This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Learning Spatial Common Sense with Geometry-Aware Recurrent Networks

Hsiao-Yu Fish Tung[1*] Ricson Cheng [2*†] Katerina Fragkiadaki [1]
[1]Carnegie Mellon University [2]Uber Advanced Technologies Group
{htung,katef}@cs.cmu.edu, ricsonc@uber.com

Abstract

We integrate two powerful ideas, geometry and deep visual representation learning, into recurrent network architectures for mobile visual scene understanding. The proposed networks learn to "lift" and integrate 2D visual features over time into latent 3D feature maps of the scene. They are equipped with differentiable geometric operations, such as projection, unprojection, egomotion estimation and stabilization, in order to compute a geometrically-consistent mapping between the world scene and their 3D latent feature state. We train the proposed architectures to predict novel camera views given short frame sequences as input. Their predictions strongly generalize to scenes with a novel number of objects, appearances and configurations; they greatly outperform previous works that do not consider egomotion stabilization or a space-aware latent feature state. We train the proposed architectures to detect and segment objects in 3D using the latent 3D feature map as input—as opposed to per frame features. The resulting object detections persist over time: they continue to exist even when an object gets occluded or leaves the field of view. Our experiments suggest the proposed space-aware latent feature memory and egomotion-stabilized convolutions are essential architectural choices for spatial common sense to emerge in artificial embodied visual agents.

1. Introduction

Current state-of-the-art visual systems [11] accurately detect object categories that are rare and unfamiliar to many of us, such as *gyromitra*, a particular genus of mushroom (Figure 1 top left). Yet, they neglect the basic principles of object permanence or spatial awareness that a one-year-old child has developed: once the camera turns away, or a person walks in front of the gyromitra, its detection disappears and it is replaced by the objects detected in the new visual frame. We believe the ability of current visual systems to detect rare and exquisite object categories and their inability to carry out elementary spatial reasoning is due to the fact that they are trained to *label object categories* from *static Internet photos* (in ImageNet and COCO datasets) using a *single frame* as input. Our overexposure to Internet photos makes us forget how pictures captured by mobile agents look. Consider Figure 1. Internet photos are skillfully captured by human photographers, are well framed and show objects unoccluded, in canonical locations, scales and poses (top row). Instead, photos captured by NAO robots during a soccer game show objects in a wide variety of scales, poses, locations, and occlusion configurations (bottom row). Often, it would not even make sense to label objects in such images, as most objects appear only half-visible. In the case of Internet vision, the picture is the **output** of visual perception of a well-trained visual agent, the human photographer. In the case of mobile robotic vision, the picture is the **input** to such visual perception. Thus, different architectures may be needed for each.

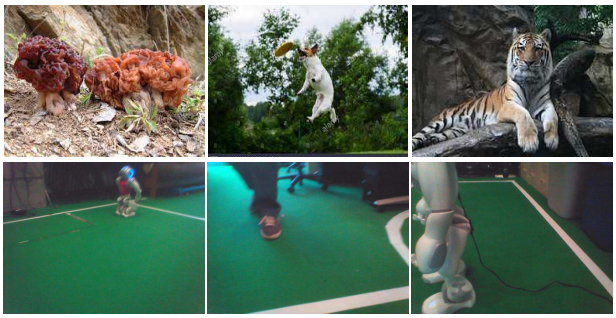

Figure 1. **Internet vision versus robotic vision**. Pictures taken by humans (top row) (and uploaded on the web) are the *output* of visual perception of a well-trained agent, the human photographer. The content is skillfully framed and the objects appear in canonical scales and poses. Pictures taken by mobile agents, such as a NAO robot during a robot soccer game (bottom row), are the *input* to such visual perception. The objects are often partially occluded and appear in a wide variety of locations, scales and poses. We present recurrent neural architectures for the latter, that integrate visual information over time to piece together the visual story of the scene.

