

Bottom-up modeling of repeated elements via single image analysis-by-synthesis

Syrine Kalleli^{1*}, Alexei A. Efros², and Mathieu Aubry¹

¹ LIGM, CNRS, Univ Gustave Eiffel, ENPC, Institut Polytechnique de Paris, France

² UC Berkeley

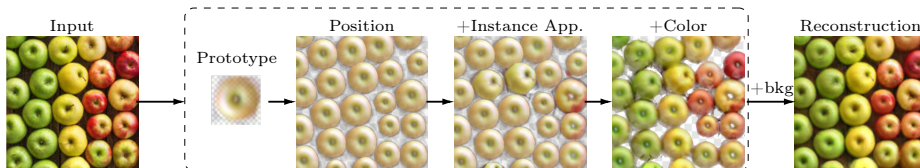


Fig. 1: Element and image models. Our goal is to learn a single image-space prototype that captures the common appearance of repeated elements within one image. Our method reconstructs the input image by transforming a shared prototype with instance-specific pose and appearance parameters. The resulting foreground is composited with a low-resolution background to produce the final reconstruction.

Abstract. We address the problem of discovering repeated elements from a single image. In contrast to existing approaches that depend on large annotated datasets, curated multi-image collections, or object segmentation masks, we show that a single image can suffice to learn a meaningful object model in a completely bottom-up fashion, without any prior knowledge beyond a coarse scale prior. Our method learns a tunable image-space prototype of the repeated elements through a reconstruction objective, enabling the model to identify and synthesize consistent object instances within the same image. Experiments on 116 real images from the FSC-147 dataset demonstrate that our method successfully learns coherent element models and captures intra-category variation on challenging images. Qualitative results reveal superior reconstructions and interpretable decompositions compared to classical decomposition, joint alignment, and 3D object modeling methods, while maintaining a simple 2D formulation. These results suggest that meaningful object discovery can emerge from single-image learning alone.

1 Introduction

Pattern recognition – discovering approximately repeating elements within the sensory data – is one of the most fundamental building blocks of animal and

* Corresponding author: syrine.kalleli@enpc.fr

human intelligence. It is what allows organisms to start building representations of their sensory world.

Bottom-up pattern recognition has been a bedrock of early computer vision, including unsupervised clustering, bottom-up segmentation, texton modeling, and unsupervised object discovery in image collections. In contrast, modern computer vision has largely forgone bottom-up methods in favor of top-down supervised approaches (supervised segmentation, supervised object counting, *etc.*). Such methods perform very well on benchmarks but often fail when presented with novel, out-of-distribution data. Moreover, their reasoning is often difficult to interpret, resulting in a black-box effect, which is problematic, in particular, for scientific applications.

In this paper, our goal is to revisit the bottom-up tradition of modeling visual data, starting from pixels, without supervision or annotation beyond a coarse scale prior. We additionally find that this object-scale prior matters only for the smallest objects. Our task is a special case of the pattern recognition problem – identifying and modeling repeated elements within a single image. We take the analysis-by-synthesis approach, using a reconstruction objective to jointly optimize a parametric model of the repeated element, together with each instance’s occurrence and parameters. Our resulting model is not only effective compared to previous works, but is also highly explainable, as visualized in Figure 1.

From the technical point of view, our architecture is inspired by deep object-centric image decomposition methods, and in particular capsule networks [33, 53] and sprite-based methods [44, 60]. However, their setting is quite different from ours – they assume access to multiple images of multiple objects in different configurations. More importantly, they struggle to work beyond simple synthetic data, except in much simpler settings, such as a single object per image, where they become similar to congealing approaches. Critical to the success of our method are a low-dimensional parametric element generation module, which enables the modeling of slight element variations without sacrificing interpretability and controllability, and a crop-based curriculum learning strategy.

We demonstrate our results on a curated subset of the FSC-147 counting dataset [52] of real images, assessing not only counting but also the modeling of elements and images.

2 Related Work

In this section, we give an overview of methods that focus on discovering and modeling single-image and multiple-image repetitions. While our work falls into the single-image category, it also builds on ideas that have been developed for multiple images, from which we also build most of our baselines. We summarize distinctions with related work in Table 1.

Table 1: Comparison of our method with related approaches.

Method	Object discovery	Object model	No regularity assumption
Single-Image Generative Models [34, 56, 57, 64]	✗	✗	✓
Regular pattern discovery [14, 22, 41, 46, 54]	✓	✗	✗
3D inverse rendering [12, 18, 68]	✗	✓	✓
Ours	✓	✓	✓

2.1 Repeated elements in a Single Image

Using repetition to detect and model important image elements is an idea that dates back to early work on visual texture. Early methods often attempt to explicitly model repeated elements but rely on strong geometric assumptions. For example, [37] assumes that the elements are close to each other, while [54] assumes that the elements lie on a common plane. Other approaches focus on texel discovery [1] through feature matching or exploit structured regularity, such as 2D lattices [14, 22, 41] and 1D linear patterns [46].

Identified recurrent elements were often used for image processing purposes, such as editing [11], super-resolution [19, 26], texture synthesis [15] and novel-view synthesis [63]. [10] revisits this line of work and performs near periodic patterns image completion by optimizing a continuous MLP-based implicit representation of repeated patterns. A related line of work learns a generative model of a scene from a single image by exploiting its internal patch statistics, using GANs [56, 57] to match multi-scale patch statistics or diffusion models [34, 64] for higher-quality, more diverse generation. These methods capture recurrence only implicitly, as a distribution over patches. In contrast, we learn an explicit, parametric model of the repeated element. More recently, several works use repetition for 3D modeling, learning from exact duplicates [12] or category-specific instances [18, 68, 69]. Similarly to our work, [68] learn a parametric object model of the repeated object. However, these 3D methods are expensive and require clean ground-truth segmentation masks. Furthermore, they struggle with 2D scenarios, such as aerial imagery, or cases where the lack of diverse viewpoints prevents robust 3D reconstruction.

