, 18

[22] S. M. A. Eslami, N. Heess, T. Weber, Y. Tassa, K. Kavukcuoglu, and G. E. Hinton. Attend, Infer, Repeat: Fast Scene Understanding with Generative Models. *NIPS*, 2016. 1, 4, 5, 9, 18, 20, 22, 23, 24, 26, 27

[23] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge: A Retrospective. *IJCV*, 111:98–136, 2015. 5

[24] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative Adversarial Nets. *NIPS*, 2014. 4

[25] K. Greff, R. L. Kaufman, R. Kabra, N. Watters, C. Burgess, D. Zoran, L. Matthey, M. Botvinick, and A. Lerchner. Multi-object representation learning with iterative variational inference. *ICML*, 2019. 4, 5, 10, 18, 20, 21, 22, 23, 24, 26, 27

[26] K. Greff, A. Rasmus, M. Berglund, T. H. Hao, J. Schmidhuber, and H. Valpola. Tagger: Deep unsupervised perceptual grouping. *NIPS*, 2016. 5

[27] K. Greff, S. Van Steenkiste, and J. Schmidhuber. Neural Expectation Maximization. *NIPS*, 2017. 5

[28] K. Greff, S. van Steenkiste, and J. Schmidhuber. On the binding problem in artificial neural networks. *arXiv:2012.05208*, 2020. 1

[29] K. He, G. Gkioxari, P. Dollar, and R. Girshick. Mask R-CNN. *ICCV*, 2017. 4, 18, 21

[30] O. J. Hénaff, S. Koppula, E. Shelhamer, D. Zoran, A. Jaegle, A. Zisserman, J. Carreira, and R. Arandjelović. Object discovery and representation networks. *ECCV*, 2022. 2, 10

[31] J. Huang and K. Murphy. Efficient inference in occlusion-aware generative models of images. *ICLR Workshops*, 2016. 5

[32] J. Jiang and S. Ahn. Generative neurosymbolic machines. *NeurIPS*, 2020. 5

[33] J. Johnson, B. Hariharan, L. van der Maaten, L. Fei-Fei, C. L. Zitnick, and R. Girshick. CLEVR: A Diagnostic Dataset for Compositional Language and Elementary Visual Reasoning. *CVPR*, 2017. 1, 4, 5, 18, 19

[34] R. Kabra, C. Burgess, L. Matthey, R. L. Kaufman, K. Greff, M. Reynolds, and A. Lerchner. Multi-Object Datasets. *https://github.com/deepmind/multi-object-datasets/*, 2019. 4, 5, 19

[35] L. Karazija, I. Laina, and C. Rupprecht. ClevrTex: A Texture-Rich Benchmark for Unsupervised Multi-Object Segmentation. *NeurIPS*, 2021. 2

[36] D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. *ICLR*, 2014. 4

[37] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár. Panoptic Segmentation. *CVPR*, 2019. 5

[38] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common Objects in Context. *ECCV*, 2014. 1, 4, 5, 19

[39] Z. Lin, Y.-F. Wu, S. V. Peri, W. Sun, G. Singh, F. Deng, J. Jiang, and S. Ahn. SPACE: Unsupervised Object-Oriented Scene Representation via Spatial Attention and Decomposition. *ICLR*, 2020. 1, 5

[40] F. Locatello, D. Weissenborn, T. Unterthiner, A. Mahendran, G. Heigold, J. Uszkoreit, A. Dosovitskiy, and T. Kipf. Object-Centric Learning with Slot Attention. *NeurIPS*, 2020. 1, 2, 4, 5, 10, 18, 20, 22, 23, 24, 26, 27

[41] S. Löwe, P. Lippe, M. Rudolph, and M. Welling. Complex-Valued Autoencoders for Object Discovery. *arXiv:2204.02075*, 2022. 5

[42] L. Matthey, I. Higgins, D. Hassabis, and A. Lerchner. dSprites - Disentanglement testing Sprites dataset. *https://github.com/deepmind/dsprites-dataset/*, 2017. 1, 4, 5, 19

[43] S. Papa, O. Winther, and A. Dittadi. Inductive Biases for Object-Centric Representations in the Presence of Complex Textures. *arXiv:2204.08479*, 2022. 2

[44] D. D. Polsby and R. D. Popper. The third criterion: Compactness as a procedural safeguard against partisan gerrymandering. *Yale L. & Pol'y Rev.*, 9, 1991. 16

[45] W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971. 5

[46] D. J. Rezende and F. Viola. Taming vaes. *arXiv:1810.00597*, 2018. 18

[47] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 2015. 2

[48] J. E. Schwartzberg. Reapportionment, gerrymanders, and the notion of compactness. *Minn. L. Rev.*, 50:443, 1965. 16

[49] C. E. Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948. 16

[50] Shengxin Qian. R-MONet: Region-Based Unsupervised Scene Decomposition and Representation via Consistency of Object Representations. *OpenReview.Net*, 2021. 5

[51] I. Sobel and G. Feldman. A 3x3 Isotropic Gradient Operator for Image Processing. *Pattern Classification and Scene Analysis*, pages 271–272, 1973. 3, 15

[52] Z. Song and B. Yang. OGC: Unsupervised 3D Object Segmentation from Rigid Dynamics of Point Clouds. *NeurIPS*, 2022. 10

[53] E. S. Spelke, K. Breinlinger, J. Macomber, and K. Jacobson. Origins of knowledge. *Psychological Review*, 99(4):605–632, 1992. 1

[54] K. Stelzner, R. Peharz, and K. Kersting. Faster attend-infer-repeat with tractable probabilistic models. *ICML*, 2019. 5

[55] M. Tangemann, S. Schneider, J. von Kügelgen, F. Locatello, P. Gehler, T. Brox, M. Kümmerer, M. Bethge, and B. Schölkopf. Unsupervised Object Learning via Common Fate. *arXiv:2110.06562*, 2021. 10

[56] S. van Steenkiste, K. Kurach, J. Schmidhuber, and S. Gelly. Investigating object compositionality in Generative Adversarial Networks. *Neural Networks*, 2020. 4

[57] J. von Kügelgen, I. Ustyuzhaninov, P. Gehler, M. Bethge, and B. Schölkopf. Towards causal generative scene models via competition of experts. *ICLR Workshops*, 2020. 5

[58] A. Voynov, S. Morozov, and A. Babenko. Object Segmentation Without Labels with Large-Scale Generative Models. *ICML*, 2021. 4

[59] W. Wang, Q. Lai, H. Fu, J. Shen, H. Ling, and R. Yang. Salient Object Detection in the Deep Learning Era: An In-Depth Survey. *TPAMI*, 2021. 2

[60] M. A. Weis, K. Chitta, Y. Sharma, W. Brendel, M. Bethge, A. Geiger, and A. S. Ecker. Benchmarking Unsupervised Object Representations for Video Sequences. *JMLR*, 2021. 2

[61] M. Wertheimer. Untersuchungen zur Lehre yon der Gestalt. *Psychologische Forschung*, 1923. 1

[62] K. Xu, C. Li, J. Zhu, and B. Zhang. Multi-object generation with amortized structural regularization. *NeurIPS*, 2019. 5

[63] J. Yuan, T. Chen, B. Li, and X. Xue. Compositional Scene Representation Learning via Reconstruction: A Survey. *arXiv:2202.07135*, 2022. 2

[64] J. Yuan, B. Li, and X. Xue. Generative modeling of infinite occluded objects for compositional scene representation. *ICML*, 2019. 5

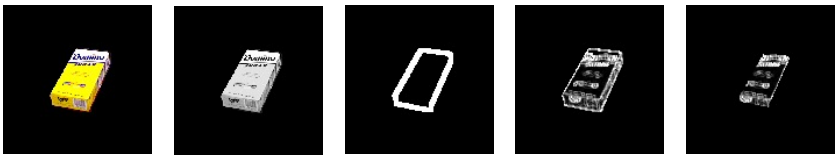
1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 5.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 5.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See **Contributions and finding** in Section 1.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See **Implementation Details** in Appendix.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] In the tables for qunatitative results, we report standard deviation of performances over three runs.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See **Implementation Details** in Appendix.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] See **Contributions and finding** in Section 1.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

# A Appendix

## A.1 Details of the Four Complexity Factors

We have introduced four complexity factors in both object- and scene-level appearance and geometry. Details of these factors are as follows.

- **Object Color Gradient:** As shown in Figure 11, given an RGB image of an object, we first convert it to grayscale by applying $Y = 0.299R + 0.587G + 0.114B$. Then Sobel filter [51] with kernel size $3 \times 3$ is applied horizontally and vertically to compute the image gradient for each pixel. Since this factor should only be related to the appearance inside an object regardless of the background, we remove the gradients at the object boundary. The object boundary is computed from its mask, following the practice of [14]. The factor score is calculated as the average gradient of all object pixels that are not on the boundary. This value ranges between 0 and 255, which is then divided by 255 so as to be normalized to the range of $[0, 1]$.



Object Image | Grayscale Image | Object Boundary | Gradient | Inner Gradient

Figure 11: The calculation of Object Color Gradient.

- **Object Shape Concavity:** As shown in Figure 12, given the binary mask of an object ($\boldsymbol{M}_{obj} \in \mathbb{R}^{H \times W}$), we first calculate its smallest surrounding convex polygon mask ($\boldsymbol{M}_{cvx} \in \mathbb{R}^{H \times W}$) using the existing algorithm [19]. Basically, this polygon mask is the smallest region that can cover any lines between two points on the original object mask. The object shape concavity factor is calculated as $1 - \sum \boldsymbol{M}_{obj} / \sum \boldsymbol{M}_{cvx}$. This factor naturally takes a value between 0 and 1. No further normalization is needed.

(a) Object Mask (b) Smallest Convex Polygon Mask

Figure 12: An illustration of the Object Shape Concavity.

- **Inter-object Color Similarity:** As shown in Figure 13, we first calculate the average RGB color of each object. Each color corresponds to a point in the RGB space. We then calculate the Euclidean distance between each pair of object colors. The average value of all pairwise distances is divided by $255 \times \sqrt{3}$, which is the largest distance between two colors in the RGB space, so as to be normalized to the range of $[0, 1]$. The final score for inter-object color similarity is calculated as $1 - normalized\ RGB\ distance$.

Figure 13: An illustration of the Inter-object Color Similarity.

- **Inter-object Shape Variation:** We first compute an axis-aligned bounding box for each object, and calculate its diagonal length. Then the differences of diagonal lengths between each pair of objects are calculated. The average value of such pairwise differences is the raw value of inter-object shape variation. This raw value is divided by $255 \times \sqrt{2}$, which is the largest possible difference of diagonal length, so as to be normalized to the range of $[0, 1]$.

