Invariant Slot Attention: Object Discovery with Slot-Centric Reference Frames

Ondrej Biza, Sjoerd van Steenkiste, Mehdi S. M. Sajjadi, Gamaleldin F. Elsayed,
Aravindh Mahendran, Thomas Kipf
NeurIPS 2022 Workshop, arXiv 2023 (ICML 2023 submission)

STRUCT Group Seminar
Presenter: Rundong Luo
2023.03.19

Outline

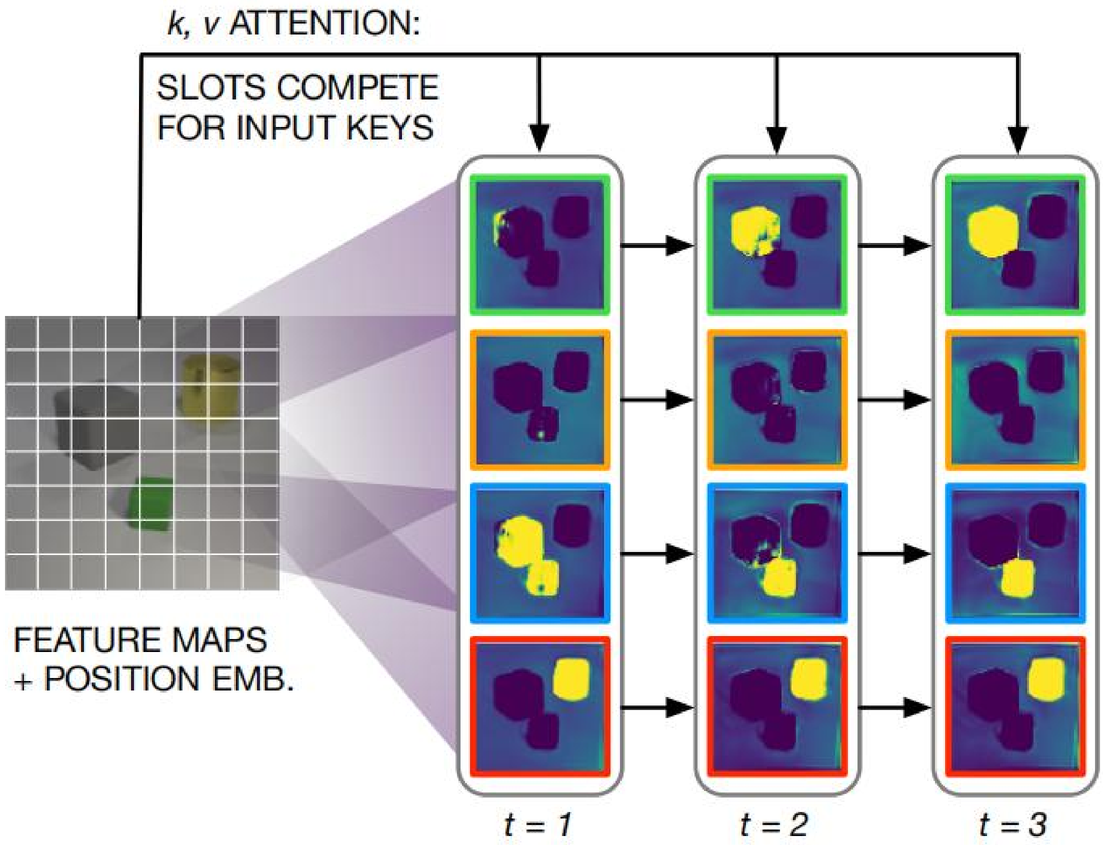
- Authorship
- Background
- Method
- Experiments
- Conclusion

Outline

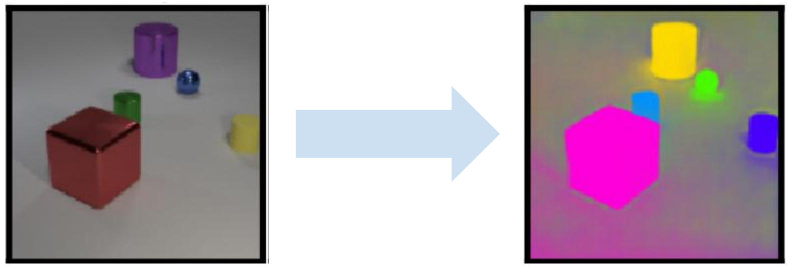
- Authorship
- Background
- Method
- Experiments
- Conclusion

Slot Attention (1/4)

- Object-Centric Learning with Slot Attention (NeurIPS 2020)

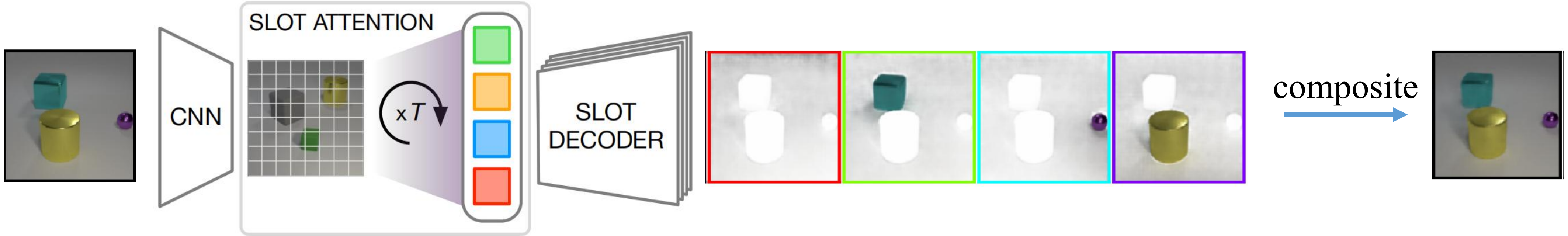


- Task: unsupervised object discovery, i.e., uncovering patterns that define objects and discriminates them against the background.
- More specifically, seperate an image to sets of pixels.



