Spatially Invariant Unsupervised 3D Object Segmentation with Graph Neural Networks

Tianyu Wang  
School of Computer Science  
The Australia National University  
tianyu.wang2@anu.edu.au

Kee Siong Ng  
School of Computer Science  
The Australia National University  
keesiong.ng@anu.edu.au

Miaomiao Liu  
School of Engineering  
The Australian National University  
miaomiao.liu@anu.edu.au

Abstract

In this paper, we tackle the problem of unsupervised 3D object segmentation from a point cloud without RGB information. In particular, we propose a framework, SPAIR3D, to model a point cloud as a spatial mixture model and jointly learn the multiple-object representation and segmentation in 3D via Variational Autoencoders (VAE). Inspired by SPAIR, we adopt an object-specification scheme that describes each object's location relative to its local voxel grid cell rather than the point cloud as a whole. To model the spatial mixture model on point clouds, we derive the Chamfer Likelihood, which fits naturally into the variational training pipeline. We further design a new spatially invariant graph neural network to generate a varying number of 3D points as a decoder within our VAE. Experimental results demonstrate that SPAIR3D is capable of detecting and segmenting variable number of objects without appearance information across diverse scenes.

1 Introduction

Reinforcement learning agents that need to deal with 3D scenes often have to grapple with enormous observation spaces that result from the combinatorial interactions between objects and scene layouts. This phenomenon renders naïve representations all but infeasible except on the simplest of 3D scenes. Motivated in part by cognitive psychology studies [1] that suggest human brains organize observations at an object level, recent advances in reinforcement learning research [2, 3] and physical prediction [4] have demonstrated superior robustness with environments modeled in an object-oriented manner. To exploit these techniques for 3D scenes, reinforcement learning agents need fast and robust ways to segment objects from 3D observations to achieve an object-centric representation of the scene. This is the high-level research problem that motivates the algorithm presented in this paper.

There is a good body of existing literature on supervised object segmentation techniques for 2D [5, 6] and 3D [7, 8] scenes. To get around the downside of having to rely heavily on human annotations, unsupervised object-segmentation methods from RGB images have received an increasing amount of attention [9, 10, 11, 12] in recent times. In particular, generative models based on Variational Autoencoders (VAE) [13] have been employed to model the pixel intensities of an image with spatial Gaussian mixture models [9, 11, 14, 15]. The encoder-decoder structure of a VAE forms an information bottleneck by compressing regions of highly correlated appearance, and the notion of objectness (i.e. the property of being an object) is then identified with image regions that exhibit strong appearance correlation. Unfortunately, extending existing image-based generative models...
to achieve unsupervised 3D point cloud segmentation is not straightforward. It is known that 2D image-based VAEs mainly rely on coordinate-dependent intensity modeling and reconstruction [16]. In contrast, finding correspondences between input point clouds and predicted ones for 3D point cloud-based generative models is far from trivial.

Existing works have shown that learning object-centric representations is crucial for unsupervised 2D object segmentation [9,10,14]. SPAIR [14], in particular, makes use of an object-specification scheme which allows the method to scale well to scenes with a large number of objects. Inspired by SPAIR [14] and existing works on 3D point cloud object generation requiring centred-points [17], we propose in this paper a VAE-based model called Spatially Invariant Attend, Infer, Repeat in 3D (SPAIR3D), a model that generates spatial mixture distributions on point clouds to discover and segment 3D objects from a point cloud of static scenes.

Here, in summary, are the key contributions of this paper:

- We propose, to the best of our knowledge, the first generative-model-based unsupervised point cloud segmentation pipeline, named SPAIR3D, capable of performing unsupervised 3D point cloud segmentation on static scenes.
- We also propose a new Chamfer Likelihood function tailored for learning mixture probability distributions over point cloud data with a novel graph neural network called Point Graph Flow network (PGF) that can be used to model and generate a variable number of 3D points for segmentation across diverse scenes.
- We provide qualitative and quantitative results that show SPAIR3D can perform object segmentation on point clouds in an unsupervised manner using only spatial structure information across diverse scenes with an arbitrary number of objects.