### A.2 Other Candidates of Complexity Factors

In addition to the primary four complexity factors, we also explore other potential complexity factors to quantitatively measure the distributions of object- and scene-level biases in appearance and geometry. Basically, we aim to consider as many aspects as possible to investigate key factors underlying the distribution gaps between synthetic and real-world datasets. However, we empirically find that these candidate factors do not show significant discrepancy between synthetic and real-world datasets. Details are shown below.

#### A.2.1 Candidates of Object-level Complexity Factors

- **Object Color Count:** This factor is defined as the total number of unique colors within an object mask. Basically, this is to simply measure the diversity of object color.
- **Object Color Entropy:** Inspired by Shannon entropy [49], we calculate the entropy value at each pixel by applying a $3 \times 3$ filter on the grayscale image concerted from RGB. In particular, for each pixel, its color value becomes a discrete value in $[0, 255]$. We compute its entropy score: $H(x) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i)$, where $p(x_i)$ denotes the probability of a specific color value $x_i$ within the $3 \times 3$ neighbourhood. Basically, this factor aims to measure the color diversity within $3 \times 3$ image patches. The higher this factor, the more frequently the object color changes in small local areas.
- **Object Shape Non-rectangularity:** Given the binary mask of an object ($\boldsymbol{M}_{obj} \in \mathbb{R}^{H \times W}$), we first calculate its axis-aligned bounding box ($\boldsymbol{M}_{bbox} \in \mathbb{R}^{H \times W}$). Object shape non-rectangularity is calculated as $1 - \sum \boldsymbol{M}_{obj} / \sum \boldsymbol{M}_{bbox}$. Similar to object shape concavity, this factor is also designed to measure the complexity of object shapes. However, this factor is more likely to be affected by the object orientation since it takes axis-aligned bounding boxes as reference.
- **Object Shape Incompactness:** There are two similar methods to quantify the compactness of object shapes. The first one is Polsby–Popper test [44]: $PP(\boldsymbol{M}_{obj}) = 4\pi A(\boldsymbol{M}_{obj}) / P(\boldsymbol{M}_{obj})^2$. The other is Schwartzberg [48] compactness score: $S(\boldsymbol{M}_{obj}) = (2\pi\sqrt{A(\boldsymbol{M}_{obj})/\pi}) / P(\boldsymbol{M}_{obj})$. In both formula, $P(\boldsymbol{M}_{obj})$ is the object perimeter and $A(\boldsymbol{M}_{obj})$ is the object area. For simplicity, we choose $PP(\boldsymbol{M}_{obj})$ to calculate the object shape incompactness score: $1 - PP(\boldsymbol{M}_{obj})$.
- **Object Shape Discontinuity:** Given an object mask ($\boldsymbol{M}_{obj} \in \mathbb{R}^{H \times W}$), we first find the largest connected component ($\boldsymbol{M}_{lcc} \in \mathbb{R}^{H \times W}$) in its binary mask. The discontinuity of shape is calculated as: $1 - \sum \boldsymbol{M}_{lcc} / \sum \boldsymbol{M}_{obj}$. This factor is to evaluate how continuous an object shape is.
- **Object Shape Decentralization:** Given an object mask, we first calculate its centroid $(\bar{x}, \bar{y})$ by averaging all pixel coordinates in the object. Then, the second moment of this object is calculated as: $\sum_x \sum_y (x - \bar{x})^2 (y - \bar{y})^2$, where $(x, y)$ is the coordinates of pixels within the object. The higher this factor, the object shape is less likely to be centralized.

As shown in Figure 14, we compare the distributions of the object-level factor candidates on both synthetic and real-world datasets. It can be seen that the majority of these factors do not show significant gaps between the simple synthetic and the challenging real-world datasets. Therefore, we do not conduct relevant ablation experiments.

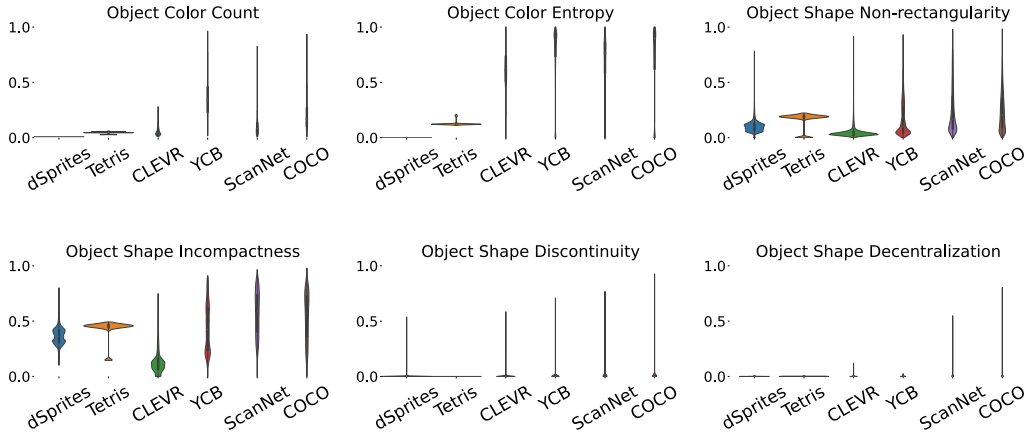



Figure 14: Distributions of Object-level Complexity Factors candidates.

### A.2.2 Candidates of Scene-level Complexity Factors

- **Inter-object Color Similarity with Chamfer Distance:** In the calculation of this factor, we first convert each pixel into a point in RGB space. In this way, each object can be represented by a point set in the RGB space. This factor is calculated between each pair of objects by measuring the Chamfer distance of two point sets in the RGB space. Since Chamfer distance is an asymmetric measurement, we calculate and average out the bidirectional Chamfer distances. Compared with Euclidean distance, this measurement favors the most similar colors between two objects.
- **Inter-object Color Similarity with Hausdorff Distance:** This factor is similar to the previous one. The only difference is that we replace Chamfer distance with Hausdorff distance. Hausdorff distance is also a directed and asymmetric measurement, so the final score is the average of distance values in both direction.
- **Inter-object Shape Similarity over Boundaries:** For each object mask, we first find its boundary using the method in [14], and then crop it with its axis-aligned bounding box. Each bounding box is scaled and fit into a unit box with its original aspect ratio. Lastly, we calculate the IoU between the boundaries of two objects to measure their shape similarity.
- **Inter-object Shape Entropy between Boundaries:** We first combine all object masks into a single image by assigning different indices to different objects. Then we compute the entropy of each pixel with a $3 \times 3$ filter. The final factor score is calculated by averaging all non-zero entropy values. Note that, the interior part of objects and background will not be considered because their entropy values will always be zeros. Basically, this factor is designed to evaluate how crowded an image is. The higher this factor, the more objects are spatially adjacent.
- **Inter-object Proximity between Centroids:** We first calculate the centroid $(\bar{x}, \bar{y})$ of each object by averaging all pixel coordinates in the object mask. Euclidean distances between object centroids are then computed pair-wisely before they are averaged to be the final factor score. This factor is designed to measure the spatial proximity of multiple objects in a single image.
- **Inter-object Proximity with Chamfer Distance:** In order to measure the spatial proximity between objects, we also calculate the spatial Chamfer distance between objects. Specifically, each object is represented by a set of $x - y$ coordinates, and then the average of pair-wise bidirectional Chamfer distances is calculated as the proximity score for each image.

As shown in Figure 15, we compare the distributions of the scene-level factor candidates on both synthetic and real-world datasets. We can see that both the inter-object color similarity with Chamfer and Hausdorff distances share similar distributions gaps with our primary inter-object color similarity factor defined in Section 2. The remaining four candidate factors relating to inter-object shape complexity do not show significant distribution gaps between the synthetic and real-world datasets. In this regard, we choose not to conduct ablation experiments on these six candidate factors.

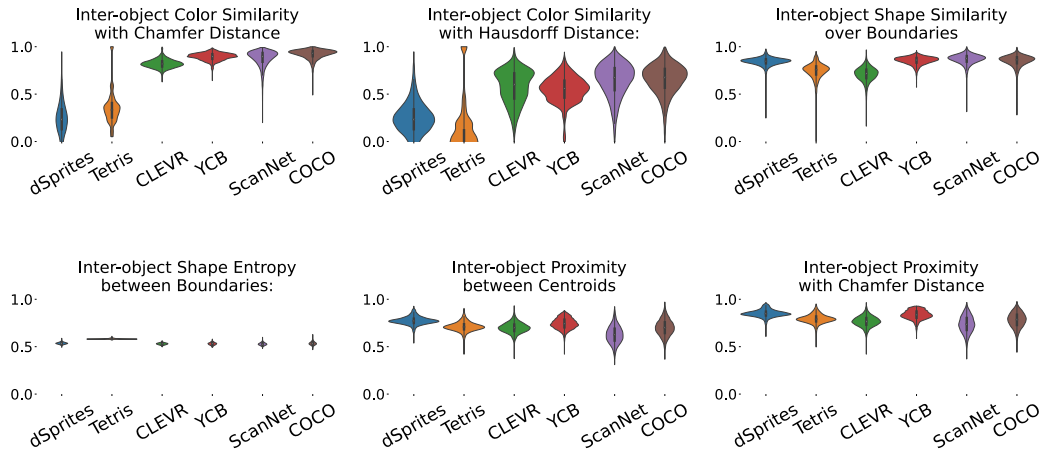



Figure 15: Distributions of Scene-level Complexity Factors candidates.

### A.3 Implementation Details

In this section, we present the implementation details of four representative models.

**AIR [22]**

- **Source Code:** We refer to https://pyro.ai/examples/air.html and https://github.com/addtt/attend-infer-repeat-pytorch for the implementation.
- **Important Adaptations:** We use an additional parameter to weight the KL divergence loss and reconstruction loss. For each experiment, we choose the weight for KL divergence from 1, 10, 25 and 50. The highest AP score is kept.
- **Training Details:** All experiments of AIR [22] are conducted with a batch size of 64. The learning rate is set to $1\mathrm{e}{-4}$ for training inference networks and decoders, $1\mathrm{e}{-3}$ for baselines which is the same as the original paper. Since the number of objects in our datasets ranges between 2 and 6, we set the maximum number of steps at inference to be 6 for all experiments. All models are trained on a single GPU for 1000 epochs. We perform evaluation every 50 epochs and select the one with the highest AP score.

**MONet [8]**

- **Source Code:** We refer to [21]'s re-implementation at: https://github.com/applied-ai-lab/genesis.
- **Important Adaptations:** We train MONet [8] with the GECO objective [46] following the protocol mentioned in [21].
- **Training Details:** All experiments on MONet [8] are conducted with a batch size of 32 and learning rate of $1\mathrm{e}{-4}$. Since the maximum number of components is 7, including 1 background and 6 objects, we set the number of steps to be 7 for all experiments. All models are trained on a single GPU for 200 epochs with the training loss converged. We perform evaluation every 10 epochs and select the one with the highest AP score.

