Unsupervised Image Representation Learning with Deep Latent Particles

Tal Daniel [1] **Aviv Tamar** [1]

Abstract

We propose a new representation of visual data that disentangles object position from appearance. Our method, termed Deep Latent Particles (DLP), decomposes the visual input into low-dimensional latent "particles", where each particle is described by its spatial location and features of its surrounding region. To drive learning of such representations, we follow a VAE-based approach and introduce a prior for particle positions based on a spatial-softmax architecture, and a modification of the evidence lower bound loss inspired by the Chamfer distance between particles. We demonstrate that our DLP representations are useful for downstream tasks such as unsupervised keypoint (KP) detection, image manipulation, and video prediction for scenes composed of multiple dynamic objects. In addition, we show that our probabilistic interpretation of the problem naturally provides uncertainty estimates for particle locations, which can be used for model selection, among other tasks. Videos and code are available: `https://taldatech.github.io/deep-latent-particles-web/`.

1. Introduction

The spatial positions of various parts of an image contain useful information for decision making. Examples include the positions of objects in video games (Smirnov et al., 2021), rigid object poses in robotic manipulation (Byravan & Fox, 2017), and image landmarks for human pose estimation (Jakab et al., 2018a). This observation motivates us to seek image representations where position is disentangled from other visual properties of objects in the scene.

In different image types, however, the definition of objects and their positions may vary (e.g., contrast the position of facial parts such as eyes and nose with the position of computer game sprites), leading us to pursue an unsupervised approach, where representations are data-dependent. In particular, we follow the generative approach – learn to reconstruct an image from its disentangled latent representation (Burgess et al., 2019).

This problem has recently gained attention in the variational autoencoding (VAE) literature, motivated by the observation that the typical single-vector representation of the entire scene has issues in scenes with multiple, varying number of objects. A common remedy is decomposing the scene into a pre-determined number of objects (Burgess et al., 2019; Watters et al., 2019; Kipf et al., 2019), and then learning a separate representation for each object. The main caveats with these methods are their strong assumptions on the number of objects, and their complexity, as the object discovery process is usually iterative and initialization-dependent (Burgess et al., 2019; Kipf et al., 2019).

An alternative representation is based on object landmarks, or *keypoints*, an idea that dates back to classical computer vision (Lowe, 1999). A keypoint (KP) representation is an unordered set of geometrical points (i.e., (x, y) locations in 2D scenes and (x, y, z) in 3D scenes). Several recent works investigated learning keypoints with deep learning methods (Jakab et al., 2018a; Thewlis et al., 2017a; Zhang et al., 2018; Lorenz et al., 2019; Dundar et al., 2021), exploiting the translational-invariance and spatial-locality of convolutional-based architectures. Predominantly, Jakab et al. (2018a) apply the spatial-softmax over feature maps extracted from a convolutional neural network (CNN) to determine the location of the keypoints. This idea became a building block in other methods that use KP for downstream tasks (Kulkarni et al., 2019; Gopalakrishnan et al., 2020; Boney et al., 2021; He et al., 2021), and has proven to be a promising alternative to the single-vector representation.

In this work, we propose a new representation method that draws inspiration from both KP and VAEs. Our idea is to *view the keypoints themselves as the latent space of the VAE*. To sufficiently capture image information, we accompany each KP with a set of features to describe the content in its vicinity, and refer to it as a "particle". Our method, termed Deep Latent Particles (DLP), inherits favorable traits from the VAE approach, such as a natural decoding scheme for

[1] Department of Electrical and Computer Engineering, Technion - Israel Institute of Technology, Haifa, Israel. Correspondence to: Tal Daniel <taldanielm@campus.technion.ac.il>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

reconstructing the image, and an uncertainty estimate for the particles that is based on the probabilistic interpretation of the latent variables. Meanwhile, as particles are jointly encoded and decoded, DLP does not require iterative inference, and can work with a larger number of objects (> 30) than prior VAE-based methods. Our method is also more flexible than recent patch-based approaches (Smirnov et al., 2021; Lin et al., 2020), which anchor objects to the center of the patch – our encoder allows particles to be freely located on the canvas, and for multiple particles to jointly model a single object (e.g., a large object).

Key for making our method work are two novel ideas. The first is to add a prior on particle positions that is based on the spatial-softmax architecture. The second is to learn the posterior of particle positions based on a modification of the evidence lower bound (ELBO) that is inspired by the Chamfer distance between particles, which we term the *Chamfer KL*. Building on these two ideas, we propose a VAE-inspired model where particles act as latent variables.

We demonstrate DLP on datasets that contain scenes with and without explicit objects. Our results show that the learned latent space effectively disentangles position from appearance. For example, when trained on CelebA (Liu et al., 2015) data, moving the particle located on the nose only controls the nose region in the output image. Importantly, we show that incorporating the particle uncertainty can benefit downstream tasks such as supervised landmark regression, where we demonstrate state-of-the-art results. Finally, we demonstrate that our method can be used to manipulate scenes with multiple objects by changing the location of the particles, and how this idea can be used for video prediction, by training a graph neural network (GNN) to predict the change in the particles.

Our contributions are summed as follows: (1) we propose a new unsupervised particle-based latent representation, trained with a novel modification of the VAE loss function based on the Chamfer distance for a set of points; (2) we show that our method is capable of extracting objects and their masks from multiple-object scenes without any supervision; (3) we experiment with various image datasets, showing the method's applications in keypoint discovery, image manipulation and video prediction; and (4) we demonstrate the benefits of incorporating the learned uncertainty information for model selection in the task of KP regression.

2. Related Work

Our work is inspired by ideas from unsupervised keypoint detection and unsupervised scene decomposition to objects.

Unsupervised keypoint detection: Thewlis et al. (2017a) represent the structure of an object as a set of keypoints learned via image deformations under equivariance constraints. Zhang et al. (2018) proposed an autoencoding-based landmark discovery approach with a constrained bottleneck, to improve upon Thewlis et al. (2017a). The method does not require pairs of images, but introduces several equivariance and separation constraints, making it more complex. Jakab et al. (2018a) proposed KeyNet[1] to learn KP using a tight bottleneck. KeyNet is simpler – the constraints of Zhang et al. (2018) are removed, but it requires pairs of images (video frames or augmented images). KeyNet outperformed Zhang et al. (2018) by a large margin. Transporter (Kulkarni et al., 2019) extends KeyNet by learning to transport image features between two frames for tracking objects and object parts across long time-horizons. The aforementioned methods learn deterministic keypoints, without uncertainty information as our DLP. Moreover, we show that in the single-image setting we outperform Zhang et al. (2018) without requiring their constraints, and outperform Jakab et al. (2018a) in the image-pairs setting as well.

Recently, several methods complimented KP discovery with modules that model object parts (Lorenz et al., 2019; Dundar et al., 2021). These methods disentangle shape and appearance by using various types of augmentations, and introduce components that mask foreground and background. This additional complexity proved useful, demonstrating outstanding results in various computer applications. Our method is simpler, as it uses standard reconstruction loss terms, and provides a natural uncertainty measure for the keypoints. Additionally, as DLP represents the whole image using the latent particles, it allows to explicitly control the generated scene, unlike previous methods that rely on feature maps from the original image, and can therefore only modify small local regions around the KP.

Unsupervised object-centric representations: unsupervised discovery of objects in scenes has mainly relied on sequential inference of objects, where in each iteration a new part of the input is attended to, or patch-based inference, where each patch can contain an object that needs to be represented. AIR (Eslami et al., 2016), SQAIR (Kosiorek et al., 2018), R-SQAIR (Stanić & Schmidhuber, 2019) and SPAIR (Crawford & Pineau, 2019) are based on sequential inference of objects and explicitly representing objects as 'what', 'where', and 'presence' latent variables, where the latter also adds a 'depth' variable. APEX (Wu et al., 2021b) leverages a similar approach but exploits temporal information in videos for better performance. These models are limited to a moderate number of objects, and typically struggle with modelling the scene's background. In contrast, our inference happens all at once, and the 'where' representation is replaced with a spatial prior over keypoint locations. MONet (Burgess et al., 2019) uses a sequential attention

[1] While Jakab et al. (2018a) did not officialy name their method, it is often referred to as KeyNet (e.g., Gopalakrishnan et al., 2020).

mechanism to allocate objects to slots, while Slot Attention (Locatello et al., 2020) replaces MONet's multi-step procedure with a single step using iterated attention between slots. IODINE (Greff et al., 2019) adds iterative refinement to objects, exhibiting similar performance to MONet, but requires more memory and is arguably more complex. GENESIS (Engelcke et al., 2019) and GENESISv2 (Engelcke et al., 2021) build on MONet and IODINE and introduce a generative model that captures relations between scene components with an autoregressive prior. Our method does not contain autoregressive components or iterative inference.

Finally, MarioNette (Smirnov et al., 2021) and SPACE (Lin et al., 2020) are non-sequential patch-based approaches. SPACE factors patches into 'what', 'where', 'depth', and 'presence' in parallel and is thus more scalable than the aforementioned methods, but has a tendency to embed objects in the background. MarioNette takes a different approach and learns a deterministic dictionary of objects or sprites that can appear in a scene. However, the dictionary approach is discrete in nature and is limited to objects seen during training. Our method is also non-sequential, but is not limited to patches or a static dictionary of learned objects.

Latent video prediction: recent advances in generative modelling (e.g., Razavi et al., 2019; Karras et al., 2020; Daniel & Tamar, 2021) have inspired a large body of video prediction methods that employ prediction in a learned latent space (Minderer et al., 2019b; Wu et al., 2021a; Walker et al., 2021; Villegas et al., 2019; Yan et al., 2021). Walker et al. (2021); Yan et al. (2021) model a sequence of discrete latent variables in the latent space of a vector-quantized VAE, while Wu et al. (2021a) and Villegas et al. (2019) focus on scaling-up latent autoregressive video prediction, the first via hierarchical VAEs and the latter via large stochastic recurrent neural networks (RNNs). V-CDN (Li et al., 2020) performs video prediction of physical interaction by building a causal graph from keypoints learned with Transporter (Kulkarni et al., 2019). Closely related to our work, Minderer et al. (2019b) uses KeyNet (Jakab et al., 2018a) to learn keypoints, and propose a variational RNN to model stochastic dynamics. We empirically compare with this approach, and report improved performance on datasets with varying number of objects, which we attribute to the GNN in our method. However, the dynamics model in Minderer et al. (2019b) is orthogonal to our work, and can potentially be used with our DLP representation as well.

3. Background

Variational Autoencoders (VAEs): VAEs (Kingma & Welling, 2014) learn an approximate model of data $p_\theta(x)$ using variational inference by maximizing the evidence lower bound (ELBO), which states that for any approximate posterior distribution $q(z|x)$:

$$\log p_\theta(x) \geq \mathbb{E}_{q(z|x)} \left[\log p_\theta(x|z)\right] - KL(q(z|x) \| p(z)) \\ \doteq ELBO(x), \quad (1)$$

where the Kullback-Leibler (KL) divergence is $KL(q(z|x)\|p(z)) = \mathbb{E}_{q(z|x)}\left[\log \frac{q(z|x)}{p(z)}\right]$. Typically, Gaussian distributions are used to model the approximate posterior $q_\phi(z|x)$, likelihood $p_\theta(z|x)$, and prior $p(z)$. The approximate posterior $q_\phi(z|x)$ is also known as the *encoder*, while $p_\theta(x|z)$ is termed the *decoder*. The ELBO can be maximized using the *reparameterization trick*, and in what follows, the term *reconstruction error* refers to $\log p_\theta(x|z)$.