2 Related Work

Unsupervised Generative Model-based 2D Object Segmentation. Unsupervised object segmentation approaches have attracted increasing attention recently. A major focus of these methods is on joint object representation-learning and segmentation from images and videos via generative models [9,10,11,14,18]. In particular, spatial Gaussian mixture models are commonly adopted to model pixel colors, and the unsupervised object segmentation problem is then framed as a generative latent-variable modelling task. For example, IODINE [9] employs amortized inference that iteratively updates the latent representation of each object and refines the reconstruction. GENESIS [18] and MONET [10] sequentially decode each object, while Neural Expectation Maximization (NEM) [19] realizes Expectation Maximization algorithms with RNNs. Such iterative processes allow IODINE and NEM to handle dynamic scenes where viewpoints or objects may change slowly.

Instead of treating each component of the mixture model as a full-scale observation in images, Attend, Infer, Repeat (AIR) [15] confines the extent of each object to a local region. To improve the scalability of AIR, SPAIR [14] employs a grid spatial attention mechanism to propose objects locally, which has proven effective in object-tracking tasks [20]. To achieve a complete scene segmentation, SPACE [11] includes MONET in its framework for background modeling.

While the works discussed above achieved promising results for object representation and segmentation from 2D images, they cannot be directly applied to handle 3D point cloud due to the lack of input-prediction correspondences. Spatial attention model is employed to reconstruct 3D scenes in the form of meshes or voxel in an object-centric fashion from a sequence of RGB frames [21]. This method similarly relies heavily on appearance and motion cues. In contrast, our work aims to segment 3D objects from point clouds of static scenes. Inspired by the aforementioned works employing VAE [15,22] structure to define the objectness by compressing image regions of high correlations, we explore in this paper correlations beyond appearance using point cloud with VAE.

Graph Neural Network for Point Cloud Generation. Generative models such as VAEs [23] and generative adversarial networks (GANs) [24] have been successfully used for point-cloud generation. However, these generative models are constrained to generate a pre-defined fixed number of points, limiting their applications. Yang et.al [25] proposed a VAE and normalizing flow-based approach that models object shapes as continuous distributions. While the proposed approach allows the generation of a variable number of points, it could not be integrated into our framework naturally due to its requirement of solving an ODE. We therefore propose to build a graph neural network specialised for our VAE-based framework as a decoder to generate an arbitrary number of points at run time.
3 SPAIR3D

SPAIR3D is a VAE-based generative model for 3D object segmentation via object-centric point-cloud generation. It takes a point cloud as input and generates a structured latent representation for foreground objects and scene layout. In the following, we first describe latent representation learning. We then leverage variational inference to jointly learn the generative model (§3.2) and inference model (§3.4). We also discuss the particular challenges arising for generative models in handling a varying number of points with a novel Chamfer Likelihood (§3.3) and Point Graph Flow (§3.4).

3.1 Object-centric Latent Representation

As shown in Fig. 1a, SPAIR3D first decomposes the 3D scene uniformly into a voxel grid named spatial attention voxel grid. Due to the irregular distribution of the point cloud in 3D, there can be empty voxel cells with no point. After discarding those empty cells, we associate an object proposal with each non-empty spatial attention voxel cell, specified in the form of a bounding box. The set of input points captured by a bounding box is termed an object glimpse. Besides object glimpses, SPAIR3D also defines a scene glimpse covering every point in an input scene. Later, we show that we encode and decode (reconstruct) points covered by each glimpse and generate a mixing weight for each point to form a probability mixture model over the scene. The mixture weights for each point naturally defines the segmentation of the point cloud.

3.2 Generative Model

Similar to SPAIR, each grid cell generates posterior distributions over a set of latent variables defined as $z_{cell}^{i} = \{z_{where}^{i}, z_{apothem}^{i}\}$, where $z_{where}^{i} \in \mathbb{R}^3$ encodes the relative position of the center of the $i$th object proposal to the center of the $i$th cell, $z_{apothem}^{i} \in \mathbb{R}^3$ encodes the apothem of the bounding box. Thus, each $z_{cell}^{i}$ induces one object glimpse associated with the $i$th cell. Each object proposal is then associated with posterior distributions over a set of latent variables specified as $z_{object}^{i} = \{z_{what}^{i}, z_{mask}^{i}, z_{pres}^{i}\}$, where $z_{what}^{i} \in \mathbb{R}^A$ encodes the structure information of the corresponding object glimpse, $z_{mask}^{i} \in \mathbb{R}^B$ encodes the mask for each point in the glimpse, and $z_{pres}^{i} \in \{0, 1\}$ is a binary variable indicating whether the proposed object should exist ($z_{pres}^{i} = 1$) or not ($z_{pres}^{i} = 0$).