**IODINE [25]**

- **Source Code:** We use the official implementation at: https://github.com/deepmind/deepmind-research/tree/master/iodine .
- **Important Adaptations:** The architecture is set the same as what is used for CLEVR dataset [33] in the original paper [25].
- **Training Details:** Since we use a single GPU for the training of all models, the batch size is adjusted to be 4 and learning rate $0.0001 \times \sqrt{1/8}$. The number of slots $K$ is set as 7 and the inference iteration $T$ as 5. We train each model for $500K$ iterations until the loss is fully converged.

**SlotAtt [40]**

- **Source Code:** We use the official implementation at: https://github.com/google-research/google-research/tree/master/slot_attention.
- **Important Adaptations:** The architecture is set the same as what is used for CLEVR dataset [33] in the original paper [40].
- **Training Details:** All experiments of SlotAtt [40] are conducted with a batch size of 32 and learning rate selected from [$4\mathrm{e}{-4}$, $4\mathrm{e}{-5}$]. The number of slots $K$ is set as 7 and the number of iterations $T$ is set as 3. All models are trained on a single GPU for $500K$ iterations until the loss is fully converged.

**Mask R-CNN [29]**

- **Source Code:** We use the implementation at: https://github.com/matterport/Mask_RCNN.
- **Important Adaptations:** We use the same settings as training for COCO in above repository.
- **Training Details:** Training for all datasets starts from the pre-trained COCO weights (mask_rcnn_coco.h5) from https://github.com/matterport/Mask_RCNN/releases. All models are trained on a single GPU for 30 epochs until the loss is fully converged.

### A.4 Details of six Benchmark Datasets

In this section, we present the details of three synthetic and three real-world datasets.

**dSprites [42]** To generate a specific image for this dataset, we first sample a random integer $K$ from a uniform distribution with interval $[2, 6]$ as the number of objects in that image. Then, $K$ object shapes are selected from the binary dsprites dataset [42] also in a uniformly random manner. Each object is assigned with a random RGB color by sampling three random integers from a uniform distribution with interval $[0, 255]$. In total, we generate 10000 images for training, 2000 for testing.

**Tetris [34]** For each image in this dataset, we first sample a random integer $K$ from a uniform distribution with interval $[2, 6]$ as the number of objects in this image. To render one tetris-like object onto the canvas, we randomly pick up a tetris object from a randomly selected image from [34]. Each object is resized to be $88 \times 88$ and then placed onto the canvas. The position of each object is also sampled from a uniform distribution with 2 criteria: 1) all objects shall be on the canvas with complete shapes; 2) all objects shall not overlap with each other.

**CLEVR [33]** We first generate CLEVR images following `https://github.com/facebookresearch/clevr-dataset-gen`, where the number of objects per image is restricted between 3 to 6. Given generated images with a resolution $640 \times 480$, we perform center-cropping and then resize them to be $128 \times 128$. Then, we remove tiny objects which have less than 35 pixels from each image. Subsequently, the images with less than 2 objects are removed. Being consistent with previous 2 synthetic datasets, all images have a black background.

**YCB [9]** We sample single frames from the YCB video dataset [9] every 20 images. Given the sampled frames with a resolution of $640 \times 480$, we first center crop and then resize them to be $128 \times 128$. Then, the images consisting of less than 2 or more than 6 objects are removed. Similarly, all background pixels are replaced by a black color.

**ScanNet [17]** We sample single frames from the ScanNet dataset [17] every 20 images. Given the selected frames with a resolution of $1296 \times 968$, we first center crop the images with a size of $800 \times 800$ and then resize them to be $128 \times 128$. For each resized image, we remove objects that contain more than $128 \times 128 \times 0.2$ pixels or less than $128 \times 128 \times 0.007$ pixels. The images with less than 2 or more than 6 are also dropped. All background pixels are replaced by the black color.

**COCO [38]** Given images in COCO-2017 [38] with various resolutions, we first center crop and then resize images to be $128 \times 128$. For each resized image, we use the same criteria applied to ScanNet to remove too large and too small objects. The images with $2 \sim 6$ objects are kept. All background pixels are replaced by the black color. Since the number of images that could meet all requirements is less than 2,000 (only 1,597) in the official validation split, we additionally select 403 different images from its official training split for testing. There is no overlap between our training and testing splits.

Figure 16 shows three example images for each dataset. The distribution of images with different number of objects is also presented.

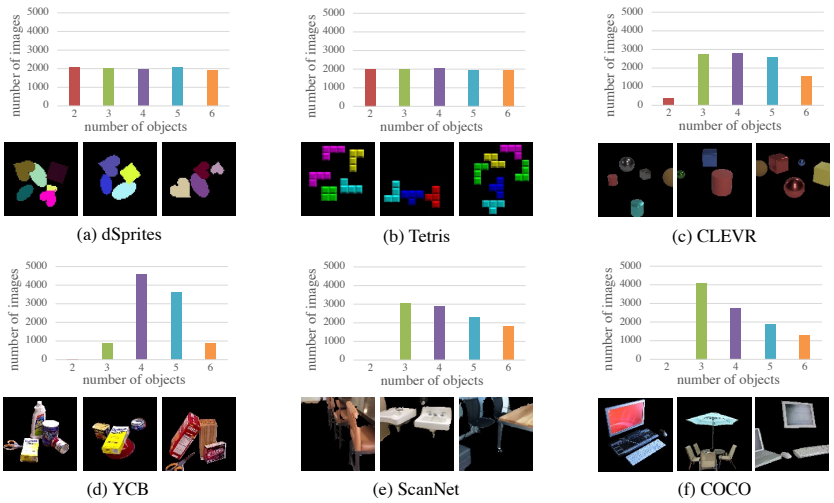



Figure 16: Example images and the statistics of the six benchmark datasets.

### A.5 Details and Results of Main Experiments

In this section, we present the result of four unsupervised methods and one supervised method on six datasets. Both quantitative evaluation and qualitative results are provided.

#### A.5.1 Four Unsupervised Methods

Figure 17 shows the qualitative results of 4 representative methods on 6 datasets. The results of AIR [22] are presented with predicted bounding boxes while others are represented with predicted segmentation masks. Different colors in a segmentation mask indicate different objects. For the same object, the assigned color may not be the same. We can see that all methods give reasonable results on the three synthetic datasets in spite of some performance gaps between them due to the different capability of these methods. However, they all fail on the three real-world datasets. Specifically, AIR [22] cannot reconstruct reasonable images. MONet [8] always performs segmentation based on color. IODINE [25] cannot recover meaningful object shapes and locations. SlotAtt [40] can roughly locate objects but cannot identify the real object shapes.

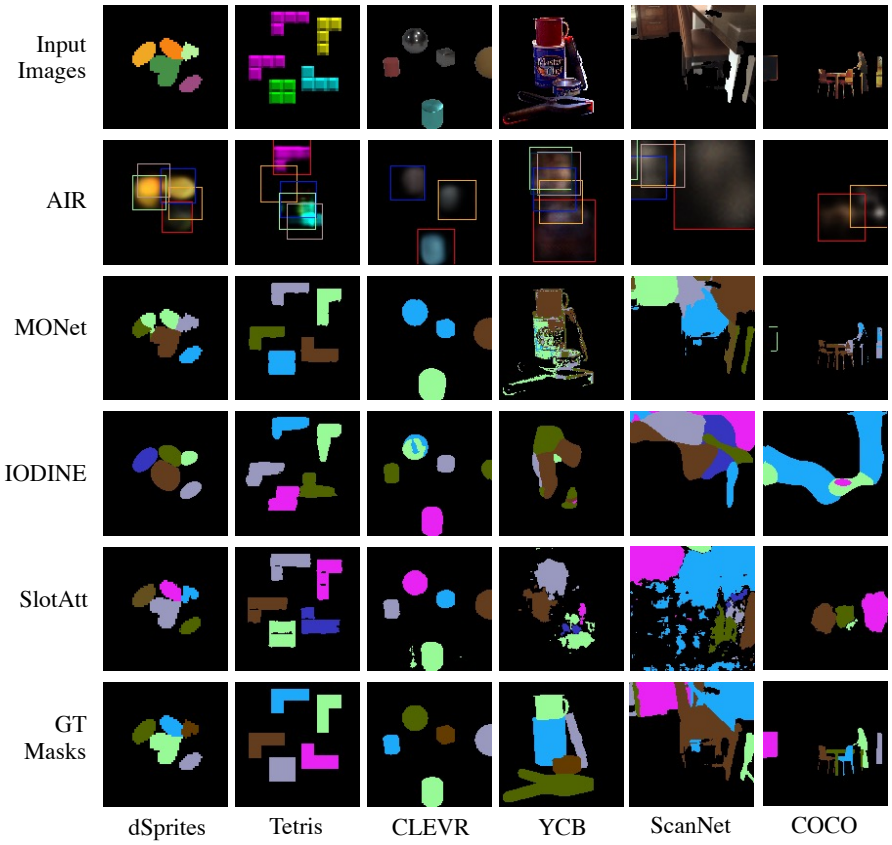


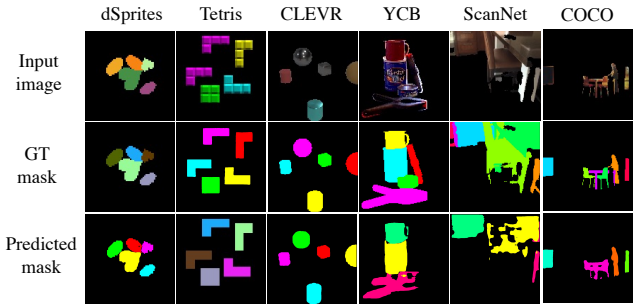

Figure 17: Qualitative results of object segmentation from the four methods on six datasets.