Moving away from attempts at modeling, counting methods focus on estimating the number of repeated elements, and often identifying their location. Counting methods can be broadly categorized depending on the type of supervision they require. Early approaches were fully supervised, and limited to a single object class of interest [36]. Class-agnostic counting, introduced in [43], reformulates the problem as locating repetition of a given exemplar, enabling generalization to unseen classes at test time. Many recent works [32, 40, 61] leverage transformers to predict density maps. In particular, [32] generate synthetic training data by copy-pasting objects within the same feature cluster defined by DINO representations. With the extension of one of the main datasets of the field, FSC-147 [52], to few-shot counting and detection (FSCD) [47], several follow-up methods adopt detection-based formulations, including C-DETR [47],

Dave [50], PseCo [27], GeCo [50] and TMR [17]. For our baselines and ablations, we use TMR [17] which provides state-of-the-art results, can segment patterns that are not associated with object categories, and provides as output a count, bounding boxes, and optionally segmentations. These approaches can be further adapted to zero-shot counting, using a textual description of the object instead of an exemplar [2]. Some methods enable having both text queries or exemplars as input [3, 27]. Others tackle the more challenging exemplar-free setting, such as RepRPN [51]. Recently, the large-scale plant counting dataset TPC-268 [66] was released, providing fine-grained and biodiverse images. Going further, ABC123 [24] proposes multi-class, exemplar-free counting through pre-training on labeled synthetic data. Since it does not require exemplars or text prompts, we argue that it is the most relevant baseline for our unsupervised approach in terms of counting results. However, unlike our method, it does not provide any model of the repeated element or the image.

2.2 Repeated elements in Image Collections

Single element type. Approaches to align images to a shared canonical template can be traced back to congealing [25, 35] where an image set is iteratively aligned to a shared image through entropy minimization. More recent works rely on Spatial Transformer Networks [28] (STN) to differentially warp instances. Some use the latent space of GANs [45, 49] to learn a model across a large dataset depicting a single object class. Others leverage the rich representations of self-supervised models such as DINO [9], which provide powerful cues for semantic correspondence [4, 5, 21, 23, 48]. In particular, SpaceJAM [5] proposes a fast, lightweight, joint alignment with an implicit atlas. Most recently, FastJAM [23] accelerates convergence by training a graph neural network on sparse keypoints from a supervised matcher. Similar to these works, our discovered instances are implicitly aligned. However, our approach does not rely on access to a collection of images of a single object, which provides strong implicit supervision.

Multiple element types. Object-centric image decomposition methods [62], aim at discovering structured, object-centric decompositions of images. While these works successfully handle multiple objects, they are generally evaluated on controlled, synthetic datasets [30]. Methods such as AIR [16] and SPAIR [13], factorize scenes into explicit latent variables for "what", "where", and "presence." These methods attend to local regions or "glimpses" and typically rely on a costly sequential inference. To improve scalability, SPACE [39] introduced a parallelized patch-based decomposition. Other works, such as MoNet [7] and IODINE [20] compute full-image pixel-level masks to isolate objects. However, these models suffer from high computational overhead and struggle to scale due to their reliance on iterative refinement or sequential attention. Slot Attention [42] and follow-ups [6, 29, 55, 59] improve this by using a single step of iterative slot attention. Closest to our work are sprite-based methods [44, 60], which learn an image-space dictionary of objects that can be transformed and placed onto a canvas. This paradigm is particularly attractive for our task since it compresses

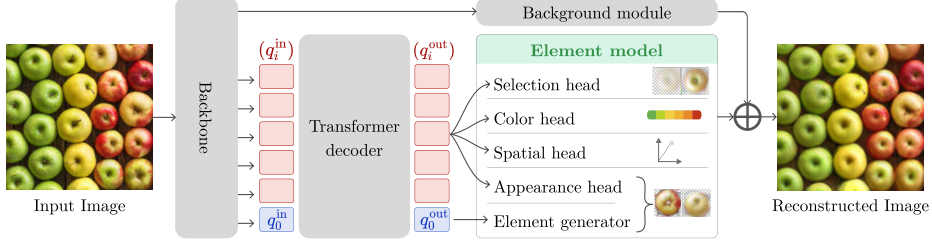


Fig. 2: Architecture overview. The input image is encoded into patch features and processed by two parallel branches: a background module for low-resolution predictions and an element prediction module. The latter uses a transformer decoder with an appearance query q_0^{in} and instance queries $\{q_i^{in}\}_{i=1}^N$ to produce output tokens $\{q_i^{out}\}_{i=0}^N$. While the q_0^{out} captures shared features, each q_i^{out} independently predicts existence probability, spatial parameters (position, rotation, and scale), a color transform, and an appearance code for element generation. Finally, these generated elements are transformed and composited onto the background via alpha blending, weighted by their existence probabilities.

the scene into a set of controllable, learnable objects. However, DTI-Sprites [44] is not designed to handle a large number of objects or a single-image setup, and MarioNette [60] is patch-based and is limited by its grid resolution. Additionally, in both of these prior works, the prototype corresponding to a single object is fixed. In contrast, our approach allows for appearance variations for each instance beyond those captured by colorimetric and simple geometric transformations, which enables it to handle real images.

3 Method

In this section we first present the core of our approach, namely in Section 3.1 our element-based image model and the associated architecture, and in Section 3.2 our single-image fully unsupervised optimization. Then, in Section 3.3, we describe how to incorporate annotations as exemplar bounding boxes or element locations such as those provided by counting models. Finally, in Section 3.4, we present a synthetic pretraining that significantly speeds up single-image optimization and improves our results.

3.1 Element-based modeling and architecture

Overview. Given a single input image $I \in \mathbb{R}^{h \times w \times 3}$ containing multiple repeated instances of the same element, our objective is to jointly learn an element model and discover its instantiations in the scene. We represent the scene as the composition of a background image $I_B \in \mathbb{R}^{h \times w \times 3}$ and L foreground layers. Each foreground layer l_i corresponds to one element instance and is obtained by applying a predicted spatial and color transformation to a shared low-dimensional

element model. An overview of our image modeling approach is presented in Figure 1 and an overview of our architecture can be found in Figure 2.

General architecture. More specifically, the input image is first processed by a backbone into patch features. These features are then fed into two parallel networks: a background module, which predicts a low-resolution background image, and an element prediction module, which models all foreground elements.

The element prediction module starts with a transformer decoder, which takes as input a set of learnable instance queries $\{q_i^{in}\}_{i=1}^N$ and a single global appearance query q_0^{in} and cross-attends to the image features. It produces output tokens $\{q_i^{out}\}_{i=0}^N$. Intuitively, the goal of the global appearance token q_0^{out} is to capture the general element appearance, while the instance queries $\{q_i^{out}\}_{i=1}^N$ aim at learning element occurrence and instance-specific pose and appearance.