Different from object glimpse, scene glimpse is defined by only one latent variable $z_{scene} = \{z_{what}^{0}\}$. The mixing weight of the scene glimpse as well as those of object glimpses completes the mixture distribution of the point cloud. We assume $z_{pres}^{i}$ follows a Bernoulli distribution. The posteriors and priors of other latent variables are all set to isotropic Gaussian distributions (see the supplementary material Sec. A for details).
Given latent representations of objects and the scene, the complete likelihood for a point cloud $X$ is formulated as $p(X) = \int p(x)p(X|z)dz$, where $z = (\bigcup_i z_i^{\text{cell}}) \cup (\bigcup_i z_i^{\text{object}}) \cup z_{\text{scene}}$. $p(X|z)$ is the Chamfer Likelihood defined below. As maximising the objective $p(X)$ is intractable, we resort to variational inference method to maximise its evidence lower bound (ELBO).

### 3.3 Chamfer Likelihood

To maximize the ELBO, we maximize the reconstruction accuracy of the point cloud. Unlike generative model-based unsupervised 2D segmentation methods that reconstruct the pixel-wise appearance conditioning on its spatial coordinate, the reconstruction of a point cloud lost its point-wise correspondence to the original point cloud. To measure the reconstruction quality, Chamfer distance is commonly adopted to measure the discrepancy between the generated point cloud ($\hat{X}$) and the input point cloud ($X$). Formally, Chamfer distance is defined by $d_{CD}(X, \hat{X}) = \sum_{x \in X} \min_{\hat{x} \in \hat{X}} \|x - \hat{x}\|^2_2 + \sum_{\hat{x} \in \hat{X}} \min_{x \in X} \|x - \hat{x}\|^2_2$. We refer to the first and the second term on the r.h.s as forward loss and backward loss respectively.

Unfortunately, the Chamfer distance does not fit into the variational-inference framework. To get around that, we propose a Chamfer Likelihood function tailored for training probability mixture models defined on point clouds. The Chamfer Likelihood has a forward likelihood and a backward likelihood corresponding to the forward and backward loss respectively, which we describe next.

Denote the $i^{\text{th}}$ glimpse as $G_i$, $i \in \{0, \ldots, n\}$ and its reconstruction as $\hat{G}_i$, $i \in \{0, \ldots, n\}$. Specifically, we treat the scene glimpse as the $0^{\text{th}}$ glimpse that contains all input points, that is, $G_0 = X$. Note that one input point can be a member of multiple glimpses. Below we use $N(\mu, \sigma)(x)$ to denote the probability density value of point $x$ evaluated at a Gaussian distribution of mean $\mu$ and variance $\sigma$. For each input point $x$ in the $i^{\text{th}}$ glimpse, the glimpse-wise forward likelihood of that point is defined as $L_i^F(x) = \frac{1}{\pi \cdot \sigma_i^2} \exp\left(-\frac{1}{2} \frac{(x - \mu_i)^2}{\sigma_i^2}\right)$, where $\mu_i = \int_{x \in X} \max_{\hat{x} \in \hat{G}_i} N(\hat{x}, \sigma_c)(x)dx$ is the normalizer and $\sigma_i$ is a hyperparameter. For each glimpse $G_i$, $i \in \{0, \ldots, n\}$, $\alpha_i^F \in [0, 1]$ defines a mixing weight for point $x$ in the glimpse and $\sum_{i=0}^n \alpha_i^F = 1$. In particular, $\alpha_i^F = 1$ is the mask decoder $\theta(\hat{z})$ and $\alpha_i^F = 0$ otherwise. Here, $f_\theta$ is the mask decoder network (see §3.4), and $T(\cdot)$ is the Sigmoid function. The mixing weight for the scene layout points complements the distribution through $G_0 = X$. Thus, the final mixture model for an input point $x$ is $L^F(x) = \sum_{i=0}^n \alpha_i^F L_i^F(x)$. The total forward likelihood of $X$ is then defined as $L^F(X) = \prod_{x \in X} L^F(x)$. For each predicted point $\hat{x}$, the point-wise backward likelihood is defined as $L_i^B(\hat{x}) = \max_{\hat{x} \in \hat{G}_i} N(x, \sigma_c)(\hat{x})$, where $i(\hat{x})$ returns the glimpse index of $\hat{x}$. We denote $x(\hat{x}) = \arg \max_{\hat{x} \in \hat{G}_i} N(x, \sigma_c)(\hat{x})$ and $X = \bigcup_{i=0}^n \hat{G}_i$. The backward likelihood is then defined as $L^B(X) = \prod_{i=0}^n \prod_{\hat{x} \in \hat{G}_i} L_i^B(\hat{x})^{\alpha_i^F \cdot \pi_i^F}$. The exponential weighting, i.e. $\alpha_i^F \cdot \pi_i^F \in [0, 1]$, is crucial. As each predicted point $\hat{x}$ belongs to one and only one glimpse, it is thus difficult to impose a mixture model interpretation on the backward likelihood. The exponential weighting encourages the generated points in object glimpse to be close to input points of high probability belonging to $G_i$. The backward likelihood provides the necessary regularization to enforce a reconstruction of high quality. Combining the forward and backward likelihood together, we define Chamfer Likelihood as $L_{CD}(X, \hat{X}) = L^F(X) \cdot L^B(\hat{X})$. After inference, the segmentation label for each point $x$ is naturally obtained by arg $\max_{\hat{x} \in \hat{G}_i} L^B(\hat{x})^{\alpha_i^F \cdot \pi_i^F}$.