Table 3: Quantitative results of object segmentation from the four methods on six datasets. Standard deviations of performance are calculated over 3 runs (marked with blue).

| | dSprites | Tetris | CLEVR |
|---|---|---|---|
| | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec |
| AIR [22] | 45.4 1.8 / 38.2 3.0 / 57.6 7.4 / 58.1 7.5 | 25.2 13.9 / 23.4 12.4 / 36.8 20.9 / 39.9 12.9 | 46.4 14.0 / 44.3 12.4 / 67.4 9.9 / 52.5 15.9 |
| MONet [8] | 69.7 4.1 / 61.6 6.0 / 70.4 8.1 / 73.9 1.9 | 85.9 13.0 / 75.8 13.6 / 85.1 16.4/ 89.7 8.2 | 39.0 8.5 / 37.3 6.3 / 65.6 11.8 / 42.8 10.8 |
| IODINE [25] | 92.9 4.3/ 71.3 6.1/ 82.6 2.3/ 96.0 5.2 | 52.2 2.3 / 37.9 4.6 / 48.0 2.3 / 61.8 1.7 | 82.8 2.8 / 73.0 5.7 / 77.5 3.1 / 87.4 2.0 |
| SlotAtt [40] | 92.8 1.4 / 82.8 1.6 / 88.8 3.4 / 92.9 1.6 | 94.3 1.2 / 79.9 6.4 / 90.5 3.3 / 94.4 1.3 | 91.7 6.4 / 82.9 10.9 / 90.8 9.7 / 92.7 5.3 |
| | YCB | ScanNet | COCO |
| | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec |
| AIR [22] | 0.0 0.1 / 0.6 0.3 / 1.1 0.4 / 0.8 0.2 | 2.7 1.4 / 6.3 1.7 / 15.6 2.8 / 7.3 1.6 | 2.7 0.1 / 6.7 0.5 / 14.3 2.6 / 8.6 0.8 |
| MONet [8] | 3.1 1.6 / 7.0 2.6 / 9.8 3.6 / 1.2 0.8 | 24.8 1.6 / 24.6 1.6 / 31.0 1.6 / 40.7 1.8 | 11.8 2.0 / 12.5 1.1/ 16.1 0.9/ 21.9 1.7 |
| IODINE [25] | 1.8 0.2 / 3.9 1.3 / 6.2 2.0 / 7.3 1.9 | 10.1 2.9 / 13.7 2.7 / 18.6 4.2 / 24.4 3.8 | 4.0 1.2 / 6.3 1.2 / 9.9 1.8 / 10.8 2.0 |
| SlotAtt [40] | 9.2 0.4 / 13.5 0.9 / 20.0 1.3 / 26.2 6.8 | 5.7 0.3 / 9.0 1.5 / 12.4 2.5 / 18.3 2.7 | 0.8 0.3 / 3.5 1.2 / 5.3 1.7 / 7.3 2.2 |

### A.5.2 One Supervised Method

Apart from above four unsupervised approaches, we also include Mask R-CNN [29] as a supervised object segmentation baseline. As shown in Figure 18, Mask R-CNN exhibits quite successful segmentation on three synthetic datasets. Real-world datasets are more challenging for Mask R-CNN, but the results are much better than unsupervised baselines.




| | MaskRCNN |
|---|---|
| | AP / PQ / Pre / Rec |
| dSprites | 98.4 / 90.2 / 99.6 / 98.4 |
| Tetris | 99.8 / 90.3 / 99.8 / 99.8 |
| CLEVR | 98.2 / 90.0 / 97.8 / 99.5 |
| YCB | 62.9 / 58.4 / 83.3 / 66.9 |
| ScanNet | 41.4 / 43.3 / 65.2 / 50.5 |
| COCO | 46.0 / 47.9 / 71.7 / 53.2 |

Figure 18: Qualitative and quantitative results of Mask-RCNN on six datasets.

## A.6 Details and Results of Ablation Experiments

In this section, we shows details of ablated datasets including example images. Both qualitative and quantitative evaluation are also presented. Table 4 is a look-up table for individual ablated factor.

Table 4: A look-up table for ablations.

| | Which level | | Which aspect | | Target factor | | | |
|---|---|---|---|---|---|---|---|---|
| Ablation | Object | Scene | Color | Shape | Object Color Gradient | Object Shape Concavity | Inter-object Color Similarity | Inter-object Shape Variation |
| C | ✓ | | ✓ | | ✓ | | | |
| S | ✓ | | | ✓ | | ✓ | | |
| T | | ✓ | ✓ | | | | ✓ | |
| U | | ✓ | | ✓ | | | | ✓ |

### A.6.1 Ablations on Object-level Factors

**Ablated Datasets**

- *Ablation of Object Color Gradient*: In the three ablated datasets: YCB-C / ScanNet-C / COCO-C, each object is represented by a single average color.
- *Ablation of Object Shape Concavity*: In the three ablated datasets: YCB-S / ScanNet-S / COCO-S, each object is represented by a convex shape.
- *Ablation of both Object Color Gradient and Shape Concavity*: In the three ablated datasets: YCB-C+S / ScanNet-C+S / COCO-C+S, each object is represented by a convex shape with a single average color.

Figure 19 shows example images from the three real-world datasets with different types of object-level factor ablations.

**Qualitative and Quantitative Results** As shown in Figure 20 and Table 5, all four methods have a significant improvement in segmentation performance on the ablated datasets with object color gradients being removed. By comparison, the ablation of object shape concavity is less effective. This shows that both object-level factors are relevant to the success of object segmentation, although object color gradient is more important for the four methods. Specifically, MONet [8] and IODINE [25] are more sensitive to color gradient compared with the other two methods.

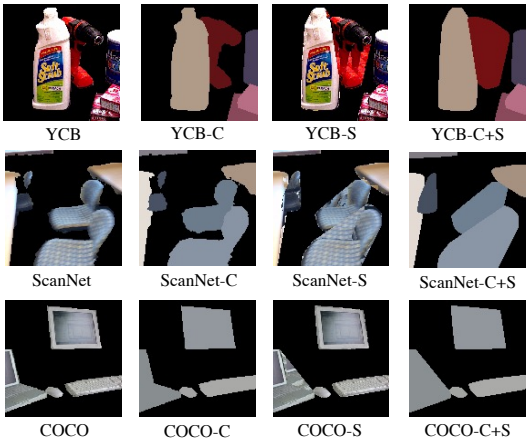



Figure 19: Example images of real-world datasets ablated with object-level factors.

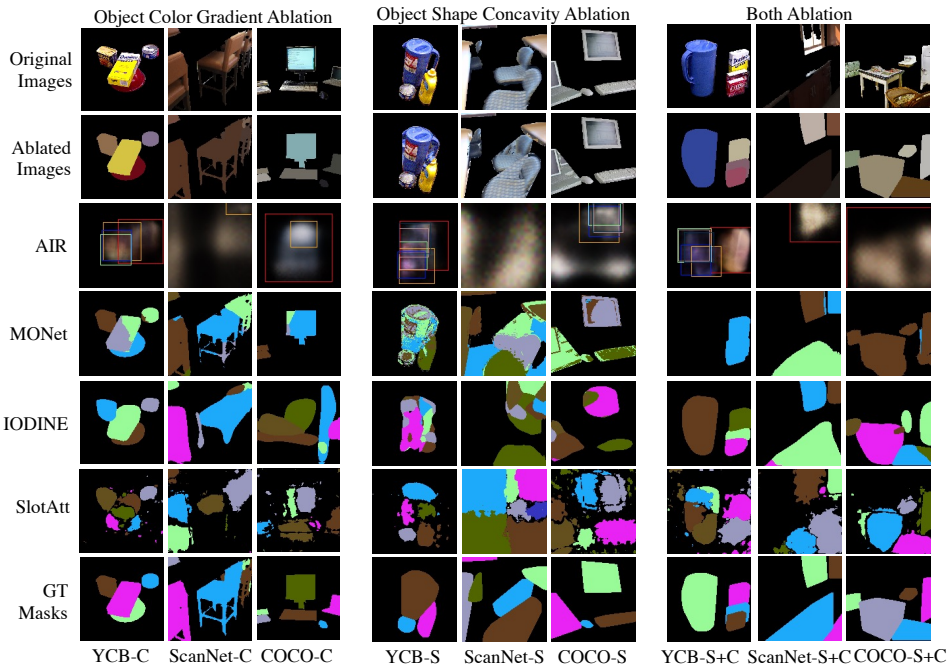



Figure 20: Qualitative results on the datasets ablated with object-level factors.

Table 5: Quantitative results on the datasets ablated with object-level factors. Standard deviations of performance are calculated over 3 runs (marked with blue).

| | YCB-C | ScanNet-C | COCO-C |
|---|---|---|---|
| | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec |
| AIR [22] | 4.4 0.2 / 1.0 7.3 / 20.7 7.1 / 12.5 1.2 | 2.7 1.0 / 7.6 2.1 / 29.1 20.2 / 7.4 1.5 | 2.9 1.8 / 7.9 2.5 / 32.0 29.3 / 7.7 4.5 |
| MONet [8] | 55.1 6.0 / 49.5 5.2 / 64.8 1.7 / 58.8 7.8 | 31.1 15.9 / 33.1 9.6 / 41.6 10.3 / 45.6 11.1 | 34.5 7.2 / 34.6 3.3 / 45.8 2.0 / 42.6 11.0 |
| IODINE [25] | 76.8 0.3 / 64.4 0.1 / 76.5 0.1 / 79.5 0.0 | 64.4 6.4 / 54.3 5.9 / 66.3 4.6 / 71.0 5.9 | 42.8 14.7 / 34.0 11.8 / 42.5 14.4 / 55.8 11.9 |
| SlotAtt [40] | 39.2 1.9 / 30.2 0.0 / 40.2 0.8 / 50.4 0.7 | 5.7 8.2 / 9.0 6.1 / 12.4 7.6 / 18.3 8.9 | 7.2 1.9 / 9.8 1.7/ 13.9 2.2 / 18.7 3.4 |
| | YCB-S | ScanNet-S | COCO-S |
| | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec |
| AIR [22] | 3.6 3.0 / 6.8 3.2 / 12.5 6.7 / 9.7 2.7 | 3.3 2.5 / 8.5 1.1 / 36.6 23.0 / 8.1 8.5 | 3.3 0.0 / 7.1 1.1 / 16.8 5.6 / 8.3 1.0 |
| MONet [8] | 2.5 1.1 / 6.4 1.8 / 8.9 2.6 / 11.3 3.2 | 18.3 10.4 / 22.2 3.9 / 28.4 4.4 / 37.5 5.3 | 8.7 3.2 / 11.2 1.7 / 14.4 3.0 / 21.0 0.7 |
| IODINE [25] | 2.5 2.5 / 5.0 5.0 / 7.9 7.8 / 1.1 1.0 | 8.3 1.1 / 13.7 0.4 / 18.4 0.4 / 24.8 0.7 | 4.2 0.4 / 8.3 0.4 / 12.1 0.3 / 14.8 0.6 |
| SlotAtt [40] | 19.1 0.7 / 21.0 1.7 / 30.4 2.8 / 37.7 0.8 | 1.9 8.5 / 6.6 7.5 / 8.7 9.8 / 13.7 11.9 | 2.6 1.0 / 5.9 0.9 / 8.3 1.3 / 11.9 10.9 |
| | YCB-C+S | ScanNet-C+S | COCO-C+S |
| | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec |
| AIR [22] | 2.0 0.3 / 6.4 0.2 / 15.1 5.2 / 7.3 4.1 | 3.5 1.0 / 9.1 0.5 / 38.3 25.8 / 8.5 5.9 | 4.8 0.1 / 10.8 0.1 / 41.5 10.2 / 10.1 1.7 |
| MONet [8] | 9.1 13.6 / 12.7 10.5 / 34.1 4.6 / 12.6 24.0 | 48.1 1.0 / 47.9 6.8 / 70.4 23.3 / 51.5 6.5 | 10.5 24.3 / 16.5 18.1 / 56.5 6.9 / 14.9 28.2 |
| IODINE [25] | 69.9 30.9 / 55.7 29.6 / 60.3 28.3 / 71.5 26.4 | 73.8 11.4 / 63.2 10.1 / 73.5 6.1 / 76.8 10.8 | 73.5 2.5 / 61.2 2.4 / 70.0 2.9 / 77.6 1.2 |
| SlotAtt [40] | 39.1 22.3 / 30.2 13.3 / 42.9 12.2 / 54.5 12.7 | 22.5 7.6 / 21.5 6.2 / 28.8 6.0 / 39.6 5.4 | 20.3 3.0 / 19.9 0.4 / 26.1 0.4 / 34.8 0.7 |