Slot Attention (2/4)

- Slots: Latent vector that describe a single object



- Encoder: Vanilla CNN, ResNet, etc. Output features of size $D_{inputs}*H_0*W_0$
- Slot attention module. Output k slots, each with size D_{slots}

1) Sampling: Sample all object latents (i.e., slots) from the same prior distribution to encourage representational uniformity across all slots. $\texttt{slots} \sim \mathcal{N}(\mu, \operatorname{diag}(\sigma)) \in \mathbb{R}^{K \times D_{\texttt{slots}}}$

2) Binding: Bound each slot to an object region via an attention mechanism.

3) Updating: Each slot gets updated by the bound object features to specialize for that object.

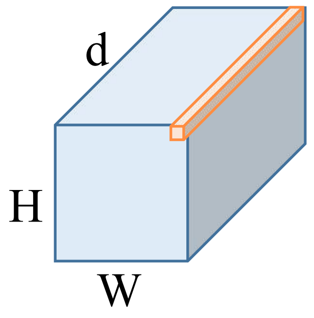
- Decoder: TransposeConvNets. Given a slot latent, broadcast it into an initial size (e.g, 8*8), then upsample to the original size. For each slot, output: $4*H*W$ (R, G, B, alpha). Alpha is a mask for compositing slots and determine the attribution of pixel.

Slot Attention (3/4)

- Primary Idea: Attention machanism
  - *Slots compete for explaining parts of the input*
  - Slots as queries
  - Features as keys&values (per input position, extracted by CNN, with positional embeddings)

add positional encoding

keys = $\mathcal{K}\Bigg($

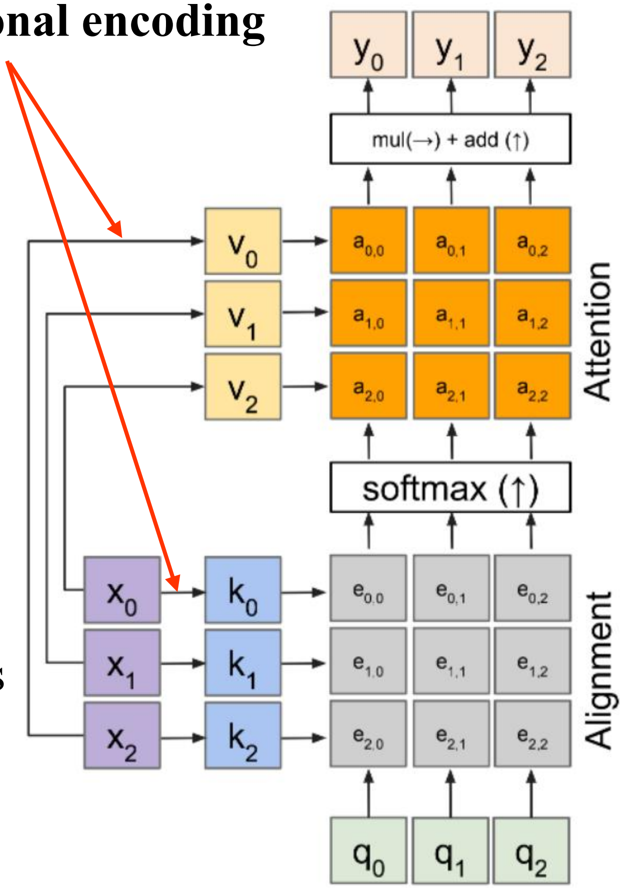


$$+ \; g\left(\begin{bmatrix} (-1,-1) & (-0.9,-1) & \cdots & (1,-1) \\ (-1,-0.9) & (-0.9,-0.9) & \cdots & (1,-0.9) \\ \vdots & \vdots & \ddots & \vdots \\ (-1,1) & (-0.9,1) & \cdots & (1,1) \end{bmatrix}\right)\Bigg)$$

input features

positional encoding

input features



slots

Slot Attention (4/4)