KeyNet: The objective in Jakab et al. (2018a) is to produce a set of K 2D coordinates (a.k.a. keypoints/landmarks) $y = (u_1, ..., u_K), u_k \in \Omega$ for a given image, where $\Omega = [-1, 1]^2$ denotes the normalized space of 2D positions in the image. Consider an image $x \in \mathcal{R}^{H\times W\times 3}$. To extract keypoints, an encoder network (CNN) outputs K feature maps $S_u(x; k) \in \mathcal{R}^{H'\times W'}$, $k = 1, ..., K$. The keypoint u_k is finally generated from $S_u(x; k)$ using a spatial softmax (SSM) layer (Finn et al., 2016).

Gaussian Heatmaps: To backpropagate through the keypoint positions, Jakab et al. (2018a) broadcast each keypoint u_k into a Gaussian-like 2D heatmap centered around u_k with a small and constant standard deviation σ: $\Phi_u(x; k) = \exp\left(-\frac{1}{2\sigma^2}||u - u_k(x)||^2\right)$. These maps are then used as part of the image reconstruction process.

Chamfer distance: the distance between point clouds S_1 and S_2 of arbitrary size can be calculated with the Chamfer distance: $d_{CH}(S_1, S_2) = \sum_{x\in S_1} \min_{y\in S_2} ||x - y||_2^2 + \sum_{y\in S_2} \min_{x\in S_1} ||x - y||_2^2$.

4. Method

Our objective is to design a VAE for images $x \in \mathcal{R}^{H\times W\times C}$ where the latent representation is structured as a set of particles $z \in \mathcal{R}^{K\times(2+d)}$, where K is the number of particles, the first two components of each particle contain information about the positions of objects, and d additional features encode information about object appearance. Explicitly, we denote $z = [z_p, z_a]$, where $z_p \in \mathcal{R}^{K\times 2}$ and $z_a \in \mathcal{R}^{K\times d}$ denote the position and appearance components, respectively.

There are several challenges in doing so. The first is how to disentangle the positional information from other content in the particle features, without sacrificing reconstruction quality. Our idea is to exploit the probabilistic interpretation of the VAE, and view each particle position as coming from a distribution, where its prior is given by the standard method of Jakab et al. (2018a), which we already know produces reasonable keypoints. The posterior neural network, in contrast, is not restricted to any particular structure, to allow maximal expressiveness for accurately reconstructing the scene. The KL term in the VAE loss, which forces the posterior to be close to the prior, will drive the posterior to have positional meaning. This idea brings about another challenge – the prior can generate more keypoints than required by the posterior, so we need a method to enforce similarity between two sets of keypoint of different sizes. Finally, we need to also encode (and decode) the appearance information *around* the predicted keypoints. For this we propose the appearance encoder, which encodes 'glimpses' of the image around the keypoint in a differentiable way. In the following, we explain each of these components in detail. The model is illustrated in Figure 1.

4.1. Patch-wise Conditional Prior

In the standard VAE setting, the prior distribution is fixed and usually set to $p(z) = \mathcal{N}(0, I)$. Here we consider a conditional setting (Sohn et al., 2015), where we explicitly learn a prior for the keypoints given an image x, $p_\psi(z|x) = p_\psi(z_p|x) \times p_\psi(z_a|x)$.[2] We set $p_\psi(z_a|x)$ to the unit Gaussian prior $\mathcal{N}(0, I_d)$. The prior for the positions, however, requires more sophistication, as we describe next.

First, note that for computing the ELBO in Eq. (1), we do not need the explicit form of the prior, but only the KL divergence between the prior and approximate posterior. In the following, the prior distribution will be defined implicitly, through a set of prior keypoint proposals, and a particular KL divergence term to be described in Section 4.3.

To generate keypoint proposals, the input $x \in \mathcal{R}^{H \times W \times 3}$ is split into K_p (in general $K_p \neq K$) patches of size $D \times D$ (in our experiments $D \in \{8, 16\}$), and for each patch, the prior network outputs a distribution for the coordinates of a single keypoint proposal. This distribution is Gaussian $\mathcal{N}(\mu_p, \sigma_p)$, where the standard deviation is chosen to be a fixed small constant $\sigma_p = 0.1$, and the mean is the output of SSM, as described in Section 3. Therefore, for each image we obtain a set of K_p unordered keypoint proposals. In practice, as the set of proposals can grow large with the number of patches, we found it useful to consider only a subset of L prior keypoints, where L is a hyper-parameter and we term the set of prior keypoints *keypoint proposals*. To filter out L keypoints, we explored uniform sampling of L from the set of K_p proposals, and a heuristic where we keep the top-L distant keypoints from the center of their respective center. The logic behind this heuristic is that applying SSM in smooth patches (e.g., a solid color background) will result in a keypoint in the center of the patch, which might be uninformative. Both filtering methods resulted in similar performance, with a slight advantage to the heuristic.

4.2. Position Encoder and Appearance Encoder

The encoder models the approximate posterior, $q_\phi(z|x) = q_\phi(z_p|x) \times q_\phi(z_a|x, z_p)$. The *position encoder* $q_\phi(z_p|x)$ has a similar architecture to KeyNet (Jakab et al., 2018a): the input image is downsampled with convolutional layers ending with K feature maps $\Phi_{\text{enc}}(x) \in \mathcal{R}^{H' \times W' \times K}$. Unlike KeyNet, however, we do not apply SSM on these feature maps, but flatten them and use a fully-connected (FC) layer to map to K keypoints. The output of the FC layer is $\mu, \log(\sigma^2) \in \mathcal{R}^{K \times 2}$, the means and log variances of K independent Gaussians that make up $q_\phi(z_p|x)$.

For the *appearance encoder*, $q_\phi(z_a|x, z_p)$, we desire the features to encode the visual properties of the vicinity of each keypoint, in a differentiable manner. To that end, we use a Spatial Transformer Network (STN Jaderberg et al. 2015), similarly to SPAIR (Crawford & Pineau, 2019), to extract regions of size $S \times S$ from the original input at the locations specified by z_p, where the region size S around the keypoint is a hyperparameter (in our work, $S \in \{16, 32\}$). These *glimpses* go through a small CNN ending with a FC layer mapping to the parameters $\mu_f, \log(\sigma_f^2) \in \mathcal{R}^d$ of a Gaussian distribution for the features of each particle.

4.3. Chamfer-KL Distance

The use of a FC layer instead of SSM in the encoder gives the model additional freedom when choosing the ideal keypoint locations for reconstruction. Note, however, that the number of posterior keypoints K does not necessarily equal L, the number of prior keypoint proposals from $p_\psi(z_p|x)$.

We constrain the posterior to be close to the prior despite differences in the number and ordering of elements in each set, using a novel loss that we term the Chamfer-KL distance. Chamfer-KL views each point cloud as a set of Gaussian distributions and calculates the KL divergence between a keypoint in set S_1 and every keypoint in set S_2:

$$d_{CH-KL}(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} KL(x \| y) + \sum_{y \in S_2} \min_{x \in S_1} KL(x \| y).$$

Note that unlike the L_2 distance, the KL is asymmetrical and thus not a metric, a property which is maintained in the Chamfer-KL as $d_{CH-KL}(S_1, S_2) \neq d_{CH-KL}(S_2, S_1)$.

4.4. Decoder

The decoder in a VAE maps the latent variable z into an image. For our particle-based latents, we found that different decoder architectures work better for different types of scenes, depending on the variation of the background in the data, and whether the scene is composed of many separated objects or not. We next describe three basic decoder components, and how to combine them for different scenes.

The basic decoder component is an upsampling CNN

[2] Sohn et al. (2015) show that such a conditional prior complies with the ELBO formulation.

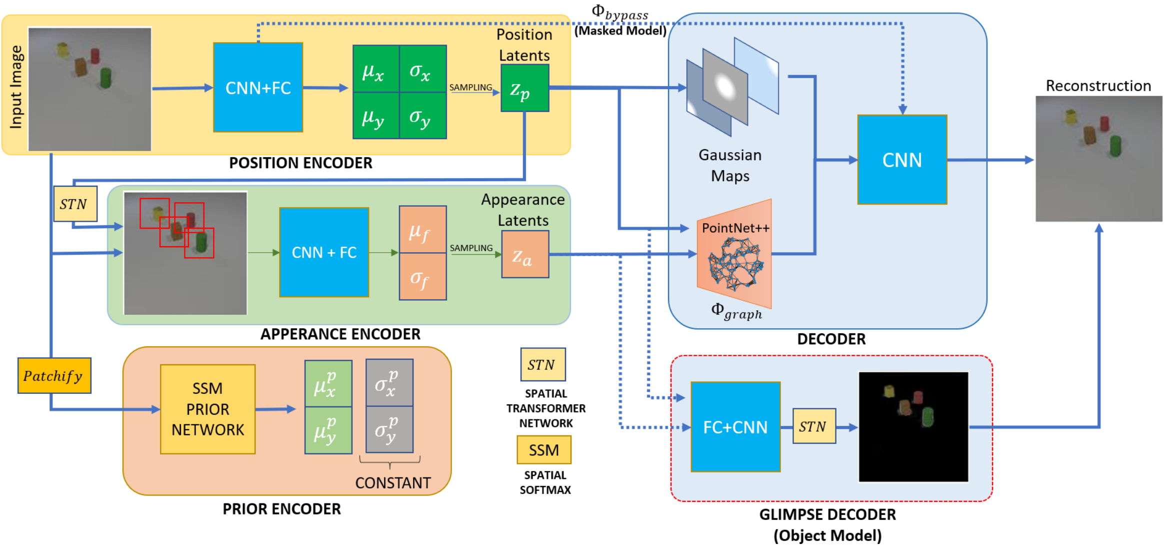

Figure 1. DLP architecture. Image is processed by the position encoder to produce the posterior probability of latent particle positions. These positions are used to extract glimpses from the original images using a STN, which are then processed by an appearance encoder to produce the appearance features for each particle. The input image (or an augmented view of it) is also processed by the prior network, producing keypoint proposals via SSM. To reconstruct the image, the particles are (1) transformed to differentiable Gaussian heatmaps and (2) go through a PointNet++ to produce feature maps Φ_{graph}. For the `Masked` model, the heatmaps are used as binary masks to combine local regions from Φ_{graph} with bypass features from the encoder. For the `Object` model, a separate glimpse decoder is used to decode RGBA patches, which are then combined with feature maps Φ_{graph} to produce the output image. See text for full details.

D_{upsample} taking in input feature maps and outputting the reconstructed image $\tilde{x} \in \mathcal{R}^{H \times W \times 3}$. The input feature maps of D_{upsample} are comprised of a concatenation of Gaussian heatmaps Φ_{heatmap} constructed from the coordinates of each particle as described in Section 3, and feature maps. We explored two methods for generating the feature maps, the *graph component* Φ_{graph} and the *bypass component* Φ_{bypass}.

