Search for Concepts: Discovering Visual Concepts Using Direct Optimization

BMVC 2022 Submission # 810

S.1 Convolution-based Grid Search

When optimizing the element parameters in a given image $\tilde{I}$, we maximize the normalized cross-correlation [1] instead of minimizing the L_2 distance:

$$\arg\max_{\theta_1,\ldots,\theta_n} \frac{\sum_p (I\tilde{I})_p}{\sqrt{\sum_p I_p^2}\ \sqrt{\sum_p \tilde{I}_p^2}} \quad \text{with } \tilde{I} = h(e_{\mathcal{V}}(\theta_1),\ldots,e_{\mathcal{V}}(\theta_n)), \tag{S1}$$

where the products are element-wise and p is a two-dimensional pixel index. We can then formulate a grid search over translation parameters as a convolution. In the single-element case, $\tilde{I} = E_1$, and we can rewrite Eq. S1 as:

$$\arg\max_{\theta_1} \frac{(I \circledast \hat{E}_1)_{t_1}}{\sqrt{\sum_p I_p^2}\ \sqrt{(\mathbf{1} \circledast \hat{E}_1^2)_{t_1}}}, \tag{S2}$$

where $\hat{E}$ is the element E with translation parameters $t \in \theta$ zeroed out, so the transformed visual concept is at the origin and acts as a convolution kernel, and $\mathbf{1}$ is an image of all-ones. The result of the convolution is an image where each pixel corresponds to the correlation at one 2D translation t.

In the general case with multiple elements, we optimize the parameters of one element at a time. At each element, we need to account for occlusions from previous elements when performing the convolution. Re-writing the alpha-composite defined in Eq. 1 to isolate the contribution of a single element E_i we get:

$$\tilde{I} = \tilde{I}_1^{i-1} + E_i O_1^{i-1} + (1 - E_i^{\mathrm{A}}) O_1^{i-1} \tilde{I}_{i+1}^{n} \tag{S3}$$
$$\text{with } \tilde{I}_a^b := \sum_{i=a}^{b} E_i O_a^{i-1} \text{ and } O_a^j := \prod_{i=a}^{j} (1 - E_i^{\mathrm{A}}).$$

Intuitively, $\tilde{I}_a^b$ is the partial composite of the elements a through b, without taking into account other elements, and O_a^j is the occlusion effect of elements a through j on the following elements. Only the second and third terms depend on L_i, where the second term is the occluded contribution of L_i to the image and the third term describes the occlusion caused by

L_i on the following elements. Substituting into Eq. S1, and using convolutions for searching over translations we arrive at:

$$\begin{aligned}\arg\max_{\theta_i} & \frac{\sum_p (IC_1)_p + (IC_2 \circledast \hat{E}_i + IC_3 \circledast (1-\hat{E}_i^A))_{t_i}}{\sqrt{\sum_p I_p^2}\ \sqrt{\sum_p (C_1^2)_p + (\mathcal{N}_i)_{t_i}}} \\ \text{with } \mathcal{N}_i &= C_2^2 \circledast \hat{E}_i^2 + C_3^2 \circledast (1-\hat{E}_i^A)^2 \\ &\quad + 2C_1C_2 \circledast \hat{E}_i + 2C_1C_3 \circledast (1-\hat{E}_i^A) \\ &\quad + 2C_2C_3 \circledast \hat{E}_i (1-\hat{E}_i^A) \\ \text{and } C_1 &= \tilde{I}_1^{i-1} \quad C_2 = O_1^{i-1} \quad C_3 = O_1^{i-1} \tilde{I}_{i+1}^n,\end{aligned} \tag{S4}$$

which we solve as efficient update step for element parameters. Section S.2 provides the derivation.