Each output instance token $\{q_i^{out}\}_{i=1}^N$ is processed independently by four specialized heads: (a) a selection head that predicts an existence probability; (b) a spatial head that predicts scale, rotation, and translation; (c) a color head that estimates a linear color transformation in RGB space; and (d) an appearance head, that predicts a low-dimensional appearance code (1D in all experiments), which conditions the element generation module, described in the next paragraph, to produce instance-specific elements. The instance-specific elements produced by the element generation module and the transformation parameters (b-c) are then used to model each instance. Finally, all generated instances are composited through alpha-blending, with opacity weighted by their probability (a), onto the predicted background to produce the reconstructed scene.

Element model. The element generation module can be thought of as extending the element model beyond a single element prototype to a low-dimensional (1D in our case) parametric family of appearances. Its goal is to model small variations in appearance across instances and 3D effects. The RGBA image of a specific element $i \in \{1, \dots, N\}$, $I_i \in \mathbb{R}^{w_P \times h_P \times 4}$, is generated by a small generator network from the concatenation of the global appearance token $\{q_0^{out}\}$ and the instance appearance tokens $\{q_i^{out}\}$ are predicted by the appearance head.

Architecture details. The backbone is a frozen pre-trained ViT-B/16 DINOv3 [58] model, selected for its rich self-supervised object-centric representations, whose features are projected to 64 dimensions by a learnable linear layer. The background module applies global average pooling followed by a 2-layer MLP with 64 hidden neurons predicting a background color. The element prediction module uses 60 instance queries and one global appearance query, all of dimension 64. It consists of 3 standard cross-attention/self-attention blocks. Each of the selection, color, spatial, and appearance heads is implemented as a 2-layer MLP with 64 hidden neurons. The element generation module is also a 2-layer MLP with 64 hidden neurons.

The prototype resolution is $(w_P, h_P) = (64, 64)$. We use images of size $(224, 224)$ but only reconstruct them at $(128, 128)$ resolution to limit computational cost. The background and the object layers are composited via alpha

blending in a fixed order. The spatial module predicts the *inverse* of the transform fed to the STN [28] as we found it leads to much more stable training.

3.2 Unsupervised single-image optimization

Loss. Our unsupervised loss $\mathcal{L}_{\text{unsup}}$ is the combination of a reconstruction loss $\mathcal{L}_{\text{rec}}$ and a sparsity regularization loss $\mathcal{L}_{\text{reg}}$ which prevent the element module from modeling background regions:

$$\mathcal{L}_{\text{unsup}} = \mathcal{L}_{\text{rec}} + \lambda_{\text{reg}} \mathcal{L}_{\text{reg}}, \quad (1)$$

where λ_{reg} is a scalar hyperparameter. The reconstruction loss combines a pixel-wise Huber loss and a structural similarity term:

$$\mathcal{L}_{\text{rec}} = \mathcal{L}_{\text{Huber}} + \lambda_{\text{SSIM}}(1 - \text{SSIM}(I, \hat{I})), \quad (2)$$

where I and $\hat{I}$ denote the target and reconstructed images, and the Huber loss is defined as

$$\mathcal{L}_{\text{Huber}} = \frac{1}{N_{\text{pix}}} \sum_x \begin{cases} \frac{1}{2}(I_x - \hat{I}_x)^2 & \text{if } |I_x - \hat{I}_x| \leq \delta, \\ \delta \left(|I_x - \hat{I}_x| - \frac{\delta}{2} \right) & \text{otherwise,} \end{cases} \quad (3)$$

where the sum is over pixels x in the image, N_{pix} is the total number of pixels, I_x and $\hat{I}_x$ are the values of the target and reconstructed images at pixel x , and $\delta > 0$ is a scalar hyper-parameter.

The regularization loss penalizes the number of elements used for the reconstruction:

$$\mathcal{L}_{\text{reg}} = \frac{1}{K} \sum_{k=1}^K |p_k|, \quad (4)$$

where K is the number of element tokens and p_k is the probability predicted by the selection head for the k -th element token.

Curriculum learning. Jointly training all modules from the start can cause ambiguities (*e.g.* the element generation module can learn to move, scale, and color the elements or reconstruct the background). To avoid this, we adopt a 3-step curriculum learning strategy. First, we train the background model keeping all other modules frozen. Second, we train all other components, except the appearance prediction head, whose output we fix at zero to enforce a shared appearance across all instances. Third, we unfreeze the appearance prediction head, which allows for instance-specific adaptation of the element appearance.

Crop-based optimization. Single-image optimization leads to uninteresting solutions, such as optimizing colored blobs to reconstruct the input image. To avoid this, we apply strong spatial data augmentation as a form of regularization. For each input image, we generate fixed-size training crops using random cropping and rotation. This augmentation is such that the smallest side of the object is between 10% and 50% of the crop side. We obtain the scale prior needed to define the crop size range for each image from the exemplar instance annotations available in the dataset.

Full image reconstruction. Since our model is trained on crops, reconstructing the full image is non-trivial: its scale and number of elements differ from the crops seen during training. We therefore extract overlapping crops with a sliding window and predict element parameters and existence probabilities per crop, which may model the same element multiple times. To remove duplicates, we map predicted transformations to global image coordinates, discard elements below a probability threshold δ_{prob} , and select the remaining ones sequentially in order of decreasing probability: a prediction is kept only if it reduces the reconstruction loss by more than δ_{loss} and its overlap with previously selected elements stays below δ_{overlap} . Finally, we run a full-image refinement stage, jointly optimizing the selected elements’ parameters, the average predicted background, and the prototype generation to improve reconstruction quality.

Optimization details. In our experiments, we use $\lambda_{\text{SSIM}} = 0.01$, $\lambda_{\text{reg}} = 0.001$, and $\delta = 1$ in the Huber loss. We train our network on image crops using the Adam optimizer with a learning rate of 10^{-3} , a linear warmup of 500 iterations, and a batch size of 64. To speed up training, we initialize the prototype generator with a gaussian blob. We use 100 iterations, 500 iterations, and 2500 iterations for the three steps of the curriculum training, respectively. For the full image reconstruction, we use a sliding window of average crop size, a stride of 25%, a probability threshold $\delta_{\text{prob}} = 0.5$, a reconstruction threshold $\delta_{\text{loss}} = 10^{-5}$, and an overlap threshold $\delta_{\text{overlap}} = 0.1$. The final refinement is performed for 200 iterations with a learning rate of 10^{-4} .

3.3 Optional annotation inputs

Our method can of course benefit from additional inputs. We consider two types of input: (i) the bounding box of an element, which is typically provided by counting datasets and can disambiguate element definition; (ii) location supervision, in the form of a set of points corresponding to all the elements’ centers, which is a typical output for counting methods. These are particularly beneficial for cases with part-whole ambiguity, e.g., between a pair of glasses and its half. These are often impossible to disambiguate from purely visual cues, in particular, in the common cases of regular arrangements and symmetric objects.