Additionally, for multi-object scenes, we introduce a separate *glimpse decoder* D_{glimpse}, taking in a single particle and outputting an RGBA patch of the surrounding region around the particle. The output of D_{upsample} and the output of D_{glimpse} can be stitched together to create the final reconstructed image. We next overview each component; a full technical description is in the Appendix.

Graph component Φ_{graph}: The goal of this component is to use the particles for modelling the global structure of the scene. We create a KNN graph, where nodes are the particles, and edges are connected based on the Euclidean distance between particle positions. This graph is processed by a PointNet++ (Qi et al., 2017), which, using global max pooling, outputs feature maps $\Phi_p(x) \in \mathcal{R}^{H' \times W' \times M}$.[3]

Bypass component Φ_{bypass}: The goal of this component is to supply the background features that are not modelled by the particles. We simply take the feature maps from the encoder $\Phi_{\text{enc}}(x)$. A similar component was employed in Jakab et al. (2018a), but was used to model the whole scene without considering the features around the keypoints.

Glimpse decoder D_{glimpse}: This component reconstructs each object's appearance independently, as a combination of RGB values and a mask (alpha channel). We use a fully-connected layer followed by a small upsampling CNN that takes in a single latent particle z_i and decodes an RGBA patch of the surrounding region around the particle $\tilde{x}_i^p \in \mathcal{R}^{S \times S \times 4}$. The decoded patches are positioned in the full $H \times W$ canvas according to their respective particle's position, using a STN.

Equipped with the decoder components defined above, we now describe two decoder combinations that we explored:

1. `Masked` model: $\texttt{concat}[\Phi_{\text{heatmap}}, \alpha \times \Phi_{\text{graph}}, (1 - \alpha) \times \Phi_{\text{bypass}}] \rightarrow D_{\text{upsample}}$. The masks α are generated from the Gaussian heatmaps Φ_{heatmap}, and represent foreground.
2. `Object-based` model: $\alpha \times D_{\text{glimpse}} + (1 - \alpha) \times (\texttt{concat}[\Phi_{\text{heatmap}}, \Phi_{\text{graph}}] \rightarrow D_{\text{upsample}})$. The masks α are given by the alpha channel of D_{glimpse}.[4]

The `Object-based` model has a *pure* bottleneck – all information from the encoder to the decoder flows through the latent representation. In the `Masked` model, however, the bypass component skips the bottleneck. As we show in our experiments, the pure bottleneck model allows the

[3] In all our experiments we arbitrarily chose $M = K$.

[4] The careful reader may wonder why we used Φ_{graph} instead of Φ_{bypass} for the background. Empirically, we found that Φ_{bypass} can be too expressive, causing some of the objects to be represented by it instead of by D_{glimpse}.

particles more control over the output image, while the Masked model excels in reconstruction quality.

4.5. Training the Model

The training procedure is based on maximizing the ELBO (1). Since all of our distributions are Gaussians, the KL divergence has a closed form solution. Similar to the β-VAE (Higgins et al., 2017), we multiply the KL and Chamfer-KL terms in the loss by hyperparameters β_{KL} and β_{CKL}, respectively. To obtain better image quality, we replace the typical MSE reconstruction error with a variant of the VGG perceptual loss (Hoshen et al., 2019) that calculates the L_2 distance between the extracted features instead of the L_1, similarly to Jakab et al. (2018a).

The model is optimized end-to-end by Adam (Kingma & Ba, 2014), using the reparametrization trick, and is implemented in PyTorch (Paszke et al., 2017). Extended implementation details can be found in the Appendix. Our code is available publicly[5].

5. Experiments

Our method produces latent particles to represent an input image. We design our set of experiments to answer the following questions: (1) does our method effectively disentangle position from appearance in various scene types; (2) how important are our novel Chamfer-KL and the conditional SSM prior components; (3) what is the quality of our particles compared to other unsupervised keypoint discovery methods; (4) can our approach be used for downstream tasks such as image manipulation; and (5) can we exploit our probabilistic formulation to infer uncertainty estimates for the particles.

5.1. Linear Regression on Face Landmarks

A standard quantitative evaluation of unsupervised keypoint discovery is based on the error in predicting annotated keypoints from the discovered keypoints.

The common benchmark for this task uses the CelebA train set while excluding the MAFL (Zhang et al., 2014b) subset which includes annotations for 5 facial landmarks– eyes, nose and mouth corners. We train our Masked-model with a similar architecture and pre-processing as Jakab et al. (2018a) on 128 × 128 face images from CelebA (Liu et al., 2015). For the prior keypoint proposals we follow Jakab et al. (2018a) and use their proposed thin-plate-spline (TPS) augmentation for the prior image x_p, which is split to patches of size 8×8 ($D = 8$), resulting in a total of $K_p = 256$ which are then filtered to $L = 50$ KP proposals. We follow Thewlis et al. (2017a;b); Jakab et al. (2018a) and use the unsupervised keypoints to regress from $K = \{25, 30, 50\}$ to the annotated keypoints in the MAFL dataset. The linear regressor is learned without the bias term. For the regressor input, we experiment with using just the mean μ as features (deterministic KP, as in all previous works) and using the mean μ and the log-variance $\log(\sigma^2)$ as features. In Table 1 we report the results in terms of the standard MSE normalized by the inter-ocular distance expressed as a percentage. It can be seen that **our method improves upon the SOTA in unsupervised keypoint discovery**. Complete results, hyperparameters and an extended comparison can be found in Appendix E.

Method	K	MAFL
Zhang (Zhang et al., 2018)	30	3.16
KeyNet (Jakab et al., 2018a)	30	2.58
	50	2.54
Ours	25	2.87
	30	**2.56**
	50	**2.43**
Ours+ (with log-variance)	25	**2.52**
	30	2.49
	50	2.42

Table 1. **Comparison with state-of-the-art on MAFL.** K is the number of unsupervised landmarks. We report the MSE in % between predicted and ground-truth (lower is better) obtained from the Masked-model.

Information from uncertainty: to test whether the learned variance of each particle contains meaningful information, we performed two experiments. First, we trained our model with $K = 25$ particles and used the mean μ and the log-variance $\log(\sigma^2)$ as features for the supervised regression task described above. As seen in Table 1, we outperform Jakab et al. (2018a) with $K = 50$, even though the number of input features to the regressor is the same. In the second experiment, we used the model trained on $K = 30$ keypoints and chose the 10 keypoints with the highest variance and 10 keypoints with lowest variance, and used their means μ as the input features to the regressor. For the low-variance batch, we report an error of 5.75%, while for the high-variance batch the error was 7.54%. The results from both experiments indicate that **the posterior variance is related to uncertainty in the location of the keypoints, and can be useful for decision making in downstream tasks**. We further illustrate the connection between the location of the particles and their variance in Appendix E.2.

5.2. Scene Decomposition and Image Manipulation

A hallmark of latent space generative models is the ability to change the image by controlling the latents (Daniel & Tamar, 2021; Karras et al., 2020). Our model allows to modify the image in an intuitive way, by simply moving

[5] https://github.com/taldatech/deep-latent-particles-pytorch

around the particles.

We first demonstrate the latent control with the Masked model, trained with $K = 30$ particles on CelebA (Liu et al., 2015) (cf. Section 5.1), and compare with KeyNet (Jakab et al., 2018a), using their published pre-trained model with $K = 30$ keypoints (Jakab et al., 2018b). To perform manipulation, we visually locate the keypoints on the nose and mouth, slightly change their coordinates (leaving their features the same), and decode a new image. For a fair comparison, as keypoints differ between the models, we manually chose and moved the keypoints of Jakab et al. (2018a) to produce the most visually pleasing results. In Figure 2 we show the keypoints detected by each model, the reconstruction, and the resulting reconstruction after performing the above manipulation. As our latent space is structured to disentangle position and appearance, changing the position only affects the respective area of the particle by performing a smooth interpolation. KeyNet, on the other hand, has limited controllability as the latent space is only represented by the keypoints, and the features are propagated from the encoder, resulting in a blurry area near the keypoint position. Note that the manipulation in our model had a semantic effect – moving the lip particle closed the mouth and hid the teeth, while moving the nose particle up exposed the nostrils.

As can be observed in Figure 2, several particles may be located in a small region (e.g., multiple particles near the nose). This is an attribute of the model, when the number of keypoints chosen is larger than the natural number of keypoints required to represent the variation in the data. In such a case, not all particles have ‘control’: manipulating them will have no effect on the reconstructed image. Interestingly, we found that controllable, or salient, particles are assigned lower uncertainty, and a simple filtering heuristic can be used to automatically select the top-K salient particles, as we show in the supplementary material.

Next, we train our Object model on two multiple-object datasets: CLEVRER (Yi et al., 2019) dataset and Traffic – a self-collected traffic camera dataset. The CLEVRER dataset is composed of 5-second (128 frames) video of rigid objects colliding, where each frame can contain up to 8 objects of various shapes and colors. For this dataset, we learn $K = 10$ particles with feature dimension $d = 5$. Traffic is composed of 44,000 frames containing cars of different sizes and shapes. For this dataset, we learn $K = 15$ particles with feature dimension $d = 20$. We emphasize that while these datasets contain videos, our method works on single images, and therefore ignores any temporal relation between the frames. We downscale the frames in both datasets to 128×128, use a glimpse size $S = 32$, and do not use augmentations, i.e., $x_p = x$.

In Figure 3 we visualize a sample of the detected KP, the reconstructed images, the detected objects and their masks, and image manipulation by moving the KP (features remain the same). Evidently, **our method can learn to decompose scenes with a varying number of objects of different shapes and sizes, and allows for particle-based manipulation of scenes where particles control objects**. Additional results can be found in Appendix E.3[6].

We finally remark that empirically, we found that applying the Masked model in object-based scenes resulted in worse reconstructions, where objects were reconstructed as blurry blobs. Alternatively, using the Object-based model for non-object-based scenes reduced the manipulation ability. We noticed that the model tended to assign ‘objects’ to high-contrast parts of the image, such as the hairline and eyebrows, and ignored smoother parts such as the nose. We believe this is since high-contrast objects are easier to reconstruct using the alpha channel approach in the Object-based model.

5.3. Particle-based Video Prediction

The image manipulation results above suggest that our particles effectively control scene generation. We capitalize on this observation, and suggest to use particles for video prediction. Recall that particles are learned per image; our idea is to also learn a predictor for the *temporal change* in particles, from video sequence data. A natural predictor that can exploit the disentangled position and appearance is a Graph Convolutional Network (GCN, Kipf & Welling 2016), where each particle is a node, and connectivity is based on the Euclidean distance between particle positions. In this work, for simplicity, we chose a deterministic prediction model. Stochastic prediction, such as in Minderer et al. (2019b), can also be used with our model, and we leave that to future work.