The evidence lower bound is $L = L_{CD}(X, \hat{X}) - L_{KL}(z^{\text{cell}}, z^{\text{object}}, z^{\text{scene}})$, where $L_{KL}$ is the KL divergence between the prior and posterior of the latent variables (supp. material Sec. A for details).

In general, the normalizer of glimpse-wise forward likelihood $u_i$ doesn’t have a closed-form solution. For a glimpse $i$ with $n_i$ points, we have $1 \leq u_i \leq n_i$. Thus, instead of optimising $L_i^F(x)$, we optimise the lower bound $\frac{1}{n_i} \max_{\hat{x} \in \hat{G}_i} N(\hat{x}, \sigma_c)(x)$. To improve the stability of early training process where $n_i$ exhibits high variance due to inaccurate object proposals, we absorb the $\frac{1}{n_i}$ term into $\alpha_i^F$ and compute $\max_{\hat{x} \in \hat{G}_i} N(\hat{x}, \sigma_c)(x)$ instead.

### 3.4 Model Structure

We next introduce the encoder and decoder network structure for SPAIR3D. The building blocks are based on graph neural networks and point convolution operations (See supp. Sec. C for details).
Encoder network. We design an encoder network \( q_\theta(z|x) \) to obtain the latent representations \( \{z_i^{\text{cell}}\}_{i=1}^n \) and \( \{z_i^{\text{object}}\}_{i=1}^n \) from a point cloud, where \( \{z_i^{\text{cell}}\}_{i=1}^n \) encode information from points in grid cells and \( \{z_i^{\text{object}}\}_{i=1}^n \) encode information for points from object glimpses. To achieve the spatially invariant property, we group one PointConv \([26]\) layer and one PointGNN \([27]\) layer into pairs for message passing and information aggregation among points and between cells. We now provide details on how we use the encoder for voxel grids and glimpses to learn latent representations.

(a) Voxel Grid Encoding. The voxel-grid encoder takes a point cloud as input and generates for each spatial attention voxel cell \( C_i \), two latent variables \( z_i^{\text{where}} \in \mathbb{R}^3 \) and \( z_i^{\text{apothem}} \in \mathbb{R}^3 \) to propose a glimpse \( G_i \) potentially occupied by an object.

To better capture the point cloud information in \( C_i \), we build a voxel pyramid within each cell \( C_i \) with the bottom level corresponding to the finest voxel grid. We aggregate information hierarchically using PointConv-PointGNN pairs from bottom to top through each level of the pyramid for each cell \( C_i \). For each layer of the pyramid, we aggregate the features of all points and assign it to the centre of mass of points within a voxel cell. Then PointGNN is employed to perform message passing on the radius graph built on all new points. The output of the final aggregation block produces \( z_i^{\text{where}} \) and \( z_i^{\text{apothem}} \) via the re-parametrization trick \([13]\).

We obtain the offset distance of a glimpse center from its corresponding grid cell center using \( \Delta g_i = \tanh(z_i^{\text{where}}) \cdot L \), where \( L \) is the maximum offset distance. The apothems of the glimpse in the \( x, y, z \) direction is given by \( \Delta g_i^{\text{apo}} = T(z_i^{\text{apothem}})(r_{\max} - r_{\min}) + r_{\min} \), where \( T(\cdot) \) is the sigmoid function and \([r_{\min}, r_{\max}]\) defines the range of apothem.