Exemplar supervision. To take advantage of exemplar bounding boxes, we add for each available exemplar box region R an additional loss, $\mathcal{L}_{\text{ex}}$. It is defined by selecting the predicted element which best fits the box and encouraging it to reconstruct the exemplar region when combined with only the background. Let us assume that for a target image $I \in \mathbb{R}^{h \times w \times 3}$ the model predicts a background image and N elements with probabilities $p^n \in [0, 1]$, spatially transformed masks $M^n \in [0, 1]^{h \times w}$, and colored appearances $A^n \in \mathbb{R}^{h \times w \times 3}$. We first select the best-fitting instance n^* with the exemplar box R as

$$n^* = \arg \max_{n \in \{1, \dots, N\}} \sum_{x \in R} p^n M_x^n, \quad (5)$$

where the sum is over the pixels x in R and M_x^n is the value of the mask M^n at pixel x . We then compute a single-exemplar reconstruction of the image with $\hat{I}_{n^*}$ by alpha-blending the selected element appearance A_{n^*} with the background, using the mask $p_{n^*}M_{n^*}$. The exemplar loss $\mathcal{L}_{\text{ex}}$ is then defined to encourage $\hat{I}_{n^*}$ to be close to the target I in the annotated exemplar bounding box region R and the probability p_{n^*} to be close to 1:

$$\mathcal{L}_{\text{ex}} = \mathcal{L}_{\text{rec} \mid R}(I, \hat{I}_{n^*}) + \lambda_{\text{pres}}(1 - p_{n^*}) \quad (6)$$

where $\mathcal{L}_{\text{rec} \mid R}$ is the reconstruction loss defined in the previous section restricted to the region defined by R , and λ_{pres} is a scalar hyperparameter set to $\lambda_{\text{pres}} = 0.1$ in our experiments.

Location supervision. In images with complex contrasted backgrounds, our unsupervised approach might use colored elements to approximate the background. This could be solved by using a detection-based counting network, trained on large-scale data, to produce a set of pseudo ground-truth element location annotations. In that case, we use the Hungarian algorithm to find an optimal one-to-one assignment between the instance locations predicted by the element prediction module and the pseudo-labels provided by the counting model. With this optimal matching, we define a loss:

$$\mathcal{L}_{\text{loc}} = \lambda_{\text{pos}}\mathcal{L}_{\text{pos}} + \lambda_{\text{prob}+}\mathcal{L}^{\text{prob}+} + \lambda_{\text{prob}-}\mathcal{L}^{\text{prob}-}, \quad (7)$$

where $\mathcal{L}_{\text{pos}}$ is the sum squared distance between pseudo-labels and their matched predictions, λ_{pos} , $\lambda_{\text{prob}+}$ and $\lambda_{\text{prob}-}$ are scalar hyperparameters, and $\mathcal{L}^{\text{prob}+}$ and $\mathcal{L}^{\text{prob}-}$ are losses on the probabilities similar to the ones of DETR [8] which encourage matched elements to have high probabilities and non-matched elements to have low probabilities. In our experiments, we use $\lambda_{\text{pos}} = 0.5$, $\lambda_{\text{prob}+} = 0.5$ and $\lambda_{\text{prob}-} = 0.2$ and remove the sparsity regularization loss $\mathcal{L}_{\text{reg}}$ when using location supervision.

3.4 Synthetic pretraining

Optimizing a per-image model from scratch is computationally inefficient, and most of the optimization time is spent finding coarse object locations with simple blob objects, and the optimization sometimes ends up in bad local minima. Thus, in order to speed up the optimization and improve results, we pretrain the network on many synthetic images with repeated elements. Our architecture and training objectives remain the same, but the task is significantly more difficult since the element generation module now needs to learn a general element model. Thus, this pretraining provides very poor image and object models when applied directly to a real image but a good initialization for our single-image optimization.



Fig. 3: Samples from TPC-268 [66] (left) and our learned prototypes (right).

Synthetic Data Generation Pretraining our network with real images would be appealing, but no large scale dataset of images with repeated objects is available, and real data is often too difficult, leading the model to reconstruct the scene with colored blobs. Synthetic data provides an appealing alternative as the difficulty can be controlled, unlimited training examples can be generated, and the model can be supervised without cost. We leverage images and segmentations from COCO [38] to generate our synthetic images. To generate a synthetic image, we first choose randomly a segmented object, and a background. The background can be a uniform color or a heavily blurred crop of an image from the dataset. The foreground is constructed by pasting copies of the selected object several times with random position, scale, rotation, and color jitter. In practice, we pretrain on 5,000 synthetic images. Examples can be seen in our supplementary material.

Pretraining details. The pretraining is done in a supervised fashion with the reconstruction, exemplar, and location supervision objectives. We pretrain the network on 5,000 synthetic images with the Adam optimizer and a learning rate of 10^{-3} starting with 500k iterations using only images with uniform background, then with an equal mix of uniform and blurred background for 500k iterations.

4 Experiments

4.1 Dataset and Evaluation Protocol

Dataset. To evaluate and compare methods for our task, we build a dataset by curating images from the 29 test categories of the standard FSC-147 [52] counting dataset. For each category, we select four images to maintain a balanced representation of classes. We favor images with simpler backgrounds and diverse object poses and counts. This results in 116 images, annotated with object count and three exemplar bounding boxes. We additionally show the learned prototype model for a sample of images from the plant dataset TPC-268 [66] in Figure 3.

Evaluation and metrics. We report metrics for element modeling, image modeling, and object count. To assess element modeling quality, we measure how accurately a method reconstructs each of the three annotated exemplars with a single element. More precisely, for each exemplar, we select for each method the predicted element whose mask overlaps the most with the exemplar bounding box, which defines a foreground and a background mask. We then build

Table 2: Quantitative results. $\uparrow$ indicates higher is better, $\downarrow$ lower is better. The *Ex* column indicates methods that require an exemplar or a text prompt to be provided, relying on user input at test time. The *Sup* column indicates methods that rely on supervised training from human annotations.