We demonstrate our approach on the Traffic dataset, using our DLP model from Section 5.2. We employ a 2-layer Gated GCN (Bresson & Laurent, 2017) to predict the change in position Δz_p and appearance features Δz_a for each particle. To reduce drift in appearance, we constrain the maximal Δz_a to a small value. The GCN is trained to predict particles of two consecutive frames $[t, t+1]$ from particles in two previous frames $[t-1, t]$. Since DLP is trained per-image, particles in consecutive frames do not necessarily match. Therefore, we do not have a ground truth for $\Delta z_p, \Delta z_a$.[7] Instead, we decode an image from the predicted particles, and train the GCN to minimize the perceptual loss (Hoshen et al., 2019) with the ground truth future frame. Video pre-

[6] We have implemented a graphical use interface (GUI) for manipulating the images, please visit https://taldatech.github.io/deep-latent-particles-web/ for videos.

[7] In principle, the Chamfer distance can be used to resolve this, but in practice it only worked well for short horizon predictions.

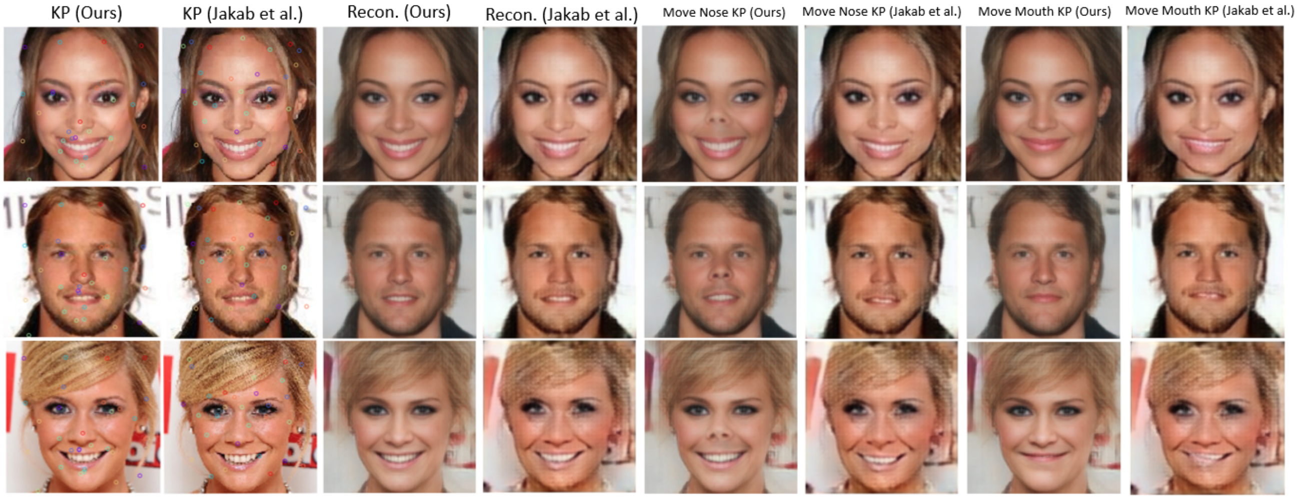

Figure 2. Image manipulation comparison with KeyNet (Jakab et al., 2018a). We visualize the keypoints learned by each model, the reconstruction, and the effect that moving keypoints on the nose and the mouth has on the output image.

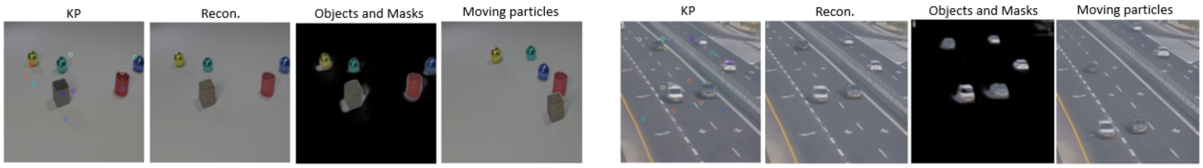

Figure 3. Scene decomposition and manipulation with `Object` model. Left - CLEVRER, right - Traffic. We show the detected particles, the reconstructed images, the objects and masks – output of the glimpse decoder, and image manipulation based on moving the particles.

dictions are generated by rolling out the GCN, starting from the particles of the first two images in the video, and using the decoder to generate images from predicted particles. We provide the complete technical description of our method in the supplementary. As a baseline, we trained the method of Minderer et al. (2019b), where keypoints are learned with KeyNet (Jakab et al., 2018a), on the Traffic dataset with the recommended hyperparameters (Minderer et al., 2019a).

As can be seen in Figure 4, our approach produces sharp predictions, even for significantly longer horizons than trained on. We see this as a promising approach to video prediction, which is often prone to blurriness (Lee et al., 2018; Yan et al., 2021; Hafner et al., 2020). We provide more details and results in the appendix. Note that the video prediction quality of the Minderer et al. (2019b) baseline is significantly worse. In the appendix, we verify that the baseline obtains good reconstructions per single frames. We hypothesize that the reason for the poor video prediction is the *varying number of objects* (cars) in the scene: in all of the experiments in Minderer et al. (2019b), the number of objects in the scene was fixed, and thus their recurrent neural network approach was reasonable. Our GNN approach can better account for a variable number of objects. Additionally, as the baseline uses KeyNet, it is less capable of manipulability, as we demonstrated above.

For more complex scenes such as CLEVRER, we found that our simple approach does not work well enough, as we describe in Section 6.

5.4. Ablative Analysis

We evaluate the importance of our novel components, the Chamfer-KL and SSM prior. The ablation of Chamfer-KL uses the conventional KL calculation performed by flattening the particle representation to a vector. For this comparison, we chose a model with the same number of prior and posterior keypoints, $K_p = K = 30$. For an ablation of the SSM-based conditional prior, we experimented with two alternative priors: (1) a Gaussian prior $\mathcal{N}(0, 0.1^2)$, similar to the standard VAE setting; and (2) prior keypoint proposals sampled uniformly $\mathcal{U}[-1, 1]$ instead of using the SSM. We experiment with the supervised KP regression task using the `Masked`-model. We run the training for 50 epochs and keep the rest of the hyperparameters similar to Section 5.1. As shown in Table 2, using **the Chamfer-KL and the SSM-based prior significantly improves the performance**.

6. Limitations and Future Work

We illustrate several limitations of our method, which we observed when trying to predict video on CLEVRER.

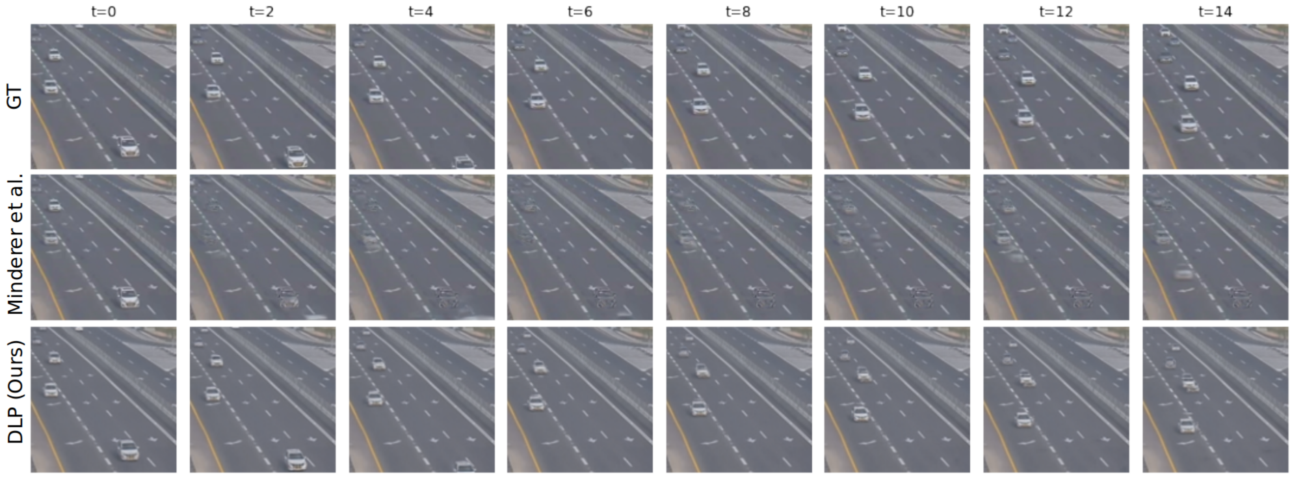

Figure 4. Video prediction on the Traffic dataset. We use the pre-trained `Object` model to provide the particle-representation and a GNN to predict the temporal change in particles. Top - ground truth, middle - Minderer et al. (2019b) prediction, bottom - DLP (ours) prediction.

Prior/ KL	Standard KL	Chamfer-KL
Constant $\mathcal{N}(0, 0.1^2)$	3.18	3.07
Random $\mathcal{U}[-1, 1]$	3.39	2.99
SSM	3.24	**2.65**

Table 2. **Ablation study.** The effect of the choice of prior and KL implementation on the supervised KP regression on the MAFL dataset. Reported results are the MSE in % between predicted and ground-truth KP (lower is better). Setting is different than in Table 1, see text for full information.

A particle shared between two objects: when the data contains objects that frequently appear together, a single particle can be assigned to more than one object. This is due to our single-image formulation – there is no signal for our method to separate the two objects. We believe that our method can be extended to a sequential setting, using the prior x_p, which would potentially resolve this issue for objects that change position during the video. **Large objects and occlusions:** several particles are assigned to large objects. This can become a problem when objects get partially occluded, requiring drastic changes to the particle features. More expressive GCNs trained on longer horizons may potentially mitigate this problem. An alternative is to modify the model to better account for occlusions. **Changing background:** the proposed model will have difficulty when the background changes dramatically, such as when the camera is moving. Incorporating some background detection into our method is one direction for addressing this.

7. Conclusion

In this work we showed that the classical concept of keypoint detection can be viewed in the lens of deep generative modelling, by viewing the keypoints themselves as the latent variables in a variational autoencoder model. Beyond the elegance of the formulation, we showed that our method can generate SOTA results in keypoint discovery, and be used for intriguing image manipulations.

Many questions remain. For example, extending the method to handle more complex video prediction, where objects change appearance dramatically, or occlude other objects. Another exciting direction is to leverage developments in VAEs, such as the vector-quantized VAE (Razavi et al., 2019) for improved performance. We also intend to explore the use of the uncertainty estimate in our model for decision making. More broadly, we are hopeful that this new connection between generative models and keypoint detection will spur up interesting developments in image representations.

8. Acknowledgements

The authors would like to thank Ron Mokadi for collecting and sharing the Traffic dataset used in this work. This work is partly funded by the Israel Science Foundation (ISF-759/19), the Open Philanthropy Project Fund, an advised fund of Silicon Valley Community Foundation, and by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

References

Boney, R., Ilin, A., and Kannala, J. End-to-end learning of keypoint representations for continuous control from images. *arXiv preprint arXiv:2106.07995*, 2021.

Bresson, X. and Laurent, T. Residual gated graph convnets. *arXiv preprint arXiv:1711.07553*, 2017.

Burgess, C. P., Matthey, L., Watters, N., Kabra, R., Higgins, I., Botvinick, M., and Lerchner, A. Monet: Unsupervised scene decomposition and representation. *arXiv preprint arXiv:1901.11390*, 2019.

Burgos-Artizzu, X. P., Perona, P., and Dollár, P. Robust face landmark estimation under occlusion. *2013 IEEE International Conference on Computer Vision*, pp. 1513–1520, 2013.