(b) Glimpse Encoding. Given the predicted glimpse-centre offset and the apothems, we can associate one glimpse with each spatial attention voxel cell. We adopt the same encoder structure to encode each glimpse \( G_i \) into one point \( a_i = (c_i, f_i) \), where \( c_i \) is the glimpse center coordinate and \( f_i \) is the glimpse feature vector. We then generate \( z_i^{\text{what}} \) and \( z_i^{\text{mask}} \) from \( a_i \) via the re-parametrization trick.

The generation of \( z_i^{\text{pres}} \) determines the glimpse rejection process and is crucial to the final segmentation quality. Unlike previous work \([11][13]\), SPAIR3D generates \( z_i^{\text{pres}} \) from glimpse features instead of cell features based on our observation that message passing across glimpses provides more benefits in the glimpse-rejection process. To this end, a radius graph is first built on the point set \( \{c_i, f_i\}_{i=1}^n \) to connect nearby glimpse centers, which is followed by multiple PointGNN layers with decreasing output channels to perform local message passing. The \( z_i^{\text{pres}} \) of each glimpse is then obtained via the re-parametrization trick. Information exchange between nearby glimpses can help avoid over-segmentation that would otherwise occur because of the high dimensionality of point cloud data.

(c) Global Encoding. The global encoding module adopts the same encoder as that for the glimpse encoding to encode all points in the scene, which is treated as a single glimpse \( G_0 \). The learned latent representation is \( z_0^{\text{what}} \) with \( z_0^{\text{pres}} = 1 \).

Decoder network. We now introduce the decoders that are used for point-cloud reconstruction and assignment of mask value to each input point.

(a) Point Graph Flow. Given the \( z_i^{\text{object}} \) of each glimpse, the decoder is used for point-cloud reconstruction as well as segmentation-mask generation. Most existing decoder or point-cloud generation framework can only generate a pre-defined fixed number points, and this can lead to under- or over-segmentation because the number of generated points has a direct effect on the relative magnitudes between the forward and backward terms in the Chamfer likelihood.

To balance the forward and backward likelihood, the number of predictions for each glimpse must be approximately the same as the number of input points. Inspired by PointFlow \([25]\), which allows the sampling of an arbitrary number of points forming a point cloud within a normalizing-flow framework, we propose a Point Graph Flow (PGF) network that allows the generation of a variable number of points at run time. The input to the PGF is a set of 3D points with coordinates sampled from a zero-centered Gaussian distribution, with the population determined by the number of points in the current glimpse. Features of the input points are set uniformly to the latent variable \( z_i^{\text{what}} \). PGF is composed of several PointGNN layers, each of which is preceded by a radius graph operation. The output of each PointGNN layer is of dimension \( f + 3 \), with the first \( f \) dimensions interpreted as the updated features and the last 3 dimensions interpreted as the updated 3D coordinates for estimated
points. Since we only focus on point coordinates prediction, we set \( f = 0 \) for the last PointGNN layer. Unlike PointFlow, PGF has a simple structure and does not require an external ODE solver.

**(b) Mask Decoder.** The Mask Decoder decodes \((c_i, z^\text{mask}_i)\) to the mask value, \(\pi^\text{c}_i \in [0, 1]\), of each point within a glimpse \(G_i\). The decoding process follows the exact inverse pyramid structure of the Glimpse Encoder. To be more precise, the mask decoder can access the spatial coordinates of the intermediate aggregation points of the Glimpse Encoder as well as the point coordinates of \(G_i\). During decoding, PointConv is used as deconvolution operation.

**Glimpse VAE and Global VAE.** The complete Glimpse VAE structure is presented in Fig. [1b]. The Glimpse VAE is composed of a Glimpse Encoder, Point Graph Flow, Mask Decoder and a multi-layer PointGNN network. The Glimpse Encoder takes all glimpses as input and encodes each glimpse \(G_i\) individually and in parallel into feature points \((c_i, f_i)\). Via the re-parameterization trick, \(z_i^\text{what}\) and \(z_i^\text{mask}\) are then obtained from \(f_i\). From there, we use the Point Graph Flow to decode \(z_i^\text{what}\) to reconstruct the input points, and we use the Mask Decoder to decode \(z_i^\text{mask}\) to assign a mask value for each input point within \(G_i\). Finally, \(x_i^\text{prex}\) is generated using message passing among neighbouring glimpses. The processing of all glimpses happens in parallel. The Global VAE consisting of the Global Encoder and a PGF-based decoder outputs the reconstructed scene layout.

