LEARNING WHAT AND WHERE – UNSUPERVISED DISENTANGLING LOCATION AND IDENTITY TRACKING

Anonymous authors
Paper under double-blind review

ABSTRACT

Our brain can almost effortlessly decompose visual data streams into background and salient objects. Moreover, it can anticipate object motion and interactions, which are crucial abilities for conceptual planning and reasoning. Recent object reasoning datasets, such as CATER, have revealed fundamental shortcomings of current vision-based AI systems, particularly when targeting explicit object encodings, object permanence, and object reasoning. Here we introduce a self-supervised LOCation and Identity tracking system (Loci), which excels on the CATER tracking challenge. Inspired by the dorsal-ventral pathways in the brain, Loci tackles the binding problem by processing separate, slot-wise encodings of 'what' and 'where'. Loci's predictive coding-like processing encourages active error minimization, such that individual slots tend to encode individual objects. Interactions between objects and object dynamics are processed in the disentangled latent space. Truncated backpropagation through time combined with forward eligibility accumulation significantly speeds up learning and improves memory efficiency. Besides exhibiting superior performance in current benchmarks, Loci effectively extracts objects from video streams and separates them into location and Gestalt components. We believe that this separation offers an encoding that will facilitate effective planning and reasoning on conceptual levels.

1 INTRODUCTION

Human reasoning about visual scenes is characterized by segmenting a scene into individual entities and their interactions [4; 13; 23]. This ability presents a non-trivial challenge for computational models of cognition [76; 33]: the binding problem [71]. Originally proposed and discussed mainly in cognitive neuroscience, the binding problem describes the fundamental challenge of how to bind features into objects, thereby segregating them from the background, and encoding them by means of compressed stable neural attractors [5; 40; 65].

Recent years have seen revolutionary progress in the ability of connectionist models to operate on complex natural images and videos [29; 44; 59]. Yet, neural network models are not capable of addressing the binding problem in its full generality [33]. Indeed, recent work on synthetic object reasoning datasets like CLEVR, CLEVRER, or CATER [39; 85; 30] suggests that state-of-the-art video reasoning systems [81; 84; 16] still struggle to model fundamental physical object properties, such as hollowness, blockage, or object permanence—concepts that children develop during the first few months of their lives [2; 57].

In a comprehensive review on the binding problem in the context of neural networks, Greff et al. [33] define three main challenges for solving the problem: *representation*, *segregation*, and *composition*. *Representation* refers to the challenge to build encodings of the essential properties of an object, including its appearance and potential dynamics. We will refer to these properties as the 'Gestalt' of an object [42; 78; 79]. Object representations should be separable to avoid interference. Moreover, their shape and location should be disentangled to enable compositional generalizations. Meanwhile, they should share a common format to enable general purpose reasoning. *Segregation* describes the challenge to extract particular object representations from a perceived scene. This extraction should be done context- and task-dependently in order to identify currently relevant entities. The essential feature of a good segregation is that it allows effective dynamic predictions of the whole, rather than the parts. *Composition* characterizes the challenge to develop object representations that

enable meaningful re-combinations of object properties. Moreover, these properties should enable the prediction of object interaction consequences, depending on the respective object properties. As a result, compositional representations allow conceptual reasoning about both object properties as well as relations and interactions between objects.

Our goal is to develop an architecture that solves the binding problem-respective challenges. We thus propose a novel **Loc**ation and **I**dentity tracking system in the context of tracking and reconstructing the behavior of objects from video data. A key motivation for our model is that the representation of an object's properties—'what it is'—should be explicitly disentangled from its location—'where it is'—mimicking ventral and dorsal processing in our brain, respectively [61; 72].

2 Related Work

Previous work by [53] has emphasized that, in general, unsupervised object representation learning is impossible because infinitely many variable models are consistent with the data. Inductive biases are thus necessary to ensure the effective learning of a system that segregates a visual stream of information into effective, compositional representations [33]. Accordingly, we review related work in the light of the binding problem and their relation to the proposed Loci system.

Representation A powerful choice of representation format is the formulation of 'slots', which share a representational space but keep the codes of individual objects separate from one another. To ensure a common format between slot representations, typically, slot-respective encoder modules share their weights [54; 8; 75]. To assign individual objects to individual slots, though, the system needs to break slot symmetry. Recurrent neural networks have been used to disentangle representations or assignments [54; 27; 11; 26; 31]. Other mechanisms explicitly separate spatial slot locations [20; 48; 38], which we also do in Loci. However, instead of treating every spatial location as a potential object, each slot has a spotlight, which is designed to approximate the object's center. Another important challenge towards a compositional object representation is the disentanglement of salient features. Loci enforces the disentanglement of object slot locations (where it is), Gestalt codes (object properties), size (visual extent), and priority (current visibility with respect to other objects).

Segregation Segregating object instances from raw input images is traditionally solved via bounding box detection [66; 52], where more advanced techniques extract additional masks for instance segmentation [21; 36; 15]. Through slot-attention mechanisms, recent unsupervised approaches partition image regions into separate slots to represent distinct objects [32; 55]. To segregate objects from images, Burgess et al. [10] and von Kügelgen et al. [77] combine a soft attention (mask) approach with encoder, recurrent attention, and decoder modules, where slots compete for encoding individual objects. Loci pursues a similar approach but segregates objects even further to encourage object-respective 'what' and 'where' encodings, where the latter additionally disentangle location, size, and approximate depth. Slot-respective masks compete via softmax attention to actively minimize the prediction error of the visual content. Moreover, a pre-trained background model separates potentially interesting from uninteresting regions. While we keep the background modeling rather simple in this work, more advanced background modeling techniques may certainly be applied in the near future [74; 62; 24; 63].

Composition Compositional reasoning in our model builds on two modules, which process object-to-object interactions and object dynamics. Object-to-object interactions are modelled using Multi-Head Self-Attention (MHSA) [75], in close relation with [25; 41]. Alternative approaches employ message passing neural networks to simulate object interactions [6; 18; 37]. Object dynamics are modelled using a recurrent GateL0RD module [35]. GateL0RD is specifically designed to apply latent state updates sparsely, which encourages the maintenance of stable object representations over long periods of time. Previous approaches have also employed recurrent structures to propagate slot representations forward in time [38; 43]. Most recent frameworks, though, tend to employ fully auto-regressive designs based on transformers without explicit internal state maintenance [25; 41; 58].

Tracking models While our primary goal is to separate object location and Gestalt representations, a successful extraction of object positions will likely facilitate object tracking; a wide area of research

on its own [83]. Modern state-of-the-art methods for object tracking rely on features extracted via attention modules that are typically applied autoregressively on individual frames [58; 19].

Again, these approaches do not explicitly foster a separation of location and Gestalt, which potentially limits their applicability and accuracy in distractive environments, where humans still maintain high tracking skills as recently shown in [49]. Loci maintains separate object representations and thus copes with the presence of distractors more readily. The tracking model proposed in [49] is similar in spirit to Loci, extracting object representations and propagating them with a competition mechanism and a recurrent module. However, to model complex objects their model relies on additional supervision.

Disentangling movement and appearance is a common principle in video models with a notion of optical flow [51]. But optical flow based methods notoriously struggle with occlusions as they can only represent parts of the scene that are currently visible. In contrast, Loci is able to maintain a representation of an object even under occlusion.

Previous work on CATER We focus Loci's evaluation on the CATER challenge [30]. Previous SOTA methods on this challenge, like Multi-Hopper or OpNet [68; 88], have focused on reusing well-established neural network building blocks. Others have attempted to build their system on top of a supervised pre-trained bounding-box-based object detector [15; 89]. In contrast, Loci effectively combines various representational formats, slot-oriented operations, and several kinds of interactive neural processing modules. It is trained in a fully self-supervised manner, but can be further fine-tuned with supervision to minimize location-specific prediction error.

3 METHODS

The core idea of Loci is to split the encoding of a moving entity into its 'where' and 'what' components. We use the Siamese network idea implemented in slot-attention [54], sharing weights across slots. In fact, Loci employs the same encoding and same decoding network for every slot starting from the raw image. Each encoder slot k generates output Gestalt vectors G_k and position codes P_k. Entity dynamics and interactions are then processed based on these pairs of latent Gestalt and location encodings. We predict the future states of latent encodings in a transition module and allow cross-slot interactions via multi-head attention. Figure 1 shows the main components; information processing in one iteration from left to right. A detailed algorithmic description is provided in Appendix B. Here, we provide an overview and then detail the loss computation and training procedure.

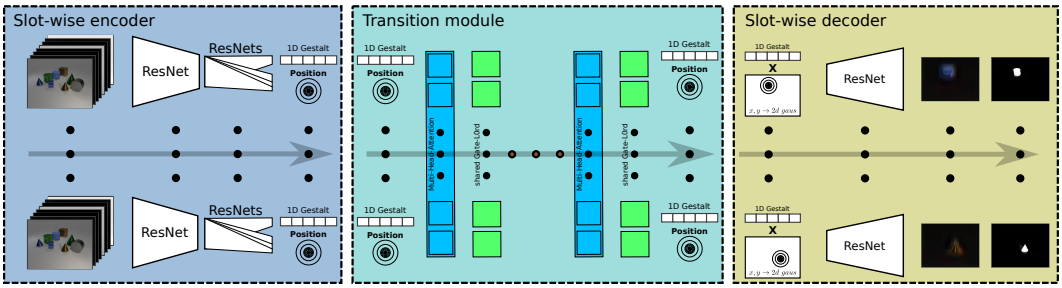

Figure 1: Loci's main processing loop has three components: The tree-structured slots of the slot-wise encoder process individual entities k to generate the disentangled latent Gestalt code G_k and position code P_k. The transition module models slot-to-slot interactions and slot-dynamics. It is implemented by multiple alternating layers of multi-head attention and GateL0RD—a recurrent neural network that fosters the stable maintenance of hidden latent states [35]. The resulting Gestalt and position codes are finally used in the slot-wise decoder to predict slot-specific RGB entity-respective reconstructions R_k and masks M_k.

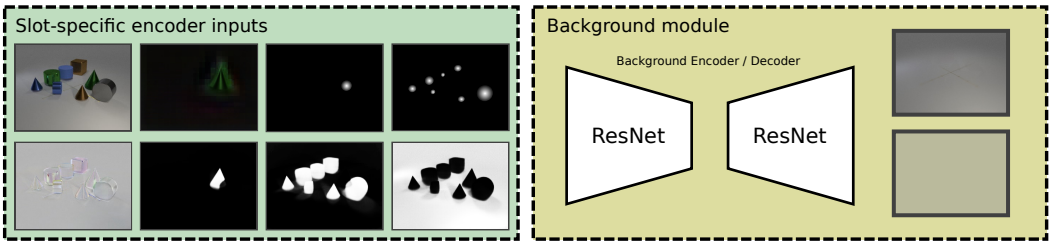

Figure 2: Eight sources of information are fed into each slot-wise encoder (left side): top row: the current RGB image I^t, the reconstructed RGB slot image R_k^t, the Gaussian slot position map Q_k^t, as well as the summations over other Gaussian position maps $Q_k^{t,s}$; bottom row: the MSE E^t, slot mask M_k^t, as well as the summations over other masks $M_k^{t,s}$, and the background mask M_{bg}. The background module (right side) is either implemented as a Gaussian mixture model or a simple autoencoder.

3.1 Loci Overview

The slot-wise encoder is based on a conventional ResNet architecture. The input consists of eight main feature maps, which encode the image state I^t at current iteration t, the prediction error map E^t, a common background mask M_{bg} as well as—slot k specifically—a mask M_k^t, a position map P_k^t, and maps of both the other masks $M_k^{t,s}$ and the other positions $P_k^{t,s}$ (cf. Figure 2). The computation path of Gestalt and position for a slot is tree-shaped, starting with a shared ResNet-encoded trunk after which the path is split into four pathways, which compute Gestalt, location, size, and priority. This encoder design encourages that separately moving entities are encoded in individual ResNet activity dynamics as well as that each slot encodes 'what' (Gestalt) and 'where' (location, size, and priority) of the slot-encoded moving entity separately but conjointly. Thereby, the error map E^t encourages active error minimization, following the principle of predictive coding fostering attractor dynamics [40; 65]. Appendix B specifies further details.