S.2 Layer Parameter Objective

We derive the objective for the layer parameter optimization with alpha compositing and convolutions defined in Eq. 7 of the main paper. The objective is obtained in two steps: (i) we substitute the alpha composite defined in Eq. 6 into layer parameter objective in Eq. 4, and (ii) we search over position parameters using convolutions, analogous to Eq. 5. For clarity, we first define:

$$C_1 = \tilde{I}_1^{i-1} \quad C_2 = O_1^{i-1} \quad C_3 = O_1^{i-1} \tilde{I}_{i+1}^n,$$

so that Eq. 6 becomes:

$$\tilde{I} = C_1 + E_i C_2 + (1 - E_i^{\mathrm{A}}) C_3. \tag{S5}$$

Substituting Eq. S5 into Eq. 4 and restricting the optimization over the parameters θ_i of layer E_i:

$$\begin{aligned}\arg\max_{\theta_i} & \frac{\sum_p (IC_1)_p + \sum_p (IC_2E_i)_p + \sum_p (IC_3(1-E_i^A))_p}{\sqrt{\sum_p I_p^2}\ \sqrt{\sum_p (C_1^2)_p + \mathcal{N}_i'}} \\ \text{with } \mathcal{N}_i' &= \sum_p (C_2^2 E_i^2)_p \\ &\quad + \sum_p (C_3^2 (1-E_i^A)^2)_p \\ &\quad + 2\sum_p (C_1C_2E_i)_p \\ &\quad + 2\sum_p (C_1C_3(1-E_i^A))_p \\ &\quad + 2\sum_p (C_2C_3E_i(1-E_i^A))_p.\end{aligned} \tag{S6}$$

Finally, exchanging any term of the form $\sum_p (CE_i)_p$ with a convolution $(C \circledast \hat{E}_i)_{t_i}$ using as kernel the layer with with zeroed translation parameters $\hat{E}_i$, we arrive at Eq. 7 of the paper:

$$\arg\max_{\theta_i} \frac{\sum_p (IC_1)_p + (IC_2 \circledast \hat{E}_i + IC_3 \circledast (1 - \hat{E}_i^A))_{t_i}}{\sqrt{\sum_p I_p^2}\ \sqrt{\sum_p (C_1^2)_p + (\mathcal{N}_i)_{t_i}}}$$

$$\begin{aligned}\text{with } \mathcal{N}_i = {} & C_2^2 \circledast \hat{E}_i^2 \\ & + \sum C_3^2 - 2C_3^2 \circledast \hat{E}_i^A + C_3^2 \circledast (\hat{E}_i^A)^2 \\ & + 2C_1C_2 \circledast \hat{E}_i \\ & + 2\sum C_1C_3 - C_1C_3 \circledast \hat{E}_i^A \\ & + 2C_2C_3 \circledast \hat{E}_i - 2C_2C_3 \circledast \hat{E}_i\hat{E}_i^A.\end{aligned}$$

S.3 Datasets and Hyperparameters details

The Tetrominoes dataset contains 60k images with 3 randomly rotated and positioned Tetris blocks. The Multi-dSprites dataset consists of 60k images, each showing 2-3 shapes picked from a dictionary and placed at random locations, possibly with occlusions. We use a discretized variant of the color version and the binarized version of this dataset. We discretize the color version to eight colors, so that the colors can be discovered as composite concepts. In addition, we create a variant of this dataset that we name *Multi-dSprites adversarial*, where shapes can only appear in three discrete locations on the canvas. This intuitively simple dataset highlights shortcomings in existing methods. The Clevr6 dataset consists of 35k rendered 3D scenes with each scene is an arrangement of up to six geometric primitives with various colors and materials. Also since MarioNette data is not available, we obtain comparable data through screen captures of Space Invaders, Super Mario Bros, and I, Robot (in the latter we use only lower-case letters).

Optimization details. We learn visual concepts using AdaDelta [4] with a learning-rate of 1.0 and a batch size of 8 on a single GPU. We have tested different optimizers but found that our pipeline was robust to the choice of optimizer. A more detailed comparison of optimizers is in the supplemental. We do not use any learning rate schedulers or warm-up techniques.

In Table S1 we list the hyperparameters we used for each experiment.