Method	Ex	Sup	Counting		Image Reconstruction			Prototype Quality		
			MAE $\downarrow$	RMSE $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$
ABC123 [24]			20.92	28.58	–	–	–	–	–	–
GeCo [50]		$\times$	7.34	13.66	–	–	–	–	–	–
TMR [17]	$\times$	$\times$	4.11	6.82	–	–	–	–	–	–
TMR+SAM+Average	($\times$)	($\times$)	–	–	13.95	0.423	0.543	17.44	0.658	0.464
TMR+SpaceJAM [5]	($\times$)	($\times$)	–	–	13.63	0.405	0.544	17.77	0.674	0.430
TMR+SAM+3D [68]	($\times$)	($\times$)	–	–	–	–	–	15.53	0.568	0.500
Ours			14.11	26.93	23.76	0.737	0.242	19.40	0.700	0.353

a reconstructed exemplar by compositing the selected element with a uniform background, using as color the average ground-truth color over the background mask. Note that this does not require a ground-truth mask and avoids favoring methods that use several elements to reconstruct a single exemplar. These single-element exemplar reconstructions are also the ones we show in our qualitative results. To assess image modeling quality, we composite all elements predicted by a method with a uniform background and compare the result to the ground-truth image. Similar to what we do for the exemplar metric, we use as background color the mean color of the input image outside the predicted elements mask.

For both object and image modeling, we report PSNR, SSIM [65] and LPIPS [67]. For counting metrics, we report the standard mean average error (MAE) and root mean square error (RMSE) metrics.

4.2 Element modeling baselines

To the best of our knowledge, no stand-alone method can model repeated elements in an image, and existing methods require element bounding boxes or masks. We thus build baselines by first extracting element bounding boxes using TMR [17], a state-of-the-art counting method, and optionally obtain element masks from its SAM [31] decoder branch. Note that TMR requires exemplar annotations, so these baselines actually require more supervision than our unsupervised method. Our three baselines are as follows.

- *TMR + SAM + Average*: we define a prototype by averaging the segmented instances and their SAM masks, which can then be placed and scaled according to TMR bounding boxes to model the image.
- *TMR + SpaceJAM [5]*: we use SpaceJAM to perform a joint alignment between all instances extracted by TMR and use the average of aligned instances as prototype appearance. SpaceJAM’s mask is used for each exemplar.
- *TMR + SAM + 3D object modeling [68]*: we use [68] to obtain parametric 3D models from the segmentations obtained by TMR and SAM. This method

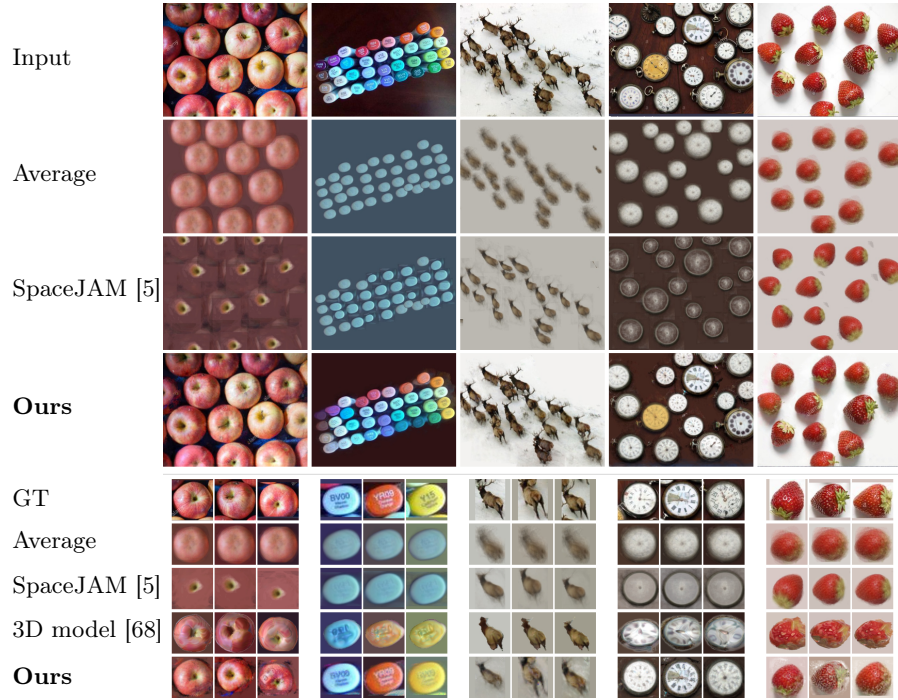


Fig. 4: Qualitative comparison. Top: Full-image reconstructions, Bottom: Exemplar reconstructions for the three exemplars annotated in the dataset.

is quite slow, requiring 10 hours to train for a single image, then more than half an hour per exemplar. Thus, for this baseline, we only evaluate element modeling.

4.3 Results

Comparisons In Table 2, we compare our exemplar-free approach to counting methods and element modeling baselines, and in Figure 4 to modeling baselines.

ABC123 [24] is an exemplar-free multiclass counting method, trained only on synthetic data, which reports 4 counts per image. To avoid penalizing class ambiguity (*e.g.* splitting red and green apples), we consider, for each image, the count with minimum error between (i) each of the 4 predicted counts and (ii) the sum of the counts. To the best of our knowledge, ABC123 is the best counting method relying neither on human supervision during training nor image prompts. GeCo [50] is a supervised counting network that supports exemplar-free counting via a pretrained latent prototype. TMR [17] is a state-of-the-art exemplar based supervised training method. Although not primarily designed for counting, our method outperforms ABC123. It is outperformed by the other supervised counting methods, which is expected.

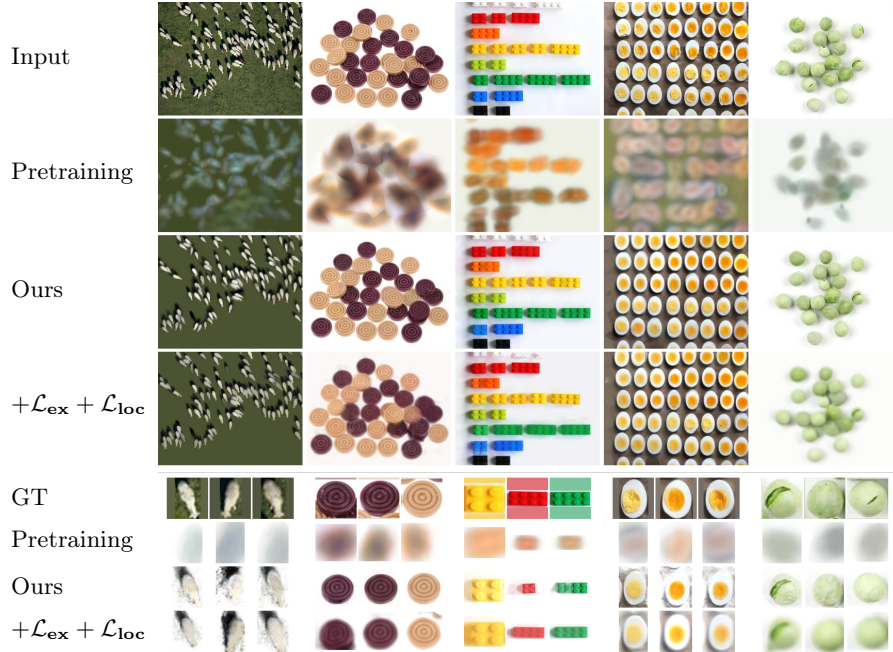


Fig. 5: Ablations and Comparison. Top: Full-image reconstructions, Bottom: Exemplar reconstructions for the three exemplars annotated in the dataset.