The transition module then predicts location P_k^{t+1} and Gestalt dynamics G_k^{t+1} as well as interactions between the slot-encoded entities. This module is loosely based on the architecture of a transformer-like encoder [75], where several multi-head attention layers are stacked together with residual feed forward layers in between. We replace the residual feed forward layers with residual GateL0RD [35] layers. GateL0RD is a recent gated recurrent neural network architecture, which is very well-suited to learn and distinctively encode sparse interaction events (events can be considered as "structured, describable, memorable units of experience." [3], cf. [86]). In accordance with our entity-focused processing approach, we apply a Siamese GateL0RD version, operating on the individual slot level while still receiving information form other slots via the attention layers.

At the end of the transition module, the Gestalt code is pushed through a binarization layer, which is inspired by the principle of vector quantization [73]. This layer enforces an information bottleneck and thus contributes to the development of disentangled entity codes.

The slot-wise decoder recombines the 'what' and 'where' output from the transition module. To do so, first the potential influence of each Gestalt code G_k is computed over the output range by means of a 2d isotropic Gaussian, yielding density maps Q_k. Next, a priority-$\hat{z}$-based attention is applied to account for the fact that only one slot-object can be visible at any location (transparent objects are left for future work, see Algorithm-2 in Appendix B). As a result, when two slots cover the same location, the one with the lower priority will have its feature maps set to zero. The rest of the decoder is based on a conventional ResNet, which increases the resolution back to the video size. Like the encoder, the decoder shares weights across all slots. Eventually, the decoder outputs an RGB entity reconstruction $\hat{R}_k$ and a mask $\hat{M}_k$ for each slot k. The masks from all slots and from the background are then combined to construct the final output image prediction $\hat{R}$.

To fully reconstruct the image, Loci employs a background encoder-decoder architecture, which generates background image estimates R_{bg} and a corresponding background mask M_{bg}. In case of a static background, a Gaussian mixture model over the whole training set is used. For more dynamic backgrounds, we employ a simple auto encoder. A 2d-bias value is used for the background mask.

3.2 TRAINING

Loci is trained using a binary cross-entropy loss (L_{BCE}) pixel-wise on the frame prediction applying rectified Adam optimization [RAdam, cf. 50]. Several regularizing losses are added to foster object permanence. Additionally, to speed-up learning, we use truncated backpropagation trough time and enhance the gradients with an e-prop-like accumulation of previous neural activities [7].

Empirical evaluations showed that backpropagating the gradients through the networks' inputs creates instabilities in the training process. Thus, we detach the gradients for the latent states. As a result, the only part of the network that needs to backpropagate gradients trough time are the GateL0RD layers. Here, we found that using the described combination of e-prop and backpropagation is not only comparable in terms of network accuracy, it also greatly decreases training wall-clock time, as it allows the use of truncated backpropagation with length 1, effectively updating the weights after each input frame (see supplementary material for details).

Another important aspect for successful training is the use of a warmup phase, where we mask the loss of the network with a foreground mask computed with a threshold τ:

$$M = \tau < \texttt{Mean}\left(\left(I^t - R_{bg}\right)^2, axis = \text{'rgb'}\right), \tag{1}$$

where R_{bg} is the background model, detailed above. The foreground mask enforces the network to initially focus on reconstructing the foreground objects only, that is, to focus on the visual signals that differ at least τ-strongly from the background module predictions. This supports learning to use the position-constrained encoding and decoding mechanism. After around 30 000 updates—when the network has sufficiently learned to use the position encodings—we gradually blend the loss from the masked foreground reconstruction to the full reconstruction.

Object Permanence Loss To encourage object permanence, an additional loss is computed based on Equation 2, which favors a slot that keeps encoding the same object, even if the object is temporarily occluded and thus invisible:

$$L_o = \sum_k \left(\mathfrak{D}_k(P_k^t, \hat{G}_k^t) - \mathfrak{D}_k(P_k^t, G_k^t)\right)^2, \tag{2}$$

$$\hat{G}_k^t = \hat{G}_k^{t-1}(1 - max(M_k^t)) + G_k^t max(M_k^t), \tag{3}$$

where P_k and G_k denote location and Gestalt encoding in slot k, $\mathfrak{D}_k$ refers to the RGB representation part of the decoder network, while $\hat{G}_k^t$ denotes the Gestalt code averaged around the last time step in which the entity was visible, and M_k^t denotes the mask of slot k at time step t. As a result, L_o is only applied when the object becomes invisible, which is the case when M_k^t is approaching zero.

Time Persistence Loss A second mechanism to enforce object permanence and to also regularize the network towards temporal consistent slot encodings, is a time persistent regularization loss:

$$L_t = \sum_k \left(\mathfrak{D}_k(p_0, G_k^{t-1}) - \mathfrak{D}_k(p_0, G_k^t)\right)^2, \tag{4}$$

where again $\mathfrak{D}_k$ refers to the RGB representation part of the decoder network and p_0 is the center position in the image. L_t essentially penalizes large visual changes in the decoded object between two consecutive time steps.

Position Change Loss In order to encourage the network to predict small slot-position changes over time, a simple L_2 regularization loss is based on the position change between two time steps:

$$L_p = 0.01 \sum_n (P_k^{t-1} - P_k^t)^2 \tag{5}$$

Supervision Loss Finally, for the experiments that are using a supervised target object location signal for fine-tuning, detailed in Equation 6, we added a gating network Φ that operates on the latent Gestalt codes G_k^t before binarization and predicts a single softmax probability, which is used to

decide which entity corresponds to the Snitch in the CATER dataset. The location of the selected entity is then used in an L_2-loss to foster regression to the target location provided in the dataset.

$$L_s = \mu_s \left(\sum_k P_k^t \Phi(\hat{G}_k^t) - p_{snitch}^t \right)^2 \tag{6}$$

The final loss for the network results from adding the individual loss components, denoting the unsupervised and supervised losses, respectively:

$$L_{unsup} = L_{BCE} + L_o + L_t + L_p, \tag{7}$$

$$L_{sup} = L_{BCE} + L_o + L_t + L_p + L_s. \tag{8}$$

4 Experiments & Results

We performed unsupervised training in all experiments. Only for the CATER challenge we additionally evaluated a version where we fine-tuned the network via supervision using Equation 6.

4.1 CATER-Tracking Challenge

In the CATER tracking challenge (cf. Figure 3), the task is to locate a unique yellow object, called Snitch, at the end of a video sequence. During the video, different geometric objects rotate, move, and hop in a scripted random way. By doing so, cone objects can engulf other objects and move them to another location before releasing them again, which can lead to situations where the Snitch remains hidden in the last frame of the video. Therefore, the challenge is not only to *recognize containment* events, but also to *track* the specific cone that contains the Snitch. For classification purposes, the 3D Snitch position is partitioned into a 6×6 grid, resulting in a total of 36 classes. In order to account for the fact that a small location error could lead to miss-classification, a common reported metric is the L_1 grid distance. We additionally report the continuous L_2 distance with grid length 1, such that the distance is comparable to, but more informative than the L_1 grid distance.

As described in Girdhar & Ramanan [30], we split the dataset with a ratio of 70:30 into a training and test set and further put aside 20% of the training set as a validation set, leaving 56% of the original data for training. To blend-in the supervision loss, we first set the supervision factor $\mu_s = 0.01$ for the first 4 epochs, fostering mostly unsupervised training, and then set $\mu_s = 0.1$ for the rest of the training process, to give the Snitch location a weak pull towards the target location.

In order to produce labels from the purely unsupervisedly trained network, we train a separate small classifier with around 17k parameters, which only operates on the latent states of the trained Loci network with a correction and a gating network. The correction network computes a residual for the location of each entity. The gating network computes softmax probabilities, which are used to select the location belonging to the Snitch object similar to Equation 6. To prevent supervised gradient flow, the classifier network module is trained once the unsupervised training has finished. Eventually, the location and Gestalt codes are computed for the whole dataset and extracted into a separate data file, from which the classifier module is trained, again using a 70:30 train/test split.

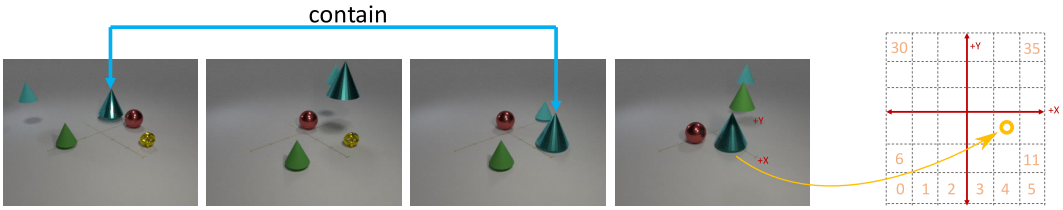

Figure 3: The CATER Snitch localization challenge: The task is to locate the yellow spherical object called Snitch within the last frame. The challenge is that the Snitch might be contained and moved by a cone. So its location has to be inferred by recognizing and remembering a containment event and then tracking the position of the container. Image adapted from [30].

Table 1: Quantitative results of the CATER Snitch localization challenge. Referenced results are from [88] and from the other system-respective papers.

Method	Parameters (M)	Top 1	Top 5	L_1 (grid)	L_2
Random	-	2.8	13.8	3.9	
Transformer [75]	15.01	13.7	39.9	3.53	
SINet [56]	138.69	21.1	47.1	3.14	
TSN (RGB) + LSTM [80]		25.6	67.2	2.6	
DaSiamRPN [89]		33.9	40.8	2.4	
I3D-50 + LSTM [16]		60.2	81.8	1.2	
Hopper-transformer [88]	15.01	61.1	86.6	1.42	
TSM-50 [47]		64.0	85.7	0.93	
TPN-101 [84]		65.3	83.0	1.09	
Hopper-sinet [88]	139.22	69.1	91.8	1.02	
Inferno [17]	-	71.7	88.9	-	
Hopper-multihop [88]	6.39	73.2	93.8	0.85	
Aloe [22]		74.0	94.0	0.44	
OPNet [68]	-	74.8	-	0.54	
TFC V3D Depthwise [87]	24.64	79.7	95.5	0.47	
Loci supervised (ours)	**2.96**	**90.7**	**98.5**	**0.14**	0.14
Loci unsupervised (ours)	4.14	78.4	92.0	0.45	0.44

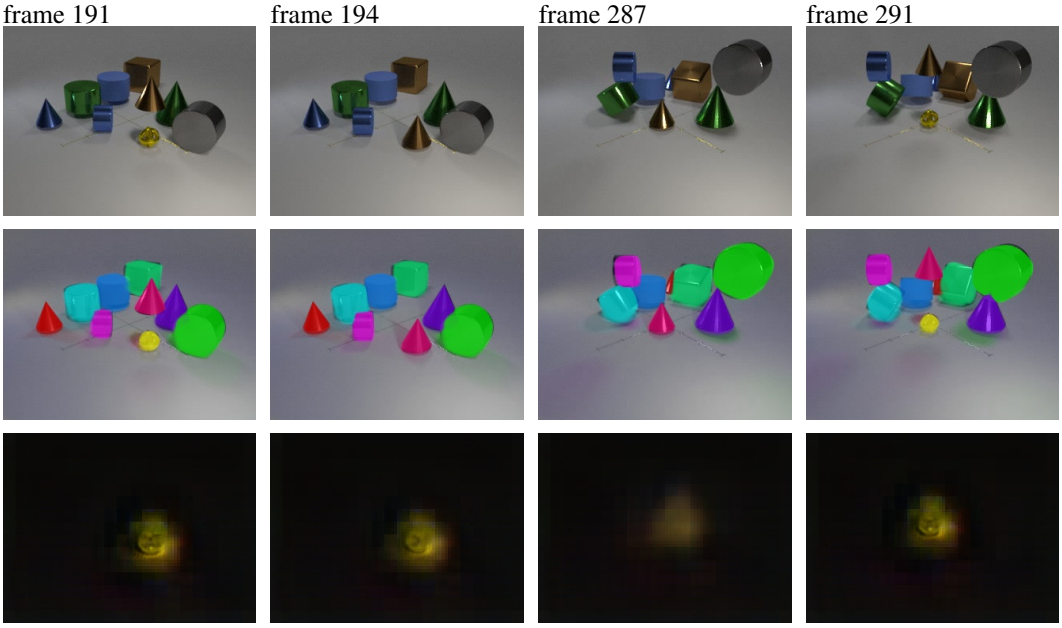

Figure 4: Object permanence shown on a CATER tracking example. While the network (after completely unsupervised training) struggles to keep the shape of the Snitch when contained for a longer time span, color and, importantly, the position of the Snitch are preserved during containment. **Top row**: input frames. **Middle row**: tracked objects visualized trough colored masks. **Bottom row**: RGB representation of the Snitch.