We present Geometry-aware Recurrent Neural Network

*Indicates equal contribution
†Work done while at CMU

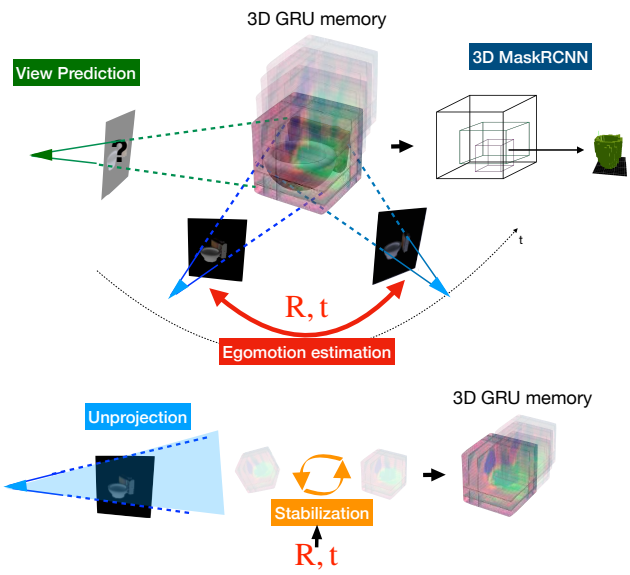

Figure 2. **Geometry-aware Recurrent Neural Networks (GRNNs)** integrate visual information over time in a 3D geometrically-consistent deep feature memory of the visual scene. At each frame, RGB images are *unprojected* into corresponding 3D feature tensors, which are oriented to the coordinate frame of the memory map built thus far (2nd row). A 3D convolutional GRU memory is then updated using the egomotion-stabilized features as input.

architectures, which we call GRNNs, that learn to "lift" and integrate over time 2D image features into 3D feature maps of the scene, while stabilizing against the egomotion of the agent. They update over time a 3-dimensional latent feature state: the latent feature vectors are arranged in a 3D grid, where every location of the grid encodes a 3D physical location in the scene. The latent state is updated with each new input frame using egomotion-stabilized convolutions, as shown in Figure 2. GRNNs learn to map 2D input visual features to a 3D latent feature map, and back, in a differentiable manner. To achieve such differentiable and geometrically-consistent mapping between the world scene and the 3D latent feature state, they are equipped with differentiable geometric operations, such as egomotion estimation and feature stabilization, 3D-to-2D projection, and 2D-to-3D unprojection, as shown in Figure 2. Beyond being space-aware, we do not impose any other constraints on the learned representations: they are free to encode whatever is relevant for the downstream task.

We train GRNNs in a self-supervised manner to predict image views from novel camera viewpoints, given short frame sequences as inputs. We empirically show GRNNs learn to predict novel views and **strongly generalize** to novel scenes with different number, appearances and configuration of objects. They greatly outperform geometry-unaware networks of previous works that are trained under the exact same view-prediction loss, but do not use egomotion-stabilized convolutions or a 3D latent space. We argue strong generalization is a necessary condition for claiming the ability to spatially reason. Furthermore, we show learnt representations of GRNNs support scene arithmetics: adding/subtracting latent scene 3D feature maps and decoding them from a particular viewpoint matches the result of adding/subtracting 3D world scenes directly.

We train GRNNs in a supervised manner to detect and segment objects in 3D, given short frame sequences as inputs (Figure 2). We use the latent 3D feature map as input to a 3D version of maskRCNN [11], a state-of-the-art 2D object detector/segmentor. The resulting 3D object detections and 3D voxel segmentations *persist in time* despite instantaneous occlusions and dis-occlusions: an object that is not visible in the current frame is still present in the latent 3D feature map. By projecting the detected 3D objects in 2D we obtain *amodal* [20] object boxes and segments, even under severe occlusions. Visual recognition with GRNNs thus exhibits **object permanence**, a property which is effortless for humans, and impossible thus far for 2D visual detectors. The GRNN architecture borrows many ideas from SLAM (Simultaneous Localization and Mapping) methods, as we discuss in Section 3. GRNNs though eventually learn to "imagine" missing content of the scene without visual inspection from multiple viewpoints. Datasets and code are available at our project page https://ricsonc.github.io/grnn/.

2. Related Work

Deep geometry Simultaneous Localization and Mapping (SLAM) [23, 16] methods are purely geometric methods that build a 3D pointcloud map of the scene while estimating the motion of the camera. Our method builds multiple deep feature maps instead, which capture both the geometry and the semantics of the scene. Recently, there has been great interest in integrating learning and geometry for single view 3D object reconstruction [25, 28], 3D object reconstruction from videos [18], depth and egomotion estimation from pairs of frames [27, 32], depth estimation from stereo images [8], and estimation of 3D human keypoints from 2D keypoint heatmaps [29, 26]. Many of those works use neural network architectures equipped with some form of differentiable camera projection, so that the 3D desired estimates can be supervised directly using 2D quantities. For example, Tulsiani *et al*. [25], Wu *et al*. [28] and Zhou *et al*. [32] use a single image frame as input to predict a 3D reconstruction for a single object, or a 2D depth map for the entire scene. These works use multiple views only to obtain extra regularization for the predictions in the form of depth re-projection error. Learnt stereo machines (LSM) [15] integrate RGB information along sequences of random camera viewpoints into a latent 3D feature memory tensor, in

an egomotion-stabilized way, similar to our method. However, their goal is to 3D reconstruct a single object, as opposed to detect and 3D reconstruct multiple objects, which our model does. They assume egomotion is given, while we also propose a way to estimate egomotion. They can only be trained supervised for the object 3D reconstruction task, while GRNNs can be trained self-supervisedly through view prediction. The work of LSM has inspired though the models proposed in this paper.