Our element modeling baselines are unsupervised but require access to all individual element bounding boxes or segmentations. For fair comparison, we obtain them through the best counting method, TMR, which requires both supervision and an exemplar. Still, our method outperforms all of these baselines on element and image modeling. This advantage is also evident in the qualitative comparison shown in Figure 4, where our method learns a meaningful element and faithfully reconstructs each of the instances. While SpaceJAM [5] performs an alignment similar to ours in spirit, its saliency mask can degenerate, in which case it models only part of the object, *e.g.* in the apple example. It is also unable to model color or appearance variation, which is obvious in the watch example. The 3D method [68] can also struggle with larger variations in appearance, and more critically, is very sensitive to the quality of its input segmentation masks.

Ablations We report ablations and variants of our method in Table 3 and visualize some in Figure 5.

Removing the appearance head, i.e., modeling each image with a single prototype, has a smaller impact on the counting metrics but a more important one on full image and prototype modeling, in particular for the more perceptual SSIM and LPIPS metrics. This is showcased in the rightmost example of Figure 5 as well, where this variant of our method lacks the capacity to model the individ-

Table 3: Ablations and analysis. We put in bold any result better than our unsupervised exemplar-free method.

Method	Counting		Full Image			Prototype		
	MAE↓	RMSE↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Ours	14.11	26.93	23.76	0.737	0.242	19.40	0.700	0.353
w/o appearance head	14.99	27.31	20.58	0.606	0.384	18.53	0.673	0.432
w/o sparsity loss	15.62	28.24	22.57	0.689	0.298	18.89	0.691	0.389
<i>Optimization ablation</i>								
direct optimization	27.54	47.75	15.84	0.524	0.493	16.88	0.632	0.490
w/o pretraining	18.81	38.13	21.08	0.689	0.330	18.29	0.655	0.469
w/o crop-based optimization	15.25	28.85	17.45	0.516	0.511	16.61	0.627	0.496
w/o image refinement	14.11	26.93	20.07	0.576	0.443	18.47	0.665	0.468
<i>Pretraining analysis</i>								
pretraining only	15.25	28.85	14.86	0.396	0.789	15.48	0.603	0.659
unsup. pretraining	15.48	28.63	22.67	0.666	0.280	19.21	0.733	0.362
<i>Leveraging annotations</i>								
+ $\mathcal{L}_{\text{ex}}$	14.03	27.18	22.11	0.701	0.282	21.54	0.751	0.291
+ $\mathcal{L}_{\text{ex}} + \mathcal{L}_{\text{loc}}$	7.07	11.26	20.00	0.644	0.343	20.28	0.729	0.330

ual elements. Removing the sparsity loss, relying only on image reconstruction, degrades slightly not only counting accuracy but also our modeling metrics.

Optimization Ablations show that all of our optimization steps are actually important for our final results. Directly optimizing full image reconstruction, without either pretraining or crop-based training leads to very bad results for all our metrics. Removing either pretraining, crop-based optimization, or image refinement leads to smaller performance drops. Pretraining is particularly critical for counting, and crop-based optimization for modeling quality, while full image refinement slightly boosts modeling.

Pretraining Pretraining alone leads to relatively good counting metrics, but very bad modeling. This is because our pretraining model only provides very coarse localization of blob-like objects, as can be seen in Figure 5. Interestingly, not using bounding box and count supervision during synthetic pretraining has limited effect on the performances. This suggests that it might be possible to pretrain the model on real datasets, if they were available, which might in turn lead to better modeling results when applying the model to real images.

Annotations Leveraging additional supervision from annotations naturally boosts our results. The exemplar loss has a limited impact on counting and full image modeling metrics, but strongly improves prototype modeling. Adding the localization loss strongly boosts counting performance but it hurts full image reconstruction modeling, since fewer prototypes are used to reconstruct the background. Additionally, as shown in the lego example of Figure 5, where the lack of supervision leads to a decomposition of the red exemplar into two pieces, this setting is particularly helpful for ambiguities due to self-similarity.

Finally, we find that using a uniform prior over object scale leads to marginal degradation in reconstruction metrics and to a degradation of counting metrics with increases of 4.53 MAE and 6.1 RMSE. On 90% of the dataset, counting MAE increases by only 1.3, with the remaining 10% (smallest objects) accounting for most of the gap. The scale prior is needed only in the extreme regime of very small objects.

Image Manipulation. We demonstrate our model’s interpretability through image manipulation (Figure 6; top: input, bottom: manipulation), editing the rotation, color, and appearance of individual instances.



Fig. 6: Pose and Appearance manipulation.

5 Discussion and Conclusion

Limitations Our method assumes a non-cluttered background and elements of moderate aspect ratio, and relies on a coarse scale prior, which limits its ability to handle extreme intra-image scale variation. Because grouping is driven by visual similarity in image space rather than semantics, instances that are semantically related but visually dissimilar are not recovered. Finally, part-whole ambiguity, a problem shared with counting models operating under similar conditions, persists as inherent to data with intra-image repetition.

Conclusion Revisiting a fundamental computer vision problem with modern optimization tools, we have demonstrated an analysis-by-synthesis approach that enables completely bottom-up discovery and modeling of repeated elements in real images. Our resulting image model not only improves over baselines but is particularly explainable, providing position, color, and a 1D appearance feature for each element instance. More broadly, our results suggest that meaningful object models can emerge directly from pixels. The learned decomposition also opens a promising direction for object-centric learning: intra-image repetition can serve as a free self-supervision signal.