As shown in Table 1, Loci not only surpasses all previous methods by a large margin—achieving a top 1 accuracy of 90.7% and a top 5 accuracy of 98.5% with an L_1 distance of 0.14—it also surpasses most of the existing work when the main architecture is trained purely unsupervised. Loci learns object permanence merely by means of inductive bias and regularization. This is shown in Figure 4, where Loci keeps the representation of the contained Snitch in memory. While the shape details blur over time, importantly, the locating is correctly tracked to reconstruct the Snitch once it is revealed. Figure 5 also demonstrates the tracking performance in case of several co-occurring occlusions and double contained situations, where two cones are covering the Snitch. Here, the location output

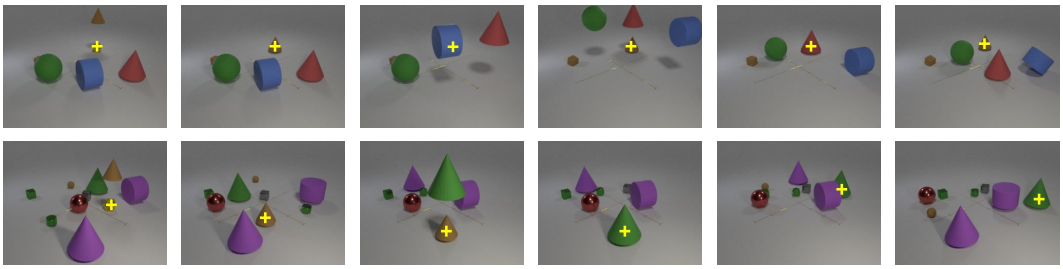

Figure 5: Two challenging CATER tracking examples with several co-occurring containments / occlusions: Video frames overlaid with the predicted Snitch location (network trained with supervision).

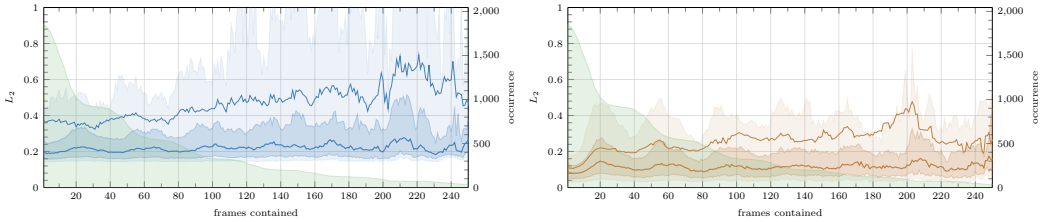

Figure 6: Comparison between unsupervised (**left**) and supervised (**right**) tracking, while the Snitch is contained progressively longer. On the x-axis are the number of frames the Snitch has been contained, while the left y-axis shows the L_2 distance to the target position. The right y-axis shows the number of times a Snitch was contained that long within the test-set (visualized in green). The thick line represents the median L_2 distance, while the thin line represents the mean L_2 distance. The weaker colored area shows the 90-10-quantile, while the stronger one corresponds to the 75-25-quantile.

of the supervised gating network is marked in the video frames. A final analysis of Loci's object permanence abilities is shown in Figure 6, where the tracking accuracy is plotted for progressively extended periods of Snitch containment. For both the supervised and unsupervised tracking, the location error remains constant after an initial increase, even over extended time spans.

In additional lesion studies (detailed in Appendix A.3.5), we furthermore demonstrate the effective disentanglement of position, Gestalt, size, and priority codes, which underlines Loci's ability to separate position and Gestalt of objects from a scene without supervision.

4.2 Moving MNIST

The moving MNIST (MMNIST) challenge is a dataset for video prediction [69]. The task is to predict the motion of two moving digits, with a maximum size of 28×28 pixels, moving independently inside a 64×64 pixel window and bouncing off the walls in a predictable manner. While the dataset is usually implemented by simply bouncing the entire 28×28 MNIST sub-image within the 64×64 window, we first crop each digit to its actual appearance in order to obtain a realistic bounding box of the digit and thus to generate more realistic bouncing effects (once the actual number instead of its bounding box touches the border). This removes the undesired bias in the dataset to remember each digit's bounding box individually, which is then the only way to correctly predict a bounce off a wall. Instead, now the network can predict the bounce based on the pixel information alone.

While the task was originally formulated to predict the next 10 frames after receiving the first 10 frames, we also evaluate the ability to generate temporal closed-loop predictions for up to 100 frames after being only trained to predict 10 frames. We compare Loci to the state-of-the-art approach PhyDNet, also designed to disentangle 'what' from 'where'; more specifically: Physical dynamics from unknown factors such as the digit's Gestalt code. While using the original code provided by the authors, we did not reach the same reported performance of PhyDNet using our unbiased MMNIST dataloader. Nevertheless, PhyDNet reaches a high accuracy for the 10 frame prediction, which is only slightly topped by Loci. For extended temporal predictions, though, PhyDNet quickly dissolves the digits, while Loci preserves the Gestalt of each digit over the 100 predicted frames, as

Table 2: Moving MNIST prediction accuracy PhyDNet vs Loci. Both networks where trained using the same dataloader to predict the next 90 frames given an input sequence of 10 frames.

	$T = 10$		$T = 20$		$T = 30$		$T = 50$		$T = 100$	
Method	MSE	SSIM	MSE	SSIM	MSE	SSIM	MSE	SSIM	MSE	SSIM
PhyDnet [34]	35.6	0.913	58.6	0.830	**80.7**	0.739	**113.9**	0.594	**155.1**	0.484
Loci	**30.7**	**0.923**	**52.5**	**0.881**	98.3	**0.814**	165.1	**0.720**	221.2	**0.639**

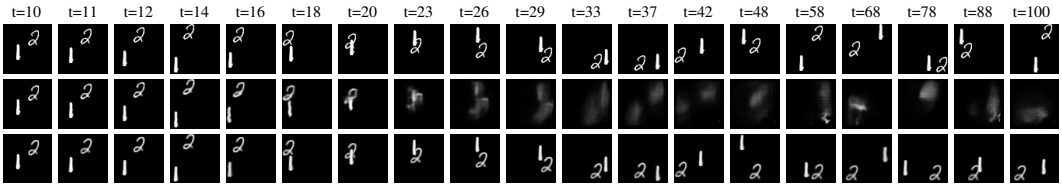

Figure 7: Comparison between ground truth **top row**, PhyDNet **center row** and Loci **bottom row** for a prediction of up to 90 frames. Both PhyDNet and Loci were trained on 10 frame prediction. While in PhyDNet, the appearance of the digits dissolves after a few frames beyond the initial training distribution, Loci manages to keep the Gestalt code and the location accurate until the fourth collision at around frame 42. The Gestalt codes remain stable until the end of the considered 100 time steps.

shown in Figure 7. Table 2 shows a qualitative comparison between Loci and PhyDNet: While Loci consistently outperforms PhyDNet for the structural similarity index measure (SSIM, [82]), PhyDNet has a lower MSE than Loci after 30 frames. This might be due to the blurring of PhyDNet, which accounts for the uncertainty in the position estimate at the cost of the digits shape.

4.3 Other Datasets & Supplementary Material

Apart from a video footage for the main experiments and further algorithmic details on Loci, we provide additional insights and tests in the supplementary material: We evaluate the real world tracking performance of Loci on a ten hour aquarium footage found on YouTube [1]. Additionally, we examine Gestalt preservation and indicators of intuitive physics in closed loop predictions on the CLEVRER dataset [85]. Furthermore, several ablation studies are provided.

5 Conclusion and Future Work

We presented Loci—a novel location and identity disentangling artificial neural network, which learns to process object-distinct position and Gestalt encodings. Loci significantly exceeds current state-of-the-art architectures on the CATER challenge, while requiring fewer network parameters and less supervision. Loci thus offers a critical step towards solving the binding problem [33; 71; 76]. We particularly hope that the mechanisms proposed in this work bear potential to enrich the field of object representation learning and highlight the importance of well-chosen inductive biases.

Currently, Loci operates on static background estimates and cameras, which we denote as its main limitation. We intend to extend the background model to incorporate rich and varying backgrounds more flexibly in future work. Potential avenues for this extension may include an explicit, separate encoding of ego-motion and depth [46]. Furthermore, image reconstructions may be fused with subsequent image inputs in an information-driven manner. We also expect to create even more compact object-interaction encodings following theories of event-predictive cognition and related conceptions and models in computational neuroscience [12; 14; 28; 70]. Moreover, we are excited to explore the potential of Loci to address problems of intuitive physics and causality [60; 64; 67], seeing that Loci offers a suitable compositional structure [45; 33] to enable symbolic reasoning. Finally, we hope that Loci will also be well-combinable with reinforcement learning and model-predictive planning in system control setups to pursue active, goal-directed environmental interactions.

[1] Aquarium footage on YouTube: `https://www.youtube.com/watch?v=9Ej-0VRWmI8&t=32347s`

REFERENCES

[1] Thomas Bachlechner, Bodhisattwa Prasad Majumder, Henry Mao, Gary Cottrell, and Julian McAuley. Rezero is all you need: Fast convergence at large depth. In *Uncertainty in Artificial Intelligence*, pp. 1352–1361. PMLR, 2021.

[2] Renée Baillargeon, Elizabeth S. Spelke, and Stanley Wasserman. Object permanence in five-month-old infants. *Cognition*, 20(3):191–208, 1985. doi: 10.1016/0010-0277(85)90008-3.

[3] Dare Ann Baldwin and Jessica E. Kosie. How does the mind render streaming experience as events? *Topics in Cognitive Science*, 13:79–105, 2021. doi: 10.1111/tops.12502.

[4] L. W. Barsalou. Perceptual symbol systems. *Behavioral and Brain Sciences*, 22:577–600, 1999.

[5] Lawrence W Barsalou. Abstraction in perceptual symbol systems. *Philos Trans R Soc Lond B Biol Sci*, 358(1435):1177–1187, Jul 2003.

[6] Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. *Advances in neural information processing systems*, 29, 2016.

[7] Guillaume Bellec, Franz Scherr, Elias Hajek, Darjan Salaj, Robert Legenstein, and Wolfgang Maass. Biologically inspired alternatives to backpropagation through time for learning in recurrent neural nets. *arXiv preprint arXiv:1901.09049*, 2019.

[8] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *European conference on computer vision*, pp. 850–865. Springer, 2016.

[9] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[10] Christopher P Burgess, Loic Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matt Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation. *arXiv preprint arXiv:1901.11390*, 2019.

[11] Christopher P. Burgess, Loïc Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matthew M. Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation. *CoRR*, abs/1901.11390, 2019. URL `http://arxiv.org/abs/1901.11390`.