Byravan, A. and Fox, D. Se3-nets: Learning rigid body motion using deep neural networks. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 173–180. IEEE, 2017.

Crawford, E. and Pineau, J. Spatially invariant unsupervised object detection with convolutional neural networks. In *AAAI*, 2019.

Daniel, T. and Tamar, A. Soft-introvae: Analyzing and improving the introspective variational autoencoder. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4391–4400, June 2021.

Dundar, A., Shih, K., Garg, A., Pottorff, R., Tao, A., and Catanzaro, B. Unsupervised disentanglement of pose, appearance and background from images and videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

Engelcke, M., Kosiorek, A. R., Jones, O. P., and Posner, I. Genesis: Generative scene inference and sampling with object-centric latent representations. *arXiv preprint arXiv:1907.13052*, 2019.

Engelcke, M., Jones, O. P., and Posner, I. Genesis-v2: Inferring unordered object representations without iterative refinement. *arXiv preprint arXiv:2104.09958*, 2021.

Eslami, S., Heess, N., Weber, T., Tassa, Y., Szepesvari, D., Hinton, G. E., et al. Attend, infer, repeat: Fast scene understanding with generative models. *Advances in Neural Information Processing Systems*, 29:3225–3233, 2016.

Fey, M. and Lenssen, J. E. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

Finn, C., Tan, X. Y., Duan, Y., Darrell, T., Levine, S., and Abbeel, P. Deep spatial autoencoders for visuomotor learning. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 512–519. IEEE, 2016.

Frankle, J., Schwab, D. J., and Morcos, A. S. Training batchnorm and only batchnorm: On the expressive power of random features in cnns. *arXiv preprint arXiv:2003.00152*, 2020.

Gopalakrishnan, A., van Steenkiste, S., and Schmidhuber, J. Unsupervised object keypoint learning using local spatial predictability. *arXiv preprint arXiv:2011.12930*, 2020.

Greff, K., Kaufman, R. L., Kabra, R., Watters, N., Burgess, C., Zoran, D., Matthey, L., Botvinick, M., and Lerchner, A. Multi-object representation learning with iterative variational inference. In *International Conference on Machine Learning*, pp. 2424–2433. PMLR, 2019.

Hafner, D., Lillicrap, T., Norouzi, M., and Ba, J. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

He, X., Wandt, B., and Rhodin, H. Latentkeypointgan: Controlling gans via latent keypoints. *arXiv preprint arXiv:2103.15812*, 2021.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. β-vae: Learning basic visual concepts with a constrained variational framework. *ICLR*, 2(5):6, 2017.

Hoshen, Y., Li, K., and Malik, J. Non-adversarial image synthesis with generative latent nearest neighbors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5811–5819, 2019.

Jaderberg, M., Simonyan, K., Zisserman, A., et al. Spatial transformer networks. *Advances in neural information processing systems*, 28:2017–2025, 2015.

Jakab, T., Gupta, A., Bilen, H., and Vedaldi, A. Unsupervised learning of object landmarks through conditional image generation. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 4020–4031, 2018a.

Jakab, T., Gupta, A., Bilen, H., and Vedaldi, A. https://github.com/tomasjakab/imm, 2018b.

Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., and Aila, T. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8110–8119, 2020.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. In Bengio, Y. and LeCun, Y. (eds.), *ICLR*, 2014.

Kipf, T., van der Pol, E., and Welling, M. Contrastive learning of structured world models. *arXiv preprint arXiv:1911.12247*, 2019.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Kosiorek, A. R., Kim, H., Posner, I., and Teh, Y. W. Sequential attend, infer, repeat: Generative modelling of moving objects. *arXiv preprint arXiv:1806.01794*, 2018.

Kulkarni, T., Gupta, A., Ionescu, C., Borgeaud, S., Reynolds, M., Zisserman, A., and Mnih, V. Unsupervised learning of object keypoints for perception and control. *arXiv preprint arXiv:1906.11883*, 2019.

Lee, A. X., Zhang, R., Ebert, F., Abbeel, P., Finn, C., and Levine, S. Stochastic adversarial video prediction. *arXiv preprint arXiv:1804.01523*, 2018.

Li, Y., Torralba, A., Anandkumar, A., Fox, D., and Garg, A. Causal discovery in physical systems from videos. *arXiv preprint arXiv:2007.00631*, 2020.

Lin, Z., Wu, Y.-F., Peri, S. V., Sun, W., Singh, G., Deng, F., Jiang, J., and Ahn, S. Space: Unsupervised object-oriented scene representation via spatial attention and decomposition. *arXiv preprint arXiv:2001.02407*, 2020.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Locatello, F., Weissenborn, D., Unterthiner, T., Mahendran, A., Heigold, G., Uszkoreit, J., Dosovitskiy, A., and Kipf, T. Object-centric learning with slot attention. *arXiv preprint arXiv:2006.15055*, 2020.

Lorenz, D., Bereska, L., Milbich, T., and Ommer, B. Unsupervised part-based disentangling of object shape and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10955–10964, 2019.

Lowe, D. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pp. 1150–1157 vol.2, 1999. doi: 10.1109/ICCV.1999.790410.

Minderer, M., Sun, C., Villegas, R., Cole, F., Murphy, K. P., and Lee, H. https://github.com/google-research/google-research/tree/master/video_structure, 2019a.

Minderer, M., Sun, C., Villegas, R., Cole, F., Murphy, K. P., and Lee, H. Unsupervised learning of object structure and dynamics from videos. *Advances in Neural Information Processing Systems*, 32, 2019b.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017.

Qi, C. R., Yi, L., Su, H., and Guibas, L. J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017.

Razavi, A., van den Oord, A., and Vinyals, O. Generating diverse high-fidelity images with vq-vae-2. In *Advances in neural information processing systems*, pp. 14866–14876, 2019.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009. doi: 10.1109/TNN.2008.2005605.

Shu, Z., Sahasrabudhe, M., Guler, R. A., Samaras, D., Paragios, N., and Kokkinos, I. Deforming autoencoders: Unsupervised disentangling of shape and appearance. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 650–665, 2018.

Smirnov, D., Gharbi, M., Fisher, M., Guizilini, V., Efros, A. A., and Solomon, J. MarioNette: Self-supervised sprite learning. *Conference on Neural Information Processing Systems*, 2021.

Sohn, K., Lee, H., and Yan, X. Learning structured output representation using deep conditional generative models. In *NIPS*, 2015.

Stanić, A. and Schmidhuber, J. R-sqair: relational sequential attend, infer, repeat. *arXiv preprint arXiv:1910.05231*, 2019.

Sun, Y. K., Wang, X., and Tang, X. Deep convolutional network cascade for facial point detection. *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3476–3483, 2013.

Thewlis, J., Bilen, H., and Vedaldi, A. Unsupervised learning of object landmarks by factorized spatial embeddings. In *Proceedings of the IEEE international conference on computer vision*, pp. 5916–5925, 2017a.

Thewlis, J., Bilen, H., and Vedaldi, A. Unsupervised learning of object frames by dense equivariant image labelling. *arXiv preprint arXiv:1706.02932*, 2017b.

Villegas, R., Pathak, A., Kannan, H., Erhan, D., Le, Q. V., and Lee, H. High fidelity video prediction with large stochastic recurrent neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

Walker, J., Razavi, A., and Oord, A. v. d. Predicting video with vqvae. *arXiv preprint arXiv:2103.01950*, 2021.

Watters, N., Matthey, L., Bosnjak, M., Burgess, C. P., and Lerchner, A. Cobra: Data-efficient model-based rl through unsupervised object discovery and curiosity-driven exploration. *arXiv preprint arXiv:1905.09275*, 2019.

Wiles, O., Koepke, A., and Zisserman, A. Self-supervised learning of a facial attribute embedding from video. *arXiv preprint arXiv:1808.06882*, 2018.

Wu, B., Nair, S., Martin-Martin, R., Fei-Fei, L., and Finn, C. Greedy hierarchical variational autoencoders for large-scale video prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2318–2328, 2021a.

Wu, Y., Jones, O. P., Engelcke, M., and Posner, I. Apex: Unsupervised, object-centric scene segmentation and tracking for robot manipulation. *arXiv preprint arXiv:2105.14895*, 2021b.

Yan, W., Zhang, Y., Abbeel, P., and Srinivas, A. Videogpt: Video generation using vq-vae and transformers. *arXiv preprint arXiv:2104.10157*, 2021.

Yi, K., Gan, C., Li, Y., Kohli, P., Wu, J., Torralba, A., and Tenenbaum, J. B. Clevrer: Collision events for video representation and reasoning. *arXiv preprint arXiv:1910.01442*, 2019.

Zhang, J., Shan, S., Kan, M., and Chen, X. Coarse-to-fine auto-encoder networks (cfan) for real-time face alignment. In *ECCV*, 2014a.

Zhang, Y., Guo, Y., Jin, Y., Luo, Y., He, Z., and Lee, H. Unsupervised discovery of object landmarks as structural representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2694–2703, 2018.

Zhang, Z., Luo, P., Loy, C. C., and Tang, X. Facial landmark detection by deep multi-task learning. In *European conference on computer vision*, pp. 94–108. Springer, 2014b.

Zhang, Z., Luo, P., Loy, C. C., and Tang, X. Learning deep representation for face alignment with auxiliary attributes. *IEEE transactions on pattern analysis and machine intelligence*, 38(5):918–930, 2015.

A. Extended Architecture Details

Encoders: all of the encoders described in this work follow a similar scheme to Jakab et al. (2018a). Unless mentioned otherwise, input images are assumed to be of 128×128 resolution. The position encoder is composed of a CNN with convolutional blocks where each block contains a convolutional layer, followed by Batch Normalization and `ReLU` activation. Downsmapling is performed by using strided ($s = 2$) convolutional layers with 'replication' padding. The channels of each convolutional block are $[32, 64, 128, 256]$ and the feature maps are of shape $16 \times 16 \times 256$. These maps are flattened and fed into 3-layer fully-connected network with hidden layers of size $[256, 128]$, each activated with `ReLU`, outputting $K \times 4$ position values, reshaped to $[\mu, \log(\sigma^2)]$. The prior encoder, operating on patches of sizes 16×16 or 32×32, is composed of a CNN with a similar structure to the position encoder with channels $[16, 32, 64]$ followed by a spatial softmax (SSM) block. Finally, the glimpse encoder has similar structure as the prior encoder, but the SSM is replaced with a FC network similar to the one in the position encoder.

Φ_{graph} - PointNet++: to decode feature maps from the latent particles, we use a PointNet++ (Qi et al., 2017) implemented as a GNN (Scarselli et al., 2009). First, a KNN graph with $K = 10$ is built from the position of the particles. This graph is then processed by a 4-layer PointNet++ layers, composed of 1-D convolution, `ReLU` activation and Batch Normalization, with channels $[64, 128, 256, 512]$. The output of the convolutional blocks is then max-pooled to produce a vector which is reshaped to $[K, 8, 8]$ maps, where K is the number of particles. This is implemented efficiently with the `torch geometric` library (Fey & Lenssen, 2019). Finally, these maps are upsampled with a convolutional layer to $[16, 16, K]$ maps.