S.4 Thresholds for Splitting and Removing Concepts

The threshold for removing a concept V_j is $N_j < 0.25\frac{n|\mathcal{I}|}{|\mathcal{V}|}$, where N_j is the number of instances of concept v_j, and n is the number of elements per image. The threshold for splitting a concept is $N_j > 0.25\frac{n|\mathcal{I}|}{|\mathcal{V}|}$, in addition to a threshold on the reconstruction error $\mathcal{E}_j < 0.95$. The reconstruction error $\mathcal{E}_j$ of a single visual concept V_j over the whole dataset is isolated as:

$$\mathcal{E}_j(\mathcal{V}, \Theta) := \sum_{\{i,k|\tau_i^k = j\}} \frac{\sum_p (M_i^k I \tilde{I} M_i^k)_p}{\sqrt{\sum_p (M_i^k I)_p^2}\sqrt{\sum_p (M_i^k \tilde{I})_p^2}} \quad \text{with } M_i^k = (E_i^A)^k (O_1^{i-1})^k. \tag{S7}$$

Table S1: **Hyperparameters.** In this table present the hyperparameters used of different experiments that are used to generate the results presented in the paper. Translation values are greater than the input image size if we allow partial placement of a concept. The first and second columns show the initial number of concepts and the max number of concepts . The rotation column indicates the number of values uniformly sampled in $[0, 2\pi]$.

	Init	Max	Translation	Rotation	nColors	nlayers	Concept Size
Tetris	1	10	35x35	4	6	3	19
Multi dSprites bin	4	22	64x64	40	1	3	33
Multi dSprites Adv	1	3	64x64	40	1	3	33
Clevr6	8	32	64x64	1	8	6	31
MNIST(128)	8	128	28x28	1	1	4	13
MNIST(512)	8	512	28x28	1	1	4	13
MNIST Sum(128)	8	128	41x41	1	1	4	13
MNIST Sum(512)	8	512	41x41	1	1	4	13
GTSRB	6	64	28x28	1	1	6	31
Xmas Pattern	2	2	85x85	1	1	45	21

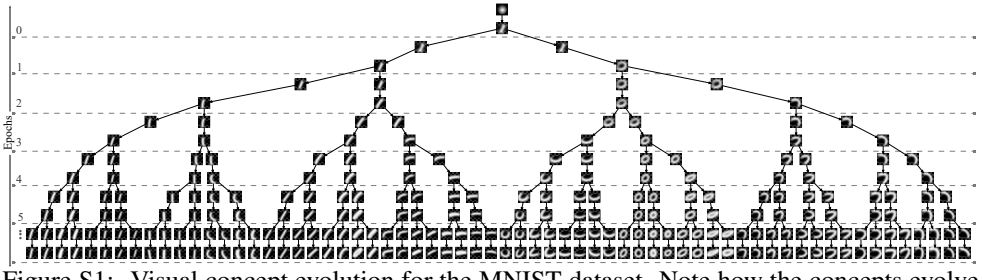

Figure S1: Visual concept evolution for the MNIST dataset. Note how the concepts evolve over optimization epochs by specializing to subtle stroke variations.

The sum is over all elements E_i^k in all images I^k that use the visual concept V_j and M_i^k is a mask that zeros out parts of the image that do not have contributions from element E_i^k.

S.5 Concept Evolution Graph

As described in the main paper, the number of concepts is determined by a concept evolution approach where concepts can be split or removed. This results in a tree of visual concepts that is grown during optimization. An illustration of such a tree is shown in Figure S1.

S.6 Learning Visual Concepts in 3D Scenes

We use our framework to learn 3D visual concepts $\mathcal{V}$ from multi-view renders of 3D scenes. In this setting, our visual concepts are 3D voxel patches instead of 2D image patches, where each voxel describes a density value. The transformation function T translates these patches and samples them at the global voxel grid of the 3D scene to obtain 3D elements E_i: $E_i = T(V_{\tau_i}, t_i)$, where t_i is a 3D translation and V_{τ_i}, a visual concept, is a 3D voxel patch containing density values. For the image formation function h, we use the orthographic projection to

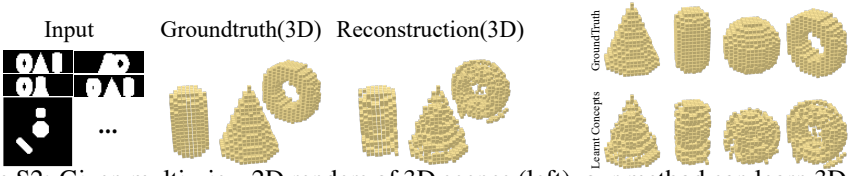

Figure S2: Given multi-view 2D renders of 3D scenes (left), our method can learn 3D visual concepts that can be used to reconstruct the 3D scenes.