[12] Martin V. Butz. Towards a unified sub-symbolic computational theory of cognition. *Frontiers in Psychology*, 7(925), 2016. ISSN 1664-1078. doi: 10.3389/fpsyg.2016.00925.

[13] Martin V. Butz and Esther F. Kutter. *How the Mind Comes Into Being: Introducing Cognitive Science from a Functional and Computational Perspective*. Oxford University Press, Oxford, UK, 2017.

[14] Martin V. Butz, Asya Achimova, David Bilkey, and Alistair Knott. Event-predictive cognition: A root for conceptual human thought. *Topics in Cognitive Science*, 13:10–24, 2021. doi: 10.1111/tops.12522.

[15] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pp. 213–229. Springer, 2020.

[16] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6299–6308, 2017.

[17] Lluis Castrejon, Nicolas Ballas, and Aaron Courville. Inferno: Inferring object-centric 3d scene representations without supervision. 2021.

[18] Michael B Chang, Tomer Ullman, Antonio Torralba, and Joshua B Tenenbaum. A compositional object-based approach to learning physical dynamics. *arXiv preprint arXiv:1612.00341*, 2016.

[19] Xin Chen, Bin Yan, Jiawen Zhu, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Transformer tracking. *CoRR*, abs/2103.15436, 2021. URL https://arxiv.org/abs/2103.15436.

[20] Eric Crawford and Joelle Pineau. Spatially invariant unsupervised object detection with convolutional neural networks. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'19/IAAI'19/EAAI'19. AAAI Press, 2019. ISBN 978-1-57735-809-1. doi: 10.1609/aaai.v33i01.33013412. URL https://doi.org/10.1609/aaai.v33i01.33013412.

[21] Jifeng Dai, Kaiming He, and Jian Sun. Instance-aware semantic segmentation via multi-task network cascades. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3150–3158, 2016.

[22] David Ding, Felix Hill, Adam Santoro, and Matt Botvinick. Object-based attention for spatio-temporal reasoning: Outperforming neuro-symbolic models with flexible distributed architectures. *arXiv e-prints*, pp. arXiv–2012, 2020.

[23] John Duncan. Selective attention and the organization of visual information. *Journal of Experimental Psychology: General*, 113:501–517, 1984. doi: 10.1037/0096-3445.113.4.501.

[24] Sébastien Ehrhardt, Oliver Groth, Aron Monszpart, Martin Engelcke, Ingmar Posner, Niloy Mitra, and Andrea Vedaldi. Relate: Physically plausible multi-object scene synthesis using structured latent spaces. *Advances in Neural Information Processing Systems*, 33:11202–11213, 2020.

[25] Gamaleldin F. Elsayed, Aravindh Mahendran, Sjoerd van Steenkiste, Klaus Greff, Michael C. Mozer, and Thomas Kipf. Savi++: Towards end-to-end object-centric learning from real-world videos, 2022. URL https://arxiv.org/abs/2206.07764.

[26] Martin Engelcke, Adam R. Kosiorek, Oiwi Parker Jones, and Ingmar Posner. GENESIS: generative scene inference and sampling with object-centric latent representations. *CoRR*, abs/1907.13052, 2019. URL http://arxiv.org/abs/1907.13052.

[27] S. M. Ali Eslami, Nicolas Heess, Theophane Weber, Yuval Tassa, Koray Kavukcuoglu, and Geoffrey E. Hinton. Attend, infer, repeat: Fast scene understanding with generative models. *CoRR*, abs/1603.08575, 2016. URL http://arxiv.org/abs/1603.08575.

[28] Nicholas T. Franklin, Kenneth A. Norman, Charan Ranganath, Jeffrey M. Zacks, and Samuel J. Gershman. Structured event memory: A neuro-symbolic model of event cognition. *Psychological Review*, 127(3):327–361, 2020. ISSN 1939-1471(Electronic),0033-295X(Print). doi: 10.1037/rev0000177.

[29] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2414–2423, 2016.

[30] Rohit Girdhar and Deva Ramanan. CATER: A diagnostic dataset for compositional actions and temporal reasoning. *CoRR*, abs/1910.04744, 2019. URL http://arxiv.org/abs/1910.04744.

[31] Klaus Greff, Raphaël Lopez Kaufman, Rishabh Kabra, Nick Watters, Chris Burgess, Daniel Zoran, Loïc Matthey, Matthew M. Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. *CoRR*, abs/1903.00450, 2019. URL http://arxiv.org/abs/1903.00450.

[32] 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. PMLR, 2019.

[33] Klaus Greff, Sjoerd Van Steenkiste, and Jürgen Schmidhuber. On the binding problem in artificial neural networks. *arXiv preprint arXiv:2012.05208*, 2020.

[34] Vincent Le Guen and Nicolas Thome. Disentangling physical dynamics from unknown factors for unsupervised video prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11474–11484, 2020.

[35] Christian Gumbsch, Martin V Butz, and Georg Martius. Sparsely changing latent states for prediction and planning in partially observable domains. *Advances in Neural Information Processing Systems*, 34, 2021.

[36] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

[37] Michael Janner, Sergey Levine, William T Freeman, Joshua B Tenenbaum, Chelsea Finn, and Jiajun Wu. Reasoning about physical interactions with object-oriented prediction and planning. *arXiv preprint arXiv:1812.10972*, 2018.

[38] Jindong Jiang and Sungjin Ahn. Generative neurosymbolic machines. *CoRR*, abs/2010.12152, 2020. URL `https://arxiv.org/abs/2010.12152`.

[39] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2901–2910, 2017.

[40] M. I. Jordan. Attractor dynamics and parallelism in a connectionist sequential machine. In *Proceedings of the Eighth Annual Meeting of the Cognitive Science Society*, pp. 531–546. Erlbaum, Hillsdale, NJ, 1986.

[41] Thomas Kipf, Gamaleldin F. Elsayed, Aravindh Mahendran, Austin Stone, Sara Sabour, Georg Heigold, Rico Jonschkowski, Alexey Dosovitskiy, and Klaus Greff. Conditional object-centric learning from video. *CoRR*, abs/2111.12594, 2021. URL `https://arxiv.org/abs/2111.12594`.

[42] K. Koffka. *Principles of Gestalt psychology*. Routledge, Abingdon, UK, 2013. first published in 1935.

[43] Adam R. Kosiorek, Hyunjik Kim, Ingmar Posner, and Yee Whye Teh. Sequential attend, infer, repeat: Generative modelling of moving objects. *CoRR*, abs/1806.01794, 2018. URL `http://arxiv.org/abs/1806.01794`.

[44] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 25*, pp. 1097–1105. Curran Associates, Inc., Red Hook, NY, 2012.

[45] Brenden M. Lake, Tomer D. Ullman, Joshua B. Tenenbaum, and Samuel J. Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 2017. doi: 10.1017/S0140525X16001837.

[46] Jianfeng Li, Junqiao Zhao, Shuangfu Song, and Tiantian Feng. Unsupervised joint learning of depth, optical flow, ego-motion from video. *arXiv*, 2021. doi: 10.48550/ARXIV.2105.14520. URL `https://arxiv.org/abs/2105.14520`.

[47] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7083–7093, 2019.

[48] Zhixuan Lin, Yi-Fu Wu, Skand Vishwanath Peri, Weihao Sun, Gautam Singh, Fei Deng, Jindong Jiang, and Sungjin Ahn. SPACE: unsupervised object-oriented scene representation via spatial attention and decomposition. *CoRR*, abs/2001.02407, 2020. URL `http://arxiv.org/abs/2001.02407`.

[49] Drew Linsley, Girik Malik, Junkyung Kim, Lakshmi Narasimhan Govindarajan, Ennio Mingolla, and Thomas Serre. Tracking without re-recognition in humans and machines. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 19473–19486. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/a2557a7b2e94197ff767970b67041697-Paper.pdf.

[50] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. *arXiv preprint arXiv:1908.03265*, 2019.

[51] Runtao Liu, Zhirong Wu, Stella Yu, and Stephen Lin. The emergence of objectness: Learning zero-shot segmentation from videos. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 13137–13152. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/6d9cb7de5e8ac30bd5e8734bc96a35c1-Paper.pdf.

[52] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pp. 21–37. Springer, 2016.

[53] Francesco Locatello, Stefan Bauer, Mario Lucic, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. *CoRR*, abs/1811.12359, 2018. URL http://arxiv.org/abs/1811.12359.

[54] 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. URL https://proceedings.neurips.cc/paper/2020/file/8511df98c02ab60aea1b2356c013bc0f-Paper.pdf.

[55] 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.

[56] Chih-Yao Ma, Asim Kadav, Iain Melvin, Zsolt Kira, Ghassan AlRegib, and Hans Peter Graf. Attend and interact: Higher-order object interactions for video understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6790–6800, 2018.

[57] Jean M. Mandler. Thought before language. *Trends in Cognitive Sciences*, 8(11):508 – 513, 2004. ISSN 1364-6613. doi: 10.1016/j.tics.2004.09.004.

[58] Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixé, and Christoph Feichtenhofer. Trackformer: Multi-object tracking with transformers. *CoRR*, abs/2101.02702, 2021. URL https://arxiv.org/abs/2101.02702.

[59] Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixé, and Christoph Feichtenhofer. Trackformer: Multi-object tracking with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8844–8854, June 2022.

[60] A. Michotte. *The Perception of Causality*. Louvain, Ed. de l'Institut Supérieur de Philosophie, 1946. translated into English by C. A. Mace, 1962.

[61] Mortimer Mishkin, Leslie G. Ungerleider, and Kathleen A. Macko. Object vision and spatial vision: Two cortical pathways. *Trends in Neurosciences*, 6:414 – 417, 1983. ISSN 0166-2236. doi: 10.1016/0166-2236(83)90190-X.

[62] Thu H Nguyen-Phuoc, Christian Richardt, Long Mai, Yongliang Yang, and Niloy Mitra. Blockgan: Learning 3d object-aware scene representations from unlabelled images. *Advances in Neural Information Processing Systems*, 33:6767–6778, 2020.

[63] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11453–11464, 2021.

[64] Judea Pearl. *The Book of Why – The new science of cause and effect*. Basic Books, New York, NY, 2018.

[65] Rajesh P. Rao and Dana H. Ballard. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience*, 2(1):79–87, January 1999. doi: 10.1038/4580.

[66] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[67] Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Towards causal representation learning. *CoRR*, abs/2102.11107, 2021. URL `https://arxiv.org/abs/2102.11107`.

[68] Aviv Shamsian, Ofri Kleinfeld, Amir Globerson, and Gal Chechik. Learning object permanence from video. In *European Conference on Computer Vision*, pp. 35–50. Springer, 2020.

[69] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *International conference on machine learning*, pp. 843–852. PMLR, 2015.

[70] David Stawarczyk, Matthew A. Bezdek, and Jeffrey M. Zacks. Event representations and predictive processing: The role of the midline default network core. *Topics in Cognitive Science*, 13:164–186, 2021. doi: 10.1111/tops.12450.

[71] Anne Treisman. The binding problem. *Current Opinion in Neurobiology*, 6(2):171–178, 1996. ISSN 0959-4388. doi: 10.1016/S0959-4388(96)80070-5.

[72] L. G. Ungerleider and J. V. Haxby. "what" and "where" in the human brain. *Current Opinion in Neurobiology*, 4(2):157–65, 1994.

[73] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[74] Sjoerd van Steenkiste, Karol Kurach, Jürgen Schmidhuber, and Sylvain Gelly. Investigating object compositionality in generative adversarial networks. *Neural Networks*, 130:309–325, 2020.

[75] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[76] C. von der Malsburg. Binding in models of perception and brain function. *Curr Opin Neurobiol*, 5(4):520–526, Aug 1995.

[77] Julius von Kügelgen, Ivan Ustyuzhaninov, Peter Gehler, Matthias Bethge, and Bernhard Schölkopf. Towards causal generative scene models via competition of experts. *arXiv preprint arXiv:2004.12906*, 2020.