- Pseudo code for attention module

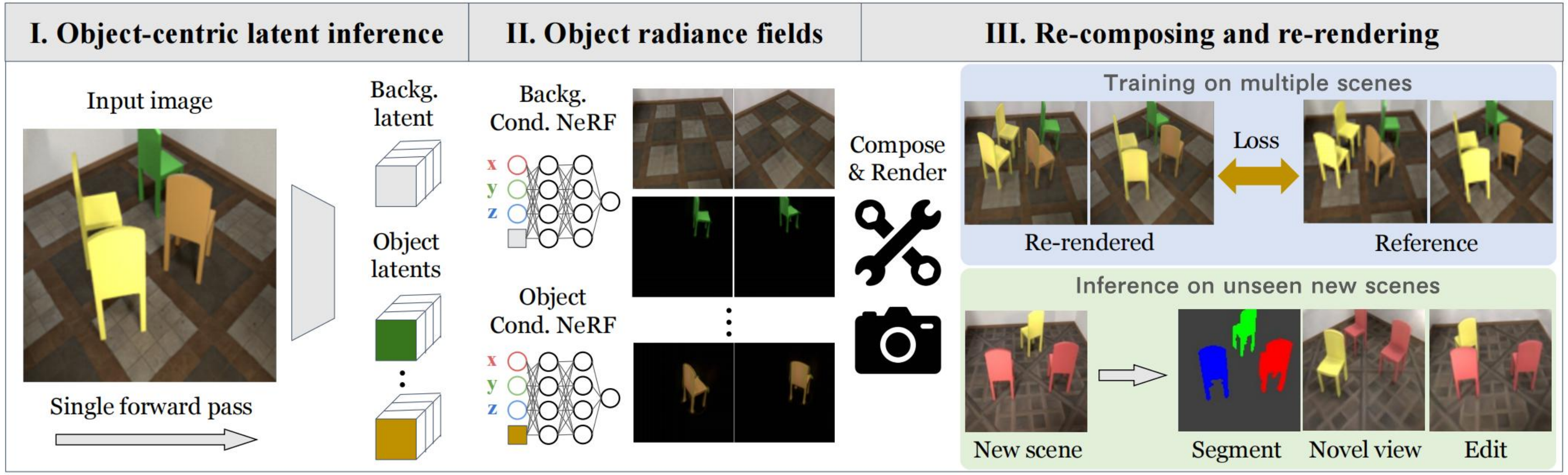
Input: $\texttt{inputs} \in \mathbb{R}^{N \times D_{\texttt{inputs}}}$, $\texttt{slots} \sim \mathcal{N}(\mu,\ \mathrm{diag}(\sigma)) \in \mathbb{R}^{K \times D_{\texttt{slots}}}$
Layer params: $k,\ q,\ v$: linear projections for attention; GRU; MLP; LayerNorm (x3)
    $\texttt{inputs} = \texttt{LayerNorm}(\texttt{inputs})$
    for $t = 0 \ldots T$
        $\texttt{slots\_prev} = \texttt{slots}$
        $\texttt{slots} = \texttt{LayerNorm}(\texttt{slots})$
        $\texttt{attn} = \texttt{Softmax}\left(\frac{1}{\sqrt{D}} k(\texttt{inputs}) \cdot q(\texttt{slots})^T,\ \texttt{axis='slots'}\right)$ # norm. over slots
        $\texttt{updates} = \texttt{WeightedMean}(\texttt{weights=attn} + \epsilon,\ \texttt{values=}v(\texttt{inputs}))$ # aggregate
        $\texttt{slots} = \texttt{GRU}(\texttt{state=slots\_prev},\ \texttt{inputs=updates})$ # GRU update (per slot)
        $\texttt{slots} \mathrel{+}= \texttt{MLP}(\texttt{LayerNorm}(\texttt{slots}))$ # optional residual MLP (per slot)
    return slots

LayerNorm: Normalize feature instead of batch

GRU: Gated Recurrent Unit (RNN with gates, similar to LSTM)

Slot Attention in 3D (1/1)

- Unsupervised Discovery of Object Radiance Fields (ICLR 2022)

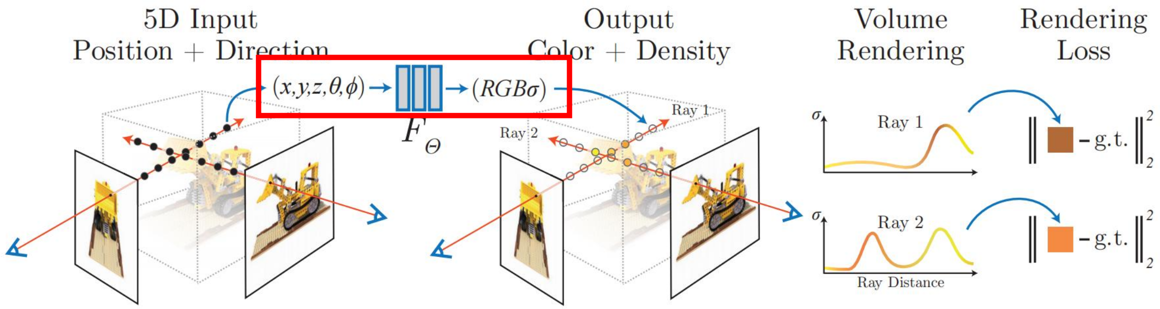


Single image NeRF for scene editing & synthesis

- Given a single reference image, extract *slot latents* for scence segmentation, decompostion, etc.
- Condition NeRF on the slot latent
- Trained on reconstruction loss

Slot Attention in 3D (2/2)