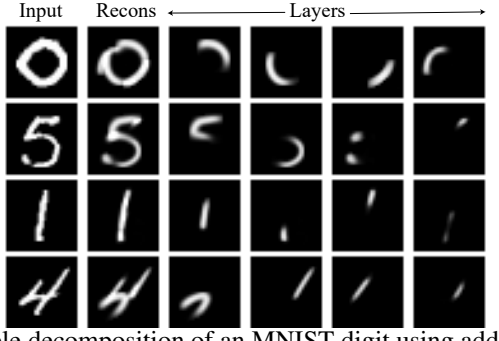

Figure S3: Example decomposition of an MNIST digit using additive compositing.

accumulate the voxel densities along the viewing direction d, giving us the image $\tilde{I}$:

$$\tilde{I} = h(E_1, \ldots, E_n | d) := \sum_l \left(S_{jkl}^2 \prod_{m=1}^{l-1} (1 - S_{jkm}) \right), \text{ with } S = \min(1, R_d\left(\sum_i E_i\right)), \tag{S8}$$

where S is the scene voxel grid, computed as a sum over all element voxel grids, rotated and re-sampled by R_d to align the last axis (indexed by l) with the viewing direction d. We clamp densities in S to have a maximum of value 1. The orthographic projection accumulates voxel densities along the viewing direction and the product attenuates voxel contributions by the occlusion effect of voxels closer to the viewer, indicated by smaller index l. We optimize element parameters t_i and visual concepts $\mathcal{V}$ to maximize the normalized cross-correlation between all reconstructed renders $\tilde{I}$ and all ground truth renders I of a 3D scene dataset, as described in the previous sections, but without using convolutions to speed up the search for element parameters. The element parameters t_i of a scene are optimized using all views of that scene. In our experiments, we use 20 views: back/front, left/right, top, and 3 additional rotations of these views by $\pi/8$ radians about the top/bottom axis.

Evaluation: We create a synthetic dataset of 16 3D scenes by randomly selecting 3 shapes from a dictionary of 4 ground truth shapes and placing them at random positions on a ground plane. We render 20 different views of each scene to form the input dataset, and use the image formation described in Section S.6 to learn 3D concepts. Reconstruction results and a comparison of the learned concepts to the ground truth is provided in Figure S2.

S.7 Additive Compositing

As mentioned in the main paper, we can use our approach with different compositing functions. Here, we present details and experiments with *additive compositing* instead of the

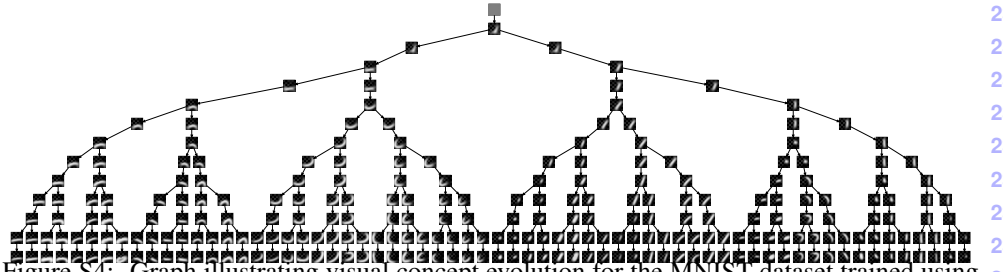

Figure S4: Graph illustrating visual concept evolution for the MNIST dataset trained using additive compositing. (Please use digital zoom for details.) Note how the method learns smaller concepts compared to alpha compositing, as shown in Figure 4 of the main paper.