D_{upsample} - Upsampling CNN: this component takes in a concatenation of feature maps; For the `Masked` model, the concatenation is of the Gaussian heatmaps $\Phi_{heatmap}$, masked feature maps from the encoder Φ_{enc} and masked feature maps from Φ_{graph}. The final shape of the input to $D_{upsample}$ is $[16, 16, K \times 3]$. For the `Object` model, the concatenation is of the Gaussian heatmaps $\Phi_{heatmap}$ and feature maps Φ_{graph}. The final shape of the input to $D_{upsample}$ is $[16, 16, K \times 2]$. The upsampling CNN network has a symmetric structure to the CNN in the position encoder, but in reverse, namely, the CNN channels are: $[256, 128, 64, 32]$. Finally, the output is aggregated with a 1×1 convolutional layer to produce a RGB image of shape: $[128, 128, 3]$.

D_{glimpse} - Glimpse decoder: this component takes in a latent vector which is mapped with a 2-layer fully-connected network with 256 hidden units each and `ReLU` activated to a vector of size $8 \times 8 \times 32$. This vector is reshaped to feature maps of shape $[8, 8, 32]$ that are upsampled with a 2-layer CNN (similar blocks as before) with 64 channels in each layer to maps of shape $[S, S, 64]$, where S is the glimpse size. This output is aggregated with a 1×1 convolutional layer and activated with a `Sigmoid` activation to produce RGBA patches of shape $[S, S, 4]$.

B. Extended Implementation and Training Details

In this section, we describe important implementation and training details for convergence of our method.

Training objective: the complete training objective follows the β-VAE (Higgins et al., 2017) formulation:

$$\mathcal{L} = \mathcal{L}_{rec}(x, \tilde{x}) + \beta_{CKL} CH - KL(q_\phi(z_p|x) \| p_\psi(z|x)) + \beta_{KL} KL(q_\phi(z_a|x) \| p(z)).$$

We found it crucial to balance the two KL terms, where usually $\beta_{CKL} > \beta_{KL}$. We report the exact values for each task in Appendix D.

Initialization, LR scheduling and latent activation: in our implementation, we initialized the convolutional layers with values from $\mathcal{N}(0, 0.01)$. We used a general multi-step learning scheduler that decreases the learning by 0.5 in each milestone. The milestones are reported in Appendix D. Moreover, to constrain the values of the particles' position to be in the range of $[-1, 1]$, we used a `TanH` activation on μ.

Binary masks from Gaussian heatmaps: for the `Masked` model, we use a binary mask $M_u(x)$ created from the Gaussian as follows:

$$M_u^i(x) = \begin{cases} 1, & \Phi^i_{heatmap}(x) \geq \tau \\ 0, & \text{else} \end{cases},$$

Algorithm 1 Stitching Algorithm

```
**Input:** alpha maps $a_{i=1}^{K}$, RGB maps $r_{i=1}^{K}$, background map $b$
Initialize $currMask = a[1]$, $masks = []$.
**for** $i = 2$ **to** $K$ **do**
    $availableSpace = 1.0 - currMask$
    $currMaskTmp = Min(availableSpace, a[i])$
    Append $currMaskTmp$ to $masks$
    $currMask = currMask + currMaskTmp$
**end for**
$alphaMask \leftarrow$ Sum($masks$)
$rec = (1 - alphaMask) * bg + masks * r$
Return $rec$
```

for some threshold τ ($\tau = 0.2$ in our experiments). Now, the input of D_{upsample} is a concatenation of $\Phi_{heatmap}$, $M \odot \Phi_{\text{graph}}$ and $(1 - M) \odot \Phi_{enc}(x)$.

Warm-up, noisy masks and transparency variable: for the `Object` model, we found it beneficial to have a short warm-up stage of 2 epochs where only the glimpse encoder and glimpse decoder are training to encode and decode patches. This warm-up step prevents D_{upsample} to take full responsibility of the reconstruction and ignore D_{glimpse}. Moreover, at the beginning of the training (5-10 epochs) we add a small Gaussian noise $\mathcal{N}(0, 0.01)$ to the alpha channel of the decoded patches following Smirnov et al. (2021). This additional step encourages learning sharper object masks. Finally, following Smirnov et al. (2021), we learn an additional 'transparency' parameter for each particle $z_{on} \in [0, 1]$ which is multiplied by the decoded glimpse prior to stitching the final reconstructed image. This implemented by extending the output dimension of the FC layer in Φ_{enc} and using `Sigmoid` activation for that variable.

Frozen prior network Interestingly, even though the prior network can be trained (the SSM is differentiable), we found that in some datasets (e.g. Traffic and CLEVRER), keeping the network frozen, with the initial random weights, worked better. This is inline with the findings of Frankle et al. (2020).

Stitching the image: we follow a layer-wise approach (Smirnov et al., 2021) and describe our stitching algorithm in Algorithm 1.

C. Datasets

In this section we provide a detailed description of the datasets we used throughout this paper.

CelebA (Liu et al., 2015): this dataset contains 200k images of celebrity faces which are cropped and resized to 128 × 128 following Jakab et al. (2018a); Thewlis et al. (2017a). The dataset provides annotations for 5 facial landmarks — eyes, nose and mouth corners, which are not required during training. As per the setting of (Jakab et al., 2018a; Thewlis et al., 2017a), the MAFL (Zhang et al., 2014b) test-set is excluded from training.

CLEVRER (Yi et al., 2019): this dataset contains 20,000 synthetic videos of moving and colliding objects, separated to 10,000 train video, 5,000 validation videos and 5,000 test videos, where each video is 5 seconds long and contains 128 frames with resolution 480 × 320. In our work, we resize the frames to 128 × 128 and use a subset of the frames created by skipping every second frame.

Traffic: a self-collected traffic camera dataset composed of 44,000 frames resized to 128 × 128 containing cars of different sizes and shapes, where we take the first 90% of the frames for train and the rest for evaluation.

D. Hyperparameters Details

In this section we provide the complete set of hyperparameters used for the experiments in this work. The shared hyperparameters between all of the experiments are described below, and the rest can be found in Table 3.

dataset	Model	K	K_p	L	β_{CKL}	β_{KL}	prior patch size	glimpse size S	feature dim d	epochs
CelebA	`Masked`	10	256	15	20	$20 * 0.001$	8	16	30	90
CelebA	`Masked`	25	256	30	30	$30 * 0.001$	8	16	10	90
CelebA	`Masked`	30	256	50	40	$40 * 0.001$	8	16	10	90
CelebA	`Masked`	50	256	50	50	$50 * 0.001$	8	16	10	90
Traffic	`Object`	15	64	64	30	$30 * 0.001$	16	32	20	100
CLEVRER	`Object`	10	64	64	40	$40 * 0.001$	16	32	5	120

Table 3. Detailed hyperparameters used for the various experiments in the paper.

Shared hyperparameters: all models were trained with an initial learning rate of $2e-4$ and a a batch size of 32 per GPU (we used 1 to 4 GPUs). For all datasets, we used a multi-step learning rate scheduler with the following milestones (in epochs): $[30, 60]$ with learning rate decreasing by 0.5 on each milestone. The warm-up stage described in Appendix B was only used for the `Object` model, where we used 1 warm-up epoch for CLEVRER and 2 for Traffic, and the number of 'noisy masks' epochs was 5 times the warm-up epochs–5 and 10 for CLEVRER and Traffic, respectively.

E. Complete Results

In this section, we provide extended results for the various tasks we presented in the main paper.

E.1. Supervised Regression on Face Landmarks

In Table 4 we present the complete set of results for the supervised KP regression task. It can be seen that our method outperforms the supervised and unsupervised benchmark for $K = \{30, 50\}$, and combined with the uncertainty information and the learned features, the results are further improved. It is worth noting that simply learning features without the notion of location results in bad performance as reported by Jakab et al. (2018a), stressing that the learned features are only informative when learned with respect to their position information.

E.2. Uncertainty Information Analysis

In this section, we demonstrate a visual connection between the location of each particle and its learned variance. Each keypoint u_k is defined by the Gaussian parameters (μ_k, σ_k), where σ_k^2 can be interpreted as the variance in the location of this keypoint. Intuitively, for common patterns in the data, we should expect the variance to be small. Accordingly, we define the per-keypoint uncertainty as follows:

$$V(u_k) \doteq \sum_i \log(\sigma_{k_i}^2),$$

where σ_{k_i} is the standard deviation in the i^{th} axis (i.e., the x and y coordinates) of u_k.[8]

To test our hypothesis, we use two trained models: (1) `Masked` model from Section 5.1 on 128 × 128 face images from CelebA (Liu et al., 2015) and (2) `Object` model from Section 5.2 on the Traffic dataset. The `Masked` was trained with $K = 30$ particles and the `Object` model with $K = 15$. In Figure 5 we plot the K keypoints learned by our model and display the top-10 keypoints with highest confidence. It can be seen that the keypoints with the highest confidence lie on locations that are common across the dataset (i.e., eyes, nose and mouth) while the rest lie in regions of higher variability (e.g., hair and background).

E.3. Scene Decomposition and Image Manipulation

We provide more results for the different experiments described in Section 5. First, we compare image manipulation of the `Masked` model with KeyNet (Jakab et al., 2018a) on CelebA in Figure 6. The experimental setting, where both models learn $K = 30$ keypoints, is the same as in Section 5.2.

[8] One may also consider the variance in the features for each particle. However, to illustrate our idea of disentangling position from appearance, we only consider position uncertainty.

Method	K	MAFL
Supervised		
RCPR (Burgos-Artizzu et al., 2013)		–
CFAN (Zhang et al., 2014a)		15.84
Cascaded CNN (Sun et al., 2013)		9.73
TCDCN (Zhang et al., 2015)		7.95
MTCNN (Zhang et al., 2014b)		5.39
Unsupervised / Self-supervised		
Thewlis (Thewlis et al., 2017a)	30	7.15
	50	6.67
Thewlis (Thewlis et al., 2017b)(frames)	–	5.83
Shu (Shu et al., 2018)	–	5.45
Zhang (Zhang et al., 2018)	10	3.46
	30	3.16
Wiles (Wiles et al., 2018)	–	3.44
KeyNet (Jakab et al., 2018a)	10	3.19
	30	2.58
	50	2.54
Lorenz (Lorenz et al., 2019)	10	3.24
Dundar (Dundar et al., 2021)	10	2.76
Ours	10	3.87
	25	2.87
	30	2.56
	50	2.43
Ours+ (with log-variance)	10	3.12
	25	2.52
	30	2.49
	50	2.42
Ours++ (with learned features)	10	2.98
	25	2.42
	30	2.36
	50	2.39

Table 4. **Comparison with state-of-the-art on MAFL.** K is the number of unsupervised landmarks. Reported results are the MSE in % between predicted and ground-truth (lower is better).