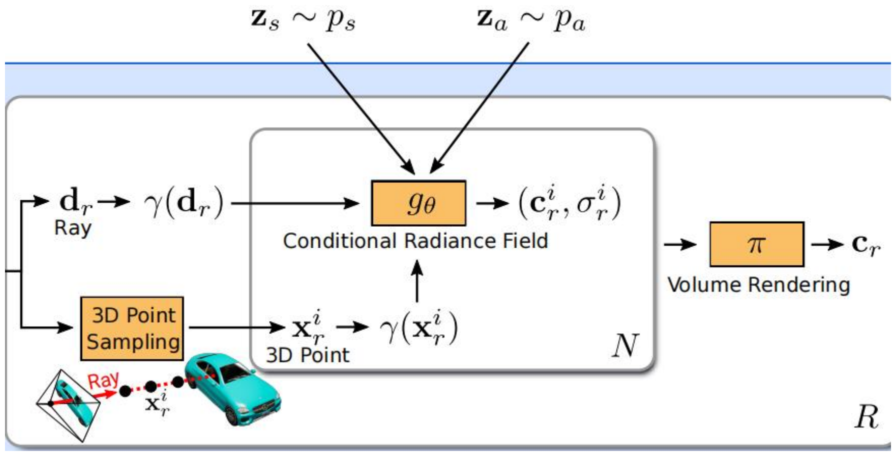
- NeRF Overview



- Volume rendering

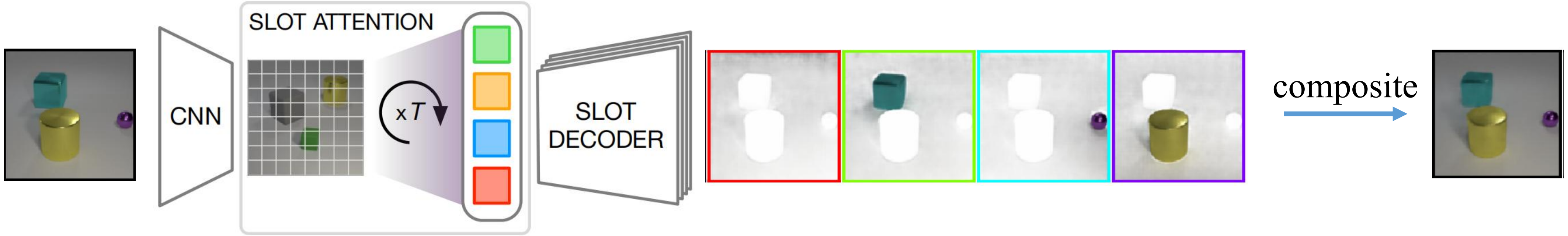
$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt\,, \text{ where } T(t) = \exp\left(-\int_{t_n}^{t} \sigma(\mathbf{r}(s))ds\right).$$

- Conditional NeRF



*

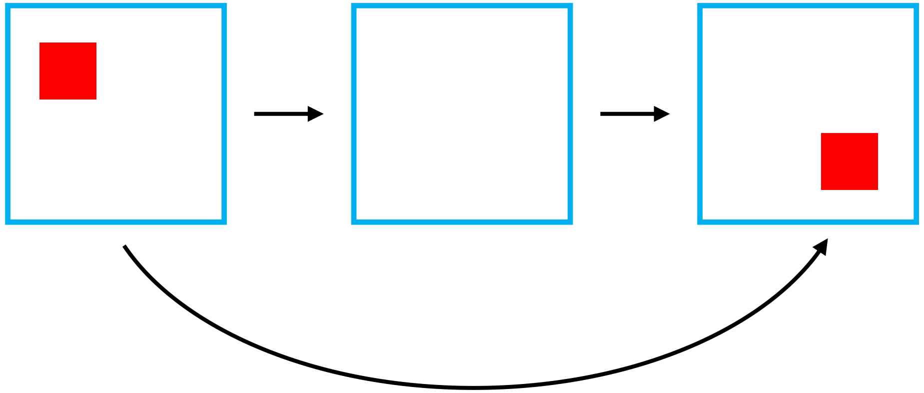
Slot Attention: Conclusion



One sentence summary: use slots to explain objects.

Problems of slot attention

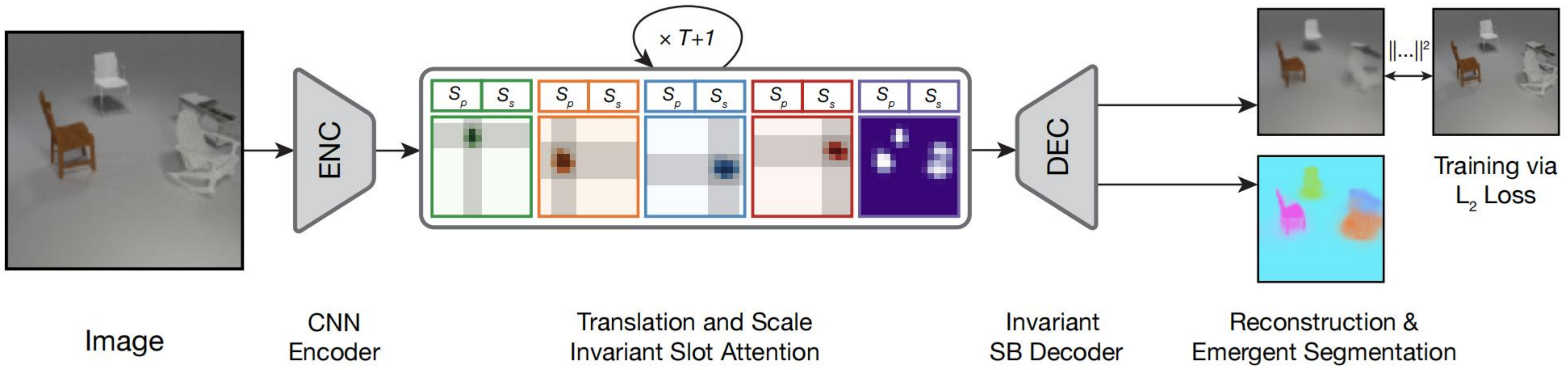
- Slot is a mixture of object information ➡ How to disentangle "slot"?



Outline

- Authorship
- Background
- Method
- Experiments
- Conclusion

Method: Overview (1/6)



- Disentangle *slot's* appearance with its position, scale, and rotation.
- Specifically, learn the position, scale, and rotation (S_p, S_s, S_r) of each slot

$$S_p \in \mathcal{R}^2, S_s \in \mathcal{R}^2, S_r \in [0, \pi]$$

- As a result, slot feature is invariant to position, scale, and rotation.