### A.6.2 Ablations on Scene-level Factors

**Ablated Datasets**

- *Ablation of Inter-object Color Similarity*: In the three ablated datasets: YCB-T / ScanNet-T / COCO-T, each object's original texture is replaced by one of the selected 6 distinctive textures from the DTD database [15], as shown in Figure 21. In particular, the 6 textures are chosen from the 'blotchy' category, and we deliberately select those with distinctive colors. The texture images are center-cropped and resized to a resolution of $128 \times 128$. Each object appearance is replaced by one of the 6 texture images, and each object in a single image is assigned with a different texture.
- *Ablation of Inter-object Shape Variation*: In the three ablated datasets: YCB-U / ScanNet-U / COCO-U, the size of each object is uniformly scaled. In particular, before generating the ablated datasets, we first calculate the average scale of objects in each dataset by calculating the mean of object bounding box diagonals (YCB: 60 / ScanNet: 70 / COCO: 57, in pixels). Each object in a specific dataset is scaled up or down to approach the average scale in this dataset. The shape and appearance are also linearly scaled, so that the shape variation is the only factor changed in the new dataset.
- *Ablation of both Inter-object Color Similarity and Shape Variation*: In the three ablated datasets: YCB-T+U / ScanNet-T+U / COCO-T+U, each object's texture is replaced and shape is uniformly scaled.

Figure 22 shows example images from the three real-world datasets with different types of scene-level factor ablations.

**Qualitative and Quantitative Results** As shown in Figure 23 and Table 6, all methods have a significant improvement in object segmentation on the ablated datasets with reduced inter-object color similarity. The effect of uniform scale ablation is limited standalone. However, when both ablations are combined, there is a larger performance gain compared with purely texture replaced ablation. Specifically, AIR [22] is sensitive to both scene-level factors, since only the combination of two factors can result in a significant improvement. MONet [8] and IODINE [25], however, are almost only sensitive to the inter-object color similarity. Both factors have a noticeable effect on the segmentation performance of SlotAtt [40].

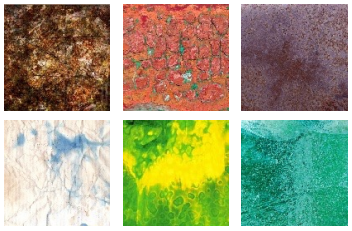


Figure 21: Six selected texture images from DTD [15].

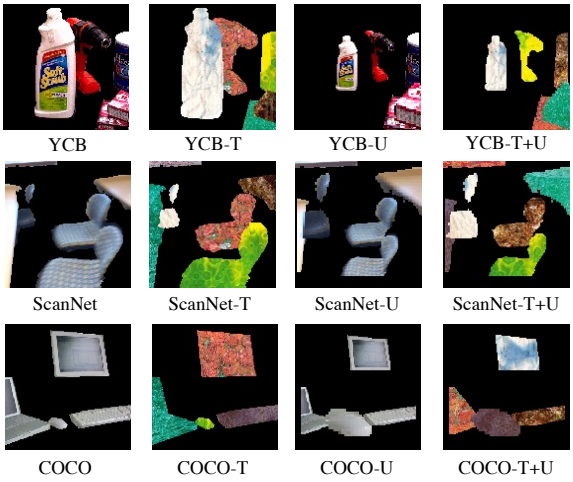



Figure 22: Example images of datasets ablated with scene-level factors.

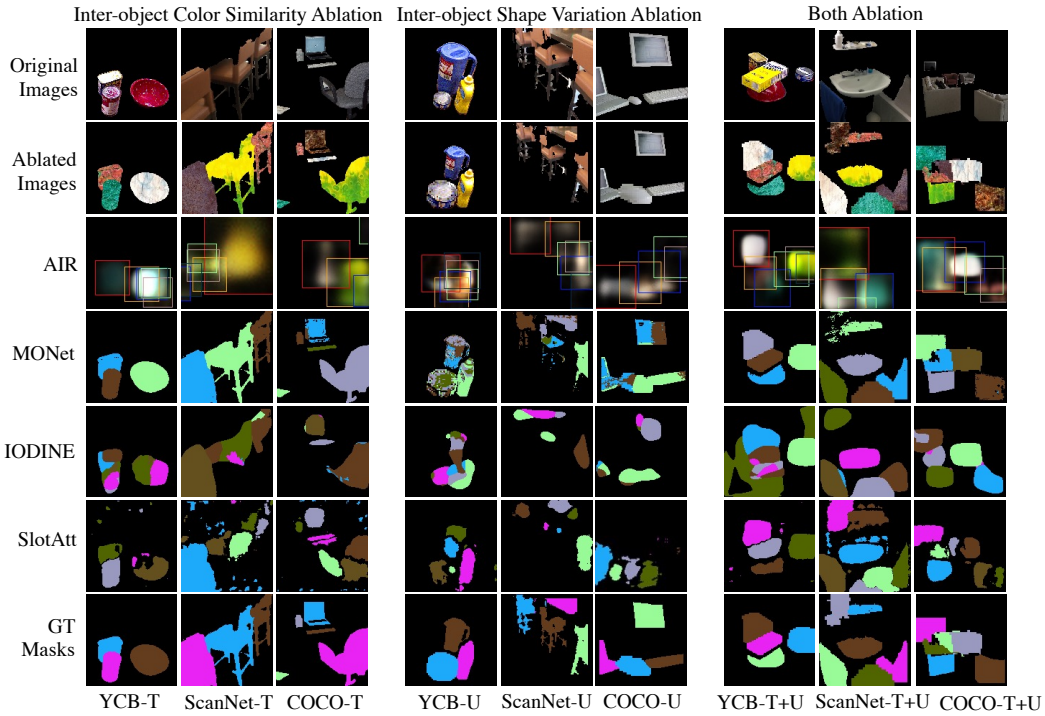



Figure 23: Qualitative results on the datasets ablated with scene-level factors.

Table 6: Quantitative results on the datasets ablated with scene-level factors. Standard deviations of performance are calculated over 3 runs (marked with blue).

| | YCB-T | ScanNet-T | COCO-T |
|---|---|---|---|
| | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec |
| AIR [22] | 5.9 3.7 / 9.6 3.3 / 19.9 10.3 / 12.7 1.3 | 2.9 1.2 / 6.3 0.1 / 13.4 3.7 / 8.6 3.2 | 7.4 3.1 / 13.0 4.0 / 29.1 9.8 / 16.4 4.3 |
| MONet [8] | 86.5 12.4 / 78.3 13.3 / 81.1 11.6 / 86.8 12.2 | 82.9 12.3 / 77.8 7.6 / 86.3 1.6 / 83.7 11.9 | 80.0 9.4 / 68.8 13.5 / 74.5 15.0 / 82.0 5.6 |
| IODINE [25] | 32.4 9.0 / 27.3 6.8 / 35.3 8.3 / 43.6 10.6 | 33.2 11.9 / 27.3 6.3 / 34.5 7.0 / 44.7 9.3 | 40.8 12.7 / 33.7 10.2 / 41.8 **10.4** / 55.9 15.8 |
| SlotAtt [40] | 64.6 5.3 / 48.5 3.2 / 58.3 3.5 / 74.94.6 | 20.5 22.3 / 18.2 20.1 / 22.9 23.1 / 33.2 31.4 | 31.8 17.8 / 28.0 11.2 / 35.2 12.4 / 50.1 19.8 |
| | YCB-U | ScanNet-U | COCO-U |
| | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec |
| AIR [22] | 9.8 3.5 / 12.2 2.6 / 18.6 4.1 / 19.9 2.8 | 7.1 1.4 / 10.4 0.2 / 19.9 2.1 / 14.0 1.0 | 12.9 4.8 / 16.3 4.6 / 28.6 1.8 / 24.3 9.3 |
| MONet [8] | 3.7 0.5 / 8.3 1.2 / 12.2 1.4 / 13.2 1.4 | 25.4 8.3 / 23.6 7.4 / 29.2 8.9 / 37.2 12.5 | 14.5 0.6 / 16.5 0.8 / 27.9 1.4 / 20.9 0.8 |
| IODINE [25] | 2.5 1.4 / 4.9 2.0 / 7.6 2.9 / 9.3 3.6 | 16.9 0.8 / 17.2 0.0 / 23.8 0.8 / 30.2 0.1 | 5.5 1.0 / 7.7 0.0 / 11.7 0.6 / 13.5 0.3 |
| SlotAtt [40] | 28.9 2.0 / 21.9 1.2 / 30.7 3.1 / 37.3 2.4 | 9.4 4.8 / 10.4 3.8 / 15.4 4.6 / 18.1 7.2 | 4.4 1.1 / 7.9 0.3 / 11.9 0.9 / 14.7 0.7 |
| | YCB-T+U | ScanNet-T+U | COCO-T+U |
| | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec |
| AIR [22] | 18.9 13.7 / 21.0 9.9 / 32.5 15.7 / 32.6 13.2 | 12.3 5.4 / 16.8 5.6 / 37.4 20.8 / 20.4 1.1 | 25.3 13.0 / 28.3 9.0 / 49.4 13.2 / 38.2 12.4 |
| MONet [8] | 89.1 0.8 / 84.7 1.5 / 91.7 4.1 / 89.3 0.8 | 87.0 8.9 / 77.5 10.3 / 80.5 11.1 / 87.4 8.7 | 88.2 2.9 / 76.2 0.1 / 79.9 3.7 / 89.0 2.9 |
| IODINE [25] | 35.3 1.2 / 26.4 0.3 / 34.9 0.6 / 42.8 0.2 | 27.4 0.8 / 25.0 0.8 / 32.1 0.5 / 41.8 1.8 | 53.3 16.7 / 35.6 8.3 / 45.0 9.7 / 59.8 13.0 |
| SlotAtt [40] | 76.8 1.2 / 57.1 3.8 / 72.4 3.4 / 77.7 1.8 | 47.3 11.2 / 33.8 9.2 / 42.6 9.1 / 57.3 8.4 | 34.5 17.8 / 25.3 7.6 / 33.7 8.3 / 43.0 14.1 |

### A.6.3 Ablations on Object- and Scene-Level Factor Joint Ablations - Part 1

#### Ablated Datasets

- *Ablation of Object Color Gradient, Object Shape Concavity and Inter-object Color Similarity*: In the three ablated datasets: YCB-C+S+T / ScanNet-C+S+T / COCO-C+S+T, for each image, we first find the smallest convex hull [19] for each object. Then each convex hull is filled with a distinctive texture as shown in Figure 21. Lastly, we calculate the average color of the distinctive texture in each convex hull and change each pixel inside to be that color.
- *Ablation of Object Color Gradient, Object Shape Concavity and Inter-object Shape Variation*: In the three ablated datasets: YCB-C+S+U / ScanNet-C+S+U / COCO-C+S+U, we use the (C+S) datasets created before and find the uniform scale of the object convex hull.
- *Ablation of all four factors*: In the three ablated datasets: YCB-C+S+T+U / ScanNet-C+S+T+U / COCO-C+S+T+U, each image is ablated with the four factors together.