[78] Johan Wagemans, James H. Elder, Michael Kubovy, Stephen E. Palmer, Mary A. Peterson, Manish Singh, and Rüdiger von der Heydt. A century of gestalt psychology in visual perception: I. perceptual grouping and figure-ground organization. *Psychological bulletin*, 138 6:1172–217, 2012.

[79] Johan Wagemans, Jacob Feldman, Sergei Gepshtein, Ruth Kimchi, James R. Pomerantz, Peter A van der Helm, and Cees van Leeuwen. A century of gestalt psychology in visual perception: Ii. conceptual and theoretical foundations. *Psychological bulletin*, 138 6:1218–52, 2012.

[80] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision*, pp. 20–36. Springer, 2016.

[81] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7794–7803, 2018.

[82] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4): 600–612, 2004.

[83] Marissa A Weis, Kashyap Chitta, Yash Sharma, Wieland Brendel, Matthias Bethge, Andreas Geiger, and Alexander S Ecker. Benchmarking unsupervised object representations for video sequences. *J. Mach. Learn. Res.*, 22:183–1, 2021.

[84] Ceyuan Yang, Yinghao Xu, Jianping Shi, Bo Dai, and Bolei Zhou. Temporal pyramid network for action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 591–600, 2020.

[85] Kexin Yi, Chuang Gan, Yunzhu Li, Pushmeet Kohli, Jiajun Wu, Antonio Torralba, and Joshua B Tenenbaum. Clevrer: Collision events for video representation and reasoning. *arXiv preprint arXiv:1910.01442*, 2019.

[86] Jeffrey M. Zacks, Nicole K. Speer, Khena M. Swallow, Todd S. Braver, and Jeremy R. Reynolds. Event perception: A mind-brain perspective. *Psychological Bulletin*, 133(2):273–293, 2007.

[87] Shiwen Zhang. Tfcnet: Temporal fully connected networks for static unbiased temporal reasoning. *arXiv preprint arXiv:2203.05928*, 2022.

[88] Honglu Zhou, Asim Kadav, Farley Lai, Alexandru Niculescu-Mizil, Martin Renqiang Min, Mubbasir Kapadia, and Hans Peter Graf. Hopper: Multi-hop transformer for spatiotemporal reasoning. *arXiv preprint arXiv:2103.10574*, 2021.

[89] Zheng Zhu, Qiang Wang, Bo Li, Wei Wu, Junjie Yan, and Weiming Hu. Distractor-aware siamese networks for visual object tracking. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 101–117, 2018.

A APPENDIX

As additional content, we first provide details on further evaluations we conducted with Loci. In order to evaluate the tracking performance of Loci in a real world example, we trained it on a 10 hour aquarium footage found on YouTube. The task poses the additional challenge to cope with backgrounds that are not fully stationary. The results are shown in Figure 8 and demonstrate that Loci is able to track 15+ objects in a complex real world environment.

The Gestalt preserving performance of Loci for closed loop predictions are also demonstrated exemplary on the CLEVRER dataset in Figure 9. Here, the effects of collisions of different geometric objects are predicted into the future. While the location deviates visibly over time, Loci is able to preserve the Gestalt code also for more complex objects in a closed loop setup, which is considered specifically challenging for RNNs.

A.1 CATER EVALUATION DETAILS

In order to produce the results from Table 1 we trained five networks with different initial seeds and and then evaluated each network five times with different initial seeds. In Table 1 we reported the mean results over the five evaluation runs from our best performing network (Network 3 from Table 3).

A.2 BACKGOUND MODEL

In order to compute an background model for datasets with a simple static background like CATER, CLEVR or CLEVRER we use a simple Gaussian Mixture Model. The specific function used is *createBackgroundSubtractorMOG2* from OpenCV [9] which we use with a learningrate of 0.00001 to compute an background image based on the training set.

For more complex backgrounds like in the aquarium example where the camera is still static, we use an autoencoder ResNet that encodes the input image into a latent vector with the same size as the gestalt code. This latent background code is then run through an residual GateL0rd in order to capture temporal dynamics, like the chancing water level, and is then run through an ResNet decoder. The background autoencoder is pretrained using a L1 reconstruction loss in order to focus on the most dominant features of the background and to not care so much about foreground objects.

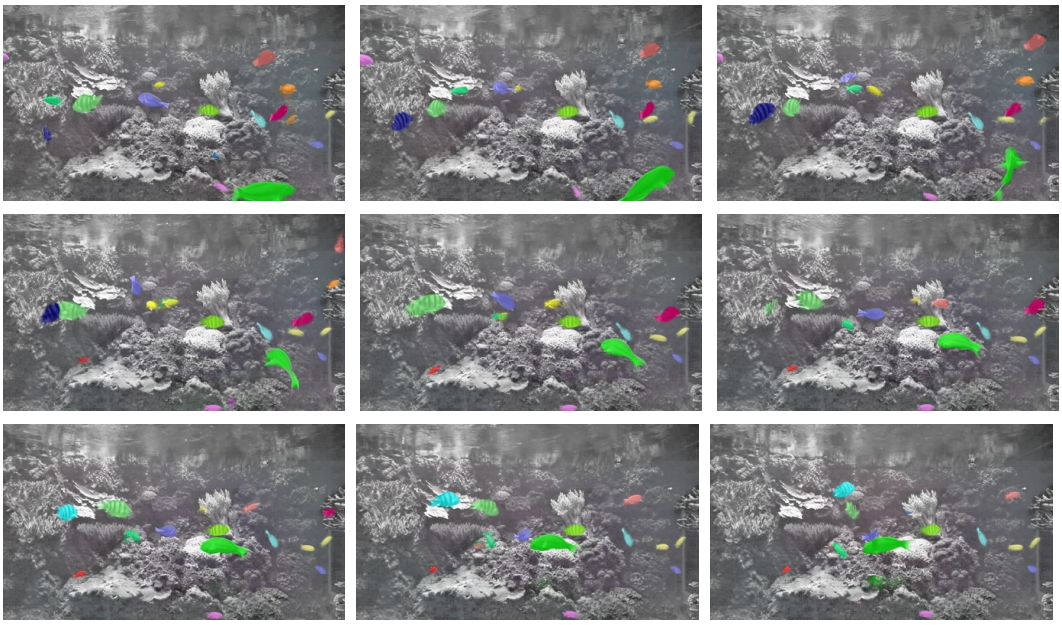

Figure 8: Fully unsupervised real world tracking example trained on 10 hour aquarium footage.

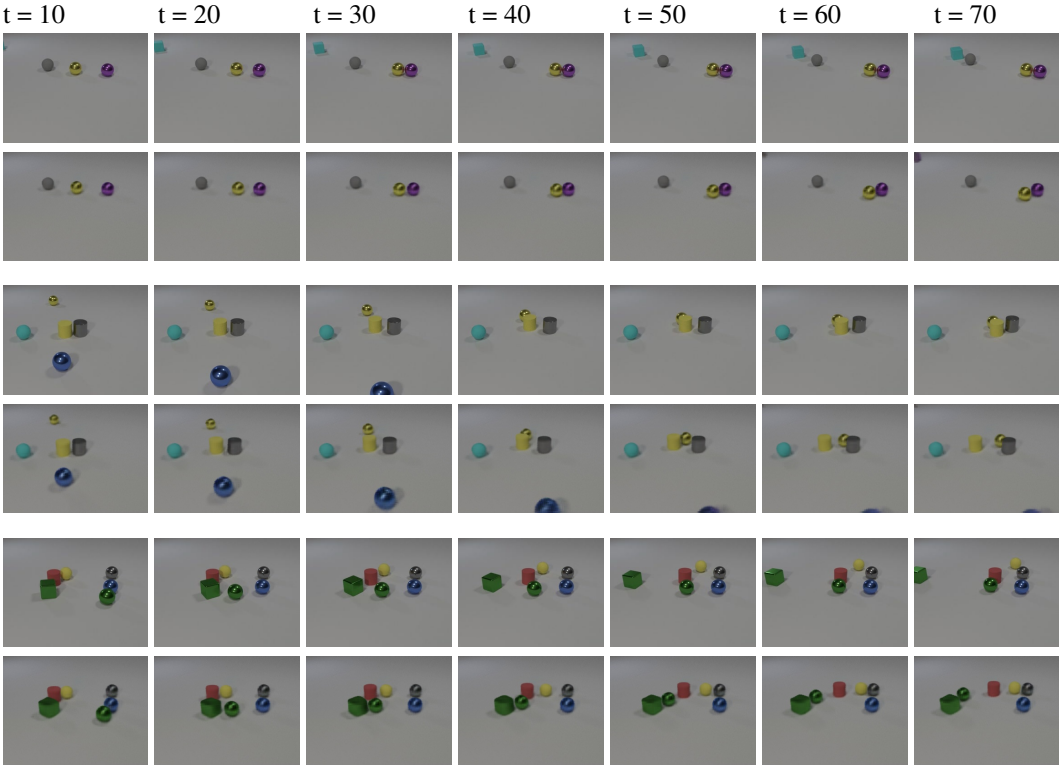

Figure 9: Predicting CLEVRER[85] 70 time-steps in closed loop after 50 times-steps of teacher forcing. Top row is ground truth wile bottom row shows Loci's closed loop predictions.

A.3 Ablations

Using the binary cross-entropy with non binary targets, while producing valid gradients, gives little insights into the network's actual performance. In order to better compare different designs and to also take the objects into focus, we use a modified L_2-loss for our ablation studies that is computed based on Equation 9:

$$L2_{object} = \sum \left((O^t_{b,c,h,w} - I^{t+1}_{b,c,h,w}) \sqrt{\sum_{c=1 \rightarrow 3} \frac{1}{3} \left(I^{t+1}_c - B^{t+1}_c \right)^2} \right)^2 \tag{9}$$

Here the MSE is masked with the error between the input and the background. As a result, we get a much higher error when the network produces a tracking error compared to a background reconstruction error. The $L2_{object}$ loss takes into account instances, where, for example, the network overlooks an object or, makes a mistake in the prediction of the movement of an object. It thus offers itself as a good metric for comparing the performance of different architectures during training.

A.3.1 GateL0RD vs LSTM

As shown in Figure 10, using GateL0RD within the predictor network significantly increases the $L2_{object}$-Loss. Thus, it appears that GateL0RD's piece-wise constant latent state regularization mechanism indeed suitably biases the network towards assuming object permanence.

A.3.2 Input channels

In Figure 12, we show the networks performance in terms of $L2_{object}$ when we zero out different input channels.

Table 3: Evaluations on the CATER Snitch challenge for supervised training. Table shows five networks trained with different initial seeds, each evaluated five times with different initial seeds.

Network	Top 1	Top 5	L_1 (grid)	L_2
1	91.7	99.0	0.118	0.122
	90.8	98.8	0.129	0.126
	91.1	98.8	0.119	0.121
	91.4	98.9	0.122	0.123
	91.3	98.8	0.123	0.126
2	91.1	98.6	0.129	0.133
	90.4	98.3	0.144	0.141
	89.8	98.3	0.151	0.144
	90.0	98.5	0.149	0.140
	89.5	98.2	0.164	0.151
3	89.6	97.8	0.173	0.170
	90.1	98.0	0.159	0.157
	89.5	97.9	0.174	0.164
	88.9	97.5	0.174	0.167
	89.8	98.1	0.162	0.157
4	91.5	99.1	0.112	0.118
	90.4	98.9	0.133	0.128
	91.0	98.7	0.125	0.129
	90.5	99.1	0.125	0.124
	90.8	98.8	0.127	0.126
5	91.7	98.2	0.130	0.139
	91.4	98.4	0.135	0.139
	91.2	98.0	0.146	0.150
	91.8	98.3	0.130	0.140
	91.3	98.1	0.150	0.151

Table 4: Evaluations on the CATER Snitch challenge for supervised training, statistics of the 25 evaluations presented in Table 3

Metric	min	mean	std	max
Top1	88.9	90.7	0.8	91.8
Top5	97.5	98.5	0.4	99.1
L_1 (grid)	0.112	0.140	0.019	0.174
L_2	0.118	0.139	0.015	0.170

A.3.3 REGULARIZATION LOSSES

In Figure 13, we show the networks performance in terms of $L2_{object}$ without certain regularization losses.

