RELATIONAL ABSTRACTIONS FOR SPATIAL REASONING WITH DIFFUSION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion models excel at image synthesis, but they remain limited in their ability to reliably satisfy structured spatial reasoning constraints. In conditional data distribution modeling tasks with implicit logical structure, such as puzzles defined by visible clues paired with consistent solutions, state-of-the-art generative models tend to approximate pixel-space distributions without learning the underlying logical rules required for inference. To address this limitation, we present a novel framework for spatial reasoning with diffusion models that leverages unsupervised object discovery and abstractions of object relations. We show that the relational knowledge derived from object-centric representations enriches diffusion models with structural primitives, allowing them to effectively guide the generative representation space during both training and inference, and enabling conditional image generation that satisfies reasoning constraints. Additionally, we introduce a large-scale generative spatial reasoning benchmark with four datasets inspired by human-solvable puzzles. Our results show that relational abstractions significantly improve reasoning capabilities of diffusion models on a variety of complex reasoning tasks, while enabling robust generalization in out-of-distribution settings.

1 INTRODUCTION

Diffusion models, including denoising-based (Sohl-Dickstein et al., 2015; Ho et al., 2020), score-based (Song & Ermon, 2019; Song et al., 2021), and flow-based formulations (Lipman et al., 2023), have emerged as powerful generative paradigms for learning and high-fidelity sampling from complex data distributions with remarkable success. Importantly, these models implicitly capture structural and semantic regularities of data, suggesting a promising foundation for reasoning in continuous domains. While reasoning in language has been extensively explored with large language models, how diffusion models can perform spatial reasoning remains an open question.

Spatial reasoning requires detecting visual patterns, inducing hypotheses, and systematically applying rules to solve complex problems (Raven, 1938; Fleuret et al., 2011). Building on the success of deep learning on image data, prior work has explored neural approaches to Abstract Visual Reasoning, where models are trained on tasks such as Raven's Progressive Matrices (RPM) (Barrett et al., 2018) and Bongard-style problems (Nie et al., 2020). These methods traditionally frame visual reasoning as a discriminative task, requiring models to make a categorical choice rather than synthesizing a solution. This formulation has been criticized for allowing models to exploit superficial visual heuristics rather than demonstrating genuine, systematic rule inference (Webb et al., 2021; Ye et al., 2025). To address this limitation, recent work has shifted toward framing spatial reasoning as a generative problem. The ARC challenge (Chollet, 2019) in particular, requires models to synthesize a complete output grid from scratch. However, existing approaches typically rely on symbolic program synthesis (Chollet et al., 2024), while their application to pixel-level image synthesis remains challenging. More recently, diffusion models have been explored in structured visual reasoning tasks, such as Sudoku-style image generation, with a particular focus on adapting inference-time sampling strategies to improve solution consistency (Wewer et al., 2025; Lee et al., 2025). In contrast, we argue that effective generative reasoning over complex visual domains requires moving beyond refined pixel-level reconstruction to incorporate higher-level context abstractions within the generation process.

Contributions: We introduce RDM (*Relational Abstractions for Diffusion Models*), a novel framework for spatial reasoning with diffusion models that leverages unsupervised object discovery via

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Slot Attention (Locatello et al., 2020) and abstractions of object relations. We show that the relational knowledge derived from object-centric embeddings via Abstractors (Mondal et al., 2024; Altabaa et al., 2024) enhances diffusion models with contextual primitives for both training and inference, enabling high-fidelity image synthesis while adhering to the reasoning constraints. We introduce a large-scale benchmark consisting of four new challenging datasets inspired by popular human-solvable puzzles for studying spatial reasoning as a conditional generative modeling task, where underlying logical rules must be inferred from training data. Each dataset contains up to one million paired examples, covering grid-based image completion and multi-context image generation tasks across multiple levels of abstraction. Our approach improves performance on a variety of complex reasoning tasks, by up to $3\times$ on the challenging multi-context reasoning problems, while demonstrating strong out-of-distribution generalization.

2 Related Work

Diffusion models: Diffusion models (Ho et al., 2020; Song et al., 2021) learn to model a data distribution $q_0(\mathbf{x})$ via a forward process that progressively transforms it into a Gaussian prior $q_1 = \mathcal{N}(0, \mathbf{I})$, and train a neural network p_θ to reverse this transformation. The forward process can be formulated either as a discrete-time Markov chain of Gaussian noising transitions, where the reverse process utilizes a neural network that predicts the added noise at each step t as in denoising models (Ho et al., 2020), or as a continuous-time stochastic differential equation (SDE) where noise is injected continuously (Song et al., 2021) and the reverse-time SDE dynamics are governed by the score function $\nabla_{\mathbf{x}} \log q_t(\mathbf{x})$, which is estimated by a neural network.

In the continuous-time setting, an alternative formulation based on ordinary differential equations (ODEs) leads to *continuous normalizing flows* (CNFs), which define the forward process via conditional probability paths $p_t(\mathbf{x} \mid \mathbf{x}_0)$ for samples $\mathbf{x}_0 \sim q_0(\mathbf{x})$. Specifically, Gaussian paths $p_t(\mathbf{x} \mid \mathbf{x}_0) = \mathcal{N}(\mathbf{x} \mid \mu_t(\mathbf{x}_0), \sigma_t(\mathbf{x}_0)^2\mathbf{I})$ across time $t \in [0, 1]$ interpolate between the data distribution and a Gaussian prior by setting $\mu_0(\mathbf{x}_0) = \mathbf{x}_0$, $\sigma_0(\mathbf{x}_0) = \sigma_{\min}$, $\mu_1(\mathbf{x}_0) = 0$ and $\sigma_1(\mathbf{x}_0) = 1$. Samples along this path can be expressed via the *flow* $\phi_t(\boldsymbol{\epsilon}, \mathbf{x}_0) = \mu_t(\mathbf{x}_0) + \sigma_t(\mathbf{x}_0)\boldsymbol{\epsilon}$, a differentiable transformation that maps noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$ to samples from $p_t(\mathbf{x} \mid \mathbf{x}_0)$. The corresponding conditional *vector field* $v_t(\mathbf{x} \mid \mathbf{x}_0) = \sigma_t'(\mathbf{x}_0)/\sigma_t(\mathbf{x}_0)(\mathbf{x} - \mu_t(\mathbf{x}_0)) + \mu_t'(\mathbf{x}_0)$ is approximated by a neural network $p_\theta(\mathbf{x}_t, t)$. In our experiments, we focus on CNFs trained via *flow matching* (Lipman et al., 2023) (see §B.3 for details), although our framework is applicable to all families of diffusion models.

Spatial reasoning with generative models: Several works explore generative models for visual reasoning. In RPM, this requires synthesizing a missing panel from scratch, requiring complete understanding of the underlying relational rules. Earlier solutions primarily used deep latent variable models (Pekar et al., 2020) with structured latent spaces (Shi et al., 2024), to abstract explicit rules on concepts. However, these solvers often require handcrafted inductive biases rooted in RPM-specific rules to facilitate abstraction and panel generation. More recently, diffusion models have been explored for spatial reasoning with sampling-time solutions (Wewer et al., 2025; Lee et al., 2025). Notably, Wewer et al. (2025) proposed SRM based on diffusion forcing (Chen et al., 2024a), to perform spatial reasoning as conditional image generation. It proposes an incremental sampling strategy based on patch-level uncertainty estimation, exploiting the grid structure of Sudoku-style tasks with state-of-the-art performance, but is inherently tailored to structured grids.

Object discovery: Object discovery is rooted in the hypothesis that human intelligence decomposes complex visual scenes into discrete, interacting entities (Battaglia et al., 2018; Greff et al., 2019), and studies aim to instill a similar inductive bias into neural networks by moving beyond dense feature maps to interpretable object representations. Locatello et al. (2020) aligned this concept into modern neural networks via *Slot Attention*, a framework that disentangles the feature and positional information of visual components into high-dimensional embeddings, named *slots*, representing the objects captured in the scene. Object discovery has been also adopted for discriminative reasoning tasks, where relational modules aggregate information across disentangled slots to model object interactions. Methods such as CoRelNet (Kerg et al., 2022), OCRA (Webb et al., 2023) and Abstractors (Altabaa et al., 2024) leverage object-centric designs in which learned relations condition a classification head, exploiting the ability of slots to disentangle scene factors under discriminative objectives. Recent object-centric diffusion models (Wu et al., 2023; Jiang et al., 2023) extend these to generative settings, with a focus on downstream vision tasks and compositional image synthesis capabilities.

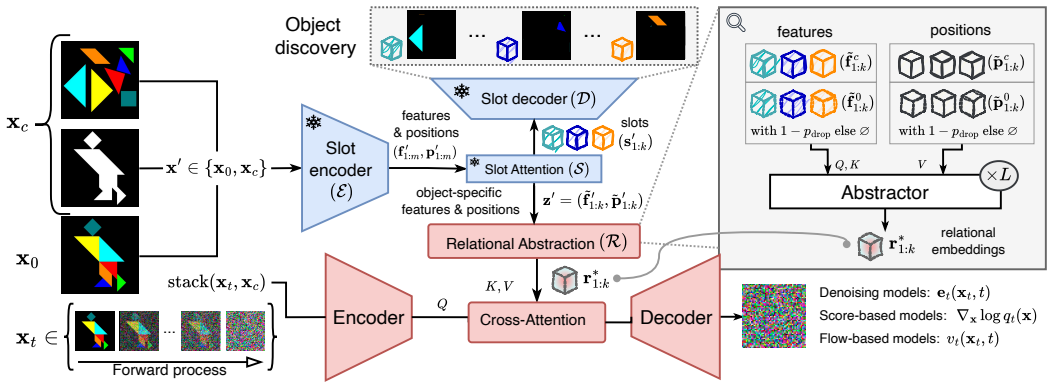


Figure 1: Illustration of RDM with *Discoverer* and the *Reasoner* blocks. During training, the clues ($\mathbf{x}_c$) and solutions ($\mathbf{x}_0$) are individually fed to the pretrained frozen (❄) slot encoder to obtain the corresponding feature ($\mathbf{f}'_{1:m}$) and positional ($\mathbf{p}'_{1:m}$) embeddings. Slot Attention yields the slots ($\mathbf{s}'_{1:k}$) for reconstruction by the slot decoder, and object-specific features and positions, $\mathbf{z}' = (\tilde{\mathbf{f}}'_{1:k}, \tilde{\mathbf{p}}'_{1:k})$ for relational abstractions. The relational abstraction module consists of L abstractor layers that optimize the relational embeddings (), which are introduced into the U-Net bottleneck through a cross-attention layer. Since the solution is unavailable at inference time, its disentangled representations ($\tilde{\mathbf{f}}^0_{1:k}, \tilde{\mathbf{p}}^0_{1:k}$) are suppressed by a default learnable embedding ($\varnothing$) with probability p_{drop}.

3 Relational Abstractions for Diffusion Models

We formalize the spatial reasoning task as modeling the conditional distribution $q(\mathbf{x}_0|\mathbf{x}_c)$, where $\mathbf{x}_c$ denotes a provided problem clue, and $\mathbf{x}_0$ represents a valid solution satisfying all logical constraints.

Our proposed framework comprises two models. Our first model, the *discoverer*, is a latent variable model p_ϕ designed to capture the objects present in the samples from $q(\mathbf{x}')$, where $\mathbf{x}' \in \{\mathbf{x}_0, \mathbf{x}_c\}$, via disentangled latent representations $\mathbf{z}'$. Our second model, the *reasoner*, is a diffusion model p_θ that learns to reverse the trajectory from a Gaussian prior to the target distribution $q(\mathbf{x}_0|\mathbf{x}_c)$, conditioned on the latents $\mathbf{z}'$ provided by the discoverer. The integration of object-centric latent variables $\mathbf{z}'$ is motivated by the inherent difficulty of learning $q(\mathbf{x}_0|\mathbf{x}_c)$ through a simple parameterization of p_θ. We propose to enrich the generative modeling process with relational abstractions inferred from object-centric representations obtained via an unsupervised object discovery model $p_\phi(\mathbf{z}'|\mathbf{x}')$.

3.1 The *Discoverer*: Unsupervised Slot-based Object Discovery

The objective of the *discoverer* is to extract informative latent representations $\mathbf{z}'$ via the encoding $p_\phi(\mathbf{z}'|\mathbf{x}')$ of a latent variable model, while simultaneously disentangling the constituent objects within $\mathbf{x}_0$ and $\mathbf{x}_c$. We use *slot attention* (Locatello et al., 2020) to achieve this, which enables unsupervised learning of disentangled object embeddings from visual data via a slot-based autoencoder.

Slot attention: The slot encoder maps input visual features into a shared latent space, where k trainable vectors (the *slots*) compete to explain parts of the input via an iterative attention process. These k slots are then passed to a spatial broadcast decoder to reconstruct the input $\mathbf{x}'$. By enforcing this reconstruction bottleneck, the model learns to partition the scene into k distinct, permutation-invariant latent variables that capture the essential objects for the downstream reasoning task.

Let $\mathbf{x} \in \mathbb{R}^{\text{H}\times\text{W}\times\text{C}}$ denote an input image, and $(\mathcal{E}, \mathcal{S}, \mathcal{D})$ denote the slot encoder, slot attention and slot decoder modules of the discoverer. The slot encoder $\mathcal{E} : \mathbb{R}^{\text{H}\times\text{W}\times\text{C}} \to (\mathbb{R}^{m\times d}, \mathbb{R}^{m\times d})$ is a feature extractor that yields m features ($\mathbf{f}_{1:m}$) and positional embeddings ($\mathbf{p}_{1:m}$) in a d-dimensional latent space. The slot attention module $\mathcal{S} : (\mathbb{R}^{m\times d}, \mathbb{R}^{m\times d}) \to \mathbb{R}^{k\times d}$ learns to iteratively update k randomly initialized slots $\mathbf{s}_{1:k}$, using η recurrent cross-attention iterations over $\mathbf{f}_{1:m}$ and $\mathbf{p}_{1:m}$ to bind object-centric information to the slots. The final attention matrix is then used to compute feature- and position-specific embeddings ($\tilde{\mathbf{f}}_{1:k}$ and $\tilde{\mathbf{p}}_{1:k}$) for each slot, which will be considered as $\mathbf{z} = (\tilde{\mathbf{f}}_{1:k}; \tilde{\mathbf{p}}_{1:k})$ (see Algorithm 2). The slot decoder $\mathcal{D} : \mathbb{R}^{k\times d} \to \mathbb{R}^{\text{H}\times\text{W}\times\text{C}}$ reconstructs the input by processing each slot independently through transposed convolutions to produce slot-specific

reconstructions $\hat{\mathbf{x}}_i \in \mathbb{R}^{\mathtt{H}\times\mathtt{W}\times\mathtt{C}}$ and associated spatial masks $\boldsymbol{\alpha}_i$ for $i = 1, \ldots, k$. The final image $\hat{\mathbf{x}}$ is computed as the weighted sum of $\{\hat{\mathbf{x}}_i\}_{i=1}^k$ with $\text{softmax}_{i=1\ldots k}(\boldsymbol{\alpha}_i)$ (see §B.1 for further details).

Scaling the number of slots: One of the main limitations of the original slot decoder is the computational overhead of the latent bottleneck when k increases. Previous works have typically restricted $k \leq 10$ due to the memory overhead of performing independent high-resolution convolutions and subsequent weighted averaging for every slot (Singh et al., 2022a;b; Mondal et al., 2023).