Method: Invariance to Translation and Scaling (2/6)

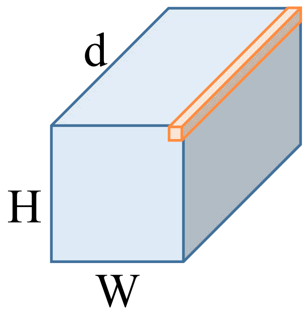
- In original slot attention, we add positional encoding to each input feature
- Relative positional encoding for each slot (K slots in total)

$$\mathrm{rel\_grid}^k = (\mathrm{abs\_grid} - S_p^k) \;/\; S_s^k \qquad k \in \{1, ..., K\}$$

$$\begin{bmatrix} (-1,-1) & (-0.9,-1) & \cdots & (1,-1) \\ (-1,-0.9) & (-0.9,-0.9) & \cdots & (1,-0.9) \\ \vdots & \vdots & \ddots & \vdots \\ (-1,1) & (-0.9,1) & \cdots & (1,1) \end{bmatrix} \xrightarrow{S_p^0 = (0.5, 0.5),\, S_s^0 = (1,1)} \begin{bmatrix} (-1.5,-1.5) & (-1.4,-1.5) & \cdots & (0.5,-1.5) \\ (-1.5,-1.4) & (-1.4,-1.4) & \cdots & (0.5,-1.4) \\ \vdots & \vdots & \ddots & \vdots \\ (-1.5,0.5) & (-1.4,0.5) & \cdots & (0.5,0.5) \end{bmatrix}$$

Uniform positional encoding — Relative positional encoding for slot 0

$\mathrm{keys}^0 = \mathcal{K}\Bigg($



$$+\; g\left(\begin{bmatrix} (-1.5,-1.5) & (-1.4,-1.5) & \cdots & (0.5,-1.5) \\ (-1.5,-1.4) & (-1.4,-1.4) & \cdots & (0.5,-1.4) \\ \vdots & \vdots & \ddots & \vdots \\ (-1.5,0.5) & (-1.4,0.5) & \cdots & (0.5,0.5) \end{bmatrix}\right)\Bigg)$$

input features — relative positional encoding

Method: Invariance to Translation and Scaling (3/6)

- Obtain slot keys and values

$$\text{keys}^k = f\left(\mathcal{K}(\text{inputs}) + g(\text{rel\_grid}^k)\right)$$

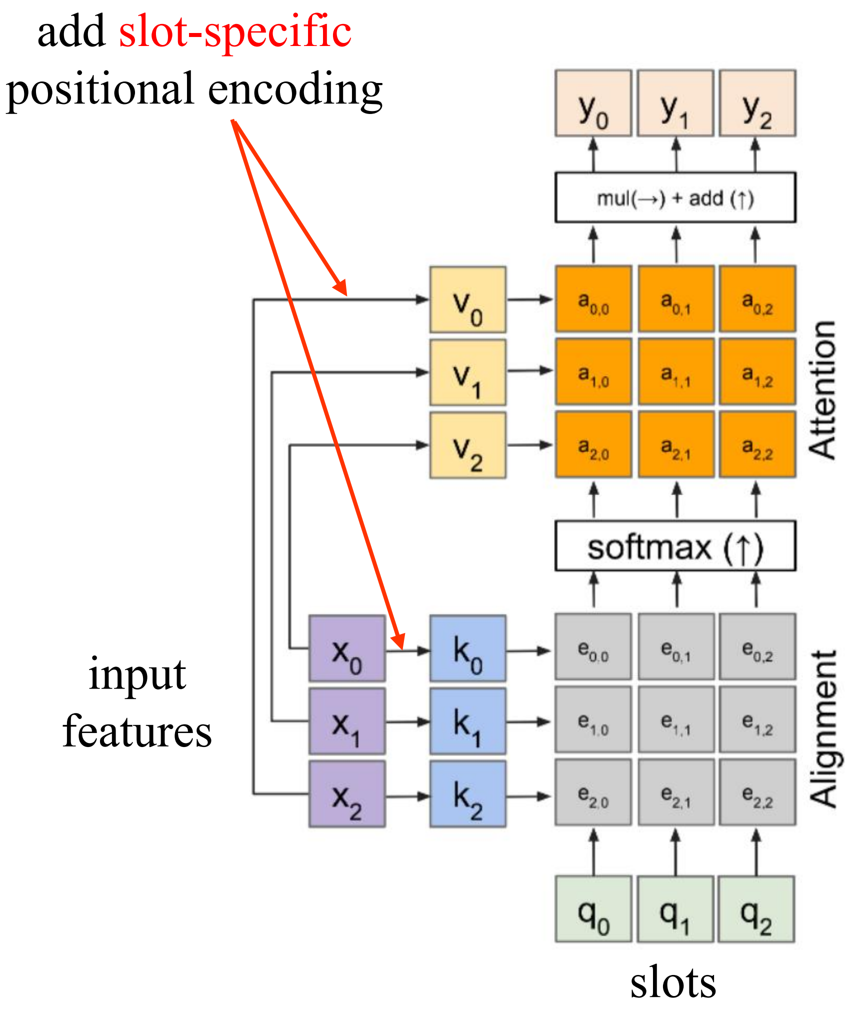
$$\text{values}^k = f\left(\mathcal{V}(\text{inputs}) + g(\text{rel\_grid}^k)\right)$$