Table S2: **Additive compositing results.** MSE reconstruction loss of EMNIST letters using additive compositing. The visual concepts are trained on the MNIST digits dataset with an additive compositing function. We show results for two dictionary sizes, $m = 128$ and $m = 512$.

	MNIST(Train)	EMNIST(Test)
Ours Additive (128)	0.0163	0.0215
Ours Additive (512)	0.0137	0.0186

alpha-compositing used for the 2D results in the main paper and the other sections of the supplementary. For additive compositing, we replace Eq. 6 with a sum over layers:

$$\tilde{I} = C + E_i \quad \text{with } C = \sum_{j \neq i} E_j. \tag{S9}$$

The layer parameter optimization objective defined by Eq. 7 for alpha composting then becomes the following for additive compositing:

$$\arg\max_{\theta_i} \frac{\sum_p (IC)_p + (I \circledast \hat{E}_i)_{t_i}}{\sqrt{\sum_p I_p^2}\ \sqrt{\sum_p (C^2)_p + (\mathbf{1} \circledast \hat{E}_i^2)_{t_i} + (2C \circledast \hat{E}_i)_{t_i}}}. \tag{S10}$$

Quantiative results measuring the MSE reconstruction loss for the cross-dataset generalization experiment are shown in Table S2 for two dictionary sizes. Note that reconstruction errors are slightly higher with additive compositing when compared to the corresponding results with alpha-compositing in Table 3 of the main paper. Figures S5 and S3 show the learned concepts and a decomposition example, respectively. Figure S4 show the evolution graph of the visual concepts.

S.8 Additional Results

We also submit an additional set of uncurated results along with the supplementary (see the contents of the zip file). We included the first b images of the respective datasets, with b being the batch-size. For each image, we show the reconstruction and the decomposed layers. Note that these results are not post-processed, so the layer decomposition may also contain layers that are completely occluded by other layers.

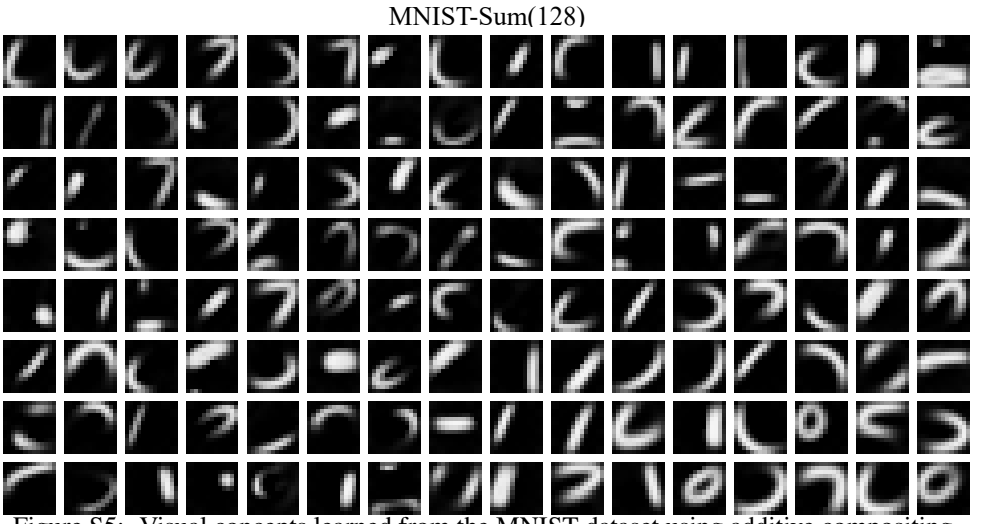

Figure S5: Visual concepts learned from the MNIST dataset using additive compositing.