To scale our discoverer to a large number of slots, we introduce the *scalable slot decoder* in Algorithm 1 that significantly reduces the memory demand during the backward pass. It relies on an additional learnable positional grid $\mathbf{g}$ that activates specific slots with specific locations in the bottleneck of the model. Rather than decoding each slot into a full-resolution image before weighting, our scalable decoder first estimates which slots are relevant to specific locations by computing the dot product between the positional grid $\mathbf{g}$ and the slot embeddings $\mathbf{s}_{1:k}$, yielding $\tilde{\boldsymbol{\alpha}}$ after softmax normalization. Subsequently, the slot information is weighted by $\tilde{\boldsymbol{\alpha}}$ and projected onto the grid, producing a single feature map $\tilde{\mathbf{s}}$, where slot information is isolated on specific positions of the grid. This aggregated representation is then passed through the remaining decoder layers to generate a unified reconstruction of the input image. Effectively, introducing this learnable positional grid creates a computationally more efficient bottleneck during optimization.

Algorithm 1: Scalable slot decoder

Input : $\mathbf{s}_{1:k} \in \mathbb{R}^{k\times d}$, $\mathbf{g} \in \mathbb{R}^{\mathtt{h}\times\mathtt{w}\times d}$
$\tilde{\boldsymbol{\alpha}} \leftarrow \text{softmax}(\mathbf{g} \cdot \mathbf{s}_{1:k}^\top)$ `#[h,w,k]`
$\tilde{\mathbf{s}} \leftarrow \sum_{i=1}^k \tilde{\boldsymbol{\alpha}}_i \cdot \mathbf{s}_i$ `#[h,w,d]`
return $\mathcal{D}(\tilde{\mathbf{s}})$

3.2 The *Reasoner*: Relational Object Abstractions as Diffusion Guidance

The *reasoner* is a diffusion model conditioned on the clue $\mathbf{x}_c$ and the embeddings $\mathbf{z}' = (\tilde{\mathbf{f}}'_{1:k}; \tilde{\mathbf{p}}'_{1:k})$ obtained by the discoverer, where $\prime \in \{0, c\}$. We formalize generative modeling of $q(\mathbf{x}_0|\mathbf{x}_c)$ through a diffusion guidance framework (Ho & Salimans, 2021) based on embeddings learned by a relational abstraction module ($\mathcal{R}$). We use a single parameterization of p_θ to learn both the *solution-guided* distribution $p_\theta(\mathbf{x}_0|\mathbf{x}_c, \mathcal{R}(\mathbf{z}^c, \mathbf{z}^0))$, and *unguided* distribution $p_\theta(\mathbf{x}_0|\mathbf{x}_c, \mathcal{R}(\mathbf{z}^c))$. This resembles the training setup of classifier-free guidance (CFG). Unlike CFG, however, RDM does not combine the guided and unguided distributions for sampling, since the solution is unavailable at inference time.

Relational abstraction: This module $\mathcal{R} : (\mathbb{R}^{K\times d}, \mathbb{R}^{K\times d}) \to \mathbb{R}^{K\times d}$ processes the latents $(\tilde{\mathbf{f}}^*_{1:k}, \tilde{\mathbf{p}}^*_{1:k})$ with *Slot Abstractors* (Mondal et al., 2024), where $* = (c, 0)$ and $K \geq 2k$ varies with the number of clues and solutions. It consists of interleaved cross-attention, self-attention and feed-forward blocks with residual connections, designed to hierarchically capture inter-slot semantics and positional relations by sequentially refining the positional signal $(\tilde{\mathbf{p}}^*_{1:k})$ with query-key feature projections $(\tilde{\mathbf{f}}^*_{1:k})$ into *relational embeddings* $\mathbf{r}^*_{1:k}$ (see §B.2 for details). After L abstractor blocks, this module returns $\mathbf{r}^*_{1:k} = \mathcal{R}(\tilde{\mathbf{f}}^*_{1:k}, \tilde{\mathbf{p}}^*_{1:k})$, encoding object relations. This ensures that the generative process will be guided not only by individual object features, but also by their contextual interactions.

Conditional generative modeling: During training, the disentangled representations of the solution $(\mathbf{z}^0)$ are nullified with probability p_{drop} to simulate the unguided distribution $p_\theta(\mathbf{x}_0|\mathbf{x}_c, \mathcal{R}(\mathbf{z}^c))$ used at inference; and maintained with probability $(1 - p_{\text{drop}})$ to learn the solution-guided distribution $p_\theta(\mathbf{x}_0|\mathbf{x}_c, \mathcal{R}(\mathbf{z}^c, \mathbf{z}^0))$. By modeling both distributions with a single parameterization p_θ, the reasoner learns: (1) the relations between the disentangled factors of the clue and the solution $(\mathbf{r}^*_{1:k})$ and (2) to generate the solution $\mathbf{x}_0$ at inference-time by relying only on the clue information $(\mathbf{x}_c, \mathbf{z}^c)$. Training can be either based on noise prediction (Ho et al., 2020), score matching (Song et al., 2021), or flow matching (Lipman et al., 2023) (which we use in our experiments, see §B.3 for details).

4 Spatial Reasoning Benchmark

We present a novel benchmark comprising four datasets with up to one million images each, inspired by classic logic puzzles, where underlying logical rules are implicit in the conditional data distribution and must be inferred from training data. We categorize them into two paradigms: (i) *grid-based image completion* tasks: Akari and Coldoku, where reasoning is framed as inpainting under global and local spatial constraints, and (ii) *multi-context image generation*: Tangram and LogicFace, where a latent rule must be inferred from context images to synthesize a consistent solution.

4.1 Grid-based Image Completion Tasks

Akari dataset: Akari is a binary-determination puzzle played on a rectangular grid of *empty cells* and *walls*. The goal is to place light bulbs in empty cells such that every cell is illuminated while ensuring no two bulbs share a line of sight. Walls obstruct light and may be numbered with a digit (0-4), specifying the exact number of bulbs in its four orthogonally adjacent cells. Figure 2 illustrates these constraints, where the *clue* (a) defines wall and digit placements. A valid solution (b) illuminates all cells without bulb interference, satisfying the wall clues. Invalid configurations (c) fail if cells remain unlit or bulbs conflict. We generated a dataset of one million 11×11 Akari puzzles with randomized wall configurations. Unlike traditional Akari, which requires a unique solution, we allow for multiple valid bulb arrangements to evaluate the model's ability to generate novel, correct solutions. Additional details on how this dataset is generated can be found in §A.1.

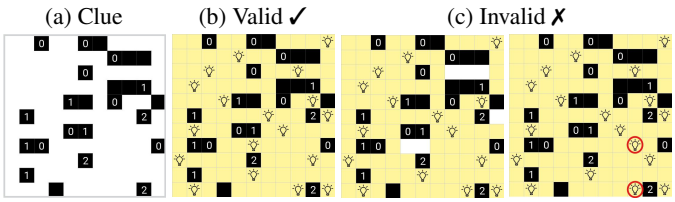


Figure 2: Akari as a grid-based image completion puzzle.

Coldoku (*Colored Sudoku*) dataset: Sudoku is a logic-based puzzle played on a 9×9 grid, divided into nine 3×3 subgrids. The objective is to fill the grid with digits from 1 to 9 such that each column, row, and subgrid does not contain repeated digits. The puzzle setter provides a partially completed grid as the *clue*, from which a well-posed solution can be derived. Recent works (Wewer et al., 2025) used MNIST-based variants of this task for generative modeling. We extend this in Coldoku, a colored variant of this puzzle, where we again have a 9×9 grid satisfying the Sudoku rules (Figure 3a), and add an extra layer of complexity by coloring the digits such that there are no repeated colors in each row, column and subgrid (Figures 3b). Coldoku requires solving two different Sudokus simultaneously, one considering the digits and the other considering colors. More details can be found in §A.2.

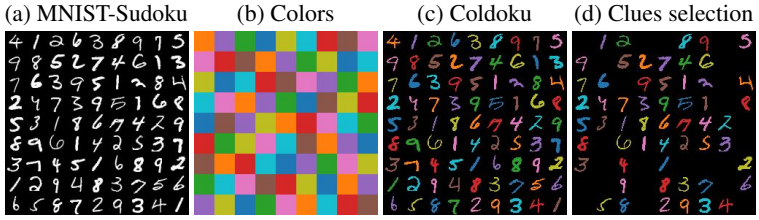


Figure 3: Coldoku puzzle, which combines Sudoku with a color arrangement that also satisfies Sudoku rules. *Tableau 10* palette is used, which is designed to have a large perceptual difference between colors.

4.2 Multi-context Image Generation Tasks

Tangram dataset: Tangram is a dissection puzzle consisting of seven flat shapes (*tans*): a square (□), a parallelogram (▱), two small triangles ($\triangle^S \times 2$), a medium triangle ($\triangle^M$) and two large triangles ($\triangle^L \times 2$), which can be arranged to form figures without overlapping. The goal is to assemble all the pieces to match a given *silhouette*, such that no piece overlaps with other, and all pieces of the puzzle are used (Figure 4). Tangram requires reasoning over shapes, sizes, rotations and displacements of the pieces to project the final solution, reflecting forms of geometric reasoning studied in humans (Bohning & Althouse, 1997; Wong et al., 2019). We rely on the Kilogram dataset (Ji et al., 2022) consisting of 1,013 uncolored Tangram solutions, and introduce the vision task of generating a valid solution (Figure 4c) by using an input pattern (Figure 4b) and a set of colored tans (Figure 4a). We generate different tan views and solutions by rotating the puzzle and randomly assigning distinct colors to tans (see §A.3 for details, as well as a discussion on Tangram task complexity in §C.2).

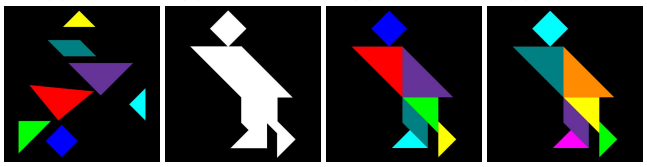


Figure 4: Tangram example. A valid solution uses all pieces matching the colors and shapes, given the (a) and (b) as *clues*.

LogicFace (*Logical Operators on Facial Features*) dataset: We present a challenging logic-based generative modeling problem that requires abstraction and reasoning over real images. Specifically, LogicFace defines a conditional generative modeling task for human faces, where certain facial features are combined through hidden logical operators that must be inferred from training examples. Figure 5 shows an example, where each data point is composed of three images from CelebA-HQ (Lee et al., 2020) associated with a subset of κ binary features retrieved from its original annotations. We consider the first two images (*clues*) with the arguments A and B, and the third as the *solution* S. Each image has a binary feature vector $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{s} \in \{0,1\}^\kappa$, and the dataset contains triples for which, for a given element-wise logical operator f_j, the relation $f_j(\boldsymbol{a}_j, \boldsymbol{b}_j) = \boldsymbol{s}_j$ holds. We set $\kappa = 4$ for the attributes to represent the feature vector of each image and a pool of 4 logical operators, each one assigned to one feature (see §A.4 for details).



Figure 5: LogicFace example with 4 features: *smiling* (`OR`), *male* (`AND`), *glasses* (`XOR`), *young* (`IMPLIES`). Features from A and B are combined in S via the corresponding logic operators (e.g., *glasses*: $A = 0$, $B = 1$, $S = 1$, satisfying `XOR`).

5 Experiments

Model configurations: For the discoverer, we use a convolutional autoencoder where the number of slots (k), slot dimension (d) and bottleneck resolution ($\mathtt{h}, \mathtt{w}$) are adapted to the specific task (see Table 3). The slot encoder maps the input into m feature and positional embeddings. Slot attention performs $\eta = 3$ recurrent iterations and feeds k slots to the decoder. For Tangram and LogicFace we chose $k = 7$ and $k = 6$, respectively. For Coldoku and Akari, where the true number of objects scales with the number of cells, i.e., 81 and 121 respectively, we propose two configurations: one where k coincides with the number of cells, hence requiring reconstruction with our scalable slot decoder (Algorithm 1), and the other with $k = 16$, such that each slot integrates information from multiple cells. For experiments with $k \in \{6, 7, 16\}$ we use the original slot decoder (Locatello et al., 2020). For the reasoner, we use a U-Net architecture with an attention-based bottleneck, where relational embeddings learned in $L = 6$ abstractor layers are incorporated via cross-attention (Chen et al., 2024b). To study the impact of model size, we conduct experiments with small (`S`), medium (`M`) and large (`L`) backbone U-Net architectures. Further architectural details are explained in §B.

Conditioning on relational abstractions: Our framework can guide diffusion models via abstraction and conditioning on information from two types of inputs during training: the clue (via $\mathbf{z}^c$) and the solution (via $\mathbf{z}^0$). During training, we drop the condition $\mathbf{z}^0$ with probability $p_{\text{drop}} = 0.5$[1] to disable the conditioning derived from the solution, since $\mathbf{x}_0$ will not be available at inference time.

We explore three different conditioning settings (alongside $\mathbf{x}_c$), where relational embeddings are injected via cross-attention in the denoiser bottleneck: (i) *clue*: conditioning on $\mathcal{R}(\mathbf{z}^c)$ during training and inference, (ii) *solution*: $\mathcal{R}(\mathbf{z}^0)$ guides training and is dropped with p_{drop}, and inference is unconditioned on any relational embedding, and (iii) *full*: both $\mathbf{z}^c$ and $\mathbf{z}^0$ (dropped with p_{drop}) are used during training, and only $\mathbf{z}^c$ is used at inference time.

Training details: We pretrained the slot autoencoder independently for each task based on a simple reconstruction loss (§B.1), providing images of clues and solutions as training samples. We assess the quality of the reconstruction and thus the slot embeddings on a validation set (see §C for the final results and disentanglement visualizations). For the reasoner, we use flow matching to create the diffusion trajectory as detailed in §B.3, where the mean and variance change linearly with t. We perform deterministic sampling from the conditional distribution by solving the reverse-time ODE using τ discretization steps. All generative models were trained with the same optimization settings.

Baseline comparisons: We use a capacity-matched U-Net as baseline (Base) by augmenting the hidden dimensions to match the same model size as RDM, accommodating for the additional parameters

[1]We observed that smaller p_{drop} values degrade estimation of the unconditional distribution of interest at inference time, although recommended in previous works (Ho & Salimans, 2021). We found the best balance at $p_{\text{drop}} = 0.5$, hypothesizing that it allows to evenly learn both the *solution-guided* and *unguided* distributions.

Table 1: Experimental results on Akari and Coldoku datasets.