- Obtain attention and updates
- Update slots by GRU
- Update slots' position and scale latent

$$S_p = \frac{\sum_{n=1}^{N} \text{attn}_n * \text{abs\_grid}_n}{\sum_{n=1}^{N} \text{attn}_n}$$

$$S_s = \sqrt{\frac{\sum_{n=1}^{N} (\text{attn}_n + \epsilon) * (\text{abs\_grid}_n - S_p)^2}{\sum_{n=1}^{N} (\text{attn}_n + \epsilon)}}$$

Intuitive explanation: move slot to the place with higher attention



Method: Invariance to Rotation (4/6)

- Encode rotation (S_r)
  - Heuristic: the orientation of a slot is given by the axis with the highest variation

$$v_1^k, v_2^k = \text{WeightedPCA}(\text{abs\_grid}, \text{attn}^k)$$

eigenvector of the covariance matrix

$$\tilde{v}_1^k, \tilde{v}_2^k = \text{post-process}\left(v_1^k, v_2^k\right) ,$$

$$S_r^k = \begin{bmatrix} | & | \\ \tilde{v}_1^k & \tilde{v}_2^k \\ | & | \end{bmatrix} .$$

$$\text{rel\_grid} = [S_r^{-1}(\text{abs\_grid} - S_p)] \,/\, (S_s \times \delta)$$

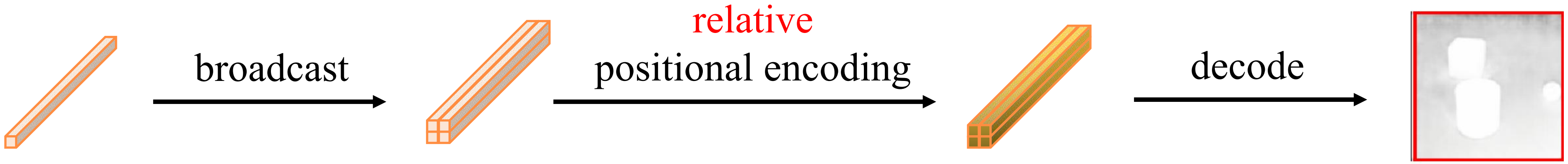
- Author's claim: Not effective enough
- My opinion: Unreasonable to encode rotation in 3D by 2D rotation matrix

Method: Decoding (5/6)

- Decode each slot
  - Calculate *rel_grid* for current slot
  - Spatially broadcast slot with relative positional encoding
  - Decode the RGB value and alpha mask (an image per slot)

$$(R, G, B, \alpha) = D_\phi(\mathrm{SB} + h(\mathrm{rel\_grid})) \qquad \mathrm{rel\_grid}^k = (\mathrm{abs\_grid} - S_p^k) \;/\; S_s^k$$

- Alpha composite the images



Method (6/6)

Algorithm 1 Translation, Rotation, and Scaling Invariant Slot Attention.

Inputs: inputs $\in \mathbb{R}^{N \times D_{inputs}}$, abs_grid $\in \mathbb{R}^{N \times 2}$, slots $\in \mathbb{R}^{K \times D_{slots}}$, Slot positions, $S_p \in \mathbb{R}^{K \times 2}$, Slot rotations, $S_r \in \mathbb{R}^{K \times 2 \times 2}$, Slot scales, $S_s \in \mathbb{R}^{K \times 2}$, T iterations, small ϵ.
Data: Encoders f, g, k, v, q, parameters of LayerNorms, MLP and GRU, δ.
Outputs: slots $\in \mathbb{R}^{K \times D_{slot}}$, $S_p \in \mathbb{R}^{K \times 2}$, $S_r \in \mathbb{R}^{K \times 2 \times 2}$, $S_s \in \mathbb{R}^{K \times 2}$.

inputs $=$ LayerNorm(inputs)
for $t = 1$ **to** $T + 1$ **do**
    slots_prev $=$ slots
    slots $=$ LayerNorm(slots)

    # Computes relative grids per slot, and associated key, value embeddings.
    rel_grid $= [S_r^{-1}(\text{abs\_grid} - S_p)] \;/\; (S_s \times \delta)$
    keys $= f\,(k(\text{inputs}) + g(\text{rel\_grid}))$
    values $= f\,(v(\text{inputs}) + g(\text{rel\_grid}))$

    # Inverted dot production attention.
    attn $= \text{softmax}(\frac{1}{\sqrt{K}}\text{keys} * q(\text{slots})^T, \text{axis} = \text{“slots”})$
    updates $= \text{WeightedMean}(\text{weights} = \text{attn}, \text{values} = \text{values})$
    attn $/= \text{Sum}(\text{attn}, \text{axis} = \text{“inputs”})$