Acknowledgments

This work was funded by the EIDA (ANR-22-CE38-0014) and ERC DISCOVER (No. 101076028) projects, and used HPC resources from GENCI-IDRIS (No. AD011015521). It was also supported by the French government via Hi! PARIS under France 2030 (ANR-23-IACL-0005). We thank Ségolène Albouy, Raphaël Baena, Sonat Baltacı, Paul Kervegan, Marta Lopez, Fei Meng, and Malamatenia Vlachou for figure feedback, fruitful discussions, and interesting datasets.

References

1. Ahuja, N., Todorovic, S.: Extracting texels in 2.1 d natural textures. In: 2007 IEEE 11th International Conference on Computer Vision. pp. 1–8. IEEE (2007)
2. Amini-Naieni, N., Amini-Naieni, K., Han, T., Zisserman, A.: Open-world text-specified object counting. arXiv preprint arXiv:2306.01851 (2023)
3. Amini-Naieni, N., Han, T., Zisserman, A.: Countgd: Multi-modal open-world counting. *Advances in Neural Information Processing Systems* **37**, 48810–48837 (2024)
4. Amir, S., Gandselman, Y., Bagon, S., Dekel, T.: Deep vit features as dense visual descriptors. In: ECCV Workshops (2022)
5. Barel, N., Weber, R.S., Muallem, N., Finder, S.E., Freifeld, O.: Spacejam: a lightweight and regularization-free method for fast joint alignment of images. In: European Conference on Computer Vision. pp. 180–197. Springer (2024)
6. Biza, O., Van Steenkiste, S., Sajjadi, M.S., Elsayed, G.F., Mahendran, A., Kipf, T.: Invariant slot attention: Object discovery with slot-centric reference frames. arXiv preprint arXiv:2302.04973 (2023)
7. Burgess, C.P., Matthey, L., Watters, N., Kabra, R., Higgins, I., Botvinick, M., Lerchner, A.: Monet: Unsupervised scene decomposition and representation. arXiv preprint arXiv:1901.11390 (2019)
8. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: European conference on computer vision. pp. 213–229. Springer (2020)
9. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 9650–9660 (2021)
10. Chen, B., Zhi, T., Hebert, M., Narasimhan, S.G.: Learning continuous implicit representation for near-periodic patterns. In: European Conference on Computer Vision. pp. 529–546. Springer (2022)
11. Cheng, M.M., Zhang, F.L., Mitra, N.J., Huang, X., Hu, S.M.: Repfinder: finding approximately repeated scene elements for image editing. *ACM transactions on graphics (TOG)* **29**(4), 1–8 (2010)
12. Cheng, T., Ma, W.C., Guan, K., Torralba, A., Wang, S.: Structure from duplicates: Neural inverse graphics from a pile of objects. arXiv preprint arXiv:2401.05236 (2024)
13. Crawford, E., Pineau, J.: Spatially invariant unsupervised object detection with convolutional neural networks. In: AAAI (2019)
14. Doubek, P., Matas, J., Perdoch, M., Chum, O.: Image matching and retrieval by repetitive patterns. In: 2010 20th international conference on pattern recognition. pp. 3195–3198. IEEE (2010)
15. Efros, A.A., Leung, T.K.: Texture synthesis by non-parametric sampling. In: Proceedings of the seventh IEEE international conference on computer vision. vol. 2, pp. 1033–1038. IEEE (1999)
16. Eslami, S., Heess, N., Weber, T., Tassa, Y., Szepesvari, D., Hinton, G.E., et al.: Attend, infer, repeat: Fast scene understanding with generative models. *Advances in neural information processing systems* **29** (2016)
17. Eunchan Jo, Dahyun Kang, S.K.Y.C., Cho, M.: Few-shot pattern detection via template matching and regression. In: International Conference on Computer Vision (ICCV) (2025)

18. Gadelha, M., Maji, S., Wang, R.: 3d shape induction from 2d views of multiple objects. In: 2017 international conference on 3d vision (3DV). pp. 402–411. IEEE (2017)
19. Glasner, D., Bagon, S., Irani, M.: Super-resolution from a single image. In: 2009 IEEE 12th international conference on computer vision. pp. 349–356. IEEE (2009)
20. Greff, K., Kaufman, R.L., Kabra, R., Watters, N., Burgess, C., Zoran, D., Matthey, L., Botvinick, M., Lerchner, A.: Multi-object representation learning with iterative variational inference. In: International conference on machine learning. pp. 2424–2433. PMLR (2019)
21. Gupta, K., Jampani, V., Esteves, C., Shrivastava, A., Makadia, A., Snavely, N., Kar, A.: Asic: Aligning sparse in-the-wild image collections. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 4134–4145 (2023)
22. Hays, J., Leordeanu, M., Efros, A.A., Liu, Y.: Discovering texture regularity as a higher-order correspondence problem. In: European Conference on Computer Vision. pp. 522–535. Springer (2006)
23. Hirsch, O., Weber, R.S., Ifergane, S., Freifeld, O.: Fastjam: a fast joint alignment model for images. arXiv preprint arXiv:2510.22842 (2025)
24. Hobley, M., Prisacariu, V.: Abc easy as 123: A blind counter for exemplar-free multi-class class-agnostic counting. In: European Conference on Computer Vision. pp. 304–319. Springer (2024)
25. Huang, G.B., Jain, V., Learned-Miller, E.: Unsupervised joint alignment of complex images. In: 2007 IEEE 11th international conference on computer vision. pp. 1–8. IEEE (2007)
26. Huang, J.B., Singh, A., Ahuja, N.: Single image super-resolution from transformed self-exemplars. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5197–5206 (2015)
27. Huang, Z., Dai, M., Zhang, Y., Zhang, J., Shan, H.: Point segment and count: A generalized framework for object counting. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 17067–17076 (2024)
28. Jaderberg, M., Simonyan, K., Zisserman, A., et al.: Spatial transformer networks. *Advances in neural information processing systems* **28** (2015)
29. Jia, B., Liu, Y., Huang, S.: Improving Object-centric Learning with Query Optimization. In: International Conference on Learning Representations (ICLR) (2023)
30. Karazija, L., Laina, I., Rupprecht, C.: Clevrtex: A texture-rich benchmark for unsupervised multi-object segmentation. arXiv preprint arXiv:2111.10265 (2021)
31. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., et al.: Segment anything. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 4015–4026 (2023)
32. Knobel, L., Han, T., Asano, Y.M.: Learning to count without annotations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 22924–22934 (2024)
33. Kosiorek, A., Sabour, S., Teh, Y.W., Hinton, G.E.: Stacked capsule autoencoders. *Advances in neural information processing systems* **32** (2019)
34. Kulikov, V., Yadin, S., Kleiner, M., Michaeli, T.: SinDDM: A single image denoising diffusion model. In: Proc. ICML (2023)
35. Learned-Miller, E.G.: Data driven image models through continuous joint alignment. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **28**(2), 236–250 (2005)
36. Lempitsky, V., Zisserman, A.: Learning to count objects in images. *Advances in neural information processing systems* **23** (2010)