### 3.5 Soft Boundary

The prior of \(z^\text{apothem}\) is set to encourage apothem to shrink so that the size of the glimpses will not be overly large. However, if points are excluded from one glimpse, the gradient from the likelihood of the excluded points will not influence the size and location of the glimpse anymore, and this can lead to over-segmentation. To solve this problem, we introduce a soft boundary weight \(b^\text{apothem}_i \in [0, 1]\) which decreases when a point \(x \in G_i\) moves away from the bounding box of \(G_i\). Taking \(b^\text{apothem}_i\) into the computation of \(\alpha\), we obtain an updated mixing weight \(\alpha^\text{c}_i = \sum_{j \in G_i} \frac{z^\text{what}_i z^\text{mask}_i}{z^\text{what}_i z^\text{mask}_i + x^\text{prex}_i b^\text{apothem}_i} \pi^\text{c}_i b^\text{apothem}_i\). By employing such a boundary loss, the gradual exclusion of points from glimpses will be reflected in gradients to counter over-segmentation. Details can be found in supp. Sec. B.

### 4 Experiments

To evaluate the performance of SPAIR3D, we first introduce two new point-cloud datasets *Unity Object Room* and *Unity Object Table* built on the *Unity* platform [28].

#### 4.1 Datasets

The Unity Object Room (UOR) dataset is built as a benchmark dataset for the evaluation of unsupervised 3D object-segmentation models. Specifically, each scene in the UOR dataset consists of multiple objects sitting on a square floor. Object are randomly sampled from a list of 8 regular geometries. The Unity Object Table (UOT) dataset approximates Robotic Object Grasping scenario where multiple objects are placed on a round table. Instead of using objects of simple geometries, we create each scene of the UOT dataset with objects selected from a pool of 9 irregular objects such as a toy car or a teddy bear. For both datasets, the number of objects placed in each scene varies from 2 to 5 with equal probabilities. During the scene generation, the size and orientation of the objects are varied randomly within a pre-defined range. We also randomly assign each object mesh a color.

We capture the depth, RGB, normal frames, and pixel-wise semantics as well as instance labels for each scene from 10 different viewpoints for both datasets. This setup aims to approximate the scenario where a robot equipped with depth and RGB sensors navigates around target objects and captures data. The point cloud data for each scene is then constructed by merging these 10 depth maps. For each dataset, we collect 50\(K\) training scenes, 10\(K\) validation scenes and 5\(K\) testing scenes. In-depth dataset specification and analysis can be found in supplementary D.

#### 4.2 Unsupervised Segmentation

**Baseline.** Due to the sparse literature on unsupervised 3D point cloud segmentation, we could not find a generative baseline to compare with. Thus, we compare SPAIR3D with PointGroup [8], a recent supervised 3D point cloud segmentation model. PointGroup performs semantic prediction and instance predictions from a point cloud and RGB data using a model trained with ground-truth semantic labels and instance labels. To ensure a fair comparison, we assign each point the same color (white) so appearance information doesn’t play a role. The PointGroup network is fine-tuned on the validation set to achieve the best performance.
Figure 2: Visualization of segmentation results on UOR and UOT dataset. First 3 columns of Fig.2i and Fig.2j give close-up visualization of masks (top row) and reconstruction (bottom row) of glimpses selected in the red box in Fig.2a and Fig.2c. The 4th column shows the foreground alpha of the scenes (top) and the reconstruction produced by the global V AE (bottom).

Table 1: 3D point cloud segmentation results on UOR dataset (blue) and UOT dataset (red). PG stands for the supervised baseline PointGroup [8]. The last two columns show the forward (CD F) and backward (CD B) Chamfer distance between the input point cloud and the reconstruction.

Performance Metric. We use the Adjust Rand Index (ARI) [29] to measure the segmentation performance against the ground truth instance labels. We also employ foreground Segmentation Covering (SC) and foreground unweighted mean Segmentation Covering (mSC) [18] for performance measurements as ARI does not penalize models for object over-segmentation [18].