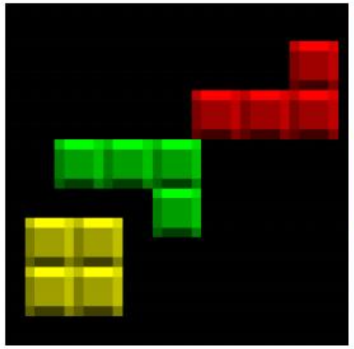
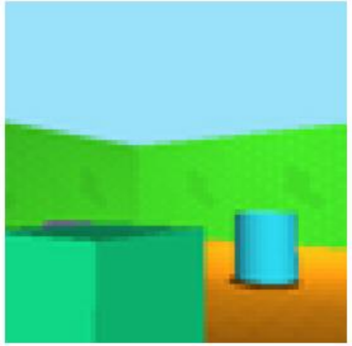
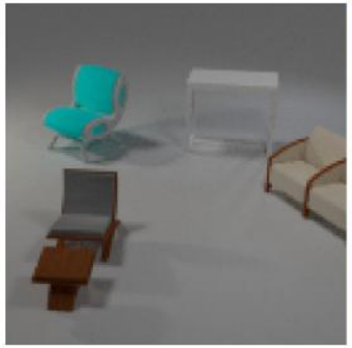
    # Updates S_p, S_s and slots.
    $S_p = \text{WeightedMean}(\text{weights} = \text{attn}, \text{values} = \text{abs\_grid})$
    $S_r = \text{Symmetrize}(\text{WeightedPCAAnalytical}(\text{inputs} = \text{abs\_grid} - S_p, \text{weights} = \text{attn})$
    $S_s = \sqrt{\text{WeightedMean}(\text{weights} = \text{attn} + \epsilon, \text{values} = [S_r^{-1}(\text{abs\_grid} - S_p)]^2)}$
    if $t < T + 1$ **then**
        slots $= \text{GRU}(\text{state} = \text{slots\_prev}, \text{inputs} = \text{updates})$
        slots $+= \text{MLP}(\text{LayerNorm}(\text{slots}))$
    end if
end for

Outline

- Authorship
- Background
- Method
- Experiments
- Conclusion

Eeperiments (1/6)

- Datasets
  - Tetrominos
  - Objects Room
  - MultiShapeNet
  - CLEVR
  - CLEVR Tex
  - Waymo Open (real-world, only for qualitative)
- Evaluation Protocol
  - Qualitative
  - Quantitative (FG-ARI)

Eeperiments (2/6)

- Rand Index: Calculate the similarity between two partitions of a set

Given a set of n elements $S = \{o_1, \ldots, o_n\}$ and two partitions of S to compare, $X = \{X_1, \ldots, X_r\}$, a partition of S into r subsets, and $Y = \{Y_1, \ldots, Y_s\}$, a partition of S into s subsets, define the following:

- a, the number of pairs of elements in S that are in the **same** subset in X and in the **same** subset in Y
- b, the number of pairs of elements in S that are in **different** subsets in X and in **different** subsets in Y
- c, the number of pairs of elements in S that are in the **same** subset in X and in **different** subsets in Y
- d, the number of pairs of elements in S that are in **different** subsets in X and in the **same** subset in Y

The Rand index, R, is:[1][2]

$$R = \frac{a+b}{a+b+c+d} = \frac{a+b}{\binom{n}{2}}$$

- Adjusted Rand Index:

$X \diagdown Y$	Y_1	Y_2	$\cdots$	Y_s	sums
X_1	n_{11}	n_{12}	$\cdots$	n_{1s}	a_1
X_2	n_{21}	n_{22}	$\cdots$	n_{2s}	a_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
X_r	n_{r1}	n_{r2}	$\cdots$	n_{rs}	a_r
sums	b_1	b_2	$\cdots$	b_s	

Definition [edit]

The original Adjusted Rand Index using the Permutation Model is

$$ARI = \frac{\sum_{ij} \binom{n_{ij}}{2} - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right] / \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right] / \binom{n}{2}}$$

Experiments: Quantitative (3/6)

Table 1. CLEVRTex FG-ARI(%) results on the test set, CAMO set (objects and backgrounds blend together) and OOD set (novel textures). Prior results taken from (Karazija et al., 2021) use 3 random seeds, we use 10 random seeds. FG-ARI is reported in %. For MSE please see the Table 8. (CNN) refers to models using a 4-layer CNN backbone, while (ResNet) models use a ResNet-34.

Method	**Main**	**CAMO**	**OOD**
SPACE	17.5 ±4.1	10.6 ±2.1	12.7 ±3.4
DTI	79.9 ±1.4	72.9 ±1.9	73.7 ±1.0
AST-Seg-B3-CT	94.8 ±0.5	87.3 ±3.8	83.1 ±0.8
SA (CNN)	54.5 ±1.6	53.0 ±1.6	54.2 ±2.6
ISA-T (CNN)	66.8 ±5.7	65.0 ±4.9	65.1 ±4.8
ISA-TS (CNN)	78.8 ±3.9	72.9 ±3.5	73.2 ±3.1
SA (ResNet)	91.3 ±2.7	84.9 ±2.9	81.4 ±1.4
ISA-T (ResNet)	87.4 ±6.6	79.0 ±5.9	78.6 ±4.9
ISA-TS (ResNet)	92.9 ±0.4	86.2 ±0.8	84.4 ±0.8

- Rotation invariance does not bring consistent improvement

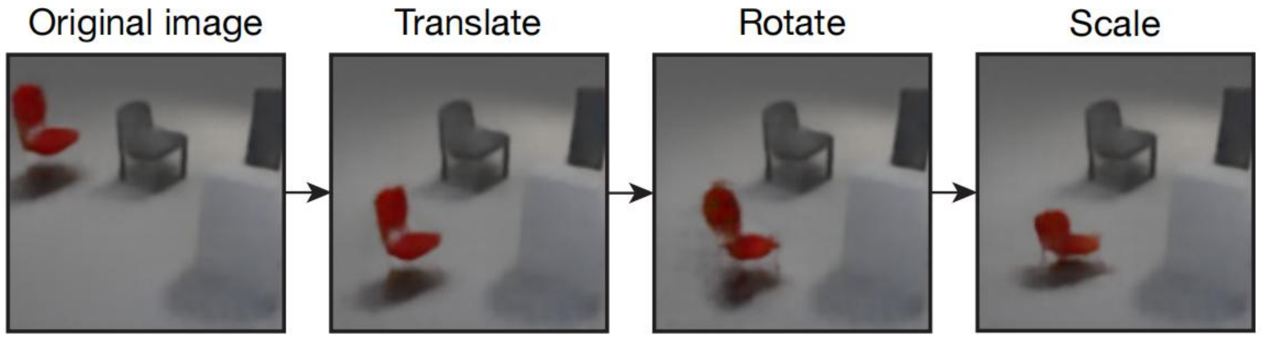
Table 2. **Rotation invariance**: Comparing ISA-TS against ISA-TSR in various benchmarks. Objects Room results are ARIs whereas all others are FG-ARIs. Remaining benchmarks evaluations are in the appendix Table 4.