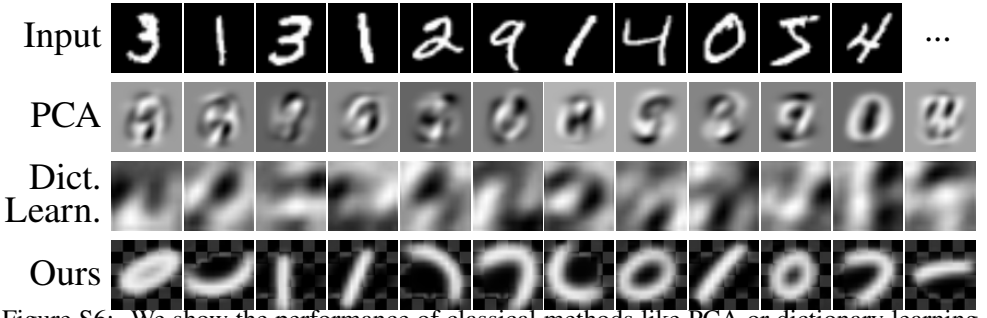

Figure S6: We show the performance of classical methods like PCA or dictionary learning to obtain dominant modes of an image dataset. Note that extracted modes are not very interpretable in terms of compositionality. In contrast, our method uses a search-and-learn strategy to extract a dictionary of interpretable *visual concepts*, in this case the underlying strokes. Checkered patterns denote transparent pixels.

S.9 Comparison with Traditional methods

We compare our approach to traditional unsupervised decomposition methods like PCA or dictionary learning in Figure S6.

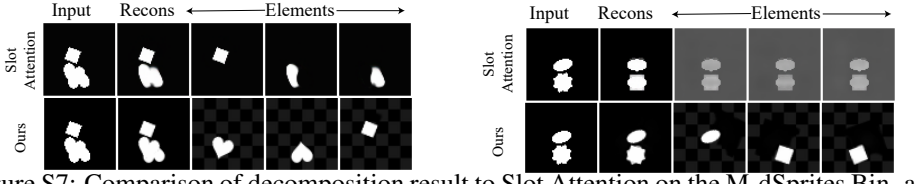

Figure S7: Comparison of decomposition result to Slot Attention on the M-dSprites Bin. and M-dSprites Adv. datasets. Slot Attention is not regularized by a global dictionary, resulting in incorrect decompositions. Ours are more interpretable.

S.10 Ablation of Optimizers

We demonstrate the robustness of our approach to the choice of optimizer and learning rates. In Figure S8, we show the MSE training loss curve for different optimizers and learning rates on the pattern images shared along with this supplementary material. We show SGD, Adam [2], Adadelta [4], and RMSprop optimizers with learning rates in $[0.001, 1]$. Our pipeline converges for all t learning rates take longer to converge

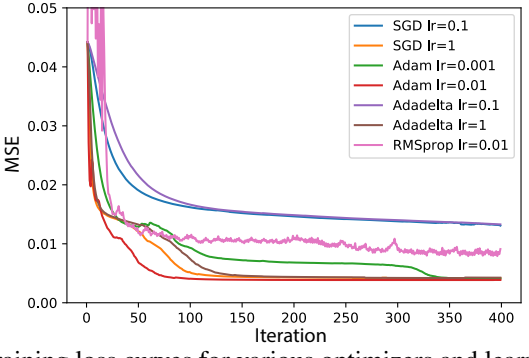

Figure S8: MSE training loss curves for various optimizers and learning rates. Note that all optimizers converge with the right learning rate.

S.11 Ablation of the Visual Concept Evolution

In Fig S9, we demonstrate the necessity for the visual concept evolution in our pipeline. Without evolution, a single optimized concept may average multiple similar ground truth concepts. Using evolution, we allow this average concept to split into multiple more specialized child-concepts, that each approximate fewer ground truth concepts. After a few evolution steps, each leaf concept eventually represents a single ground truth concept.