		Akari						Coldoku								
		in-dist.		*low*		*high*		*Easy (50-60)*			*Medium (40-50)*			*Hard (30-40)*		
		BLB	ACC	BLB	ACC	BLB	ACC	DGT	CLR	ACC	DGT	CLR	ACC	DGT	CLR	ACC
Base		86.02	80.04	43.62	39.48	92.22	88.92	93.53	94.73	89.00	61.81	78.56	49.06	11.38	25.31	3.88
SRM (Wewer et al., 2025)																
parallel		35.82	18.80	1.60	0.94	54.18	33.34	94.67	96.74	91.06	61.02	78.53	51.01	12.17	25.91	4.00
sequential		00.00	00.00	–	–	–	–	93.05	96.70	92.37	**82.21**	83.61	68.35	**40.95**	36.66	**20.78**
RDM*																
16 slots	*clue*	88.30	82.06	**88.30**	72.48	92.90	88.94	98.06	98.50	96.66	78.18	82.20	65.62	21.40	25.64	7.76
	solution	86.10	78.68	82.21	70.73	92.51	87.96	96.80	96.18	93.16	71.80	67.36	50.68	16.64	12.74	3.34
	full	**89.44**	80.42	83.32	**75.06**	**93.80**	87.48	**98.14**	**99.30**	**97.48**	80.12	**88.22**	**71.64**	22.76	**36.88**	12.58
>80 slots	*clue*	88.88	**83.12**	83.10	54.58	93.62	**89.56**	96.82	94.88	91.98	79.18	63.40	47.12	16.46	11.76	2.86
	solution	86.36	80.28	81.12	74.58	92.37	88.94	96.82	95.26	92.38	72.04	67.24	50.86	17.24	12.94	3.36
	full	85.84	77.34	77.30	56.20	92.76	87.40	91.12	99.18	90.36	51.86	86.26	46.14	6.46	32.06	2.94

*Results of RDM$^{[S]}$ for Akari, and RDM$^{[L]}$ with $\tau = 10$ for Coldoku.

of the *Discoverer* and the Abstractor stack. The Base model is trained to estimate $q(\mathbf{x}_0|\mathbf{x}_c)$ also via flow matching, with the main difference of not incorporating relational abstractions. We also trained this architecture with diffusion forcing (Chen et al., 2024a) for comparisons to SRM (Wewer et al., 2025) with *parallel* and *sequential* sampling strategies, where the latter additionally requires learning uncertainty at training time to perform patch-based autoregressive denoising. We also performed experiments on MNIST-Sudoku and Counting Polygons benchmarks for completeness (Wewer et al., 2025), and ablations with non-object-centric feature guidance (details and results in §C.2.1).

Model evaluations: We designed task-specific neural evaluators (see §A for details), and used 5,000 held-out samples per task that served as test sets. We use the overall accuracy (ACC) as the main metric in all datasets. For Coldoku, we also report individual digit (DGT) and color (CLR) accuracies across three difficulty levels: *Easy* (50-60 clues), *Medium* (40-50 clues) and *Hard* (30-40 clues). For Akari, we also measure the bulb accuracy (BLB), which only evaluates the correctness of bulb allocations by ignoring the numbered walls, and evaluate both in an *in-distribution (ID)* regime where the wall density ratio matches the training set, and *out-of-distribution (OoD)* regimes with extreme wall densities: *low* (5-10%) and *high* (35-40%). For Tangram, we measure SHAPE accuracy, which only measures the correct partition of the silhouette, and the f_1-score per tan type. For LogicFace, we measure accuracy at feature level (proportion of correctly generated attributes). We report all results with $\tau = 100$ unless specified otherwise (see §C for $\tau = 10$ and varying model sizes).

6 Experimental Results

We demonstrate the performance of our models on the proposed benchmark in Tables 1 and 2, where RDM outperforms the baselines in Akari, Tangram and LogicFace, while only lagging behind SRM in the hardest level of Coldoku. We present extended results and output examples in §C.

Results on Akari: RDM consistently outperforms baseline models across all distribution regimes and demonstrates superior generalization capabilities under the most challenging (*low*) regime. Under the in-distribution (ID) evaluation, RDM achieves 83.12 accuracy with 121 slots, compared to 80.04 for standard diffusion. We observe a performance margin between 16 and 121 slots (*clue*: 83.12 vs 82.06), potentially because the slot count allows the discoverer to freely learn a bijective mapping between slots and cells, enhancing the reasoner to better exploit bulb placement. In the *high* regime, all models benefit from an increased wall ratio, which bounds the solution space, localizes constraints, and reduces ambiguity. Conversely, the *low* regime imposes harder reasoning demands as the solution space expands beyond the training distribution. Here we find evidence of how relational abstractions capture the underlying task rules rather than just imitating pixel-wise distributions, and show that disentanglement with $k = 16$ strongly aids out-of-distribution generalization in contrast to $k = 121$, which slightly overfits the training set. While Base model plummets (in-distribution (Table 1): 80.04 $\rightarrow$ *low*: 39.48), RDM *clue* with 16 slots maintains consistency (in-distribution: 82.06 $\rightarrow$ *low*: 72.48).

Our evaluation of SRM reveals that diffusion forcing does not universally benefit grid-based puzzles. While SRM excels in MNIST-Sudoku (Wewer et al., 2025), where patch-based noise levels combined

Table 2: Experimental results on Tangram and LogicFace datasets.

	Tangram							LogicFace					
	▱	□	$\triangle^S$	$\triangle^M$	$\triangle^L$	SHAPE	ACC	*smiling* (OR)	*male* (AND)	*glasses* (XOR)	*young* (IMP.)	AVG	ACC
Base	23.06	24.22	42.82	41.50	55.60	57.72	15.46	54.60	76.52	67.34	38.90	57.84	11.24
SRM (Wewer et al., 2025)[2]													
parallel	–	–	–	–	–	–	–	37.00	38.16	49.26	39.44	40.97	2.60
sequential	–	–	–	–	–	–	–	25.04	58.38	50.58	73.86	51.94	5.70
RDM$^{[S]}$													
clue	36.71	36.96	59.59	60.33	63.29	58.41	34.37	**90.19**	**97.00**	**97.26**	**87.27**	**92.93**	**74.74**
solution	41.12	43.94	56.49	53.06	59.94	60.25	38.57	64.87	73.20	50.16	67.78	64.00	16.27
full	**49.75**	**49.98**	**61.55**	**61.25**	**65.00**	**65.12**	**46.76**	83.16	94.00	94.76	79.81	87.93	60.76

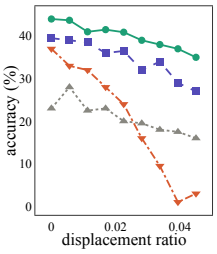
with uncertainty-based sampling align with optimal Sudoku-solving heuristics, its strategy fails to generalize to Akari, where reasoning variables are not evenly distributed across cells. During training, diffusion forcing yields a vast combinatorial space of noise levels that mostly affect illuminated cells, damaging convergence (ACC: 18.80). At inference, SRM *sequential* prematurely prioritizes denoising empty (yellow-illuminated) cells due to their persistent low uncertainty, resulting in performance collapse, where the critical task of bulb placement is deferred until the final sampling states.

Results on Coldoku: Our results from Table 1 show that RDM generally outperforms all baselines in color accuracy and overall accuracy at the easy and medium levels of Coldoku, and SRM performs better in the hard regime. The best performance was achieved with 16 slots (ACC: 97.48 *easy*, 71.64 *medium* and 12.58 *hard*), whereas the *full* 81-slot configuration caused ACC on the easy level to drop to 90.36. We hypothesize that the primary bottleneck in Coldoku is not the slot embeddings themselves, but rather the relational abstraction module ($\mathcal{R}$) shared between the clue and the solution as input with a high number of slots $k = 81$, which requires the unique $\mathcal{R}$ to reconcile complex positional and feature relations from two distinct distributions. Coldoku involves a diverse set of objects (multi-colored digits) in varying positions, while Akari presents a much smaller object vocabulary, where the complexity is exclusively positional.

Tildoku (*Tilted and Colored Sudoku*): We observed that SRM performs well on hard grid-based puzzles with sequential, cell-wise reasoning. However, its reliance on the grid's geometry and patch structure introduces an implicit task-aligned bias. To challenge this in a more general scenario, we introduce Tildoku, where Coldoku images are randomly tilted (see §A.2 for details), disrupting the alignment between grid cells and image coordinates. Thus, the spatial structure now does not necessarily align with the grid-informed reasoning mechanism that SRM utilizes.

Figure 6 shows results on Tildoku (*Medium*, $\tau = 10$) for different displacement ratios at test time (left), and their average (right), where RDM demonstrates more robust reasoning (ACC: 28.40 averaged across various displacement ratios with the *solution* approach). Notably, SRM significantly worsens when increasing the tilting degree at inference time (orange line in the plot), resulting in lower average metrics. This is due to the fact that the learned uncertainty is only consistent when patches cover objects from a stable distribution (e.g., when each patch covers exactly one digit), and it becomes unstable when a patch covers half of a digit or none. Figure 6 (left) demonstrates this effect, where SRM outperforms the baseline at low tilting degrees, but abruptly collapses when digits do not align within fixed patches (see Table 8 for extended results).

Figure 6: Performance of RDM$^{[L]}$ in Tildoku.



Medium	DGT	CLR	ACC
Base	22.62	45.92	12.48
SRM (*seq.*)	24.31	21.31	7.08
clue	33.02	52.84	19.18
solution	**43.70**	**60.12**	**28.40**
full	22.24	31.96	8.46

Results on Tangram: Table 2 shows that RDM significantly surpasses baselines across all metrics on Tangram. Notably, RDM (*full*) reaches three times the baseline performance, achieving ACC: 46.76, compared to baseline's 15.46 (see Figure 15 for visual results). In contrast to Akari and Coldoku, we

[2]Diffusion forcing led to highly unstable training in Tangram, where both SRM *parallel* and *sequential* were unable to outperform Base (similar to LogicFace). These findings also align with Wewer et al. (2025).

observe a considerable improvement when incorporating the solution into the abstractor stack (*clue*: 34.37 → *solution*: 38.57 → *full*: 46.76). This behavior arises from how relational features differ between the clue and solution distributions. RDM (*clue*) only extracts primitives from the piece view (colors and size proportions), mainly boosting accuracy in color assignment (Base: 15.46 vs *clue*: 34.37), but maintaining the shape accuracy close to the baseline (Base: 57.72 vs *clue*: 58.41). Adding information from the solution view, which provides the explicit silhouette partition, boosts the shape and overall accuracy to 65.12 and 46.76 respectively, showing that the relational stack effectively captures the structural dependencies between these two views to guide reasoning.

Results on LogicFace: Our results in Table 2 show superior performance of RDM. In particular, clue-conditioned models achieve the highest accuracies (*clue*: 74.74 and *full*: 60.76, see Figure 16 for generated solutions), whereas standard diffusion models perform poorly (ACC: 11.24). Unlike Akari, Coldoku and Tangram, where baselines remained competitive, LogicFace presents the unique challenge of simultaneously modeling a complex real-image distribution, encoding high-level features of multiple input arguments (e.g., gender, presence of glasses), and inferring logical operations to generate a consistent target image. Baseline models prioritize generating realistic human faces at the expense of satisfying the logical rules (see §C for image quality metrics), effectively ignoring any relationship between the inputs and targets. In contrast, RDM captures the underlying rules for all operators, as evidenced by accuracies that exceed trivial strategies, such as the model defaulting to a single mode (e.g., only generating a smiling female face).

Impact of relational abstractions and evidence of rule learning: In an ablation study, we isolated the impact of *relational abstractions* by evaluating a *slot-based* diffusion guidance approach, i.e., $p_\theta(\mathbf{x}_0|\mathbf{x}_c, \mathbf{s}^c_{1:k})$ at training and inference, similar to previously proposed slot-conditioned diffusion models (Jiang et al., 2023). Our results in Table 11 revealed that explicit relational abstractions are critical for complex reasoning, beyond purely slot-based diffusion guidance alone (see §C.2.1).

Finally, besides our rule-consistent image synthesis evaluations, we also investigated whether the relational abstraction module learns embeddings that capture the logical rules required for reasoning. Across controlled experiments, we found evidence that these embeddings (1) yield near-optimal solution-conditioned inference at test time, (2) can help improve reasoning without the raw clue image conditioning, and (3) remain linearly separable between valid versus rule-violating puzzles (see §C.2.1 for further details on our empirical evidence of rule learning experiments).

7 Discussion

We study a key limitation of image diffusion models regarding their emphasis on pixel-level fidelity in conditional data distribution modeling, which limits their ability to capture logical structures required for spatial reasoning. We introduce RDM as a novel framework that enhances diffusion models with unsupervised object-centric relational abstractions of the structural regularities in the training data. While RDM does not incorporate explicit symbolic computation modules, it leverages learned relational abstractions that move diffusion models toward more symbolic forms of reasoning.

We further present a large-scale spatial reasoning benchmark comprising diverse challenging tasks, on which our approach significantly improves reasoning-driven image generation capabilities. We show that harnessing abstractions from different task components (clues and solutions) enhances reasoning both when applied during training alone for diffusion guidance (*solution*), and when used jointly during training and inference (*clue* and *full*). Experiments on grid-based image completion tasks revealed that current task-informed uncertainty-driven inference-time reasoning approaches (Wewer et al., 2025) do not reliably scale across diverse reasoning tasks, where the failure modes became more pronounced in Tildoku under distorted grid structures. Nevertheless, our method is in principle compatible with any additional sampling-time reasoning approach (Wewer et al., 2025).

Limitations: Our approach relies on object discovery to extract relational patterns, utilizing neural architectures with inherently finite scalability. While these models might not scale to "in-the-wild" scenarios in unconstrained scenes with thousands of entities, we argue that it still remains aligned with the scope of human reasoning, where the number of objects is naturally bounded by cognitive constraints. We also note that the neural-based automated evaluators used in our benchmarks may be prone to minor estimation errors due to their own performance. However, the comparisons presented remain fair and valid as the identical evaluation pipeline applied consistently for all models.

AI USE STATEMENT

We used generative AI tools only to polish writing to improve readability. We have reviewed all AI-assisted text during polishing of writing. We take responsibility for the final content of this work.

We have not used generative AI tools to generate synthetic data sets, help development of theoretical models or conceptual frameworks, propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods, assist with translation, clean and reformat dataset, support qualitative and thematic data analysis, interpret results, create or modify scientific figures or images, suggest experimental parameters, create or edit software code, creation of artifacts, draft parts of a research paper, transcribe recordings of research material, identify summarize or analyse relevant literature, discover research topics or identify gaps, brainstorming, sourcing/searching for information, format references, suggest a structure for the paper, propose a title or keywords for the paper. Formulation of any mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs are not applicable to this work.

ETHICS STATEMENT

We introduce a general framework for spatial reasoning with conditional generative models, with a scope on solving puzzles with logical constraints. We do not target any safety-critical or real-world decision-making application. As a training approach for diffusion models, it might only inherit known risks associated with generative models (e.g., training data memorization). However, we do not identify direct pathways for harmful applications that are specific to our method. The datasets used are open-source and puzzle-based, alleviating concerns related to privacy or sensitive attributes.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the model architectures, training configurations and hyperparameters of the experiments reported in the paper, both partly in the main text, and in §A and §B. Our implementations are provided as supplementary materials, and will be made publicly available.

REFERENCES

Awni Altabaa, Taylor Whittington Webb, Jonathan D Cohen, and John Lafferty. Abstractors and relational cross-attention: An inductive bias for explicit relational reasoning in transformers. In *International Conference on Learning Representations (ICLR)*, 2024.

Pavel Avdeyev, Chenlai Shi, Yuhao Tan, Kseniia Dudnyk, and Jian Zhou. Dirichlet diffusion score model for biological sequence generation. In *International Conference on Machine Learning (ICML)*, pp. 1276–1301, 2023.

David Barrett, Felix Hill, Adam Santoro, Ari Morcos, and Timothy Lillicrap. Measuring abstract reasoning in neural networks. In *International Conference on Machine Learning (ICML)*, pp. 511–520, 2018.

Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Gerry Bohning and Jody Kosack Althouse. Using tangrams to teach geometry to young children. *Early Childhood Education Journal*, 24:239–242, 1997.