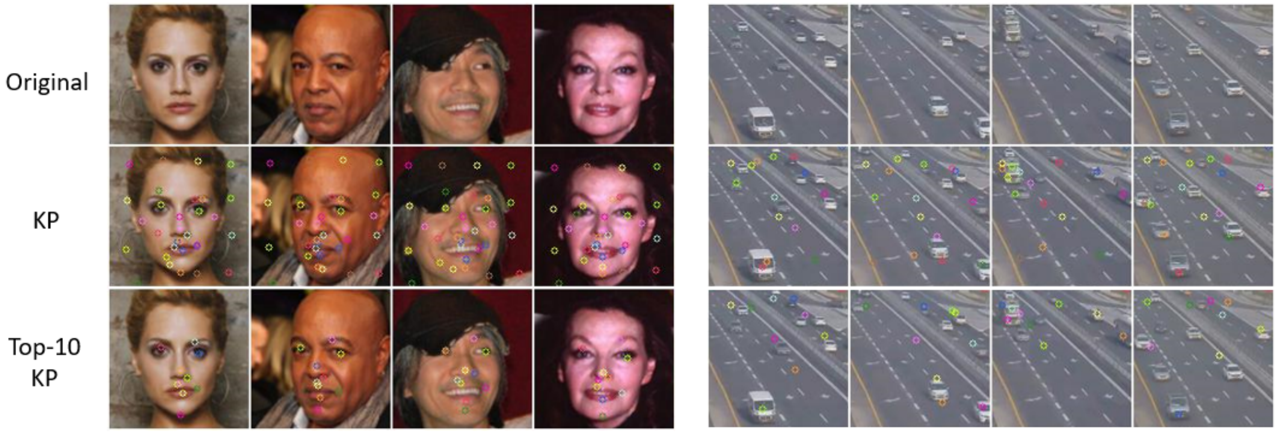

Figure 5. Information from uncertainty. We show the $K = 30$ particles (second row) learned from two models and top-10 particles with the highest confidence (third row): (1) `Masked` model trained on CelebA (left); and (2) `Object` model trained on Traffic (right).

In Figure 7 we provide more manipulations on CelebA produced by moving the particles in the face area.

Next, we provide extended results of our `Object` model. In Figures 8 and 9 we visualize the detected particles, the reconstructed images, the objects and masks – output of the glimpse decoder, and image manipulation based on moving the particles on CLEVRER and Traffic, respectively.

E.4. Video Prediction

We present more results for the video prediction experiment we presented in Section 5.3. Video predictions are generated by rolling out the GCN, starting from the particles of the first two images in the video, and using the decoder to generate images from predicted particles. As can be seen in Figures 10 and 11, our approach produces sharp predictions, even for significantly longer horizons than trained on. For animated sequences please see supplementary material. In Figure 13 we plot show single-frame reconstructions and video prediction of the method proposed in Minderer et al. (2019b) trained on the Traffic dataset.

F. Video Prediction Experiment Details

In this section, we provide technical details of the video prediction experiment we described in Section 5.3. The objective in this experiment is to learn a predictor for the temporal change of the particles, from video sequence data. Recent advances in graph neural networks (GNN, Scarselli et al. 2009) provide a natural framework to learn on graph-structured data. GNNs, such as Graph Convolutional Network (GCN, Kipf & Welling 2016), can learn to extract details relevant to interaction between nodes in the graph. With DLP, we model each particle as a node and the connectivity between nodes is based on the Euclidean distance between particle positions. Furthermore, each node is described by the particle's features, namely, its position and appearance features.

To predict the change in position Δz_p and appearance features Δz_a for each particle, we employ a 2-layer Gated GCN (Bresson & Laurent, 2017), with 128 hidden units each and activated with `ReLU`. To make the network more expressive, we add a 2-layer 1D convolutional network implemented as a shared MLP operating on each node separately, with 128 units each and activated with `ReLU`. The GCN is trained to predict particles of two consecutive frames $[t, t+1]$ from particles in two previous frames $[t-1, t]$ as follows: first, for each frame, we extract its respective particle-representation using a pre-trained DLP model and we concatenate a one-hot vector to its features indicating the time-step (e.g., $[1, 0]$ is concatenated to the features of the particle representation at time-step $t-1$). Then, we build a *radius* ($r = 0.2$) graph from the resulting set of nodes based on the Euclidean distance between the particles. This graph is fed into the GCN, which via the message-passing process outputs Δz_p and Δz_a for the consecutive time-steps $[t, t+1]$. These updates are activated with a `TanH` activation and multiplied by constants $\gamma_p = 0.2$ and $\gamma_a = 0.02$ to constrain the maximal Δz_p and Δz_a, respectively. Finally, the Δ is added to the original particles (a residual connection) and the new particle is decoded back to the image space using the

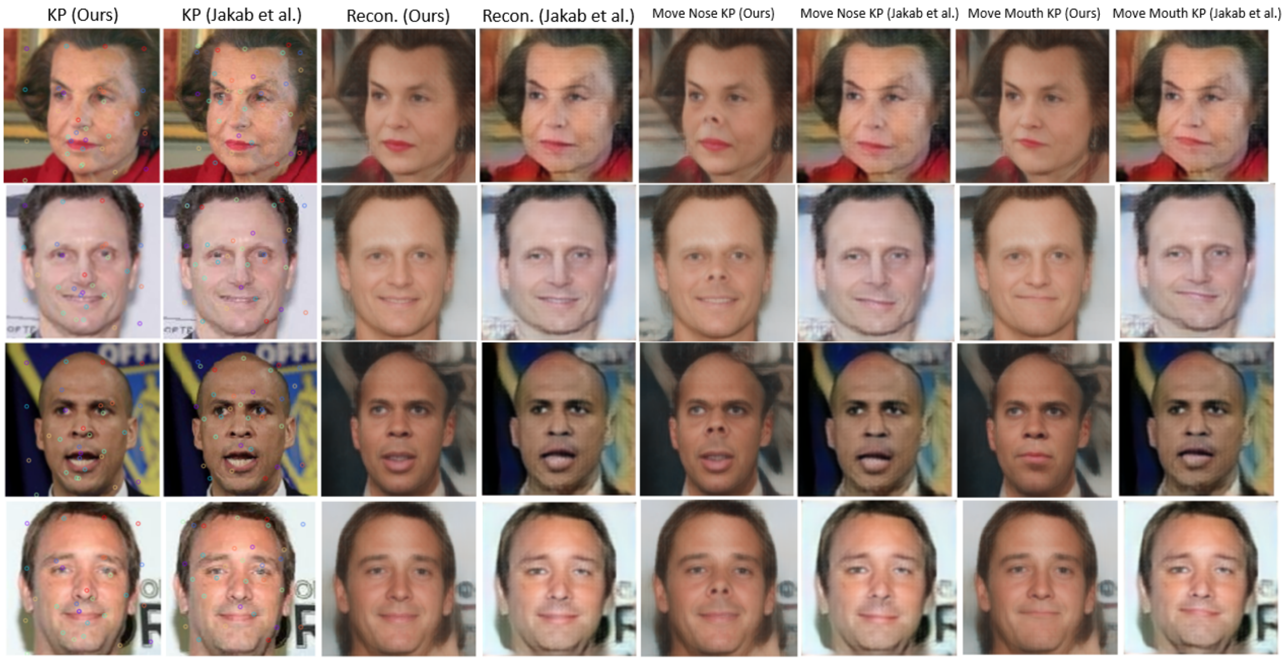

Figure 6. Image manipulation comparison with KeyNet (Jakab et al., 2018a). We visualize the keypoints learned by each model, the reconstruction, and the effect that moving keypoints on the nose and the mouth has on the output image.

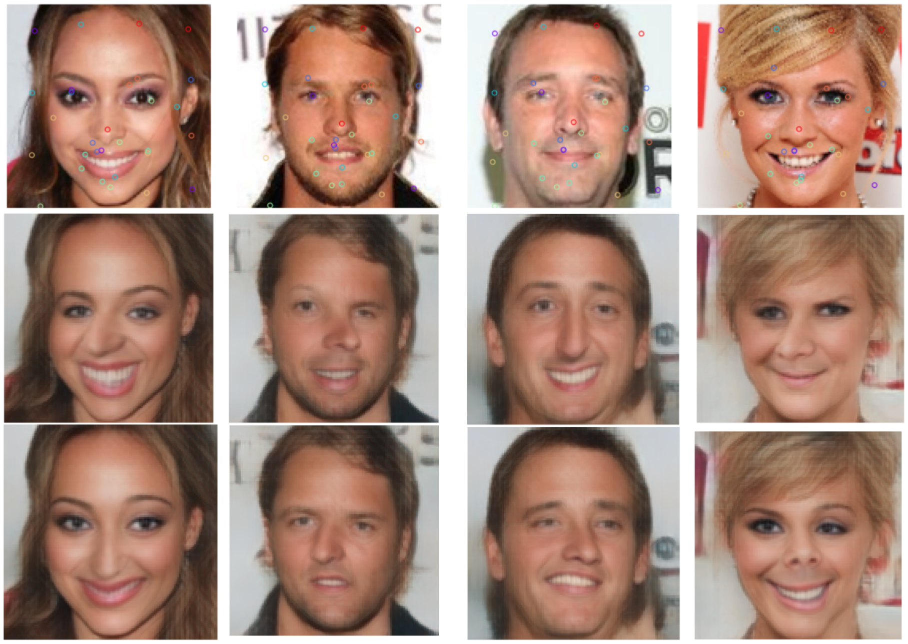

Figure 7. Image manipulation comparison with KeyNet (Jakab et al., 2018a). We visualize the effect of moving keypoints in the face area.

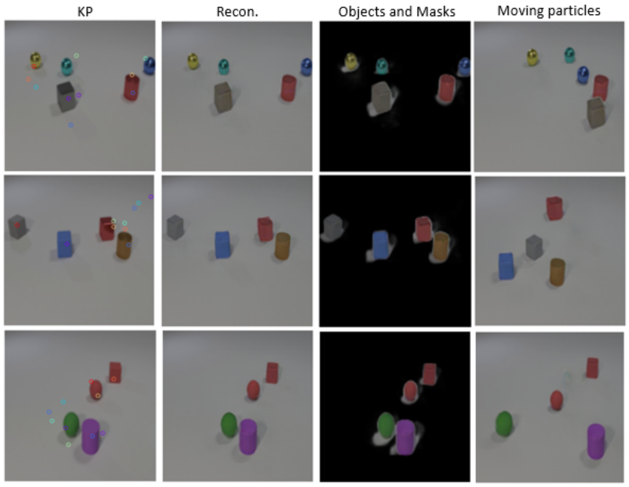

Figure 8. Scene decomposition and manipulation with `Object` model on CLEVRER. We show the detected particles, the reconstructed images, the objects and masks – output of the glimpse decoder, and image manipulation based on moving the particles.

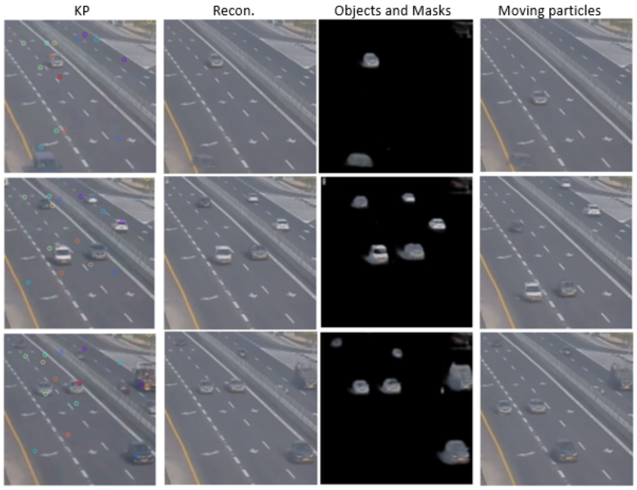

Figure 9. Scene decomposition and manipulation with `Object` model on Traffic. We show the detected particles, the reconstructed images, the objects and masks – output of the glimpse decoder, and image manipulation based on moving the particles.