Figure 24: Example images of datasets ablated with both object- and scene-level factors.

Figure 24 shows example images from the three real-world datasets with different types of joint object- and scene-level factor ablations.

**Qualitative and Quantitative Results** As shown in Figure 25 and Table 7, all methods have achieved a significant improvement in object segmentation on the datasets ablated with joint object- and scene-level factors. The datasets with all four factors ablated can lead to impressive performance similar to the synthetic datasets. This shows that the distribution gaps of the objectness biases represented by the four factors between the synthetic and real-world datasets lead to the failure of existing unsupervised models.

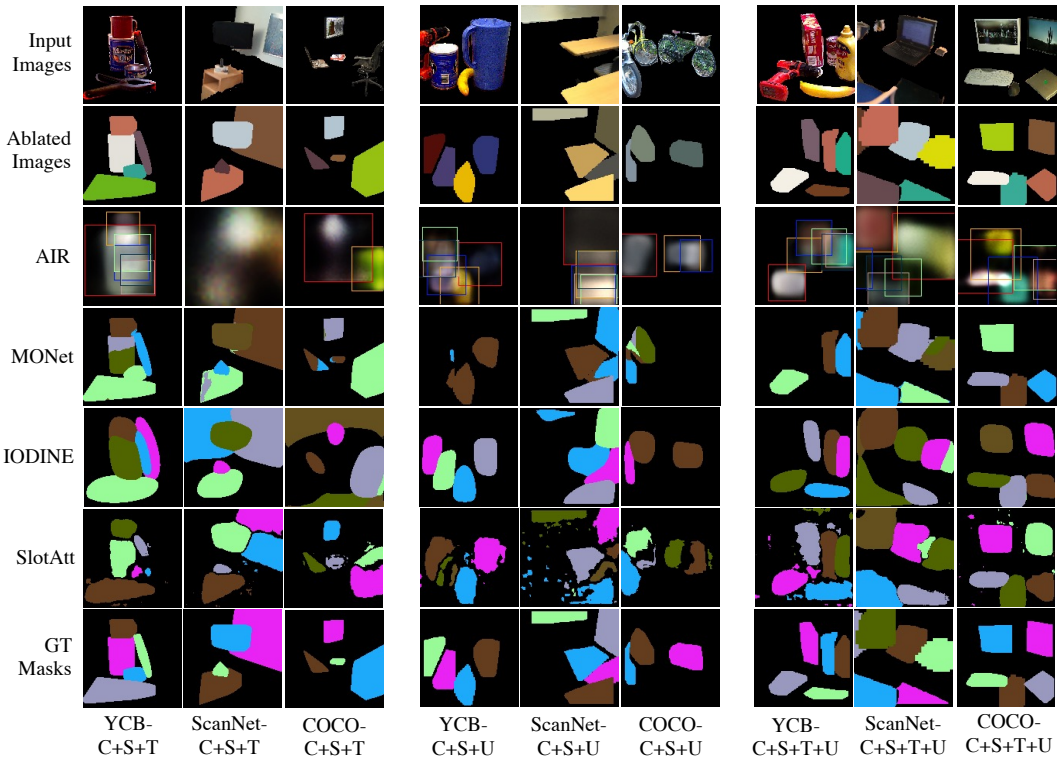



Figure 25: Qualitative results on the datasets ablated with both object- and scene-level factors.

Table 7: Quantitative results on the datasets ablated with both object- and scene-level factors. Standard deviations of performance are calculated over 3 runs (marked with blue).

| | YCB-C+S+T | ScanNet-C+S+T | COCO-C+S+T |
|---|---|---|---|
| | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec |
| AIR [22] | 6.3 3.7 / 10.7 3.3 / 24.0 13.2 / 13.1 0.5 | 1.9 3.9 / 6.4 5.2 / 27.6 10.7 / 6.3 13.9 | 10.8 5.6 / 18.6 7.0 / 45.1 14.4 / 20.3 6.9 |
| MONet [8] | 67.7 4.2 / 55.0 8.5 / 62.7 19.8 / 75.4 10.2 | 76.1 7.4 / 61.7 2.6 / 64.4 2.1 / 77.1 2.8 | 67.1 15.3 / 56.4 10.1 / 62.2 6.2 / 73.9 17.3 |
| IODINE [25] | 82.6 2.6 / 63.8 0.9 / 75.5 5.1 / 86.0 1.6 | 69.0 14.9 / 53.5 14.1 / 60.7 14.5 / 77.3 9.4 | 77.6 7.8 / 59.8 6.8 / 72.5 1.8 / 83.0 7.5 |
| SlotAtt [40] | 65.9 23.2 / 51.6 18.5 / 62.2 19.8 / 72.9 17.6 | 60.2 8.7 / 45.5 8.1 / 52.9 9.2 / 68.3 7.3 | 51.8 11.0 / 41.5 7.7 / 48.6 7.5 / 62.9 8.5 |
| | YCB-C+S+U | ScanNet-C+S+U | COCO-C+S+U |
| | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec |
| AIR [22] | 12.5 6.4 / 17.7 7.0 / 27.5 11.2 / 29.6 10.5 | 8.2 1.5 / 11.1 0.5 / 20.4 0.8 / 15.0 1.3 | 23.7 9.8 / 27.5 8.0 / 51.3 9.6 / 32.8 8.3 |
| MONet [8] | 3.5 2.5 / 4.8 3.2 / 23.5 2.6 / 5.3 6.5 | 60.1 9,5 / 56.3 13.3 / 69.7 20.7 / 64.0 6.2 | 35.6 1.4 / 33.0 3.8 / 43.4 16.6 / 38.7 0.1 |
| IODINE [25] | 65.3 9.2 / 55.9 10.4 / 73.9 4.6 / 67.7 8.4 | 64.6 3.2 / 52.1 5.1 / 61.3 7.6 / 68.2 2.2 | 64.8 7.3 / 53.4 6.5 / 65.8 8.5 / 68.2 2.5 |
| SlotAtt [40] | 58.1 3.7 / 39.1 2.8 / 52.4 2.3 / 62.0 3.6 | 50.5 2.9 / 38.2 3.6 / 51.8 6.9 / 55.0 1.3 | 20.2 5.2 / 18.1 0.3 / 26.9 0.1 / 33.4 0.6 |
| | YCB-C+S+T+U | ScanNet-C+S+T+U | COCO-C+S+T+U |
| | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec |
| AIR [22] | 24.8 20.1 / 23.2 11.9 / 33.7 16.8 / 39.1 20.1 | 10.0 1.8 / 13.5 1.8/ 24.0 1.4 / 18.2 7.4 | 36.4 19.3 / 39.4 14.5 / 63.1 16.9 / 49.2 17.9 |
| MONet [8] | 70.8 9.5 / 79.2 13.0 / 96.8 9.4 / 70.8 8.5 | 79.2 3.4 / 65.0 3.6 / 69.1 4.2 / 82.4 2.2 | 76.2 3.0 / 75.9 3.8 / 90.9 6.4 / 76.5 3.3 |
| IODINE [25] | 72.4 4.8 / 54.2 6.7 / 65.1 5.8 / 74.3 5.8 | 76.1 8.7 / 56.5 9.2 / 66.9 7.0 / 80.3 6.2 | 81.4 3.4 / 59.3 5.1 / 70.5 7.7 / 84.3 6.7 |
| SlotAtt [40] | 92.0 2.0 / 65.5 6.1 / 84.4 6.5 / 92.5 1.7 | 62.7 25.4 / 42.6 28.4 / 50.5 34.3 / 69.4 19.0 | 83.7 4.7 / 60.7 7.2 / 76.4 4.6 / 84.0 5.0 |

### A.6.4 Ablations on Object- and Scene-Level Factor Joint Ablations - Part 2

In addition to the experiments in Section A.6.3, we generate additional 6 groups of datasets ablated with different combinations of both object- and scene-level factors as detailed in Section A.6.4, and conduct extra experiments shown in Section A.6.4. Figure 26 shows example images of the additional ablated datasets.

**Ablated Datasets**

- *Ablation of Object Color Gradient and Inter-object Color Similarity*: In each image of the three real-world datasets, we replace the object color by averaging all pixels of a distinctive texture, and keep the original shape unchanged, getting three ablated datasets: YCB-C+T / ScanNet-C+T / COCO-C+T.
- *Ablation of Object Color Gradient and Inter-object Shape Variation*: In each image of the three real-world datasets, we replace the object color by averaging its own texture, and then apply size normalization on the original shape of objects, getting three ablated datasets: YCB-C+U / ScanNet-C+U / COCO-C+U.
- *Ablation of Object Color Gradient, Inter-object Color Similarity and Inter-object Shape Variation*: In each image of the three real-world datasets, we replace the object color by averaging all pixels of a distinctive texture, and then apply size normalization on the original shape of objects. The ablated datasets are denoted as: YCB-C+T+U / ScanNet-C+T+U / COCO-C+T+U.
- *Ablation of Object Shape Concavity and Inter-object Color Similarity*: In each image of the three real-world datasets, we replace the object color by a distinctive texture, and modify the object shape as a convex hull, getting three ablated datasets: YCB-S+T / ScanNet-S+T / COCO-S+T.
- *Ablation of Object Shape Concavity and Inter-object Shape Variation*: In each image of the three real-world datasets, we keep the texture of object unchanged, and modify the object shape as a convex hull followed by size normalization, getting three ablated datasetss: YCB-S+U / ScanNet-S+U / COCO-S+U.
- *Ablation of Object Shape Concavity, Inter-object Color Similarity and Inter-object Shape Variation*: In each image of the three real-world datasets, we replace the object color by a distinctive texture, and modify the object shape as a convex hull followed by size normalization. The ablated datasets are denoted as: YCB-S+T+U / ScanNet-S+T+U / COCO-S+T+U.