Evaluation. Table 1 shows that SPAIR3D achieves comparable performance with the baseline, which is a supervised method on both UOT and UOR datasets. As demonstrated in Fig. 2, each foreground object is proposed by one and only one glimpse. The scene layout is separated from objects and accurately modelled by the global VAE. It is worth noting that the segmentation errors mainly happen at the bottom of objects. Without appearance information, points at the bottom of objects can also be correlated to the ground. Since the prior of \( z_{cell} \) encourages glimpses to shrink, those points are likely segmented as part of the scene layout instead of objects.

In Fig. 3, we show the performance distributions on the test set. To draw the distribution curve, we sort the test data in ascending order of performance. As expected, the supervised baseline (Orange) performs better but SPAIR3D manages to achieve high-quality segmentation (SC score > 0.8) on around 80% of the scenes without any supervision.

More segmentation results including failure cases can be found in the supp. E.

4.3 Voxel Size Robustness and Scalability

In the literature [11,14], the cell voxel size \( l \), an important hyperparameter, is chosen to match the object size in the scene. To evaluate the robustness of our method w.r.t voxel size, we train our model
Figure 3: Test set performance distributions on UOR (first row) and UOT (second row).

Figure 4: Segmentation on scenes with 6 to 12 objects (a - d) and on object matrix (e-h)
the UOR dataset achieves ARI: 0.841, SC: 0.610, and mSC: 0.627, which is significantly worse than the full SPAIR3D model. The performance distribution of ablated SPAIR3D (Fig 5, first row) indicates that removing the multi-layer PointGNN has a broad negative influence on the entire dataset. Fig. 5 shows that the multi-layer PointGNN is crucial to preventing over-segmentation.

4.5 Empirical Evaluation of PGF

3D objects of the same category can be modeled by a varying number of points. The generation quality of the point cloud largely depends on the robustness of our model against the number of points representing each object. To demonstrate that PGF can reconstruct each object with a dynamic number of points, we train the global VAE on the ShapeNet dataset [17], where each object is composed of roughly 2000 points, and reconstruct the object with a varying number of points. To this end, for reference input point clouds of size \(N\), we force PGF to reconstruct a point cloud of size \(1.5N, 1.25N, N, 0.75N, \) and \(0.5N\) respectively. The reconstruction results are shown in Fig. 6. While the reconstructions have less details than the input point cloud, the PGF can reconstruct point clouds that capture the overall object structure in all 5 different settings.

5 Conclusion and Future Work

We propose, to the best of our knowledge, the first generative model for unsupervised 3D point-cloud object segmentation called SPAIR3D, which models a scene with a 3D spatial mixture model that exploits spatial structure correlations within each object. Our SPAIR3D model achieves the unsupervised point cloud segmentation by generation. Our model is evaluated on two newly proposed UOR and UOT datasets. We further demonstrate that SPAIR3D can generalise well to previously unseen scenes with a large number of objects without performance degeneration. Our model may tend to over-segment objects of a significant different scale from those in the training dataset, which is left as our future work. Moreover, the spatial mixture interpretation of SPAIR3D allows the integration of memory mechanism [30] or iterative refinement [9] to handle objects segmentation from video sequences, which are key directions for our future work.

References

[1] Daniel Kahneman, Anne Treisman, and Brian J Gibbs. The reviewing of object files: Object-specific integration of information. Cognitive Psychology, 24(2):175 – 219, 1992. ISSN 0010-0285. doi: https://doi.org/10.1016/0010-0285(92)90007-O. URL http://www.sciencedirect.com/science/article/pii/0010028592900070.

[2] Carlos Diuk, Andre Cohen, and Michael L. Littman. An object-oriented representation for efficient reinforcement learning. In ICML, ICML ’08, page 240–247, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 9781605582054. doi: 10.1145/1390156.1390187. URL https://doi.org/10.1145/1390156.1390187.

[3] Ken Kansky, Tom Silver, David A. Mély, Mohamed Eldawy, Miguel Lázaro-Gredilla, Xinghua Lou, Nimrod Dorfman, Szymon Sidor, D. Scott Phoenix, and Dileep George. Schema networks:
Zero-shot transfer with a generative causal model of intuitive physics. *ArXiv*, abs/1706.04317, 2017.

[4] B. Michael Chang, Tomer Ullman, Antonio Torralba, and B. Joshua Tenenbaum. A compositional object-based approach to learning physical dynamics. *ICLR*, 2017.

[5] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *ICCV*, pages 2980–2988, 2017.

[6] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, pages 779–788, June 2016. doi: 10.1109/CVPR.2016.91.

[7] Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh r-cnn. In *ICCV*, October 2019.