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *Advances in Neural Information Processing Systems*, 37:24081–24125, 2024a.

Minghao Chen, Iro Laina, and Andrea Vedaldi. Training-free layout control with cross-attention guidance. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 5343–5353, 2024b.

François Chollet. On the measure of intelligence. *arXiv preprint arXiv:1911.01547*, 2019.

Francois Chollet, Mike Knoop, Gregory Kamradt, and Bryan Landers. ARC Prize 2024: Technical report. *arXiv preprint arXiv:2412.04604*, 2024.

François Fleuret, Ting Li, Charles Dubout, Emma K. Wampler, Steven Yantis, and Donald Geman. Comparing machines and humans on a visual categorization test. *Proceedings of the National Academy of Sciences*, 108(43):17621–17625, 2011.

Nir Goren, Shai Yehezkel, Omer Dahary, Andrey Voynov, Or Patashnik, and Daniel Cohen-Or. Visual Diffusion Models are Geometric Solvers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 43187–43196, June 2026.

Klaus Greff, Raphaël Lopez Kaufman, Rishabh Kabra, Nick Watters, Christopher Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. In *International Conference on Machine Learning*, pp. 2424–2433, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Doll'ar, and Ross B. Girshick. Masked Autoencoders Are Scalable Vision Learners. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15979–15988, 2021.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Anya Ji, Noriyuki Kojima, Noah Rush, Alane Suhr, Wai Keen Vong, Robert Hawkins, and Yoav Artzi. Abstract visual reasoning with Tangram shapes. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 582–601, 2022.

Jindong Jiang, Fei Deng, Gautam Singh, and Sungjin Ahn. Object-centric slot diffusion. *Advances in Neural Information Processing Systems*, 36:8563–8601, 2023.

Giancarlo Kerg, Sarthak Mittal, David Rolnick, Yoshua Bengio, Blake Richards, and Guillaume Lajoie. On neural architecture inductive biases for relational tasks. *arXiv preprint arXiv:2206.05056*, 2022.

Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. MaskGAN: Towards Diverse and Interactive Facial Image Manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Gyubin Lee, Bao N Nguyen Truong, Jaesik Yoon, Dongwoo Lee, Minsu Kim, Yoshua Bengio, and Sungjin Ahn. Adaptive inference-time scaling via cyclic diffusion search. *Advances in Neural Information Processing Systems*, 2025.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow Matching for Generative Modeling. In *International Conference on Learning Representations (ICLR)*, 2023.

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

Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Shanka Subhra Mondal, Taylor Whittington Webb, and Jonathan Cohen. Learning to reason over visual objects. In *International Conference on Learning Representations (ICLR)*, 2023.

Shanka Subhra Mondal, Jonathan D Cohen, and Taylor Whittington Webb. Slot abstractors: Toward scalable abstract visual reasoning. In *International Conference on Machine Learning (ICML)*, pp. 36088–36105, 2024.

Weili Nie, Zhiding Yu, Lei Mao, Ankit B Patel, Yuke Zhu, and Anima Anandkumar. Bongard-logo: A new benchmark for human-level concept learning and reasoning. *Advances in Neural Information Processing Systems*, 33:16468–16480, 2020.

Maxime Oquab et al. DINOv2: Learning Robust Visual Features without Supervision. *Transactions on Machine Learning Research*, 2024.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023.

Niv Pekar, Yaniv Benny, and Lior Wolf. Generating correct answers for progressive matrices intelligence tests. *Advances in Neural Information Processing Systems*, 33:7390–7400, 2020.

J. C. Raven. *Progressive Matrices: A Perceptual Test of Intelligence*. Oxford Psychologists Press, Ltd., Oxford, 1938.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241, 2015.

Fan Shi, Bin Li, and Xiangyang Xue. Towards generative abstract reasoning: Completing raven's progressive matrix via rule abstraction and selection. In *International Conference on Learning Representations (ICLR)*, 2024.

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

Gautam Singh, Yi-Fu Wu, and Sungjin Ahn. Simple unsupervised object-centric learning for complex and naturalistic videos. *Advances in Neural Information Processing Systems*, 35:18181–18196, 2022b.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning (ICML)*, pp. 2256–2265, 2015.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-Based Generative Modeling through Stochastic Differential Equations. In *International Conference on Learning Representations (ICLR)*, 2021.

Libo Sun, James Browning, and Roberto Perera. Shedding some light on Light Up with artificial intelligence. *arXiv preprint arXiv:2107.10429*, 2021.

Taylor Webb, Shanka Subhra Mondal, and Jonathan D Cohen. Systematic visual reasoning through object-centric relational abstraction. *Advances in Neural Information Processing Systems*, 36: 72030–72043, 2023.

Taylor Whittington Webb, Ishan Sinha, and Jonathan Cohen. Emergent symbols through binding in external memory. In *International Conference on Learning Representations (ICLR)*, 2021.

Christopher Wewer, Bartlomiej Pogodzinski, Bernt Schiele, and Jan Eric Lenssen. Spatial reasoning with denoising models. In *International Conference on Machine Learning (ICML)*, pp. 66706–66725, 2025.

Simpson WL Wong, Rebecca Wing-yi Cheng, Bonnie Wing-Yin Chow, and Sandrine Man-Chi Chung. The link between a set of tangram-based tasks and chinese and english reading and related skills among chinese kindergarteners. *AERA Open*, 5(1):2332858419829723, 2019.

Ziyi Wu, Jingyu Hu, Wuyue Lu, Igor Gilitschenski, and Animesh Garg. SlotDiffusion: Object-centric generative modeling with diffusion models. *Advances in Neural Information Processing Systems*, 36:50932–50958, 2023.

Fernanda Miyuki Yamada, Harlen Costa Batagelo, João Paulo Gois, and Hiroki Takahashi. TANGAN: solving Tangram puzzles using generative adversarial network. *Applied Intelligence*, 55(7), 2025.

Takayuki Yato and Takahiro Seta. Complexity and completeness of finding another solution and its application to puzzles. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 86(5):1052–1060, 2003.

Junyan Ye, Dongzhi Jiang, Jun He, Baichuan Zhou, Zilong Huang, Zhiyuan Yan, Hongsheng Li, Conghui He, and Weijia Li. BLINK-Twice: You see, but do you observe? A Reasoning Benchmark on Visual Perception. In *Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2025.

Chi Zhang, Baoxiong Jia, Song-Chun Zhu, and Yixin Zhu. Abstract Spatial-Temporal Reasoning via Probabilistic Abduction and Execution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021.

Yikun Zong and Cheston Tan. TangramSR: Can Vision-Language Models Reason in Continuous Geometric Space? *arXiv preprint arXiv:2602.05570*, 2026.

Mehmet Çelik, Halit Erdogan, Firat Tahaoglu, Tansel Uras, and Esra Erdem. Comparing ASP and CP on Four Grid Puzzles. In *RCRA@AI*IA*, 2009.

A Details on the Spatial Reasoning Benchmark Datasets

In this section, we provide a detailed description of our spatial reasoning datasets, and motivate their use for evaluating generative models on challenging visual tasks. Our primary goal in introducing this novel spatial reasoning benchmark is to move beyond simple pattern matching by requiring generative models to satisfy discrete geometric and logical reasoning constraints.

	Akari	Coldoku	Tangram	LogicFace
$\mathtt{H} \times \mathtt{W}$	198×198	180×180	128×128	128×128
$\mathtt{h} \times \mathtt{w}$	11×11	9×9	8×8	8×8
k	121 / 16	81 / 16	7	6
# samples	1M	1M	1,013	30,000
neural evaluator	✗	✓	✗	✓

Table 3: General statistics: image ($\mathtt{H} \times \mathtt{W}$) and bottleneck ($\mathtt{h} \times \mathtt{w}$) resolution; number of slots (k) and unique samples (# samples); and neural evaluation.

A.1 Akari dataset

Akari (also known as *Light Up*) is a binary-determination puzzle played on a rectangular grid of empty cells (□) and walls (■). The goal is to place light bulbs (💡) in empty cells such that every cell is illuminated (□) while ensuring no pair of bulbs shine on one another, i.e., no two bulbs share a line of sight, horizontally or vertically, unobstructed by a wall. Walls act as obstacles to light and may serve as *numbered cells*. These cells are annotated with a digit from 0 to 4, indicating the exact number of bulbs that must be placed in the four adjacent (orthogonal) cells.

We generated a dataset of one million 11×11 Akari puzzles[3]. Our generation process follows a constraints-first approach. We first distributed walls across 20%–30% of the grid, then allocated bulbs according to the core rules of the puzzle, and finally converted 50% of the walls into clued cells by annotating them with the count of their adjacent bulbs, leaving the remaining walls unnumbered.

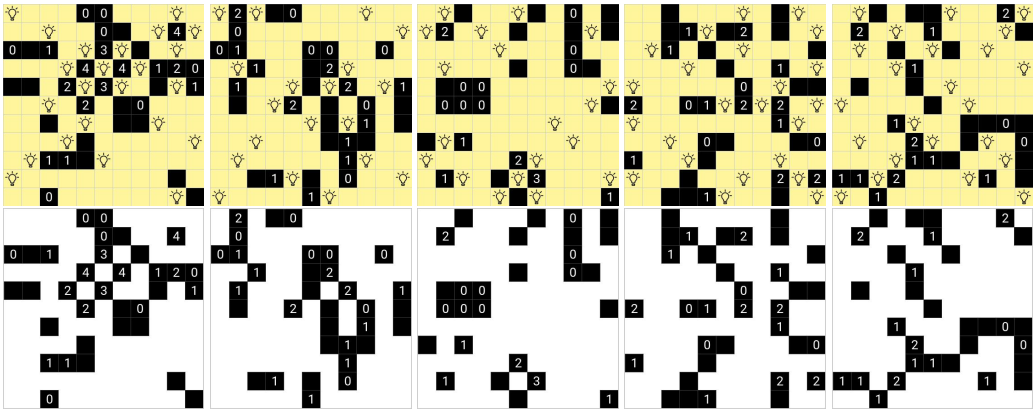


Figure 7: Akari samples from the training set.

Akari properties: Akari is an NP-complete problem (Sun et al., 2021), a complexity derived from the interplay between line-of-sight constraints and numerical wall requirements. Without clued cells, the problem reduces to polynomial time. Solving typically involves local constraint satisfaction, such as prioritizing bulbs for high-value clues or identifying bottleneck empty cells that can only be illuminated by a single potential bulb placement.

From a generative perspective, ensuring a unique solution elevates the task to a higher-order complexity ($\sum_2^P$ or ASP classes (Çelik et al., 2009)). Standard algorithms rely on backtracking solvers: a candidate grid is produced and iteratively modified with additional wall constraints if more than one solution is found. To better evaluate model novelty and rule abstraction, our dataset permits multiple valid arrangements, allowing novel solutions that may differ from training annotations.

Evaluation of Akari: We evaluate generated images with a deterministic pipeline that individually classifies each cell as a bulb (💡) or illuminated background (□) via mean squared error against target patches. A sequential algorithm then verifies the arrangement against puzzle constraints. We use five

[3] Akari grids can be of arbitrary sizes, but most commonly 11×11 or 13×13. We opted for 11×11 simply to balance the image resolution with our computational resources.

metrics: (i) *accuracy*, requiring all constraints to be satisfied, (ii) *bulb accuracy*, a relaxed metric ignoring numbered wall constraints, (iii) *unlit count*, the count of unlit empty cells, (iv) *distance*, as the absolute error between a wall's digit and its actual adjacent bulb count, and (v) *overlap*, as the count of bulbs sharing a line of sight.

A.2 COLDOKU: THE COLORED SUDOKU DATASET

Sudoku is a combinatorial number-placement puzzle played on a 9×9 grid, which is further partitioned into nine 3×3 subgrids (also known as *blocks* or *boxes*). In its classical formulation, the objective is to populate the grid with digits from 1 to 9 such that each row, column, and subgrid contains every digit exactly once. Various extensions of the puzzle have been proposed, ranging from larger grid dimensions to the addition of supplementary constraints. The complexity of solving Sudoku puzzles has been formally classified as NP-complete (Yato & Seta, 2003). While it has been mathematically proven that a minimum of 17 initial clues is necessary for a unique solution, human-solvable puzzles typically provide 20 to 35 fixed digits to guide the player through the remaining cells.

We introduce on a *multi-constraint* Sudoku variant that couples the classical digit-placement logic with a coloring challenge, named Coldoku. This task effectively requires the model to simultaneously solve a secondary Sudoku-style constraint involving nine distinct colors. We relied on the MNIST-Sudoku dataset released by Wewer et al. (2025) as our primary source for 1M solved Sudoku configurations. Following their methodology, we train an MNIST classifier to populate the grid cells with handwritten digits, thereby generating a realistic handwritten view of the Sudoku grid. To introduce the additional layer of complexity, we randomly pair each Sudoku with another independent Sudoku. The second Sudoku determines the color assignment for each digit, effectively creating a Coldoku puzzle where both digit and color placements must satisfy Sudoku constraints (Figure 3). Our dataset uses the *Tableau 10* palette, which is designed to have a large perceptual difference between colors (blue ●, orange ●, green ●, red ●, purple ●, brown ●, pink ●, olive ● and cyan ●. By cross-combining these digit and color arrangements, the resulting dataset spans a massive state space of 10^{12} ($1M^2$) unique puzzle configurations.

Coldoku properties: Coldoku inherits the formal logical properties of the classical Sudoku puzzle, as it effectively requires the joint resolution of two orthogonal Sudoku instances (one in the digit domain and one in the color domain) given the same set of initial clues. While a human solver would likely adopt a sequential strategy, first solving the digit arrangement before addressing the color-based constraints, a neural network must perform these operations as a simultaneous multi-objective task.

The properties of uniqueness and complexity (NP-completeness) are preserved for both the digit and color layers. Consequently, Coldoku represents an inherently more challenging benchmark for generative models, as it requires the latent reasoning module to maintain two independent sets of global constraints and resolve them within a single inference pass.

Tildoku (*Tilted and Colored Sudoku*): To evaluate RDM in a more realistic scenario, we introduce Tildoku, a variant of Coldoku where samples are randomly tilted within a bounded angular range (see Figure 9). While Tildoku maintains the underlying logical rules and reasoning complexity of Coldoku, it introduces a visual challenge: digit completion now requires the model to detect the boundaries of a varying puzzle geometry. To ensure the task remains approachable, we limited the tilting distortion to the 15% of corner displacements. During training, each Coldoku sample is subjected to a random tilt within these bounds before being processed by the model. For evaluation, the associated affine matrix is utilized to reverse the transformation, allowing the predicted output to be verified by the Coldoku evaluator.

Evaluation of Coldoku: We assess the generative quality of Coldoku samples using a two-stage pipeline. First, we train a CNN classifier on 28×28 colored MNIST images ($> 99\%$ test accuracy). To ensure precise classification, these training images are quantized such that each digit contains only its target color centroid against a black background. During evaluation, each generated 9×9 grid is partitioned into 81 individual cells and processed by the classifier to identify the digit-color pairs. Consistent with Wewer et al. (2025), we mitigate the impact of perception errors on our reasoning framework by constructing the input Coldoku views exclusively from correctly classified MNIST samples. Following the perception stage, we independently verify the row, column, and subgrid constraints for both the digit and color arrangements. We report three primary metrics: (i) *digit accuracy*, ratio of grids where the 81 predicted digits satisfy all Sudoku constraints, (ii)



Figure 8: Visualization of Coldoku samples from the training set (top: *solution*, bottom: *clue*).