**Qualitative and Quantitative Results** As shown in Table 8 and Figures 27/28/29/30, all 6 additional combinations of object- and scene-level factors are explored, demonstrating consistent findings as our experiments in Section 4.4. Overall, all four methods show a high sensitivity to both object- and scene-level factors relating to appearance. This can be seen from the fact that for datasets without ablations in appearance, *i.e.*, the (S+U) ablated datasets, the object segmentation performance is inferior. By contrast, the object segmentation accuracy can be greatly improved on the datasets only

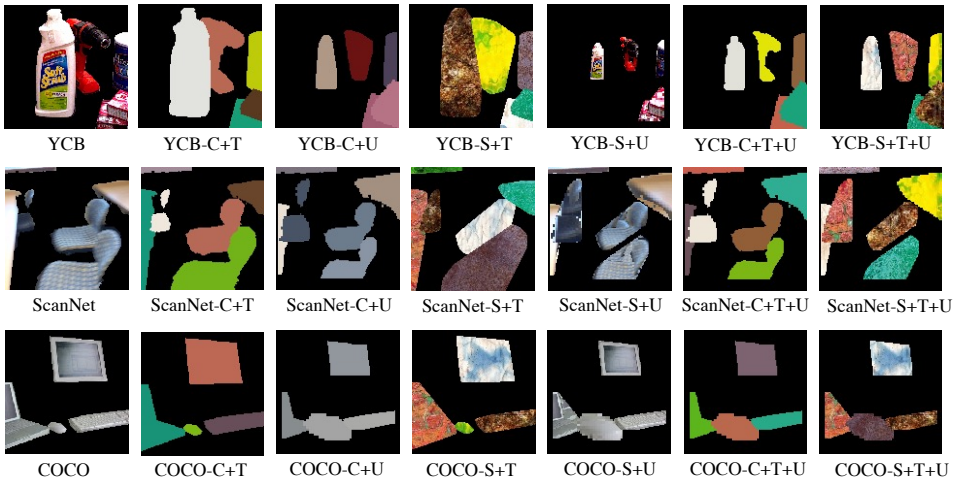



Figure 26: Example images of additional datasets ablated with both object- and scene-level factors.

Table 8: Quantitative results on additional datasets ablated with both object- and scene-level factors. Standard deviations of performance are calculated over 3 runs (marked with blue).

| | YCB-C+T | ScanNet-C+T | COCO-C+T |
|---|---|---|---|
| | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec |
| AIR [22] | 8.0 6.1 / 13.2 7.3 / 26.6 17.6 / 17.9 7.3 | 2.8 4.7 / 6.7 6.5 / 14.8 4.0 / 8.1 14.8 | 12.4 9.4 / 30.2 23.0 / 40.1 25.6 / 24.2 13.8 |
| MONet [8] | 82.9 9.6 / 66.7 0.5 / 73.0 1.2 / 87.8 12.2 | 36.5 28.4 / 33.5 21.8 / 40.4 24.1 / 50.3 19.7 | 66.1 5.6 / 53.2 3.7 / 56.2 2.3 / 74.4 6.4 |
| IODINE [25] | 78.9 2.6 / 59.7 2.1 / 71.5 0.6 / 83.6 2.7 | 65.1 2.9 / 51.1 1.0 / 62.4 2.9 / 74.2 0.5 | 55.1 10.4 / 42.2 7.5 / 56.3 6.4 / 66.2 9.2 |
| SlotAtt [40] | 58.7 12.1 / 43.2 5.0 / 57.8 9.4 / 71.0 9.4 | 29.2 2.9 / 27.6 1.3 / 34.4 1.4 / 49.2 1.2 | 22.1 8.6 / 19.6 4.1 / 25.2 4.1 / 35.8 8.1 |
| | YCB-C+U | ScanNet-C+U | COCO-C+U |
| | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec |
| AIR [22] | 11.4 4.9 / 16.4 5.4 / 25.8 8.7 / 25.1 6.5 | 5.4 0.5 / 12.3 2.6 / 35.1 15.3 / 12.6 0.4 | 20.5 10.8 / 24.9 7.1 / 47.7 10.2 / 30.5 8.8 |
| MONet [8] | 49.1 3.5 / 44.6 4.1 / 60.6 3.2 / 52.5 3.3 | 31.9 5.7 / 33.7 5.7 / 46.8 5.4 / 35.7 8.7 | 32.6 11.1 / 28.5 12.1 / 34.9 14.2 / 39.0 18.5 |
| IODINE [25] | 66.3 0.1 / 54.6 0.0 / 71.3 4.1 / 70.0 0.2 | 34.8 20.9 / 30.0 14.7 / 36.4 20.9 / 47.2 14.0 | 44.4 8.9 / 32.6 7.9 / 41.7 12.3 / 51.1 7.6 |
| SlotAtt [40] | 45.6 7.9 / 31.2 4.2 / 41.9 7.7 / 51.3 6.4 | 30.4 8.9 / 24.2 6.2 / 33.8 9.5 / 40.6 11.0 | 12.7 7.7 / 11.3 3.7 / 15.8 5.0 / 23.5 6.8 |
| | YCB-S+T | ScanNet-S+T | COCO-S+T |
| | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec |
| AIR [22] | 9.3 7.8 / 13.4 8.4 / 21.9 14.3 / 21.2 12.3 | 3.2 0.5 / 7.7 0.6 / 18.5 6.0 / 8.9 6.0 | 10.5 5.4 / 18.2 7.2 / 4.3 23.2 / 20.3 7.3 |
| MONet [8] | 86.7 2.2 / 78.9 1.6 / 81.9 1.8 / 87.0 2.2 | 83.2 3.7 / 77.8 13.0 / 86.0 20.0 / 83.8 2.9 | 80.2 0.0 / 68.5 7.2 / 73.2 11.7 / 82.3 1.0 |
| IODINE [25] | 41.9 17.9 / 33.2 11.4 / 41.5 12.0 / 51.3 14.7 | 54.9 23.1 / 44.4 16.2 / 52.0 16.7 / 68.0 21.6 | 44.1 1.1 / 34.3 0.7 / 41.0 0.3 / 56.8 0.6 |
| SlotAtt [40] | 77.3 7.8 / 60.6 1.5 / 75.0 2.3 / 69.2 19.7 | 24.9 46.1 / 21.2 35.0 / 25.5 38.3 / 37.9 43.2 | 67.4 3.1 / 50.2 1.6 / 59.2 1.6 / 76.4 0.3 |
| | YCB-S+U | ScanNet-S+U | COCO-S+U |
| | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec |
| AIR [22] | 0.8 5.4 / 2.9 7.1 / 5.0 10.0 / 5.2 12.7 | 6.3 0.2 / 13.1 3.3 / 32.4 16.3 / 14.3 1.2 | 20.0 9.5 / 25.8 10.0 / 48.3 15.6 / 31.6 10.9 |
| MONet [8] | 5.5 1.2 / 9.8 0.4 / 14.1 0.8 / 17.0 0.2 | 36.9 2.9 / 31.4 2.6 / 38.2 3.6 / 50.1 0.1 | 26.1 3.5 / 23.8 1.8 / 29.6 1.6 / 41.2 10.0 |
| IODINE [25] | 2.8 2.4 / 4.6 2.0 / 7.2 2.9 / 8.9 3.6 | 17.3 2.4 / 17.8 1.4 / 24.1 1.8 / 31.6 2.1 | 5.7 0.5 / 8.7 1.7 / 1.3 13.7 / 16.3 2.4 |
| SlotAtt [40] | 36.2 3.3 / 23.8 4.1 / 33.6 5.5 / 45.6 1.7 | 21.1 0.9 / 18.9 0.2 / 26.2 0.1 / 32.5 2.0 | 12.7 7.1 / 12.1 2.3 / 16.4 2.1 / 24.7 5.2 |
| | YCB-C+T+U | ScanNet-C+T+U | COCO-C+T+U |
| | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec |
| AIR [22] | 18.2 12.0 / 20.5 7.9 / 32.5 13.5 / 30.9 9.0 | 13.5 6.6 / 20.2 9.1 / 42.4 25.5 / 24.1 5.6 | 24.2 11.1 / 28.3 8.5 / 50.5 13.1 / 36.4 9.7 |
| MONet [8] | 63.2 6.9 / 66.5 11.8 / 86.3 13.0 / 64.4 5.7 | 74.6 5.6 / 62.1 6.6 / 67.8 7.2 / 79.2 3.2 | 73.0 3.3 / 75.7 10.3 / 94.2 23.1 / 73.1 7.5 |
| IODINE [25] | 54.9 23.1 / 38.7 18.2 / 58.0 12.0 / 59.4 20.8 | 65.5 3.8 / 48.0 3.2 / 59.8 5.9 / 70.7 2.9 | 47.9 11.7 / 35.5 7.4 / 53.8 2.1 / 53.4 14.2 |
| SlotAtt [40] | 73.8 4.6 / 53.0 2.9 / 67.6 7.8 / 75.2 4.5 | 58.5 8.1 / 45.5 10.2 / 52.9 8.9 / 68.3 11.9 | 30.5 27.6 / 20.3 18.0 / 27.8 23.0 / 38.9 23.3 |
| | YCB-S+T+U | ScanNet-S+T+U | COCO-S+T+U |
| | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec | AP / PQ / Pre / Rec |
| AIR [22] | 24.6 20.0 / 24.8 13.7 / 37.4 20.7 / 37.6 18.6 | 16.7 10.1 / 23.9 12.0 / 47.8 30.2 / 28.5 7.8 | 29.4 16.5 / 34.6 18.4 / 58.3 35.7 / 42.2 13.4 |
| MONet [8] | 88.4 0.6 / 84.6 1.5 / 92.0 1.5 / 88.6 0.7 | 97.8 0.3 / 87.3 0.8 / 85.4 0.1 / 98.1 0.4 | 86.9 0.1 / 83.8 11.6 / 89.9 13.1 / 87.0 1.5 |
| IODINE [25] | 42.3 15.0 / 32.1 9.7 / 41.0 11.5 / 52.0 15.7 | 39.1 4.7 / 29.9 1.8 / 41.8 7.2 / 47.9 2.4 | 64.3 9.7 / 45.1 6.9 / 51.4 5.6 / 72.0 11.7 |
| SlotAtt [40] | 88.6 6.5 / 67.0 9.6 / 80.9 13.0 / 89.6 5.6 | 82.6 8.5 / 61.5 10.1 / 73.2 10.2 / 83.9 7.7 | 84.2 0.9 / 59.6 6.8 / 74.7 3.8 / 85.3 0.6 |

with appearance factors ablated, *i.e.*, the (C+T) datasets. Meanwhile, more regular shapes and uniform scales of objects still have a significant positive influence on the success of object segmentation especially when the appearance factors are combined in ablated datasets. To be specific, AIR [22] is quite sensitive to the scale of objects apart from object color gradient and inter-object color similarity. MONet [8] can obtain comparable performance to the simple synthetic datasets once object color gradient and inter-object color similarity are ablated. All four factors are closely relevant the results of IODINE [25] and SlotAtt [40].