A.3.4 E-PROP VS BACK-PROPAGATION TROUGH TIME

In Figure 11 we compared the $L2_{object}$ loss for truncated back-propagation through time with different sequence lengths against a version where we used back-propagation together with e-prop. In all experiments the networks are trained on the full Cater sequence with 300 frames, which are fed into the network sequentially. We then updated and afterwards detached the gradients using truncated back-propagation through time with different time intervals. Using e-prop informed gradients not only drastically sped up training time, but it was also more sample-efficient while achieving the same performance as truncated back-propagation through time with an interval of three time steps. Especially interesting is the experiment where we tested truncated back-propagation with a sequence length of one, without using e-prop's informed gradients. In this case, the performance is significantly worse than when using e-prop's informed gradients.

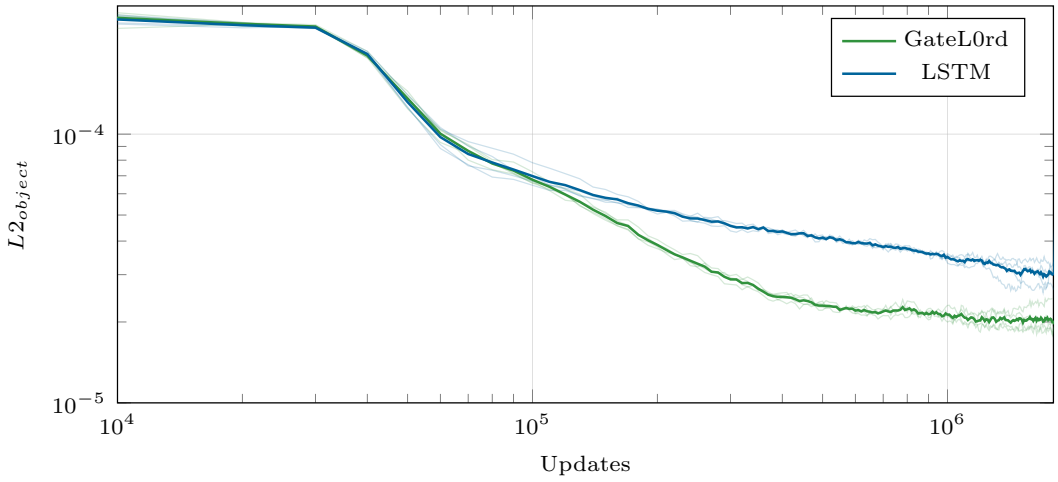

Figure 10: Comparison between GateL0RD and LSTM modules within the predictor part of the network.

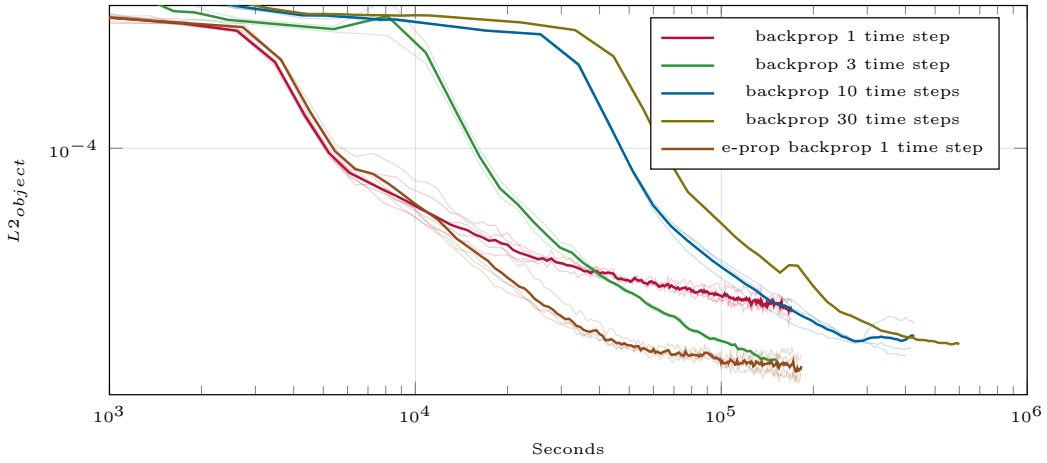

Figure 11: Comparison between different sequence lengths for truncated back-propagation trough time vs back-propagation and e-prop.

A.3.5 Lesion studies

In order to quantify the effect of the gestalt and position codes, we conducted an ablation by adding normally distributed noise with standard deviations $\sigma \in [0, 0.1, ..., 1.0]$ to the gestalt and position codes provided by the encoder before they were forwarded to the transition module and calculated the resulting accuracies and errors, which are reported in Figure 14. Furthermore, exemplary images of according position, priority, size, and gestalt code manipulations are presented in Figure 15. Effectively, manipulating the according codes result in changes in position, priority, size, or gestalt code, demonstrating the disentangled encoding of object features in the respective latent codes.

A.4 e-prop for GateL0RD

GateL0RD is defined in Equation 10 till Equation 15 following Gumbsch et al. [35] with the gating network $\mathfrak{g}$, the candidate network r, and the cell state c. Next, we detail how we applied e-prop in GateL0RD.

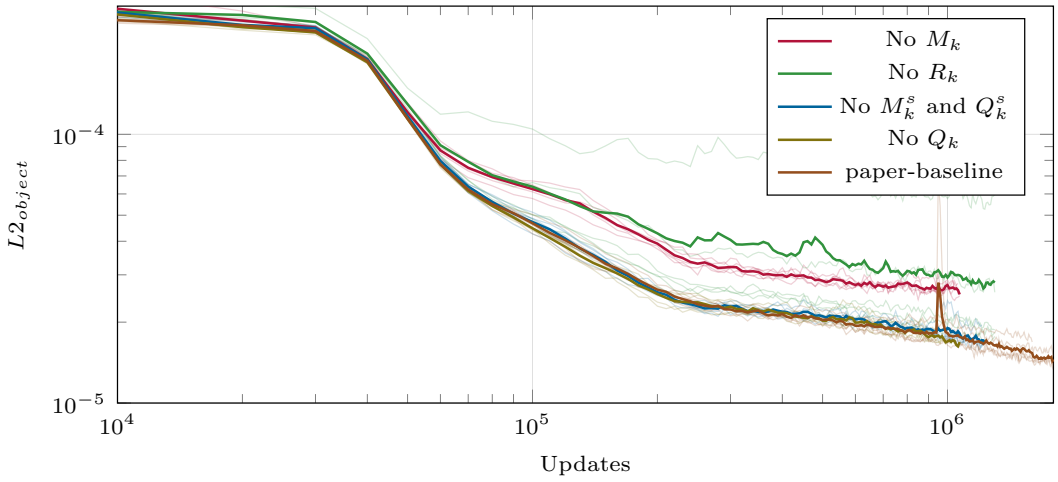

Figure 12: Ablation the importance of different input channels.

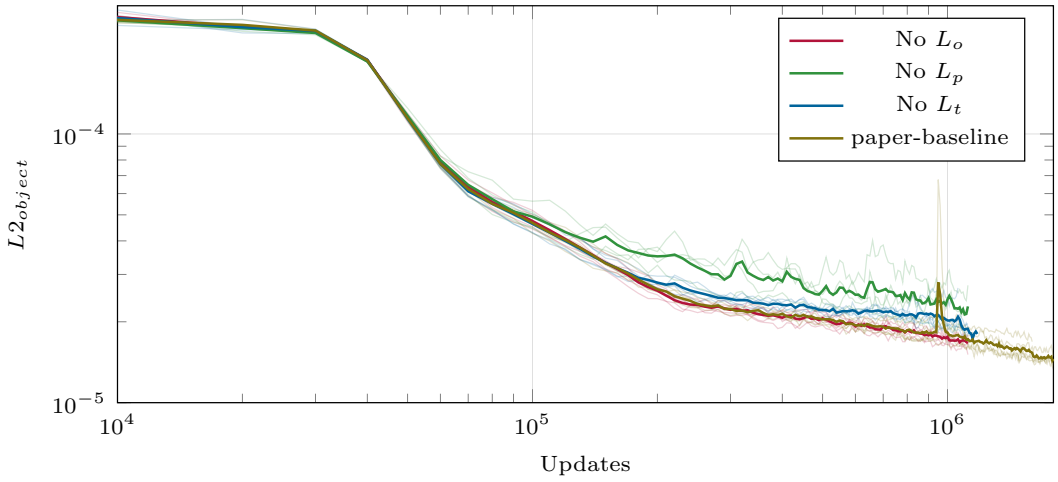

Figure 13: Ablation the importance of different regularization losses.

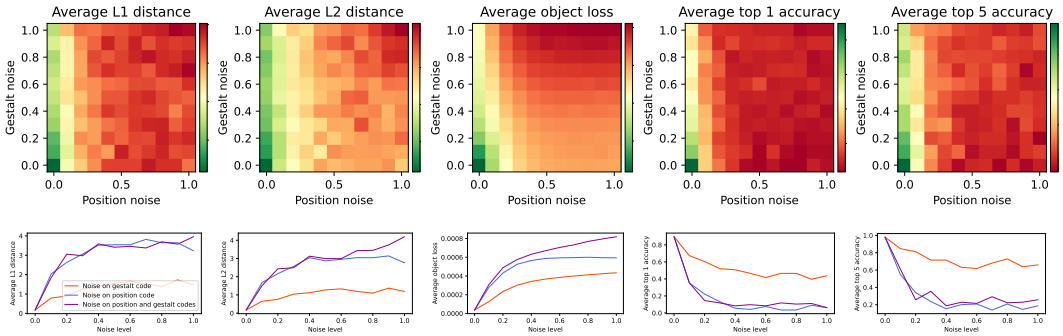

Figure 14: Top: Exhaustive evaluation of effects on the metrics resulting from different *noise* intensities *added to* the *gestalt and position codes*. Bottom: Explicit visualization of the bottom-most row (manipulating position only, blue), left-most column (manipulating gestalt only, orange), and diagonal from bottom left to top right in the plots at the top (manipulating both position and gestalt, purple).

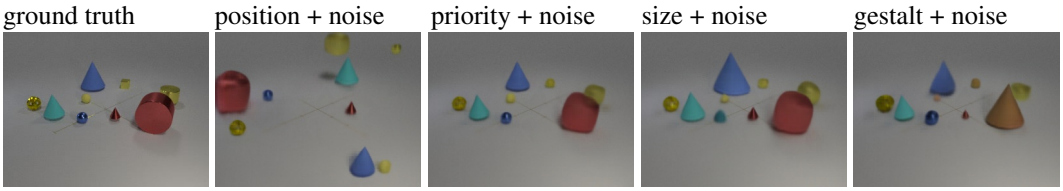

Figure 15: To study the disentanglement of the different parts of our learned codes, we perform a lesion study. The principle of the study is as follows: For each component code (position, priority, size, and gestalt) we add gaussian noise to the code before applying the decoder while keeping all other codes as predicted by the model.