(a) 0.01 (b) 0.03 (c) 0.05 (d) 0.07 (e) 0.09

Figure 9: Visualization of Tildoku *solution* samples with different tilting ratios.

color accuracy, ratio of grids where the 81 predicted colors satisfy all Sudoku constraints, and (iii) *accuracy*, joint ratio of grids where both the digit and color arrangements are simultaneously valid.

A.3 TANGRAM DATASET

Tangram is a classic dissection puzzle consisting of seven flat shapes, called *tans*, which must be arranged to form a specific target figure, or *silhouette* (see Figure 4b). The seven pieces (Figure 4a), comprising two large ($\triangle^L \times 2$), one medium ($\triangle^M$) and two small ($\triangle^S \times 2$) right triangles, one square ($\square$), and one parallelogram ($\parallel$), remain constant across all puzzles. The core challenge of the Tangram task lies in the latent inference that is required to map these seven discrete pieces onto a silhouette, where individual piece boundaries are not visible. A valid solution must satisfy three primary constraints: (i) all seven tans must be used, (ii) no two pieces may overlap or occupy the same spatial coordinates, and (iii) the final arrangement must maintain the area and size proportions[4] of the provided layout. Many Tangram layouts exhibit high visual ambiguity, i.e., cases where internal boundaries are not easily identifiable, making them a rigorous test for visual reasoning and spatial reconstruction, even for human solvers (Figure 10).

We rely on the SVG annotations from the Kilogram dataset (Zhang et al., 2021), which provides 1,013 uncolored, unique Tangram solutions. Each sample consists of an SVG file containing precise polygon coordinates for the final silhouette. We extended this dataset by randomly creating 9,200 shuffles to use as input clue (Figure 4a). To color the pieces and the solution, we randomly sample with replacement a pool of nine colors: red ●, green ●, blue ●, yellow ●, magenta ●, cyan ●, purple ●, orange ● and teal ●, and match the color of each piece of the shuffled view to a piece of the same type in the solution.

While the original Kilogram dataset was designed to evaluate the semantic associations between silhouettes and language, recent works have begun addressing the underlying spatial reasoning task (Yamada et al., 2025; Zong & Tan, 2026). However, existing approaches such as TANGAN (Yamada

[4]All tans maintain strict area ratios relative to the smallest triangle: the medium triangle, square, and parallelogram has twice the area of a small triangle, while each large triangle has four times the area.

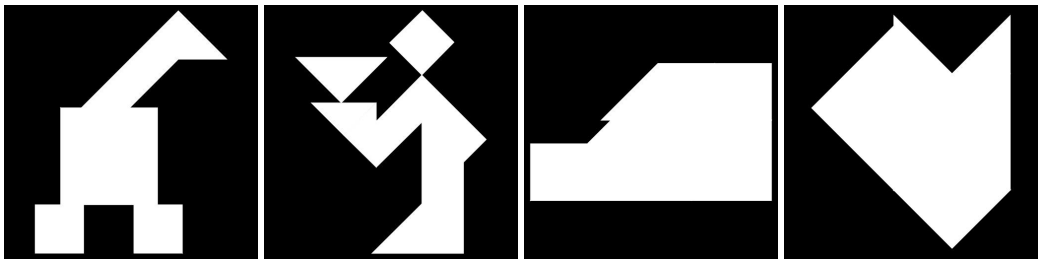


Figure 10: Tangram layouts ordered by reasoning difficulty. From easy configurations (left) containing salient geometric cues that simplify piece identification to complex silhouettes (right) characterized by high structural ambiguity, where multiple valid piece arrangements may exist for the same layout.

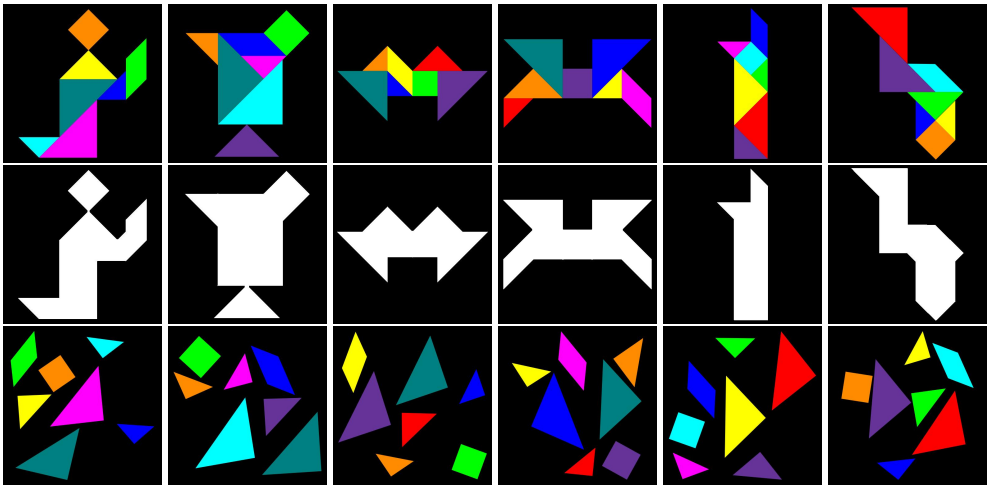


Figure 11: Visualization of Tangram samples from the training set.

et al., 2025) treat the problem as a standard generative task using Generative Adversarial Networks (GANs) to segment a silhouette into pieces. Such methods may overlook the discrete constraints of the puzzle (e.g., that a specific set of seven pieces with fixed area ratios must be utilized), and ignore the actual reasoning on the piece proportions to simply recreate a shapes.

To bridge this gap and enforce an actual human-like reasoning process, we augment the dataset by assigning distinct colors to each of the seven tans. This modification requires the model to perform joint spatial and logical reasoning: it must not only partition the silhouette correctly but also match the specific geometry of each piece (e.g., the large triangle vs. the medium triangle) to its corresponding color and position. This ensures that the *reasoner* is explicitly accounting for the size relationships and global constraints of the puzzle rather than relying on local shape generation. Additionally, this initialization (Figure 4a-4b) also resembles how humans solve the Tangram puzzle: from a random view of separated tans (the *shuffle*), *reasoning* over the color and proportions to fill the layout. Our perception is robust to different shuffles of the same seven pieces, as well as rotations of the silhouette.

Evaluation of Tangram: We use a deterministic evaluation pipeline to assess the correctness of the generated solutions with our diffusion models. First, we quantize the color space by mapping each pixel to the nearest of 10 predefined color centroids (9 tan colors plus the black background). We then leverage the input layout to define the ground-truth boundaries of the silhouette. If the pixel-level mismatch between the generated image and the ground-truth layout exceeds 5%, the sample is immediately assigned zero accuracy.

Subsequently, for samples passing the initial silhouette layout check, we use the OpenCV library to perform contour detection and polygonal approximation. Based on the extracted polygons, we evaluate the sample against two metrics:

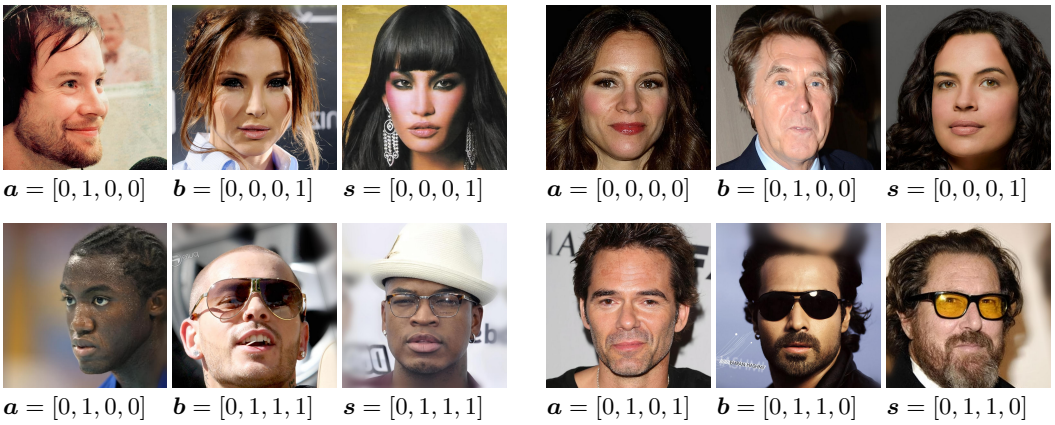


Figure 12: LogicFace examples from the training set with the four binary features: *smiling* (OR), *male* (AND), *glasses* (XOR), *young* (IMPLIES).

1. Shape accuracy (correctly-shaped): if the polygons maintain the correct area and proportions and the set of extracted shapes matches the original seven-tan composition.
2. Accuracy (correctly-colored): building upon soft accuracy, if the color of each placed piece matches the identity of the corresponding tan provided in the input piece view.

In Figure 4d of the main text, we show an example of a solution that would be classified with 100% shape accuracy but with zero standard accuracy.

A.4 LogicFace: Logic Operators on Facial Features

We challenge our spatial reasoning framework to abstract real-image patterns, and introduce LogicFace as a structured logical puzzle where facial features must satisfy specific combinations based on logical operators. The LogicFace puzzle consists of three images, sorted in a row (Figure 5), where the first two are the arguments (A and B) and the third one is the answer (S). Each image contains a clear centered human face, characterized with κ binary attributes, named as $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{s} \in \{0, 1\}^{\kappa}$ for A, B and S, respectively. For each attribute, the puzzle fixes an implicit logical operator $f_j : \{0, 1\} \times \{0, 1\} \to \{0, 1\}$ such that the statement $f_j(\boldsymbol{a}_j, \boldsymbol{b}_j) = \boldsymbol{s}_j$ holds.

We used the face images and annotations from the CelebA-HQ dataset (Lee et al., 2020) and selected $\kappa = 4$ binary attributes: *smiling*, *male*, *glasses* and *young* to characterize each input image, with the OR, AND, XOR and IMPLIES logical operators. These attributes and operators were fixed for all images of the dataset. During training, the first and second panels are used as clues, while the last panel is corrupted to task the reasoner to learn how to generate facial attributes such that the logical operators hold.

LogicFace statistics: We selected the attributes of the LogicFace puzzle after a manual examination of the CelebA annotations, where we found *smiling*, *male*, *glasses* and *young* to be less noisy than the other labels, as well as uncorrelated. Attributes like *smiling* (48.21%) and *male* (41.67%) are balanced, while *glasses* and *young* constitute the 6.5% and 77.36% of the CelebA-HQ dataset, respectively. To ensure a balanced number of LogicFace configurations, we first uniformly sample one million LogicFace configurations, and then select images whose attributes matched the specific statements. Note that with this process, the LogicFace dataset is not balanced in terms of images (indeed, faces with *glasses* are considerably repeated since they need to cover the missing ratio of the CelebA-HQ dataset to create statements of the XOR operator). However, since our purpose is to abstract on the combinations of glasses rather than the diversity of

OR			AND			XOR			IMPLIES		
A	B	S	A	B	S	A	B	S	A	B	S
0	0	0	0	0	0	0	0	0	0	0	1
0	1	1	0	1	0	0	1	1	0	1	1
1	0	1	1	0	0	1	0	1	1	0	0
1	1	1	1	1	1	1	1	0	1	1	1

Table 4: Logical operators between A and B, yielding S. Since there are four operator and each operator has four valid statements, LogicFace has 256 possible configurations.

Algorithm 2: Slot Attention module ($\mathcal{S}$).

Input : $(\mathbf{f}_{1:m}, \mathbf{p}_{1:m})$ from $\mathcal{E}$.
Params: Projections k, q, v, GRU, MLP, LayerNorm.
$\mathbf{h}_{1:m} \leftarrow \text{LayerNorm}(\mathbf{f}_{1:m} + \mathbf{p}_{1:m})$;
$\mathbf{s}_{1:k} \sim \mathcal{N}(0, \mathbf{I})$;
for $i = 1, \dots, \eta$ **do**
  $\mathbf{s}_{1:k} \leftarrow \text{LayerNorm}(\mathbf{s}_{1:k})$;
  $A \leftarrow \text{softmax}(k(\mathbf{h}_{1:m}) \cdot q(\mathbf{s}_{1:k})^\top) \in \mathbb{R}^{m \times k}$;
  $\mathbf{s}_{1:k} \leftarrow \text{GRU}(\mathbf{s}_{1:k}; A^\top \cdot v(\mathbf{h}_{1:m}))$;
  $\mathbf{s}_{1:k} \leftarrow \mathbf{s}_{1:k} + \text{MLP}(\text{LayerNorm}(\mathbf{s}_{1:k}))$
$\tilde{\mathbf{f}}_{1:k} \leftarrow A^\top \mathbf{f}_{1:m}$;
$\tilde{\mathbf{p}}_{1:k} \leftarrow A^\top \mathbf{p}_{1:m}$;
return $\mathbf{s}_{1:k}$

Algorithm 3: Relational Abstraction ($\mathcal{R}$).

Input : Latents $\mathbf{z} = [\tilde{\mathbf{f}}_{1:K}; \tilde{\mathbf{p}}_{1:K}]$ from $\mathcal{S}$.
Params: Projections k_ℓ, q_ℓ, v_ℓ and FFN_ℓ per layer ℓ.
$\mathbf{r}_{1:K} \leftarrow \tilde{\mathbf{p}}_{1:K}$;
for $\ell = 1, \dots, L$ **do**
  $\mathbf{v}_{1:K} \leftarrow v_\ell(\mathbf{r}_{1:K})$;
  $\mathbf{v}_{1:K} \leftarrow \text{softmax}(k_\ell(\tilde{\mathbf{f}}_{1:K}) \cdot q_\ell(\tilde{\mathbf{f}}_{1:K})^\top)\mathbf{v}_{1:K} + \mathbf{v}_{1:K}$;
  $\mathbf{v}_{1:K} \leftarrow \text{FFN}_\ell(\mathbf{v}_{1:K}) + \mathbf{v}_{1:K}$;
  $\mathbf{v}_{1:K} \leftarrow \text{SelfAttention}(\mathbf{v}_{1:K}) + \mathbf{v}_{1:K}$;
  $\mathbf{r}_{1:K} \leftarrow \text{FFN}_\ell(\mathbf{v}_{1:K}) + \mathbf{v}_{1:K}$;
return $\mathbf{r}_{1:k}$

the generation, we opted to use data augmentation methods (e.g., horizontal flipping) to expand the variety of the samples, but acknowledging that our diffusion models might be recreating training set replicas, especially for unbalanced attributes like (not) *young* and *glasses*.

Evaluation of LogicFace: We trained a ResNet-18 (He et al., 2016) with a binary multi-label objective corresponding to the selected attributes (*smiling*, *male*, *glasses* and *young*) on the CelebA-HQ dataset ($> 98\%$ test accuracy). At inference time, given a generated $\hat{S}$ and its predicted binary value $\hat{\boldsymbol{s}}$, the evaluator uses Table 4 to validate the constraints. Specifically, it individually evaluates $f_j(\boldsymbol{a}_j, \boldsymbol{b}_j) = \hat{\boldsymbol{s}}_j$ for the j-th feature and logical operator, and the overall correctness of a sample.