	(FG-)ARI ↑	
Dataset	ISA-TS	ISA-TSR
Objects Room (w/ bg) Val.	85.5 ±6.6	84.3 ±4.6
CLEVR	98.9 ±0.2	98.0 ±0.9
MultiShapeNet		
- All Data	69.8 ±1.1	77.7 ±5.5
- Four Objects	86.5 ±1.1	80.7 ±6.4
CLEVRTex (CNN)	78.8 ±3.9	79.6 ±5.5
CLEVRTex (ResNet)	92.9 ±0.4	93.3 ±0.7

Experiments: Qualitative (4/6)

- ISA-T improves OOD robustness

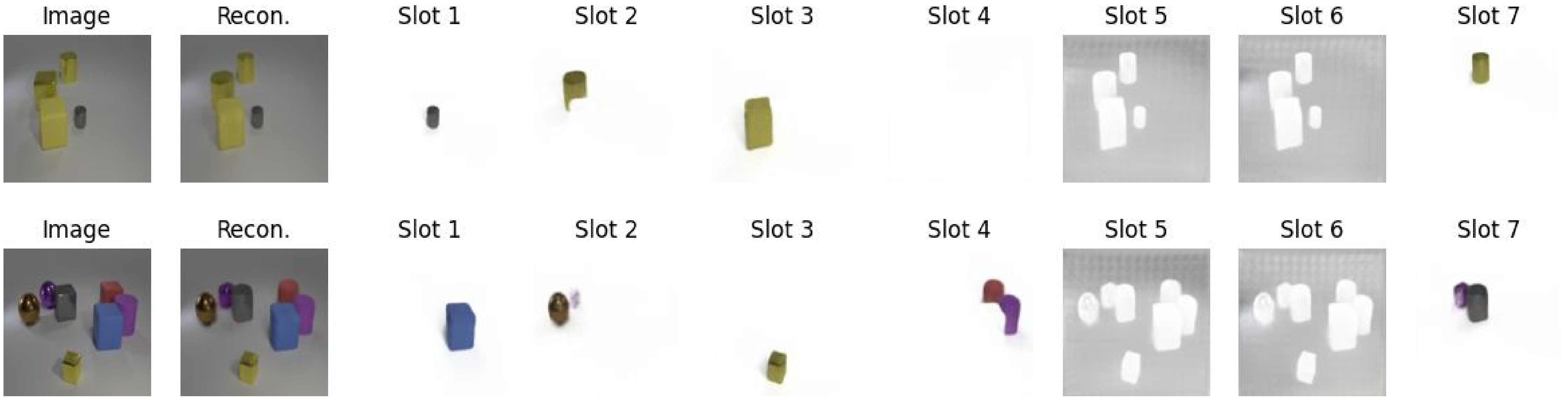
- Qualitative results on MultiShapeNet



- Qualitative results on Waymo open

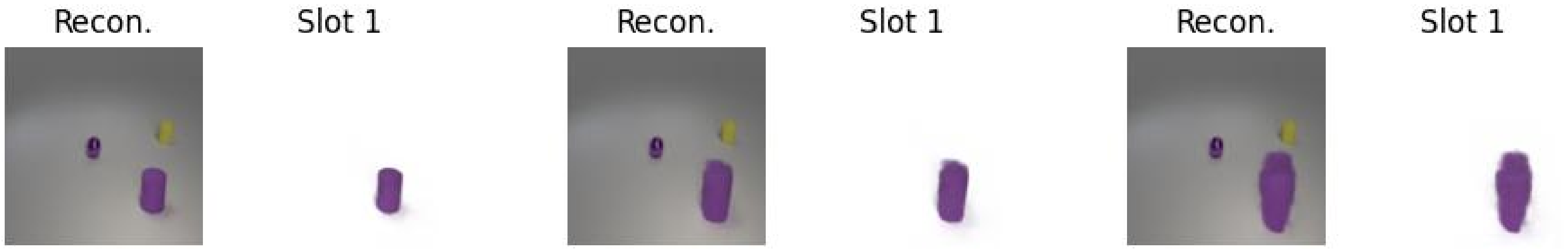
Experiments: Reproduction (5/6)

- Cannot guarantee one slot represent one object
- No explicit control on background (multiple background slot)
- Cannot handle occulusion (general problem for 2D methods)
- Undesirable reconstruction result on edges (e.g, tend to smooth sharp edges)



Experiments: Reproduction (6/6)

- Results for translation and scaling



translate   translate, scale*1.5   translate, scale*2

OUTLINE

- Authorship
- Background
- Method
- Experiments
- Conclusion

Conclusion

Pros

- The authors proposed invariant slot attention, a novel approach for obtaining object-centric representations from 2D single image

Cons

- Position and scale are defined in the image plane, while objects position is 3D
- This approach seems only work for 2D

Future research direction

- 3D object-centric representations from a single image.

Thanks for listening!