37. Leung, T., Malik, J.: Detecting, localizing and grouping repeated scene elements from an image. In: European Conference on Computer Vision. pp. 546–555. Springer (1996)
38. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European conference on computer vision. pp. 740–755. Springer (2014)
39. Lin, Z., Wu, Y.F., Peri, S.V., Sun, W., Singh, G., Deng, F., Jiang, J., Ahn, S.: Space: Unsupervised object-oriented scene representation via spatial attention and decomposition. arXiv preprint arXiv:2001.02407 (2020)
40. Liu, C., Zhong, Y., Zisserman, A., Xie, W.: Countr: Transformer-based generalised visual counting. arXiv preprint arXiv:2208.13721 (2022)
41. Liu, S., Ng, T.T., Sunkavalli, K., Do, M.N., Shechtman, E., Carr, N.: Patchmatch-based automatic lattice detection for near-regular textures. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (December 2015)
42. Locatello, F., Weissenborn, D., Unterthiner, T., Mahendran, A., Heigold, G., Uszkoreit, J., Dosovitskiy, A., Kipf, T.: Object-centric learning with slot attention. *Advances in neural information processing systems* **33**, 11525–11538 (2020)
43. Lu, E., Xie, W., Zisserman, A.: Class-agnostic counting. In: Asian conference on computer vision. pp. 669–684. Springer (2018)
44. Monnier, T., Vincent, E., Ponce, J., Aubry, M.: Unsupervised Layered Image Decomposition into Object Prototypes. In: ICCV (2021)
45. Mu, J., De Mello, S., Yu, Z., Vasconcelos, N., Wang, X., Kautz, J., Liu, S.: Coordgan: Self-supervised dense correspondences emerge from gans. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10011–10020 (2022)
46. Müller, P., Zeng, G., Wonka, P., Van Gool, L.: Image-based procedural modeling of facades. *ACM Trans. Graph.* **26**(3), 85 (2007)
47. Nguyen, T., Pham, C., Nguyen, K., Hoai, M.: Few-shot object counting and detection. In: European Conference on Computer Vision. pp. 348–365. Springer (2022)
48. Ofri-Amar, D., Geyer, M., Kasten, Y., Dekel, T.: Neural congealing: Aligning images to a joint semantic atlas. In: CVPR (2023)
49. Peebles, W., Zhu, J.Y., Zhang, R., Torralba, A., Efros, A.A., Shechtman, E.: Gansupervised dense visual alignment. In: CVPR. pp. 13470–13481 (2022)
50. Pelhan, J., Lukezic, A., Zavrtanik, V., Kristan, M.: A novel unified architecture for low-shot counting by detection and segmentation. *Advances in Neural Information Processing Systems* **37**, 66260–66282 (2024)
51. Ranjan, V., Nguyen, M.H.: Exemplar free class agnostic counting. In: Proceedings of the Asian Conference on Computer Vision. pp. 3121–3137 (2022)
52. Ranjan, V., Sharma, U., Nguyen, T., Hoai, M.: Learning to count everything. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 3394–3403 (2021)
53. Sabour, S., Frosst, N., Hinton, G.E.: Dynamic routing between capsules. *Advances in neural information processing systems* **30** (2017)
54. Schaffalitzky, F., Zisserman, A.: Geometric grouping of repeated elements within images. In: Shape, contour and grouping in computer vision, pp. 165–181. Springer (1999)
55. Seitzer, M., Horn, M., Zadaianchuk, A., Zietlow, D., Xiao, T., Simon-Gabriel, C.J., He, T., Zhang, Z., Schölkopf, B., Brox, T., et al.: Bridging the gap to real-world object-centric learning. arXiv preprint arXiv:2209.14860 (2022)

56. Shaham, T.R., Dekel, T., Michaeli, T.: Singan: Learning a generative model from a single natural image. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 4570–4580 (2019)
57. Shocher, A., Bagon, S., Isola, P., Irani, M.: InGAN: Capturing and remapping the “DNA” of a natural image. In: Proc. ICCV (2019)
58. Siméoni, O., Vo, H.V., Seitzer, M., Baldassarre, F., Oquab, M., Jose, C., Khalidov, V., Szafraniec, M., Yi, S., Ramamonjisoa, M., et al.: Dinov3. arXiv preprint arXiv:2508.10104 (2025)
59. Singh, G., Deng, F., Ahn, S.: Illiterate dall-e learns to compose. arXiv preprint arXiv:2110.11405 (2021)
60. Smirnov, D., Gharbi, M., Fisher, M., Guizilini, V., Efros, A., Solomon, J.M.: Marionette: Self-supervised sprite learning. *Advances in Neural Information Processing Systems* **34**, 5494–5505 (2021)
61. Đukić, N., Lukežič, A., Zavrtanik, V., Kristan, M.: A low-shot object counting network with iterative prototype adaptation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 18872–18881 (2023)
62. Villa-Vásquez, J.F., Pedersoli, M.: Unsupervised object discovery: A comprehensive survey and unified taxonomy. arXiv preprint arXiv:2411.00868 (2024)
63. Violante, N., Meuleman, A., Gauthier, A., Durand, F., Groueix, T., Drettakis, G.: Splat and replace: 3d reconstruction with repetitive elements. In: Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers. pp. 1–12 (2025)
64. Wang, W., Bao, J., Zhou, W., Chen, D., Chen, D., Yuan, L., Li, H.: Sindiffusion: Learning a diffusion model from a single natural image. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **47**(5), 3412–3423 (2025)
65. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing* **13**(4), 600–612 (2004)
66. Xu, J., Hu, T., Hu, X., Zhou, L., Cao, S., Zhang, M., Lu, H.: Plant taxonomy meets plant counting: A fine-grained, taxonomic dataset for counting hundreds of plant species. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 167–177 (2026)
67. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 586–595 (2018)
68. Zhang, Y., Wu, S., Snavely, N., Wu, J.: Seeing a rose in five thousand ways. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 962–971 (2023)
69. Zhao, L., Yuan, Z., Zhang, Y., Wu, S., Wu, J.: Learning generative 3d scene layouts from a single image (2024)