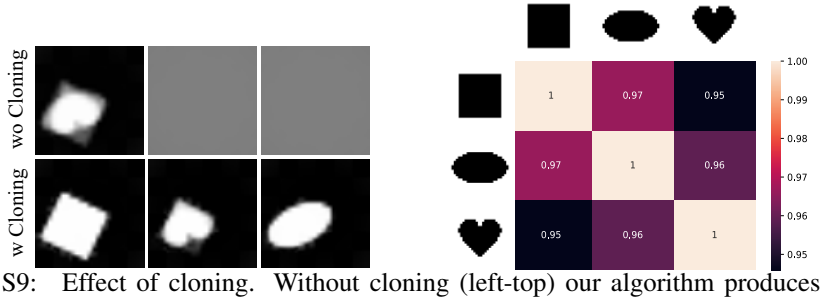

Figure S9: Effect of cloning. Without cloning (left-top) our algorithm produces an average visual concept as a proxy to collectively represent similar shapes. However, with cloning (left-bottom) our algorithm can specialize to pickup subtle differences among the visual concepts, producing individualized concepts. On the right, we show the confusion matrix to show how similar the ground truth visual concepts are, with 1 denoting perfect similarity.

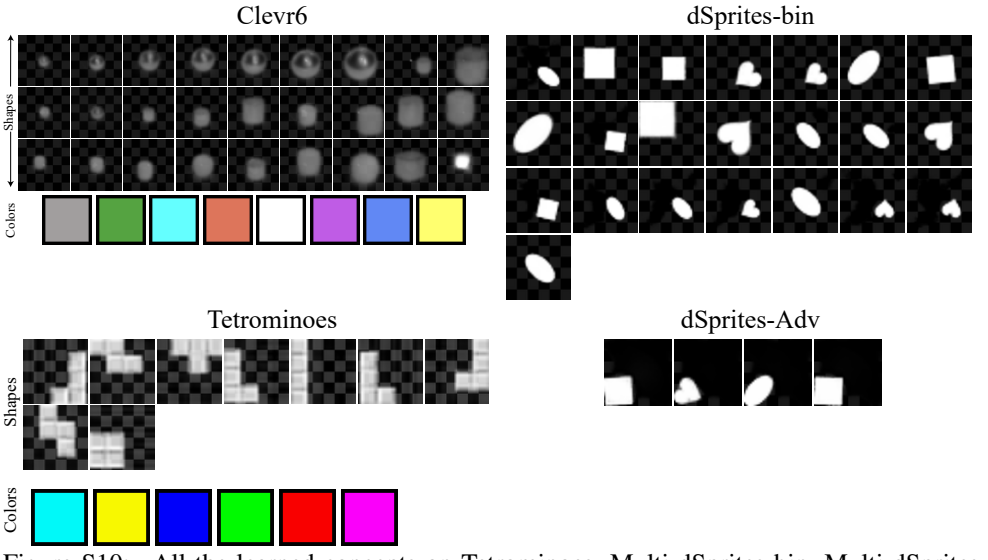

Figure S10: All the learned concepts on Tetrominoes, Multi-dSprites bin, Multi-dSprites Adv and Clevr6 datasets.

S.12 ARI calculation

The Adjusted Rand Index (ARI) measures the similarity between two clusterings. We use it to compare the decomposition found by our method to the ground truth decomposition. In images without occlusions or with visual concepts that have the same constant color, any layer ordering results in the same reconstructed image, thus the layer order is ambiguous. In cases with ambiguous ordering, we select the layer ordering that gives the highest ARI score.

S.13 Full visual concepts

In Figures S10 and S11, we show all visual concepts learned by our method from each 2D dataset used in the main paper (all visual concepts of the 3D dataset are shown in Figure 10 of the main paper). Additionally, we show visual concepts obtained from the GTSRB traffic sign dataset [3] in Figure S11, bottom.

S.14 dSprites Adv. Dataset Details

To create the dSprites Adversarial dataset, we place two or three visual concepts in each image. Each concept is placed at one of three pre-defined locations on the canvas (without overlaps). All concepts have the same scale and a random rotation. Figure S12 shows samples of the dataset.