B Details on the Model Architectures

Our spatial reasoning framework is composed of two models: (i) the *discoverer*, which is independently trained to disentangle objects in the scene through slot attention, (ii) the *reasoner*, which is a diffusion model tasked to model the conditional distribution $q(\mathbf{x}_0|\mathbf{x}_c)$ guided via *relational embeddings* learned by a *relational abstraction* module. In what follows, we describe each module in detail, as well as the specific hyper-parameters used to train them.

B.1 Architecture of the *Discoverer*

The discoverer follows a similar architecture to the one proposed by Locatello et al. (2020).

- The slot encoder $\mathcal{E} : \mathbb{R}^{\mathtt{H} \times \mathtt{W} \times \mathtt{C}} \to (\mathbb{R}^{m \times d}, \mathbb{R}^{m \times d})$ maps the input image to a set of hidden features $\mathbf{f}_{1:m} \in \mathbb{R}^{m \times d}$ through convolutional layers, alongside positional embeddings $\mathbf{p}_{1:m} \in \mathbb{R}^{m \times d}$, where $m = \mathtt{hw}$ corresponds to the reduced spatial resolution.
- The slot attention module, $\mathcal{S} : (\mathbb{R}^{m \times d}, \mathbb{R}^{m \times d}) \to \mathbb{R}^{k \times d}$, uses the extracted features ($\mathbf{f}_{1:m}$) and positions ($\mathbf{p}_{1:m}$) from the encoder to compute the slot embeddings ($\mathbf{s}_{1:k}$). This process begins by initializing the slots as $\mathbf{s}_{1:k} \sim \mathcal{N}(0, \mathbf{I})$, and iteratively refining them over $\eta = 3$ iterations using a GRU conditioned on the input features and positions. The output is an updated set of slots $\mathbf{s}_{1:k}$, whose information is decoupled into object-specific features ($\tilde{\mathbf{f}}_{1:k}$) and positions ($\tilde{\mathbf{p}}_{1:k}$) by computing the dot product of $\mathbf{f}_{1:m}$ and $\mathbf{p}_{1:m}$ with the final attention matrix of the refinement, respectively. This is formally outlined in Algorithm 2.
- The slot decoder $\mathcal{D} : \mathbb{R}^{k \times d} \to \mathbb{R}^{\mathtt{H} \times \mathtt{W} \times \mathtt{C}}$ is fed with the slot embeddings $\mathbf{s}_{1:k}$ to generate individual reconstructions and their corresponding importance masks. The decoder spatially broadcasts each slot embedding onto a 2D grid, augmented with positional embeddings $\mathbf{g} \in \mathbb{R}^{\mathtt{h} \times \mathtt{w} \times d}$. This is processed by convolutional layers to produce a candidate reconstruction per slot, named $\{\hat{\mathbf{x}}_{1:k}\} \in \mathbb{R}^{k \times \mathtt{H} \times \mathtt{W} \times \mathtt{C}}$, with an associated mask, $\boldsymbol{\alpha}_i \in \mathbb{R}^{\mathtt{H} \times \mathtt{W}}$. The final output $\hat{\mathbf{x}}$ is computed as the weighted sum of the candidate reconstructions, where the weights are determined by a softmax normalization over the masks: $\hat{\mathbf{x}} = \sum_{i=1}^{k} \text{softmax}(\boldsymbol{\alpha})_i \hat{\mathbf{x}}_i$.

The full architecture is trained end-to-end with the mean squared error between the original input $\mathbf{x}$ and the final reconstruction $\hat{\mathbf{x}}$.

Scalable decoder: To scale the number of slots k to hundreds, we propose a *scalable decoder* that avoids the memory-intensive process of generating full candidate reconstructions for every slot. Instead, this decoder estimates the attention masks (alpha weights) prior to the broadcasting step.

The mechanism uses a grid of positional embeddings, $\mathbf{g} \in \mathbb{R}^{h \times w \times d}$. Unlike the standard decoder that broadcasts every slot to the full grid resolution, our approach computes the dot product between the grid and the slot embeddings $\mathbf{s}_{1:k}$ to derive an estimated mask: $\tilde{\boldsymbol{\alpha}} = \text{softmax}(\mathbf{g} \cdot \mathbf{s}_{1:k}^{\top})$. The positional grid is then updated by aggregating the slots according to these weights: $\tilde{\mathbf{s}} = \sum_{i=1\ldots k} \tilde{\boldsymbol{\alpha}}_i \mathbf{s}_i$. The weighted feature map $\tilde{\mathbf{s}}$ serves as the sole input for the subsequent decoder layers, effectively decoupling the reconstruction from the number of slots k.

Training details: We trained our *Discoverer* module with AdamW (Loshchilov & Hutter, 2019), learning rate of 4×10^{-4}, batch size of 64, and linear warmup of 10,000 steps followed by exponential decay; using mixed precision (bfloat) for accelerated training.

Evaluation: We assess disentanglement of the *Discoverer* by manually inspecting predicted samples, and additionally measure the reconstruction performance[5] with the task-specific evaluators. For Akari, Coldoku, Tildoku and MNIST-Sudoku we use the cell accuracy (ratio of cells with matched predictions with the evaluator). For Tangram, LogicFace and Counting Polygons we run the evaluator on the reconstructed image and measure if the output satisfies the logical constraints.

B.2 Architecture of the *Relational Abstraction* module

The relational abstraction module is designed to enrich slot representations with decoupled positional and relational information via attention mechanisms. It leverages Slot Abstractors (Mondal et al., 2024) to map object-level information onto learnable *relational embeddings*. These embeddings are iteratively refined alongside the slots to produce high-level, position-aware representations.

The relational abstraction module, $\mathcal{R} : (\mathbb{R}^{K \times d}, \mathbb{R}^{K \times d}) \to \mathbb{R}^{K \times d}$, builds upon the output of the slot autoencoder. As described in §B.1, the slots from the encoder $\mathcal{E}$ can be decoupled into object-specific features ($\tilde{\mathbf{f}}_{1:k}$) and positional embeddings ($\tilde{\mathbf{p}}_{1:k}$), which together compose the conditional latents for the diffusion model $\mathbf{z}^* = (\tilde{\mathbf{f}}^*_{1:k}, \tilde{\mathbf{p}}^*_{1:k})$, where $* = (0, c)$. This operation re-projects the consolidated slot information back into separate features and position subspaces.

The slot abstractor module (Algorithm 3) learns relational information between slots by passing their feature and positional information through L abstractor layers (Mondal et al., 2024), consisting of interleaved cross-attention, self-attention and MLP blocks with residual connections. Through consecutive layers, the positional embeddings ($\tilde{\mathbf{p}}^*_{1:k}$) are updated by hierarchically pooling feature relations ($\tilde{\mathbf{f}}^*_{1:k}$), enabling the model to capture interactions between the discovered objects and learn contextualized positional-aware representations of their spatial dependencies.

B.3 Configuration of the *Reasoner*

We define the reasoning task as modeling the conditional distribution $q(\mathbf{x}_0|\mathbf{x}_c)$, where $\mathbf{x}_0$ is a valid solution for the reasoning task and $\mathbf{x}_c$ is the initial clue or statement.

The reasoner is a diffusion model that simultaneously learns two distributions: (i) the *unguided* conditional distribution $q(\mathbf{x}_0|\mathbf{x}_c, \mathcal{R}(\mathbf{z}^c))$, and (ii) the *solution-guided* conditional distribution $q(\mathbf{x}_0|\mathbf{x}_c, \mathcal{R}(\mathbf{z}^c, \mathbf{z}^0))$, where the latent variables $\mathbf{z}^c$ and $\mathbf{z}^0$ correspond to the feature-position embeddings obtained with the discoverer module. Since the goal is to model $q(\mathbf{x}_0|\mathbf{x}_c)$ at inference time, ancestral sampling on $\mathbf{z}^0$ is not possible, so the *unguided* distribution is the one sampled to generate a solution from the clue $\mathbf{x}_c$ and the latents $\mathbf{z}^c$.

Forward process: We define the diffusion process of the target distributions with continuous normalizing flows (CNFs), where the trajectory from data distribution $q_0(\mathbf{x})$ to the Gaussian prior $q_1 = \mathcal{N}(0, \mathbf{I})$ is governed by Gaussian probability density paths (Lipman et al., 2023, Eq. (1)), whose parameters depend on $\mathbf{x}_0$ and a continuous time $t \in [0, 1]$ such that convergence from $\mathbf{x}_0$ to the Gaussian prior is guaranteed by fixing $\mu_0(\mathbf{x}_0) = \mathbf{x}_0$, $\sigma_0(\mathbf{x}_0) = \sigma_{\min}$, $\mu_1(\mathbf{x}_0) = 0$ and $\sigma_1(\mathbf{x}_0) = 1$.

[5]Note that this accuracy is not correlated with the final reasoning accuracy since it only measures the pixel-level reconstruction quality of the autoencoder. In some cases, like Tangram, the accuracy is low since most of the shapes are not sharply reconstructed, causing the evaluator to not properly estimate the polygons.

The *flow* (Eq. (2)) is a differentiable transformation to reshape samples from the prior to $p_t(\mathbf{x}|\mathbf{x}_0)$, and its associated conditional *vector field* (Eq. (3)) is approximated by a neural network.

$$p_t(\mathbf{x}|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}|\mu_t(\mathbf{x}_0), \sigma_t(\mathbf{x}_0)^2\mathbf{I}), \tag{1}$$

$$\phi_t(\boldsymbol{\epsilon}, \mathbf{x}_0) = \mu_t(\mathbf{x}_0) + \sigma_t(\mathbf{x}_0)\boldsymbol{\epsilon}, \tag{2}$$

$$v_t(\mathbf{x}|\mathbf{x}_0) = \frac{\sigma_t'(\mathbf{x}_0)}{\sigma_t(\mathbf{x}_0)}(\mathbf{x} - \mu_t(\mathbf{x}_0)) + \mu_t'(\mathbf{x}_0). \tag{3}$$

We define the mean and variance of the probability paths as linear functions of t (Eq. (4)):

$$\mu_t(\mathbf{x}_0) = (1-t)\mathbf{x}_0, \quad \sigma_t(\mathbf{x}_0) = t, \tag{4}$$

which effectively sets $\sigma_{\min} = 0$. Given this choice of linear functions, the training objective based on flow matching (Lipman et al., 2023) can be written as follows[6]:

$$\mathcal{L}(\theta) = \mathbb{E}_{\mathbf{x}_0 \sim q_0,\, \boldsymbol{\epsilon} \sim q_1,\, \mathbf{x}_t \sim p_t} \|p_\theta(\mathbf{x}_t, t) - (\boldsymbol{\epsilon} - \mathbf{x}_0)\|^2. \tag{5}$$

Sampling: Since the transformation is governed by a continuous-time vector field, sampling can be framed as solving an ordinary differential equation (ODE). Specifically, once the neural network $p_\theta(\mathbf{x}_t, t)$ is trained to approximate the target vector field $v_t(\mathbf{x}|\mathbf{x}_0)$, we generate new samples by evolving the Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$ backward in time to $t = 0$. The trajectory follows the ODE:

$$\frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t} = p_\theta(\mathbf{x}_t, t). \tag{6}$$

In practice, this is computed with first-order discretization (Euler method) with step size $\mathrm{d}t$:

$$\mathbf{x}_{t-\mathrm{d}t} \approx \mathbf{x}_t + p_\theta(\mathbf{x}_t, t)\mathrm{d}t. \tag{7}$$

Architecture details: Our denoiser module uses a standard attention-based U-Net (Ronneberger et al., 2015) comprising residual convolutional blocks where input images are always stacked in the channel dimension. Self-attention is applied to the visual features exclusively within the final two blocks of the encoder. Each residual block is composed of two 3×3 convolutions followed by a 2-factor pooling. The U-Net self-attention layers incorporate an extra cross-attention layer with shared KV projections to integrate the relational embeddings with the hidden features. Skip connections are implemented via channel concatenations between the corresponding encoder and decoder layers.

In preliminary experiments, we also explored the Diffusion Transformer (DiT) architecture (Peebles & Xie, 2023) as a backbone for the reasoning mechanism. However, it proved difficult to optimize this architecture even under baseline configurations for these specific spatial tasks. This observation also aligns with findings by Wewer et al. (2025), who noted similar challenges in reasoning-heavy domains. We leave the development of hybrid architectures, leveraging the inductive biases of convolutional layers alongside the scalability of high-complexity Transformers, for future work.

Training details: We use AdamW optimizer (Loshchilov & Hutter, 2019), learning rate of 1×10^{-4}, linear warm-up of 5,000 steps, and batch size of 128. For accelerated training, all denoisers are trained with the mixed precision (bfloat16). We train all models on the same number of steps.

B.4 Model complexity and parameters

We provide three size configurations for RDM to broadly investigate the impact of training small to large diffusion models for spatial reasoning tasks. Previous work like SRM (Wewer et al., 2025) rely on large U-Net backbones (120M parameters) for puzzles like Sudoku, while others (Goren et al., 2026; Lee et al., 2025) provide empirical evidence that visual reasoning can still be unlocked with small architectures (<50M parameters). We show two key findings through our experiments: (1) RDM is compatible with multiple backbone sizes, scaling from 30M to 120M parameters, and (2) larger architectures do not necessarily benefit reasoning and smaller architectures are preferable for faster convergence. We highlight these findings as a positive result for the AI community in terms of reproducibility and feasibility.

[6]We exclude in this notation the conditioning on the clue $\mathbf{x}_c$ and latents $\mathbf{z}^c$, $\mathbf{z}^0$ for better readability.

We set RDM and capacity-matched baselines to three size configurations: small (S), medium (M) and large (L). In both S and M, RDM maintains a compact discoverer module of <8M parameters (see Table 5) while the reasoner leverages 25M and 40M parameters, respectively (see Table 6). In the L setting, we expanded the *discoverer* dimension to $d = 512$ to study the impact of richer slot embeddings, and matched the diffusion backbone in L to exactly mirror the model from (Wewer et al., 2025). Due to the computational overhead, we adapted the training limit to each task (see Table 6).

Table 5: Architecture, training and performance details of the *Discoverer* in configurations S and M.

	k	d	channels	#conv/layer	#params.	#steps	Accuracy
Slot-Attention							
Akari	16	128	[16, 32, 64, 128]	1	1.4M	100K	99.58
Coldoku	16	128	[32, 64, 128, 256]	1	5.4M	400K	88.19
Tangram	7	128	[16, 32, 64, 128]	1	1.4M	300K	16.34
LogicFace	6	256	[32, 64, 128, 256]	2	7.6M	200K	72.54
Tildoku	20	128	[16, 32, 64, 128]	2	7.6M	400K	61.15
MNIST-Sudoku	16	128	[16, 32, 64, 128]	1	1.4M	200K	84.73
Counting Polygons	12	256	[32, 64, 128, 256]	1	5.4M	300K	80.67
Scalable Slot-Attention							
Akari	121	128	[16, 32, 64, 128]	1	1.4M	100K	100
Coldoku	81	128	[16, 32, 64, 128]	1	1.4M	200K	99.52

Table 6: Architecture and training details of the *Reasoner*.

	d	channels	#conv/layer	#params of $\mathcal{D}$	#params of $\mathcal{R}$	#params of U-Net	# steps
RDM$^{[S]}$	128	[64, 128, 256, 512]	[2, 2, 1, 1]	(Table 5)	4M	25M	300K
RDM$^{[M]}$	256	[64, 128, 256, 512]	[2, 2, 2, 2]	(Table 5)	4M	35M	300K
RDM$^{[L]}$							
Akari	512	[128, 256, 512, 512]	2	40M	15M	120M	40K
Coldoku	512	[128, 256, 512, 512]	2	40M	15M	120M	45K
Tangram	512	[128, 256, 512, 512]	2	40M	15M	120M	80K
LogicFace	512	[128, 256, 512, 512]	2	40M	15M	120M	40K
Tildoku	512	[128, 256, 512, 512]	2	40M	15M	120M	45K

Computational resources: All models were implemented in PyTorch 2.9 and trained using the CUDA 13.0 toolkit. Our experiments were conducted across multiple compute clusters consisting of NVIDIA L40, A40, A100 and H100 GPU nodes.