A.4.1 GATEL0RD MODEL

$$\mathfrak{g}_j^t = \Lambda\left(\sum_i \theta_{j,i}^{rec,\mathfrak{g}} c_i^{t-1} + \sum_i \theta_{j,i}^{in,\mathfrak{g}} x_i^t + b_j^{\mathfrak{g}} + \mathcal{N}(0,\Sigma)\right) \quad \text{(gating network state)} \quad (10)$$

$$r_j^t = \phi\left(\sum_i \theta_{j,i}^{rec,r} c_i^{t-1} + \sum_i \theta_{j,i}^{in,r} x_i^t + b_j^r\right) \quad \text{(candidate network state)} \quad (11)$$

$$c_j^t = \mathfrak{g}_j^t r_j^t + (1-\mathfrak{g}_j^t) c_j^{t-1} + h_0^t \quad \text{(consequent cell state)} \quad (12)$$

$$p_j^t = \phi\left(\sum_i \theta_{j,i}^{rec,p} c_i^t + \sum_i \theta_{j,i}^{in,p} x_i^t + b_j^p\right) \quad \text{(output factor } p\text{)} \quad (13)$$

$$\mathfrak{o}_j^t = \sigma\left(\sum_i \theta_{j,i}^{rec,\mathfrak{o}} c_i^t + \sum_i \theta_{j,i}^{in,\mathfrak{o}} x_i^t + b_j^{\mathfrak{o}}\right) \quad \text{(output factor } o\text{)} \quad (14)$$

$$h_j^t = \mathfrak{o}_j^t p_j^t \quad \text{(resulting prediction at time } t\text{)} \quad (15)$$

A.4.2 E-PROP UPDATES:

core derivative and eligiblity determination:

$$\Lambda' = \frac{\partial \Lambda(x)}{\partial x} = \begin{cases} 0 & \text{if } x \leq 0 \\ (1-\Lambda(x)^2) & \text{otherwise} \end{cases} \quad \text{(pseudo-derivative of gate)} \quad (16)$$

$$\epsilon_{j,i}^t = \frac{\partial c_j^t}{\partial c_j^{t-1}} \epsilon_{j,i}^{t-1} + \frac{\partial c_j^t}{\partial \theta_{j,i}} = (1-\mathfrak{g}_j^t)\epsilon_{j,i}^{t-1} + \frac{\partial c_j^t}{\partial \theta_{j,i}} \quad \text{(cell-specific eligibility value)} \quad (17)$$

gating network forward eligibility propagation:

$$\frac{\partial E}{\partial \theta_{j,i}} = \frac{\partial E}{\partial c_j^t}\epsilon_{j,i}^t + \lambda \mathcal{H}(\mathfrak{g}_j^t)\frac{\partial \mathfrak{g}_j^t}{\partial \theta_{j,i}} \qquad \text{(actual update signal)} \qquad (18)$$

$$\epsilon_{j,i}^{rec,\mathfrak{g},t} = (1-\mathfrak{g}_j^t)\epsilon_{j,i}^{t-1} + \Lambda' c_i^{t-1}(r_j^t - c_j^{t-1}) \qquad \text{(recurrent weights-specific elig.)} \qquad (19)$$

$$\epsilon_{j,i}^{in,\mathfrak{g},t} = (1-\mathfrak{g}_j^t)\epsilon_{j,i}^{t-1} + \Lambda' x_i^t(r_j^t - c_j^{t-1}) \qquad \text{(input weights-specific elig.)} \qquad (20)$$

$$\epsilon_{j,i}^{b,\mathfrak{g},t} = (1-\mathfrak{g}_j^t)\epsilon_{j,i}^{t-1} + \Lambda'(r_j^t - c_j^{t-1}) \qquad \text{(bias weight-specific elig.)} \qquad (21)$$

candidate network forward eligibility propagation:

$$\frac{\partial E}{\partial \theta_{j,i}} = \frac{\partial E}{\partial r_j^t}\epsilon_{j,i}^t \qquad \text{(actual update signal)} \qquad (22)$$

$$\epsilon_{j,i}^{rec,r,t} = (1-\mathfrak{g}_j^t)\epsilon_{j,i}^{t-1} + (1-\left(r_j^t\right)^2)c_i^{t-1}\mathfrak{g}_j^t \qquad \text{(recurrent weights-specific elig.)} \qquad (23)$$

$$\epsilon_{j,i}^{in,r,t} = (1-\mathfrak{g}_j^t)\epsilon_{j,i}^{t-1} + (1-\left(r_j^t\right)^2)x_i^t\mathfrak{g}_j^t \qquad \text{(input weights-specific elig.)} \qquad (24)$$

$$\epsilon_{j,i}^{b,r,t} = (1-\mathfrak{g}_j^t)\epsilon_{j,i}^{t-1} + (1-\left(r_j^t\right)^2)\mathfrak{g}_j^t \qquad \text{(bias weight-specific elig.)} \qquad (25)$$

Note that the partial derivatives $\frac{\partial E}{\partial \theta_{j,i}}$ in Equation 18 and Equation 22 address the current time step t only. Also note that $\frac{\partial E}{\partial c_j^t}$ and $\frac{\partial E}{\partial r_j^t}$ carry true gradient information of the current time step, backprogated through the feedforward connections of the architecture. In contrast, original e-prop uses local approximations of the learning signal only, which does not allow to stack multiple layers without losing the exact local gradient.

Table 5: Evaluations of Loci's foreground segmentation masks. Trained networks from the Cater challenge are evaluated on CLEVR with mask by running the single CLEVR images 30 iterations through the network and then comparing the masks using the Intersecion over Union (IoU) metric. We compare a mean per mask accuracy (mask avg) and size weighted average that represents how many pixels where segmented correctly (pixel avg).

Network	mask avg (%)	pixel avg (%)
1	83.3	96.8
2	84.6	97.0
3	84.3	97.1
4	84.1	96.7
5	85.6	97.2

B LOCI ALGORITHM

Loci processes a sequence of RGB video-encoding images $I \in \mathbb{R}^{T \times H \times W \times 3}$. Processing is mostly done slot-wise, whereby the system is initialized with a variable number of K processing slots. Its main components consist of a *slot-wise encoder*, a *transition module*, and a *slot-wise decoder*. Moreover, a background processing module is implemented. The slot-wise encoder is implemented by a tree-structured ResNet-based processing encoder (see Figure B.6). The transition module processes *slot-wise temporal dynamics* and *between-slot interaction dynamics*. The slot-wise decoder is again implemented by a ResNet (see Figure B.7). For simple backgrounds, we use a Gaussian Mixture Model to obtain a default background estimate $\hat{R}_{bg}$, which is used for the whole training set. For more complex backgrounds we use an additional Auto-Encoder Module.

In the remainder, we denote scalar values by lower-case letters, tensors by upper-case letters, and vectors by bold letters. Slot-specific activities with a subscript $k \in 1, .., K$, while we denote time by the superscript t. We drop t for temporary values.

We now first define data and neural encodings sizes and types used throughout Loci's processing pipeline. We then specify neural activity initialization. Finally, we detail the unfolding overall processing loop.

B.1 SLOT-WISE ENCODER

Inputs The encoder inputs at each time step t consist of:

- RGB input image $I^t \in \mathbb{R}^{H \times W \times 3}$,
- MSE map $E^t \in \mathbb{R}^{H \times W \times 1}$,
- Slot-specific RGB image reconstructions $R_k^t \in \mathbb{R}^{H \times W \times 3}$,
- Slot-specific mask predictions $M_k^t \in \mathbb{R}^{H \times W \times 1}$,
- Slot-specific sums of the other slots' mask predictions $M_k^{t,s} = \sum_{k' \in \{1,..,K\} \setminus k} M_{k'}^t$
- Slot-specific isotropic Gaussian position maps $Q_k^t \in \mathbb{R}^{H \times W}$,
- Slot-specific sums of the other slots' Gaussian position maps $Q_k^{t,s} = \sum_{k' \in \{1,..,K\} \setminus k} Q_{k'}^t$
- Background mask $M_{bg}^t \in \mathbb{R}^{H \times W}$, which is determined via $1 - \sum_{k \in \{1,..,K\}} M_k^t$

Outputs Based on these inputs the slot-wise encoder network generates latent codes, which are forwarded to the transition module:

- Slot-specific position codes $P_k^t \in \mathbb{R}^4$ encode an isotropic Gaussian $(\boldsymbol{\mu}_k^t, \sigma_k^t)$ and a slot-priority code z_k^t
- Slot-specific Gestalt codes $G_k^t \in \mathbb{R}^{D_g}$.

B.2 TRANSITION MODULE

A transition module is used to refine these slot-codes and create a prediction for the next state, which is fed into the decoder. The input to the transition module equals P^t, G^t, which is processed across slots and per slot in the respective layers: Multi-Head Attention processes slot interactions (over slots), while GateL0RD processes slot-specific dynamics (per slot). In our main CATER implementation we use five attention layers with ten heads each with GateL0RD layers in between.

Outputs The output of the transition module has the same coding size as the input fed into it. Additionally, recurrent, slot-respective hidden states $\hat{H}_k^t$ are maintained in the time-recurrent GateL0RD layers:

- Slot-specific position codes $\hat{P}^t \in \mathbb{R}^4$,
- Slot-specific Gestalt codes $\hat{G}^t \in \mathbb{R}^{D_g}$,
- Slot-specific GateL0RD-layer-respective hidden states $\hat{H}^t \in \mathbb{R}^{D_h}$

B.3 SLOT-WISE DECODER

Inputs The outputs from the transition module $\hat{P}^t$ and $\hat{G}^t$ then act as the input to the decoder.

Outputs The output of the decoder includes the slot-respective masks and RGB reconstructions:

- Slot-specific mask outputs $\hat{M}_k^t \in \mathbb{R}^{H \times W}$,
- Slot-specific RGB image reconstructions $\hat{R}_k^t \in \mathbb{R}^{H \times W \times 3}$,

which are then used as part of the input at the next iteration as specified above, that is, $\hat{M}_k^t \rightarrow M_k^{t+1}$ and $\hat{R}_k^t \rightarrow R_k^{t+1}$.

We generate the combined reconstructed image R^t by summing all slot estimates $\hat{R}_k^{t-1}$ and the background estimate $\hat{R}_{bg}$ weighted with their corresponding masks $\hat{M}_k^{t-1}$ and $\hat{M}_{bg}^{t-1}$, as specified further in Algorithm 1.

B.4 SEQUENCE INITIALIZATION

At time step $t = 0$ we determine the network's inputs based on randomly generated, fictive position and Gestalt estimates for each slot k at time $t = -1$, which are then fed through the Loci decoder module. Initial position and Gestalt codes $\hat{P}_k^{-1}$ and $\hat{G}_k^{-1}$ are sampled from an isotropic Gaussian distribution $\mathcal{N}_{Position}\ (\boldsymbol{\mu}_p, \sigma_p)$ and a factorized Gaussian distribution $\mathcal{N}_{Gestalt}\ (\boldsymbol{\mu}_g, \boldsymbol{\sigma}_g)$ with learnable parameters $\boldsymbol{\mu}_p \in \mathbb{R}^3$, $\sigma_p \in \mathbb{R}$ and $\boldsymbol{\mu}_g \in \mathbb{R}^{D_g}$, $\boldsymbol{\sigma}_g \in \mathbb{R}^{D_g}$, respectively. The third position code value of $\hat{P}_k^{-1}$, that is, the Gaussian standard deviation $\hat{\sigma}_k^{-1}$, is initialized to zero, but lower bounded by $1/\texttt{width}$ via the algorithm cf. Line 17 in Algorithm1, that is, the distance between two neighboring pixels. The fourth position code value, that is, the priority value $\hat{z}_k^{-1}$, is set to its index value $\hat{z}_k^{-1} \leftarrow k$, inducing an ordered priority, which biases initial random slot assignments and thus bootstraps initial slot-assignment progress.

Based on the initial codes, we generate estimates of the output mask $\hat{M}_k^{-1}$, reconstruction $\hat{R}_k^{-1}$ and Gaussian positions Q_k^{-1} by calling the Loci slot-wise decoder (see Algorithm 1 line 19 and following): $\hat{M}_k^{-1}, \hat{R}_k^{-1}, Q_k^{-1} \leftarrow SlotWiseDecoder(\hat{P}_k^{-1}, \hat{G}_k^{-1})$. We finally determine the first RGB image reconstruction $\hat{R}^{-1}$.

The hidden states of the recurrent neural network GateL0RD are initialized to zero, that is, $\hat{H}^0 \leftarrow 0$.