[8] Li Jiang, Hengshuang Zhao, Shaoshuai Shi, Shu Liu, Chi-Wing Fu, and Jiaya Jia. Pointgroup: Dual-set point grouping for 3d instance segmentation. *CVPR*, 2020.

[9] Klaus Greff, Raphaël Lopez Kaufman, Rishabh Kabra, Nick Watters, Christopher Burgess, Daniel Zoran, Loïc Matthey, Matthew M Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. In *ICML*, 2019.

[10] Christopher Burgess, Loic Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matt Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation. *ArXiv*, abs/1901.11390, 01 2019. URL https://arxiv.org/abs/1901.11390.

[11] 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. In *ICLR*, 2020. URL https://openreview.net/forum?id=rk103ySYDH.

[12] Daniel Bear, Chaofei Fan, Damian Mrowca, Yunzhu Li, Seth Alter, Aran Nayebi, Jeremy Schwartz, Li F Fei-Fei, Jiajun Wu, Josh Tenenbaum, et al. Learning physical graph representations from visual scenes. Advances in *Neural Information Processing Systems*, 33, 2020.

[13] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In 2nd *International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[14] Eric Crawford and Joelle Pineau. Spatially invariant unsupervised object detection with convolutional neural networks. *AAAI*, 33:3412–3420, 07 2019. doi: 10.1609/aaai.v33i01.33013412.

[15] S. M. Ali Eslami, Nicolas Heess, Theophane Weber, Yuval Tassa, David Szepesvari, Koray Kavukcuoglu, and Geoffrey E. Hinton. Attend, infer, repeat: Fast scene understanding with generative models. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS’16, page 3233–3241, Red Hook, NY, USA, 2016. Curran Associates Inc. ISBN 9781510838819.

[16] Nicholas Watters, Loic Matthey, Christopher Burgess, and Alexander Lerchner. Spatial broadcast decoder: A simple architecture for learning disentangled representations in vaes. *ArXiv*, abs/1901.07017, 08 2019. URL https://arxiv.org/abs/1901.07017.

[17] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. *ShapeNet: An Information-Rich 3D Model Repository*. Technical Report arXiv:1512.03012 [cs,GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[18] Martin Engelcke, Adam R. Kosiorek, Oiwi Parker Jones, and Ingmar Posner. Genesis: Generative scene inference and sampling with object-centric latent representations. In *ICLR*, 2020. URL https://openreview.net/forum?id=BlxXaITVfW.

[19] Klaus Greff, Sjoerd van Steenkiste, and Jürgen Schmidhuber. Neural expectation maximization. In *NeurIPS, NIPS’17*, page 6694–6704, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.
[20] Eric Crawford and Joelle Pineau. Exploiting spatial invariance for scalable unsupervised object tracking. *AAAI*, 34:3684–3692, 04 2020. doi: 10.1609/aaai.v34i04.5777.

[21] Paul Henderson and Christoph H. Lampert. Unsupervised object-centric video generation and decomposition in 3D. In *NeurIPS*, 2020.

[22] Irina Higgins, Loïc Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew M Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2017.

[23] Matheus Gadelha, Rui Wang, and Subhransu Maji. Multi-resolution tree networks for 3D point cloud processing. In *ECCV*, pages 103–118, 2018.

[24] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3D point clouds. In *ICML*, pages 40–49. PMLR, 2018.

[25] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. Pointflow: 3D point cloud generation with continuous normalizing flows. In *ICCV*, pages 4540–4549, 10 2019. doi: 10.1109/ICCV.2019.00464.

[26] Wenxuan Wu, Zhongang Qi, and Fuxin Li. Pointconv: Deep convolutional networks on 3D point clouds. In *CVPR*, pages 9613–9622, 06 2019. doi: 10.1109/CVPR.2019.00985.

[27] W. Shi and R. Rajkumar. Point-gnn: Graph neural network for 3D object detection in a point cloud. In *CVPR*, pages 1708–1716, 2020.

[28] A. Juliani, V. Berges, E. Teng, A. Cohen, J. Harper, C. Elion, C. Goy, Y. Gao, H. Henry, M. Matar, and D. Lange. Unity: A general platform for intelligent agents. *ArXiv*, abs/1809.02627, 2020.

[29] L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2:193–218, 1985.

[30] J. Bornschein, A. Mnih, Daniel Zoran, and Danilo Jimenez Rezende. Variational memory addressing in generative models. In *NIPS*, 2017.