C Details on the Experimental Results

C.1 Results of the *Discoverer*

The reconstruction quality of the discoverer module is evaluated with the *cell accuracy* in the grid-like puzzles (Akari and Coldoku). For Tangram, the evaluation pipeline described in §A.3 is also used to evaluate the discoverer. For LogicFace, we use the ResNet-18 classifier (§A.4) on the reconstructed samples to match the real binary vector with the one predicted by the classifier.

In Akari reconstruction accuracies achieved 97% and 98% using 16 and 121 slots, respectively. For Coldoku, digit reconstruction accuracy reached 82.64% with 16 slots and 99.79% with 81 slots. However, color accuracy was significantly lower at 40.41% and 72.05%. We observed that the gap in color reconstruction correlates with observed blurriness in the outputs. Because the slot autoencoder uses a variational architecture with a simple convolutional decoder, it is prone to producing blurred digits with imprecise color values. Consequently, more errors occurred when reconstructed colors fell between two palette values, leading to misclassification by evaluator. We observed a similar behavior for Tangram, where the discoverer only achieved 57% of accuracy with our evaluation pipeline (§A.3). In practice the model showed real disentanglement (Figure 13), although reconstruction struggled to define sharp borders for the figures, leading to errors when estimating polygons with the

deterministic evaluator. For LogicFace, we evaluated the discoverer's reconstructions and obtained 95% of accuracy in the selected features (*smiling*, *male*, *glasses* and *young*).

Quality of object disentanglement: Figures 13 and 14 illustrate the decomposition performance of the trained slot autoencoders on the Tangram and LogicFace datasets. While Slot Attention is theoretically designed to ensure perfect object disentanglement, we observed that for Tangram, certain slots remained inactive while others encapsulated multiple tans. We hypothesize that because pieces vary in color and exhibit slight scale variances across samples, the slots may have fit specific chromatic patterns rather than individual geometric entities. Consequently, the model occasionally groups co-occurring tans into a single slot instead of maintaining a strictly one-to-one object correspondence.

Slot visualizations for LogicFace (Figure 14) depicts the inherent difficulty of object disentanglement in real images. We observed that salient, well-defined features from the CelebA-HQ dataset—such as glasses, smiling, and facial hair, are consistently isolated within specific slot embeddings. Conversely, the remaining slots struggle to decouple diffuse attributes, including skin tone and hair textures, or to model the high-variance backgrounds. This suggests that while Slot Attention effectively identifies discrete semantic anchors, it faces challenges in partitioning continuous or highly irregular global structures into distinct object-centric representations. We still find slot embeddings beneficial for reasoning, since key features were already represented into slots (glasses and smiling), while the others could be associated with other attributes (e.g., beard with *male*, gray hair with *not young*).

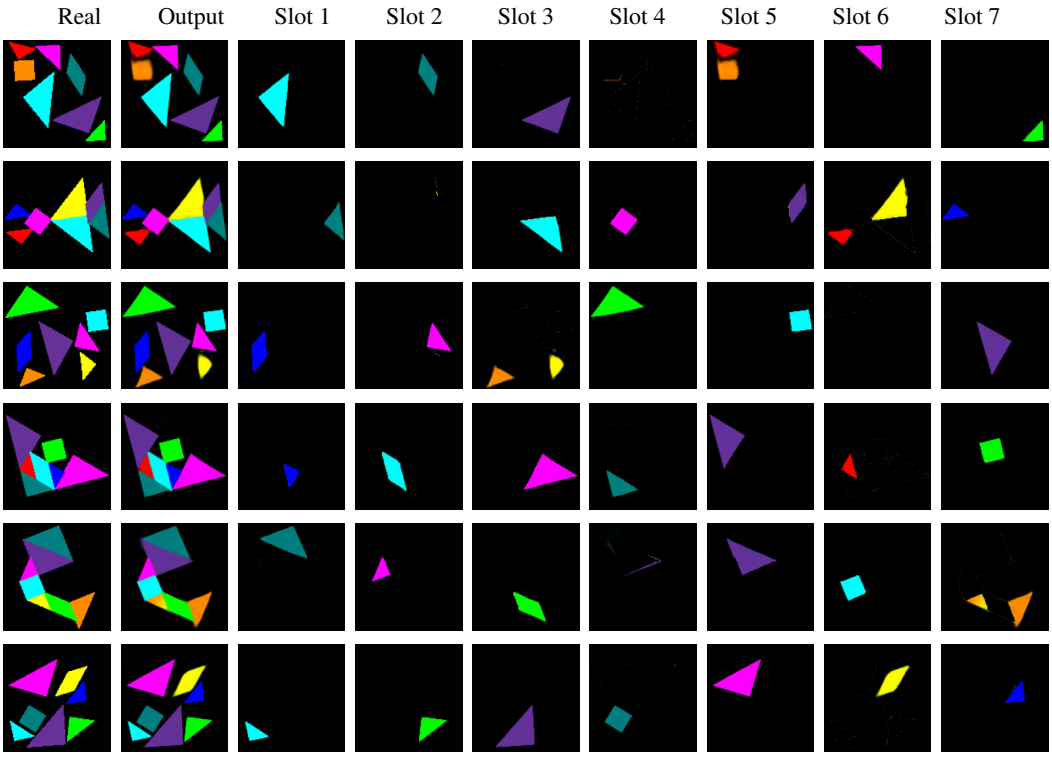


Figure 13: Disentanglement of the tans on Tangram inputs.

C.2 Results of the *Reasoner*

Tables 7–10 present detailed results for our reasoning benchmark with different sampling steps ($\tau \in \{10, 100\}$) in Akari, Tildoku, Tangram and LogicFace. We exclude sampling steps $\tau = 100$ from Tildoku since we did not appreciate significant changes from $\tau = 10$; and SRM performance in Tangram, since performance was significantly degraded in Table 2. We provide visualization of results obtained with our models on Tangram and LogicFace datasets in Figures 15 and 16.

Discussion on Tangram complexity: The majority of existing generative modeling literature focuses on Sudoku-style reasoning constraints for diffusion models (Wewer et al., 2025; Avdeyev et al.,

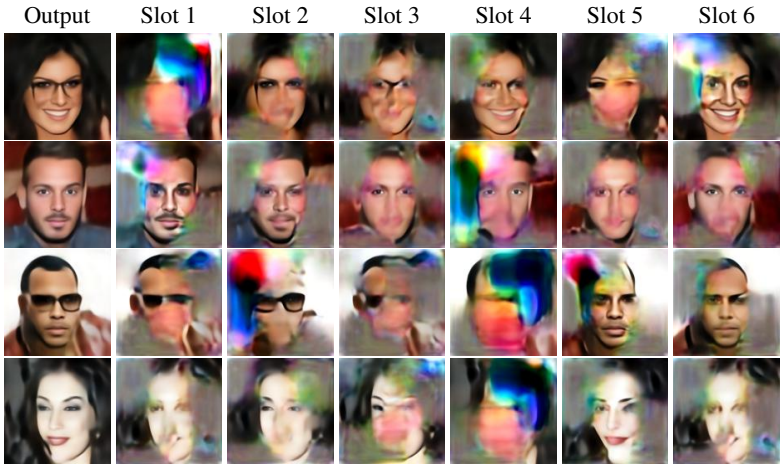


Figure 14: Disentanglement of the facial attributes on LogicFace inputs.

Table 7: Detailed results on Akari.

		ID					OoD									
							5-10% (low)					*35-40% (high)*				
		UNL	DST	OVP	BLB	ACC	UNL	DST	OVP	BLB	ACC	UNL	DST	OVP	BLB	ACC
$\tau = 10$	Base	0.46	0.10	0.03	74.08	67.35	3.22	0.22	0.48	17.02	13.80	0.17	0.04	0.01	89.78	86.80
	SRM															
	parallel	2.38	0.81	0.15	26.52	12.26	15.43	0.63	0.77	1.04	0.54	1.27	0.63	0.06	45.42	27.22
	sequential	7.01	9.81	17.79	0.00	0.00	0.26	4.64	87.15	0.00	0.00	13.87	13.47	6.28	0.00	0.00
	RDM[S]															
	16 slots clue	0.40	0.10	0.01	77.14	70.20	0.54	0.29	0.22	55.48	42.30	0.16	0.04	0.01	90.92	87.88
	16 slots solution	0.45	0.11	0.02	75.32	67.90	0.68	0.24	0.15	54.28	42.58	0.17	0.07	0.00	90.02	84.62
	16 slots full	0.30	0.13	0.04	79.14	69.98	0.45	0.21	0.41	50.06	41.32	0.12	0.07	0.00	92.64	87.02
	121 slots clue	0.42	0.09	0.02	77.73	72.17	1.69	0.59	0.09	42.57	24.96	0.15	0.04	0.01	91.00	87.94
	121 slots solution	0.52	0.13	0.02	71.70	63.02	1.19	0.22	0.14	42.74	34.16	0.29	0.07	0.01	85.16	80.78
	121 slots full	0.61	0.17	0.04	66.49	57.44	1.77	0.71	0.13	35.00	18.05	0.25	0.08	0.01	84.68	79.01
$\tau = 100$	Base	0.10	0.08	0.01	86.02	80.04	1.61	0.11	0.16	43.62	39.48	0.10	0.04	0.01	92.22	88.92
	SRM															
	parallel	1.86	0.68	0.11	35.82	18.80	15.43	0.57	0.40	1.60	0.94	0.98	0.58	0.06	54.18	33.34
	*sequential**	6.53	9.66	16.89	0.00	0.00	–	–	–	–	–	–	–	–	–	–
	RDM[S]															
	16 slots clue	0.14	0.08	0.01	88.30	82.06	0.15	0.18	0.02	85.98	72.48	0.09	0.05	0.01	92.90	88.94
	16 slots solution	0.18	0.09	0.01	86.10	78.68	0.19	0.15	0.03	82.20	70.73	0.09	0.07	0.01	93.33	87.96
	16 slots full	0.12	0.10	0.01	89.44	80.42	0.13	0.11	0.08	83.32	75.06	0.07	0.08	0.01	93.80	87.48
	121 slots clue	0.15	0.07	0.00	88.88	83.12	0.26	0.49	0.01	83.10	54.58	0.08	0.05	0.00	93.62	89.56
	121 slots solution	0.17	0.08	0.01	86.36	80.28	0.33	0.13	0.02	75.34	74.58	0.12	0.00	0.00	91.14	88.94
	121 slots full	0.18	0.09	0.01	85.84	77.34	0.27	0.34	0.04	77.30	56.20	0.10	0.07	0.01	98.76	87.40

*We do not report results of SRM (*sequential*) with $\tau = 100$ in *low* and *high* regimes due to already low performance in ID evaluation and the high computational overhead of inference.

2023; Lee et al., 2025). These works typically frame the task as an inpainting problem, where the model learns to satisfy localized relational rules within a fixed coordinate system. In our Tangram benchmark, generating valid solutions represents a significantly more complex challenge with fewer precedents in deep learning. Prior work, such as TANGAN (Yamada et al., 2025), used adversarial networks to partition silhouettes into geometric components. However, our benchmark requires the model to go beyond structural partitioning, and requires simultaneously perform geometric induction, maintain color-consistency across dynamic pieces and infer latent spatial transformations to align

Table 8: Detailed results on the Tildoku dataset.

ACC	*Easy (50-60)*			*Medium (40-50)*			*Hard (20-30)*		
	DGT	CLR	ACC	DGT	CLR	ACC	DGT	CLR	ACC
Base	59.20	66.84	40.28	22.62	45.92	12.48	1.50	9.42	0.24
SRM (*seq.*)	44.87	25.87	13.80	24.31	21.31	7.08	3.62	5.62	0.62
RDM [L]									
clue	62.52	79.74	50.28	33.02	52.84	19.18	3.78	10.12	0.64
solution	65.72	80.92	53.26	43.70	60.12	28.40	8.34	14.48	1.92
full	55.56	69.52	39.30	22.24	31.96	8.46	1.66	2.80	0.14

Table 9: Detailed results on Tangram.

		Configuration [S]							Configuration [L]						
	(*↑)	▱	□	$\triangle^S$	$\triangle^M$	$\triangle^L$	SHAPE	ACC	▱	□	$\triangle^S$	$\triangle^M$	$\triangle^L$	SHAPE	ACC
	Base	17.96	17.82	29.78	34.98	39.33	40.09	12.37	16.08	15.24	21.78	24.64	28.08	28.64	11.46
$\tau = 10$	RDM														
	clue	24.84	25.37	37.55	40.71	41.90	42.41	22.20	34.72	34.26	41.80	43.34	44.88	44.98	31.96
	solution	31.43	31.80	36.73	38.69	40.06	40.51	28.45	33.34	32.66	39.90	41.64	43.16	43.52	29.94
	full	34.49	34.69	38.65	42.67	43.82	4.31	30.18	22.20	21.38	33.26	36.28	40.48	41.44	16.74
	Base	23.06	24.22	42.82	41.50	55.60	57.72	15.46	24.74	23.06	40.34	38.62	47.74	48.02	19.00
$\tau = 100$	RDM														
	clue	36.71	36.96	59.59	60.33	63.29	58.41	34.37	42.62	41.54	60.78	59.46	63.02	63.04	40.04
	solution	41.12	43.94	56.49	53.06	59.94	60.25	38.57	42.18	40.18	40.44	60.04	58.50	62.74	38.84
	full	49.75	49.98	61.55	61.25	65.00	65.12	46.76	31.44	30.32	53.18	52.34	61.54	61.96	25.00

tans with a target configuration. The low baseline accuracy reported in Table 2 clearly reflects this increased dimensionality.

We conducted preliminary experiments to better understand the complexity of Tangram. We designed two intermediate level benchmarks to isolate the challenges introduced in the main task (geometric partitioning and color consistency). The first level resembles the TANGAN approach (Yamada et al., 2025), where the model is tasked with generating a valid partition of a given silhouette. We evaluated both a standard convolutional autoencoder (AE) and our baseline diffusion model on this task, using the silhouette as the sole conditional input and using solutions where a unique color was always associate to the same tan. Under this setup, the AE achieved only 1.93% of accuracy, whereas the diffusion model reached 82%. These results show that while traditional reconstruction-based architectures struggle with geometric arrangements, diffusion models are highly capable of learning complex partitions.

In our second level, we introduced the additional challenge of chromatic modeling. Here, the model is provided with the target silhouette alongside *a fixed piece view* of the tans, where only the colors vary across samples. The diffusion model is tasked with generating a valid partition of the silhouette, while matching the specific colors provided in the tan view. Here, the model only needs to learn the fixed proportions of the pieces and the mapping of their associated colors. We consider this level simpler than our Tangram task, since the tan view does not remain fixed across samples in our task.