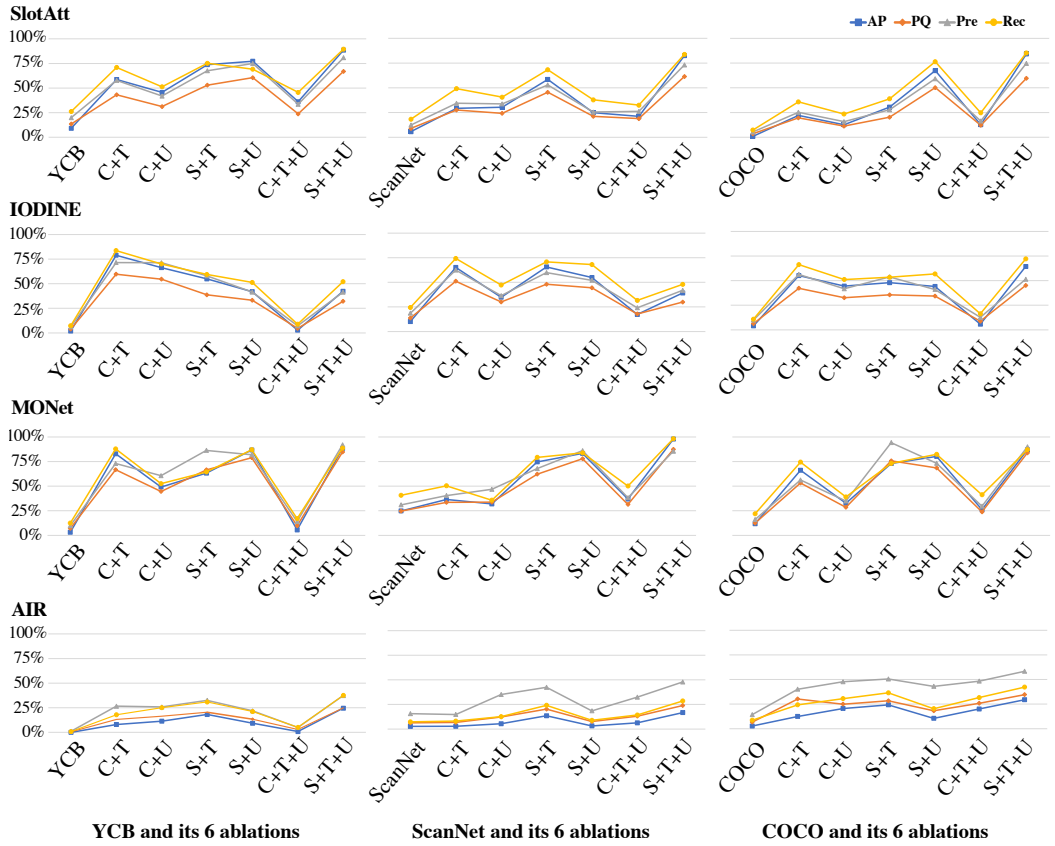



Figure 27: Quantitative results of baselines on three real-world datasets and their variants in Sec A.6.4

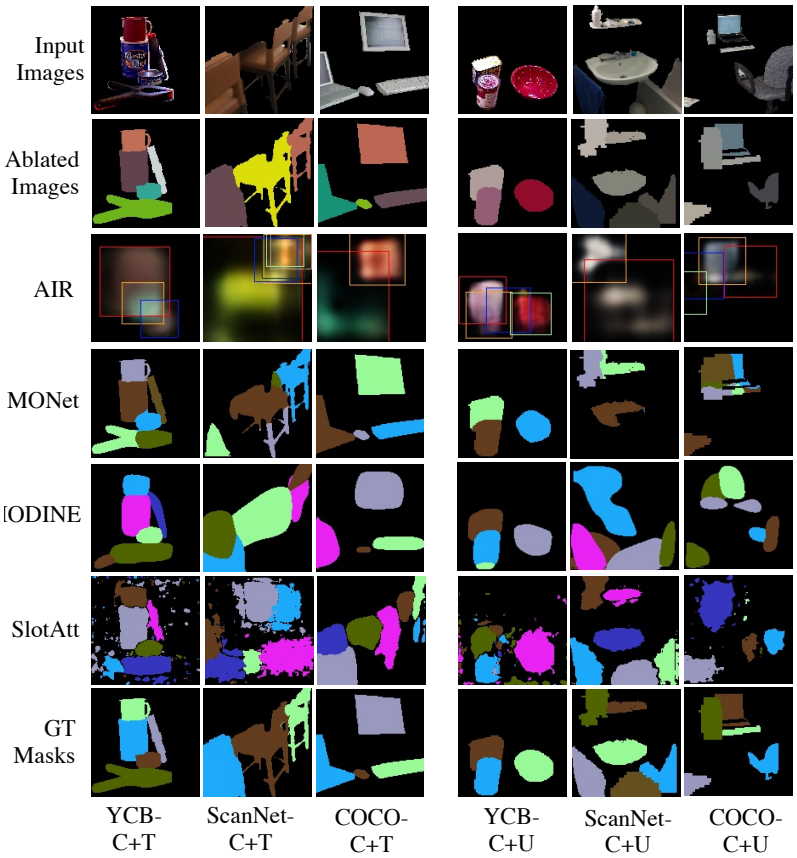



Figure 28: Qualitative results on the additional datasets ablated with object- and scene-level factors.

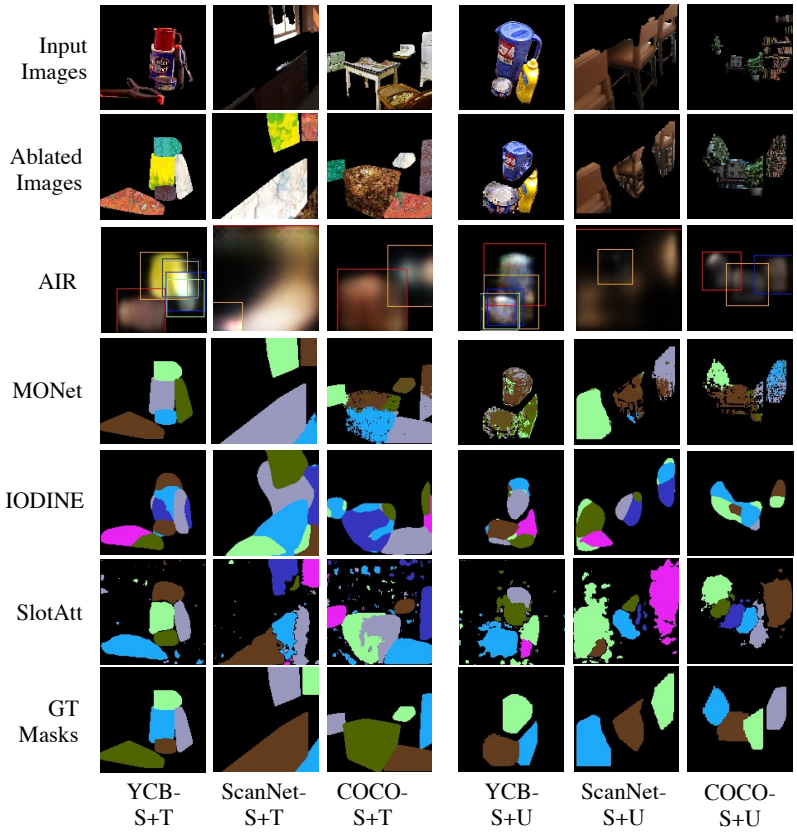



Figure 29: Qualitative results on the additional datasets ablated with object- and scene-level factors.

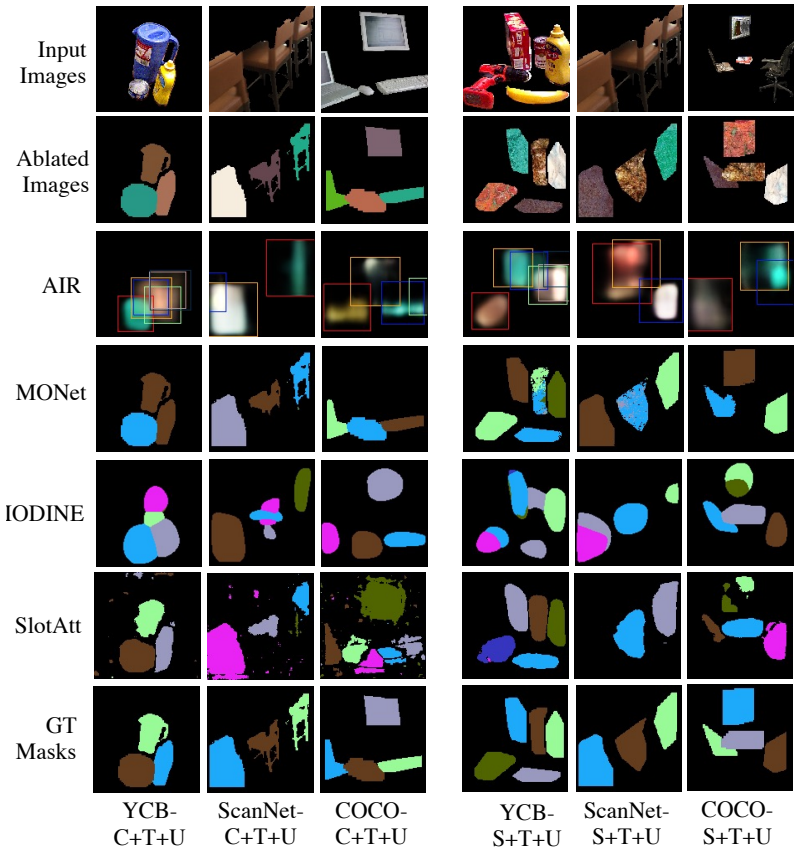



Figure 30: Qualitative results on the additional datasets ablated with object- and scene-level factors.