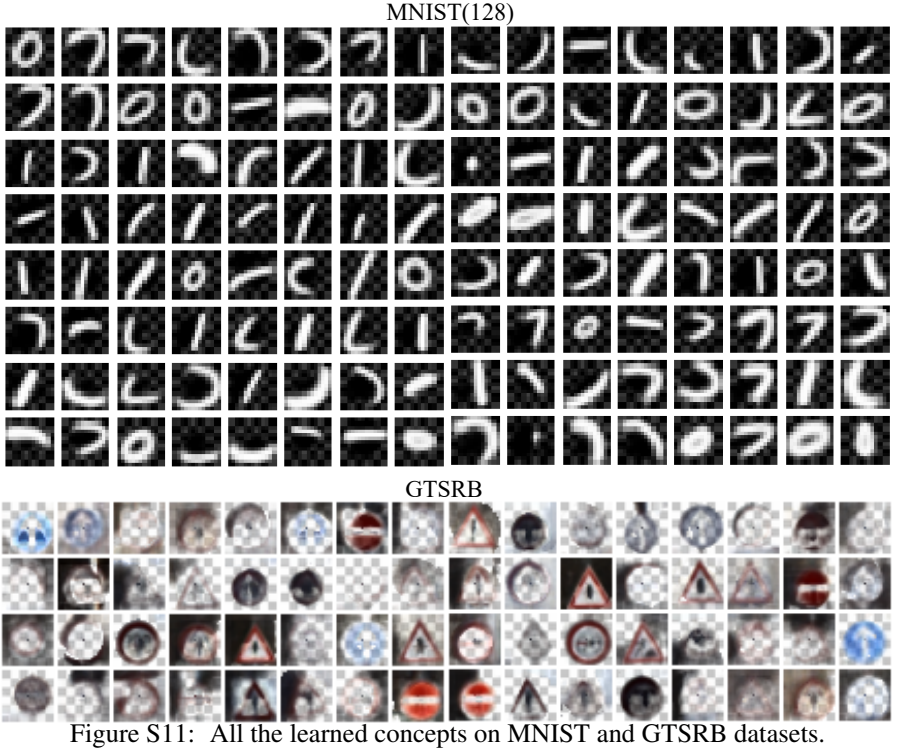

Figure S11: All the learned concepts on MNIST and GTSRB datasets.

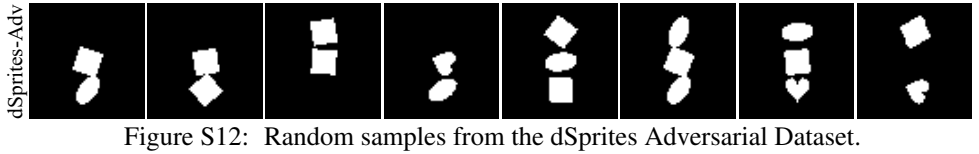

Figure S12: Random samples from the dSprites Adversarial Dataset.

S.15 Background Handling

When using alpha-compositing, we treat the background as a special layer L_n that is locked to the back of the layer stack (i.e. it is occluded by all other layers when using alpha-compositing) and does not have layer parameters. The background is represented by a special visual concept V_m that is only used by the background layer and is initialized with a constant value of 0.5 in all pixels. During optimization of a given image $\tilde{I}$, we optimize the background visual concept before the other concepts or layers, to make sure the other layers don't represent parts of the background.

S.16 3D Scene Reconstruction segmentation

We also measure the quality of our decompositions by comparing the 2D projections of the segmented 3D scene to a known ground truth. We achieve an ARI of 99.3% on this task.

S.17 Video

In the supplementary material, we include a video that visualizes the optimization process of our method, showing the optimized visual concepts in each iteration and the layer segmentation of the reconstructed image. For clarity, we demonstrate the optimization on a dataset consisting of a single image with multiple repeating visual concepts.

S.18 Code

We include a development version of our code in the supplemental that can be used to reproduce the results.

References

[1] Artan Kaso. Computation of the normalized cross-correlation by fast fourier transform. *PLOS ONE*, 13(9):1–16, 09 2018. doi: 10.1371/journal.pone.0203434. URL https://doi.org/10.1371/journal.pone.0203434.

[2] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[3] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural networks*, 32:323–332, 2012.

[4] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.