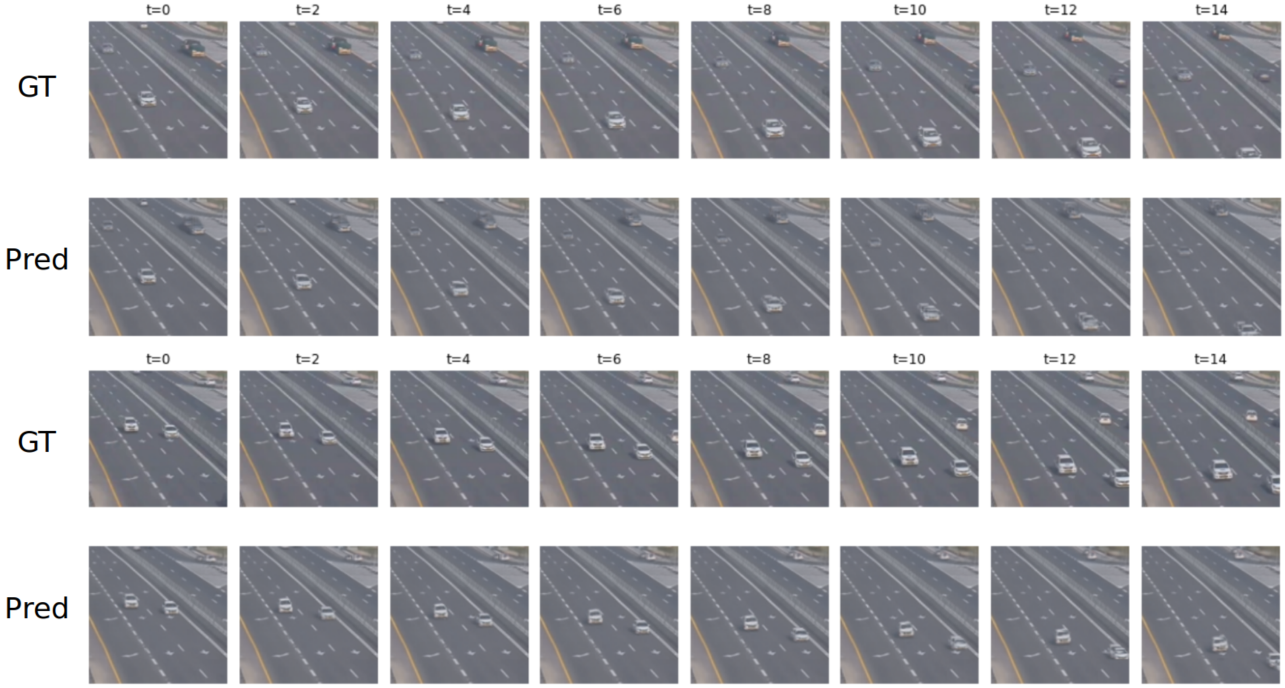

Figure 10. Video prediction on the Traffic dataset. We use the pre-trained `Object` model to provide the particle-representation and a GNN to predict the temporal change in particles. Top - ground truth, bottom - prediction.

pre-trained DLP model. We note that the pre-trained DLP model stays frozen throughout the training process of the GCN. As DLP is trained per-image, particles in consecutive frames do not necessarily match, and the GCN is trained to minimize the perceptual loss (Hoshen et al., 2019) with the ground truth future frame. Video predictions are generated by rolling out the GCN, starting from the particles of the first two images in the video, and using the decoder to generate images from predicted particles. The GCN layes are implemented efficiently with the `torch geometric` library (Fey & Lenssen, 2019).

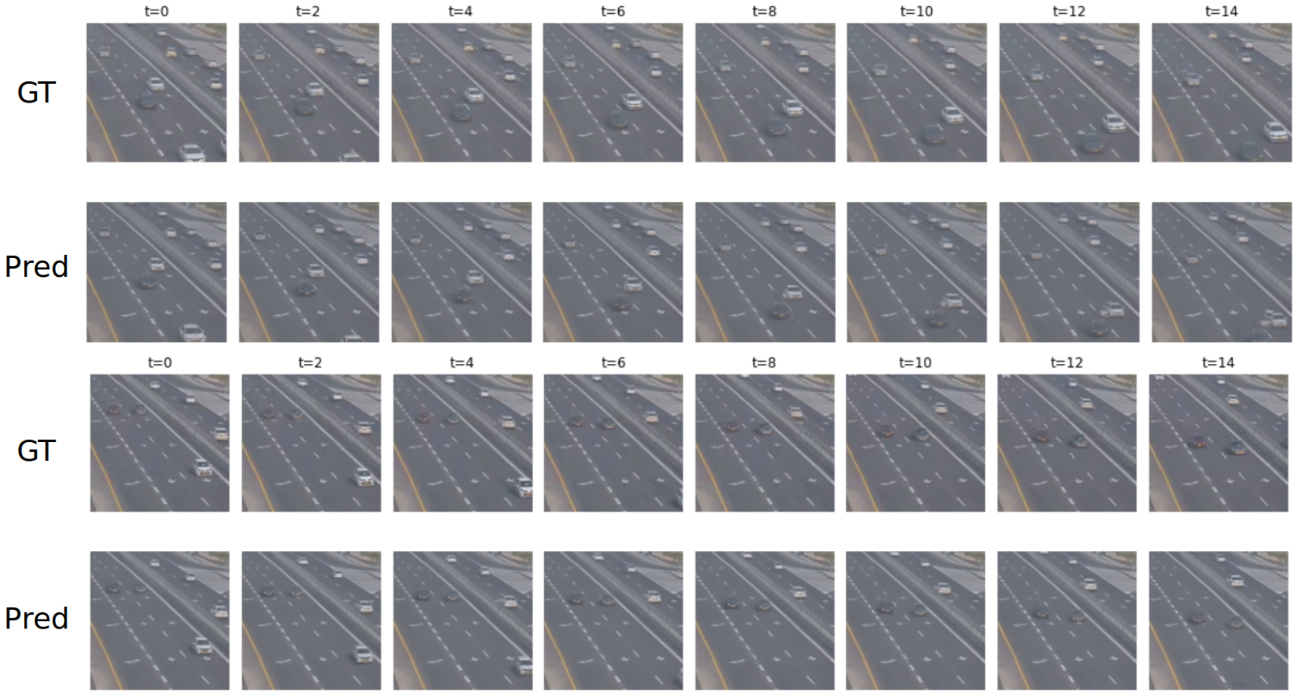

Figure 11. Video prediction on the Traffic dataset. We use the pre-trained `Object` model to provide the particle-representation and a GNN to predict the temporal change in particles. Top - ground truth, bottom - prediction.

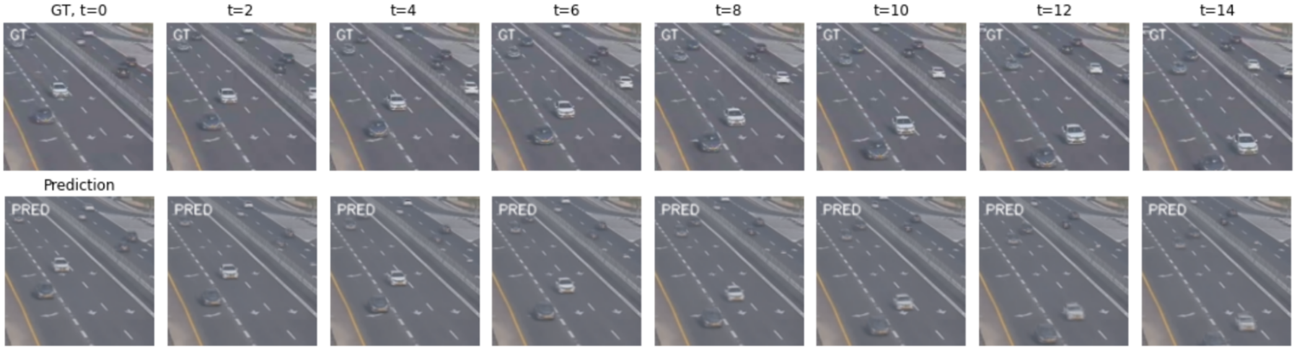

Figure 12. Video prediction on the Traffic dataset. We use the pre-trained `Object` model to provide the particle-representation and a GNN to predict the temporal change in particles. Top - ground truth, bottom - prediction.

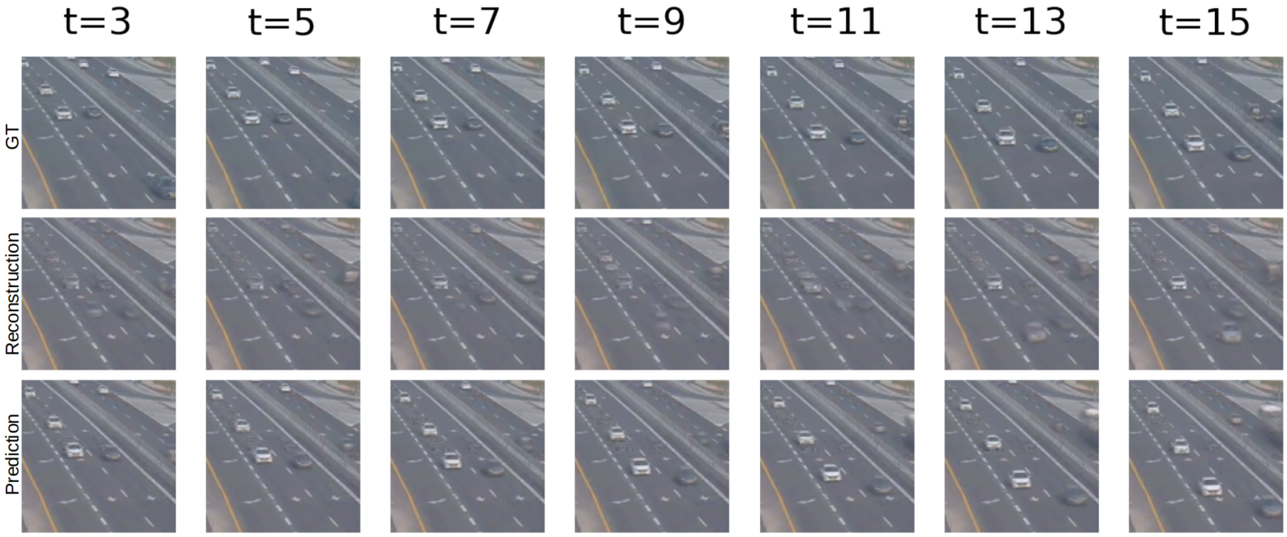

Figure 13. Video prediction baseline (Minderer et al.) on Traffic. Second row: reconstructions of KeyNet; third row: dynamics prediction by Minderer et al. (2019b)