B.5 MAIN PROCESSING LOOP

Loci first runs the main processing loop for ten time steps with the first video image given a particular image sequence. It thereby bootstraps the objects into individual slots, somewhat similar to previous slot-attention approaches [54]. After the ten initial time steps, Loci keeps re-initializing positions P_k^t of a slot k to random values (as specified above) given that $max(M_k^t)$ has been smaller than 0.5 for all time points until t. This induces an initial active search process. For longer video sequences, such as the Aquarium footage, this search process was also used for invisible slots, which increased the number of tracked objects, but negatively influenced the object permanence.

Note that the prediction error is calculated across the three RGB channels. It determines the MSE between the input I^t and the static background R_{bg}, multiplied per Hadamard-product with the forth square root MSE between the input I^t and the predicted input R^{t-1}. The fourth square root emphasizes small differences, encouraging accurate encodings of individual entities.

In the transition module we apply a single trainable parameteric bias neuron alpha, as proposed in Bachlechner et al. [1], instead of layer-normalization. Alpha is initialized to zero. Its current value is multiplied with the output vector before the residual parts of the transition module. These alpha-residuals enforce the predictor to initially compute the identity function. The mechanism bootstraps Loci's learning progress by initially focusing it on developing decoder-suitable Gestalt encodings.

Algorithm 1 `LOCI-Algorithm` (main processing loop)

Inputs: Input video $I \in \mathbb{R}^{T \times H \times W \times 3}$, static background $\hat{R}_{bg} \in \mathbb{R}^{H \times W \times 3}$
Network parameters: $\Theta_{encoder}, \Theta_{transition}, \Theta_{decoder}$
Additional parameters: initialization parameters Θ_{init}; background threshold θ_{bg}, which is encoded as a uniform offset mask $\hat{M}_{bg} \leftarrow \theta_{bg}$

Initialize $H_k^{-1}, \hat{R}_k^{-1}, \hat{R}^{-1}, \hat{M}_k^{-1}, Q_k^{-1}$ # see Section B.4 for details
for $t = 0 \dots T-1$ **do**
  # Pre-processing:
  $E^t \leftarrow \sqrt{\texttt{Mean}\left((I^t - R_{bg})^2, axis = \text{'rgb'}\right)} \circ \sqrt[4]{\texttt{Mean}\left(\left(I^t - \hat{R}^{t-1}\right)^2, axis = \text{'rgb'}\right)}$
  $R_k^t, M_k^t, Q_k^t \leftarrow \hat{R}_k^{t-1}, \hat{M}_k^{t-1}, Q_k^{t-1}$
  # Slot-wise encoder:
  $S_k^t \leftarrow Encoder_{Trunk}(I^t, E^t, R_k^t, M_k^t, M_k^{t,s}, Q_k^t, Q_k^{t,s}, M_{bg}^t)$
  $G_k^t \leftarrow Encoder_{Gestalt}(S_k^t)$
  $P_k'^t \leftarrow Encoder_{Position}(S_k^t)$
  $P_k^t \leftarrow \texttt{Cat}\,[\boldsymbol{\mu}_k^t, \sigma_k^t, z_k^t] \leftarrow \texttt{Cat}\,[Encoder_{\boldsymbol{\mu}}(P_k'^t), Encoder_{\boldsymbol{\sigma}}(P_k'^t), Encoder_z(P_k'^t)]$
  # Transition module:
  $\hat{G}^t, \hat{P}^t, H^t = TransitionModule(G^t, P^t, H^{t-1})$
  # Gestalt binarization:
  $\hat{G}^t \leftarrow \texttt{Sigmoid}(\hat{G}^t)$
  $\hat{G}^t \leftarrow \hat{G}^t + \hat{G}^t(1 - \hat{G}^t)\mathcal{N}(0, 1)$
  # Slot-wise decoder:
  # p encodes all pixel positions normalized to $[-1, 1]$;
  # `width` denotes number of pixels in a row, inducing a lower bound of one pixel distance
  $Q_k \leftarrow \exp\left(\frac{-(p - \hat{\boldsymbol{\mu}}_k^t)^2}{2\max\left(\frac{1}{\texttt{width}}, \hat{\sigma}_k^t\right)^2}\right)$
  $decode_k \leftarrow \texttt{Priority-Based-Attention}(\hat{G}_k^t, Q_k, \hat{\boldsymbol{z}})$
  $\hat{R}_k^t, \hat{M}_k^t \leftarrow Decoder_{Trunk}(decode_k)$
  # Post-processing:
  $\hat{M}^t \leftarrow \texttt{Softmax}(\texttt{Cat}[\hat{M}_0^t \dots \hat{M}_k^t, \hat{M}_{bg}], axis = \text{'K'})$
  $\hat{R}^t \leftarrow \texttt{Sum}(\texttt{Cat}\left[\hat{R}_1^t, .., \hat{R}_K^t, \hat{R}_{bg}\right] \circ \hat{M}^t, axis = \text{'K'})$
end for
return $[R^0 \dots R^T]$

Algorithm 2 `Priority-based-Attention`

Inputs: Gestalt: $G_k \in \mathbb{R}^{1, D_g}$, Gaussian 2d position: $Q_k \in \mathbb{R}^{H' \times W' \times 1}$, priority: $\boldsymbol{z} \in \mathbb{R}^K$
Additional parameters: values of the learnable $\boldsymbol{\theta^w} \in \mathbb{R}^K$ are initially set 25, while $\boldsymbol{\theta^b} \in \mathbb{R}^K = \{0, 1, \dots, (K-1)\}$ induces a default slot-order bias.

$\boldsymbol{z'} \leftarrow (\boldsymbol{z} \cdot K + \mathcal{N}(0, 0.1) + \boldsymbol{\theta^b}) \circ \boldsymbol{\theta^w}$ # Scale priorities and add Noise

Subtract Gaussian attention from other slots, scaled by priority (σ denotes the sigmoid)
$Q_k' \leftarrow \max(0, Q_k - \sum_{k' \in \{1, \dots, K\} \setminus k} \sigma(z_k' - z_{k'}') \cdot Q_i)$
$combine_k \leftarrow Q_k' \times G_k$ # $combine_k \in \mathbb{R}^{H' \times W' \times D_g}$

return $combine_k$

B.6 ENCODER RESNET

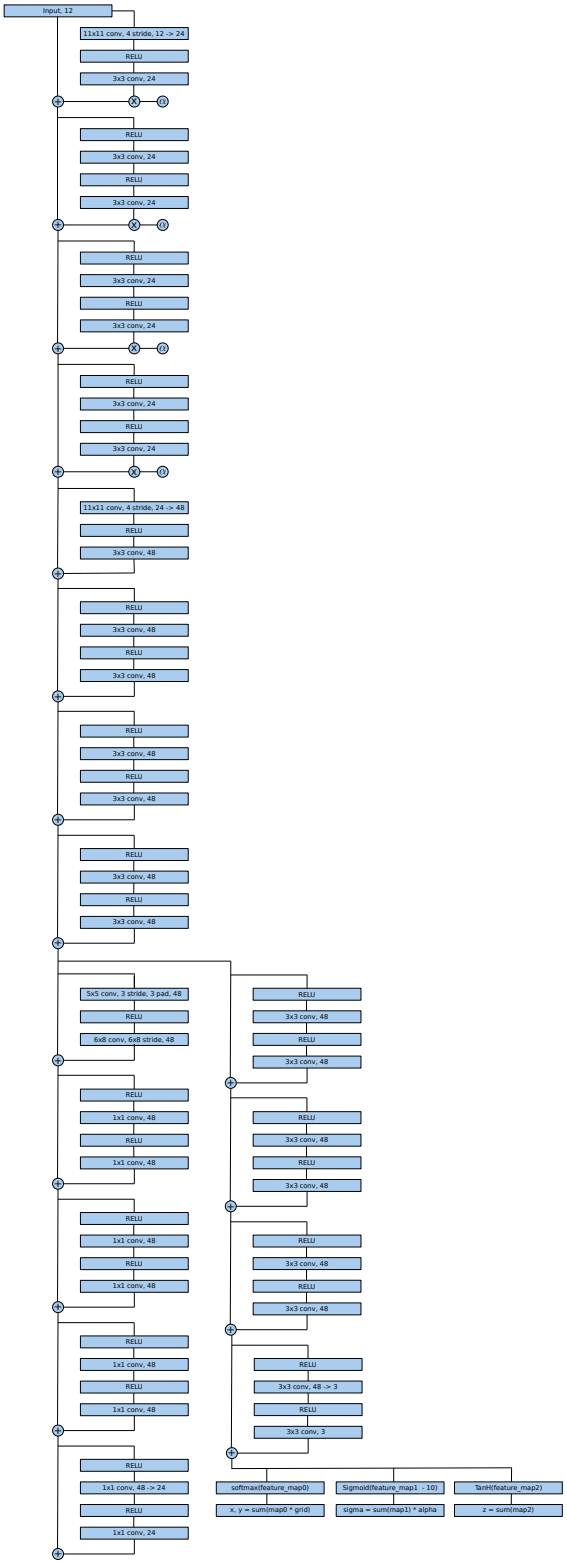

Figure 16: Encoder architecture diagram

B.7 DECODER RESNET

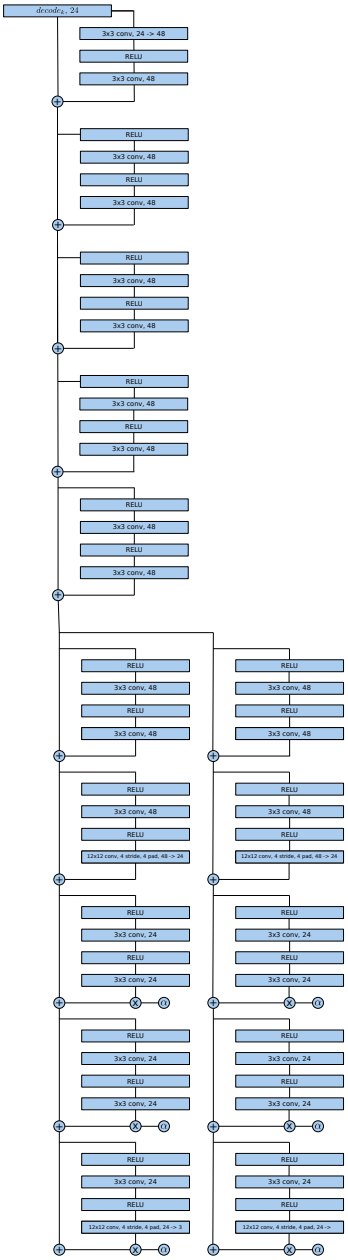

Figure 17: Decoder architecture diagram

C POTENTIAL NEGATIVE SOCIETAL IMPACTS

Here we take potential negative social impacts of Loci into consideration. Orientated on the NeurIPS Ethics Guidelines we identified two topics where Loci could potentially be used in an harmful way:

Could Loci directly facilitate injury to living beings? At its current stage it is unlikely that Loci could directly be used in any weapons systems. However, when developing Loci further, its unsupervised nature allows training on huge amounts of unlabeled data and then fine tune a weapons systems to identify and track a specific target.

Could Loci help develop or extend harmful forms of surveillance? Potentially yes; Loci already works well with static cameras and backgrounds. While we did not evaluate Loci's performance on pedestrians, it is likely that Loci could be trained to track pedestrians in a surveillance setting. Supervised fine-tuning could then potentially be used to identify a specific target group of people.

While these potential misuses of Loci are concerning, they are an unavoidable byproduct of advancing the field of (unsupervised) object tracking as a whole. Seeing that current object tracking systems are mostly black boxes, particularly when it comes to deciphering how and why they track certain entities, Loci may enable better control over what is tracked and where exactly tracking may be applied. Hopefully, Loci can thus be used to mitigate unwanted tracking biases, or, at least, to facilitate the identification of such tracking biases.