Under this configuration, the diffusion model's partitioning accuracy dropped to 54%, with a total accuracy of 52%. Accordingly, we infer that the key challenge in Tangram appears when different color permutations are associated with the same silhouette partition. This suggests that the model struggles to establish a robust cross-modal correspondence between the conditional input and the denoising target. Specifically, the increased difficulty stems from the model's inability to consistently correlate the arbitrary colors in the tan view with the final spatial arrangement of the pieces.

Table 10: Detailed results on LogicFace.

	(*↑)	Configuration [S] *smiling* (OR)	*male* (AND)	*glasses* (XOR)	*young* (IMP.)	AVG	ACC	FID	Configuration [L] *smiling* (OR)	*male* (AND)	*glasses* (XOR)	*young* (IMP.)	AVG	ACC	FID
	Base	71.28	79.39	60.77	73.50	71.24	26.03	44.95	55.08	55.42	39.24	70.28	54.51	8.62	35.92
$\tau = 10$	RDM														
	clue	87.74	96.66	93.61	81.67	89.92	66.08	37.66	83.34	94.52	49.60	70.02	74.37	27.98	37.18
	solution	68.53	78.34	49.81	72.10	67.19	18.98	50.21	65.41	66.44	49.41	66.05	61.83	14.95	39.71
	full	84.97	94.74	90.81	78.55	87.27	59.57	36.37	78.38	91.74	50.68	65.66	71.61	25.08	38.04
	Base	68.18	76.31	64.86	71.09	70.11	25.13	21.00	54.50	52.06	39.70	69.52	53.94	8.36	33.08
$\tau = 100$	SRM														
	parallel	37.00	38.16	49.26	39.44	40.97	2.60	109.48	–	–	–	–	–	–	–
	sequential	25.04	58.38	50.58	73.86	51.94	5.70	191.03	–	–	–	–	–	–	–
	RDM														
	clue	90.19	97.00	97.26	87.27	92.93	74.74	13.34	82.50	94.30	49.74	66.58	73.28	26.60	35.70
	solution	64.87	73.20	50.16	67.78	64.00	16.27	19.14	65.32	66.16	49.38	66.02	61.72	14.98	37.19
	full	83.16	94.00	94.76	79.81	87.93	60.76	14.72	82.16	92.12	50.00	68.84	73.23	25.66	36.03

C.2.1 Additional Experiments

We present further experimental analyses by: (1) isolating the impact of relational abstractions via an ablation study where we evaluate the reasoning capabilities of purely slot-conditioned diffusion guidance, (2) investigating empirical evidence of symbolic rule learning by the relational abstractions module, and (3) evaluations on previous spatial reasoning benchmarks MNIST-Sudoku and Counting Polygons (Wewer et al., 2025) for completeness.

Impact of relational abstractions: We demonstrate that diffusion is not only compatible with relational abstractions over object-centric embeddings, but also benefits from these contextual representations in solving reasoning problems. Previous works by Wu et al. (2023) and Jiang et al. (2023) have proposed slot-conditioned diffusion models for image synthesis, although these were not adapted to CFG or object disentanglement for higher-level abstraction and reasoning. To isolate the impact of *relational abstractions*, we also benchmarked a purely *slot-based* conditioned diffusion model, i.e., $p_\theta(\mathbf{x}_0|\mathbf{x}_c, \mathbf{s}^c_{1:k})$ at training and inference. In Table 11, we demonstrate an interesting result which was not investigated in previous works, that purely slot conditioning alone can already improve reasoning upon a standard baseline. Nevertheless, our results show that the slot-based approach still underperforms relative to RDM. This performance gap is particularly pronounced in the Tangram dataset, where RDM outperforms slot-conditioning (*clue*: 34.37 vs 27.09). These results suggest that explicit relational abstraction between objects and their spatial configurations is critical for complex reasoning.

Table 11: RDM vs slot-based diffusion.

ACC	Coldoku ($\tau = 10$) *Easy*	*Medium*	*Hard*	Tangram ($\tau = 100$)
Base	90.84	48.72	3.43	15.46
Slot-based	93.76	54.94	4.74	27.09
RDM* (*clue*)	**96.66**	**65.62**	**7.76**	**34.37**

*RDM[S] for Tangram, RDM[L] for Coldoku.

Empirical evidence of rule learning: Beyond our general evaluations of reasoning performance based on rule-consistent image synthesis, we further explore whether the learned embeddings by the relational abstraction modules truly capture the logical rules required for reasoning. We investigate this aspect in three different experiments with RDM. In our first experiment, we perform sampling from $p_\theta(\mathbf{x}_0|\mathbf{x}_c, \mathcal{R}(\mathbf{z}^0))$ with RDM (*solution*, by enabling access to the relational solution embeddings at inference time. Under this setting, accuracy boosted to 93% in Tangram, and 95% in Coldoku (*hard*), evidencing that RDM learns to abstract representations to almost perfectly solve both puzzles. In the second experiment, we used the Tangram dataset and removed the raw visual clues $\mathbf{x}_c$ while training RDM to model $p_\theta(\mathbf{x}_0|\mathcal{R}(\mathbf{z}^c))$. This approach yielded ACC: 37.55 by surpassing again the Base model (15.46), and showing that the relational abstractions alone are powerful enough to replace the clue itself and improve reasoning. In our third experiment, we conducted linear probing tests on Coldoku. We froze the abstractor stack and trained a linear classifier to distinguish the relational embeddings produced by 6,400 valid and 6,400 invalid puzzles, where invalid puzzles were artificially generated to violate Coldoku constraints. In the first setting, invalid puzzles were

obtained by arranging the same digit in all columns. In the second setting, the digit arrangements were randomly shuffled across the grid. We trained simple linear classifiers on these high-dimensional ($k \times d$) embeddings, and achieved 89.98% in the first setting, and 82.59% in the second setting. These results indicate that the learned relational representations encode information that makes valid versus rule-violating puzzles linearly separable. Altogether, we treat these results as additional evidence that learned relational representations encode meaningful logical rule information for reasoning, and justify their effectiveness when guiding diffusion models.

Evaluations on previous benchmarks: Table 12 presents the evaluation of RDM on previous spatial reasoning benchmarks (Wewer et al., 2025): MNIST-Sudoku (*medium* and *hard* regimes), and Counting Polygons. In MNIST-Sudoku we report the original metrics and the evaluation of SRM under our setup ($\tau = 100$ and 5,000 test samples). In Counting Polygons, we report metrics based on the original neural evaluator and our unbiased neural evaluator[7]. With less than half of parameters, RDM outperforms SRM in the *medium* and *hard* regimes, while slightly surpasses the Base model in the *medium* regime (55.80 vs 54.18). We noticed that diffusion forcing only benefits performance with a high number of sampling steps, as the accuracy gap disappears when sampling with 100 steps. In contrast, RDM considerably outperforms SRM and the Base diffusion model in Counting Polygons (30.7 vs 10.1), where pure diffusion forcing (*parallel*) and SRM *sequential* no longer benefit the denoising process.

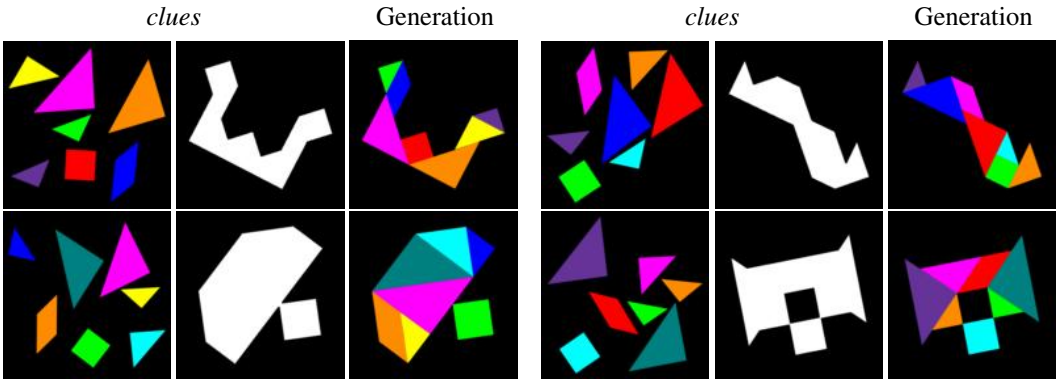
Table 12: Results on previous benchmarks.

	MNIST-Sudoku		Counting Polygons	
	Medium	*Hard*	*Original*	*Unbiased*
Base	54.18 (*53.6*)	**1.9** (*0.8*)	13.2	9.8
SRM				
parallel	31.04 (*59.0*)	0.00 (*1.0*)	16.6	10.1
sequential	40.02 (*75.4*)	0.01 (*51.6*)	18.6	5.4
Non-object-centric features				
DINOv2	44.26	0.18	–	–
ViT-MAE	44.60	0.08	–	–
RDM$^{[M]}$				
clue	47.34	0.22	–	–
solution	47.14	0.06	**51.2**	**30.7**
full	**55.80**	0.76	–	–

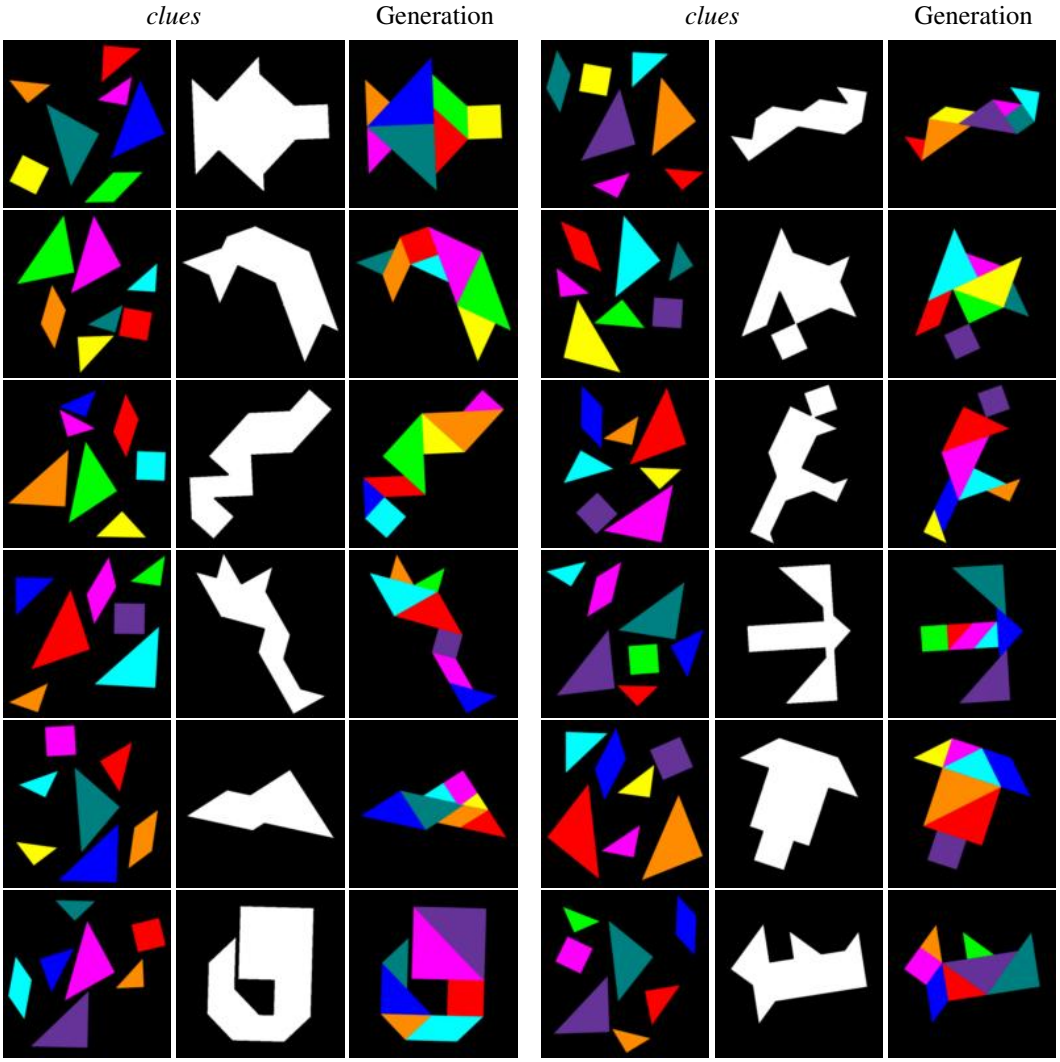
Reported results from Wewer et al. (2025) in parentheses.

Finally, we conducted additional ablations on MNIST-Sudoku to demonstrate the benefits of object-centric representations for diffusion guidance. We perform conditioning via non-object-centric visual features from DINOv2 (Oquab et al., 2024) and ViT-MAE (He et al., 2021). Table 12 shows these representations do not lead to substantial improvements in either the *medium* or the *hard* regime. Overall, performance is still worse than the Base model and RDM, demonstrating that object-centric relational representations are superior to non-disentangled image features for reasoning.

[7]We observed that the original evaluator for Counting Polygons (Wewer et al., 2025) was biased to always predict a positive number of polygons and exactly two digits, even when no clear shape or digit was present, leading to optimistic evaluations. We therefore trained an *unbiased* evaluator that handles cases when no digit or shape appears in the image, and report evaluations with both.

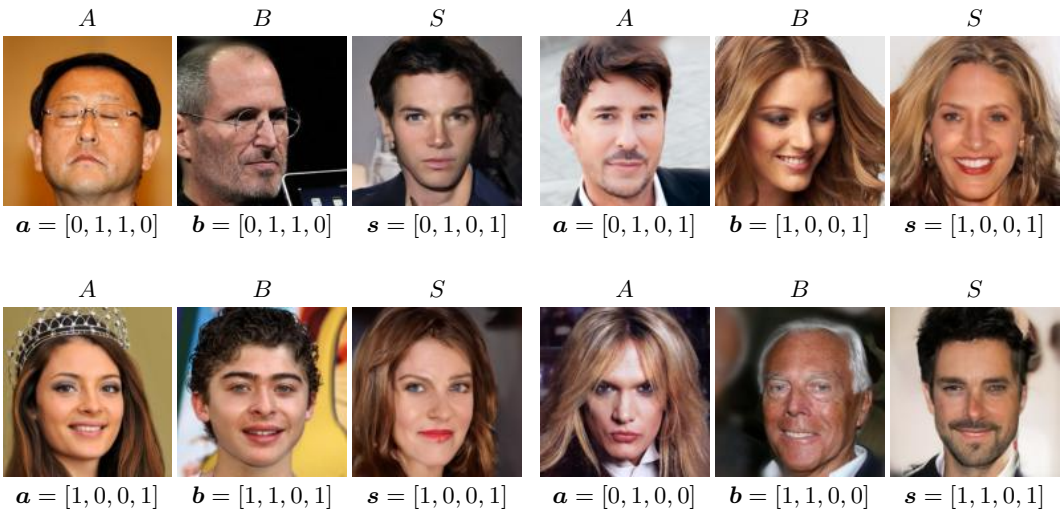


(a) Examples from the *Baseline*.



(b) Examples from RDM$^{[L]}$ with the *clue* based abstraction conditioning approach.

Figure 15: Examples of generated Tangram solutions.



(a) Examples from the *Baseline*.



(b) Examples from RDM$^{[\mathrm{L}]}$ with the *clue* based abstraction conditioning approach.

Figure 16: Examples of generated LogicFace solutions shown in S, when conditioned on A and B.