MapNet [12], Cognitive mapping and planning [10], IQA [9] and Neural Map [19] construct 2D overhead maps of the scene by taking into account the egomotion of the observer, similar to our method. MapNet further estimates the egomotion, while other methods assume it is known. In IQA, objects are detected in each frame and detections are aggregated in a birdview map, whereas we detect objects directly using the 3D feature map as input.

The closest work to ours is the work of Cheng at al. [3], which considers egomotion-stabilized convolutions and a 3D latent map for segmenting objects in 3D, like us. However, they assume egomotion is known—while we learn to estimate it—and their object detection pipeline uses heuristics in order to specify the number of objects in the scene by discretizing continuous voxel segmentation embeddings that they obtain with metric learning. We instead train 3D region proposal and segmentation networks. Most importantly, they do not consider self-supervised learning via view prediction, which is one of the central contributions of this work. Rather, they exclusively focus on supervised voxel labelling using groundtruth 3D voxel occupancies provided by a simulator.

Self-supervised visual feature learning Researchers have considered many self-supervised tasks to train visual representations without human labels. For example, works of [13, 1] train visual representation by predicting egomotion between consecutive frames, and works of [6, 24] predict novel views of a scene. In particular, the authors of generative query network (GQN) [6] argue that GQN learns to disentangle color, lighting, shapes and spatial arrangement without any human labels. We compare against their model in Section 4 and show GRNNs can strongly generalize beyond the training set, while GQN cannot. Such strong generalization suggests that 3D latent space and egomotion-stabilization are necessary architectural choices for spatial reasoning to emerge.

3D object detection When LiDAR input is available, many recent works attempt detecting objects directly in 3D using LiDAR and RGB streams [33, 17, 31]. They mostly use a single frame as input, while the proposed GRNNs integrate visual information over time. Extending GRNNs to scenes with independently moving objects is a clear avenue for future work.

3. Geometry-aware recurrent networks

GRNNs are recurrent neural networks whose latent state $\mathrm{m}_t \in \mathbb{R}^{w\times h\times d\times c}, t = 1\cdots T$ learns a 3D deep feature map of the visual scene. We use the terms 4D tensor and 3D feature map interchangeably, to denote a set of feature channels, each being 3-dimensional. The memory map is updated with each new camera view in a geometrically-consistent manner, so that information from 2D pixel projections that correspond to the same 3D physical point end up nearby in the memory tensor, as illustrated in Figure 3. This permits later convolutional operations to have a correspondent input across frames, as opposed to it varying with the motion of the observer. We believe this is a key for generalization. The main components of GRNNs are illustrated in Figure 3 and are detailed right below.

Unprojection At each timestep, we feed the input RGB image I_t to a 2D convolutional encoder-decoder network with skip-connections (2D U-net [22]) to obtain a set of 2D feature maps $\mathrm{F}_t \in \mathbb{R}^{w\times h\times c}$. We then unproject all feature maps to create a 4D feature tensor $\mathrm{V}_t^I \in \mathbb{R}^{w\times h\times d\times c}$ as follows: For each "cell" in the 3D feature grid indexed by (i, j, k), we compute the 2D pixel location (x, y) which the center of the cell projects onto, from the current camera viewpoint:

$$[x, y] = [f \cdot i/k, f \cdot j/k],$$

where f is the focal length of the camera. Then, $\mathrm{V}^I_{i,j,k,:}$ is filled with the bilinearly interpolated 2D feature vector at that pixel location (x, y). All voxels lying along the same ray casted from the camera center will be filled with nearly the same image feature vectors. We further unproject the input 2D depthmap D_t into a binary voxel occupancy grid $\mathrm{V}_t^D \in \{0, 1\}^{w\times h\times d}$ that contains the thin shell of voxels directly visible from the current camera view. We compute this by filling all voxels whose unprojected depth value equals the grid depth value. When a depth sensor is not available, we learn to estimate the depthmap using a 2D U-net that takes the RGB image as input.

We multiply each 3-dimensional channel of the feature tensor V_t^I with the binary occupancy grid V_t^D to get a final 4D feature tensor $\mathrm{V}_t \in \mathbb{R}^{w\times h\times d\times c}$. The unprojected tensor V_t enters a 3D encoder-decoder network with skip connections (3D U-net) to produce a resulting feature tensor $\bar{\mathrm{V}}_t \in \mathbb{R}^{w\times h\times d\times c}$.

Egomotion estimation and stabilization Our model orients the 3D feature memory to have 0° elevation using the absolute elevation angle of the first camera view. We assume this value is given, but it can also be estimated using a

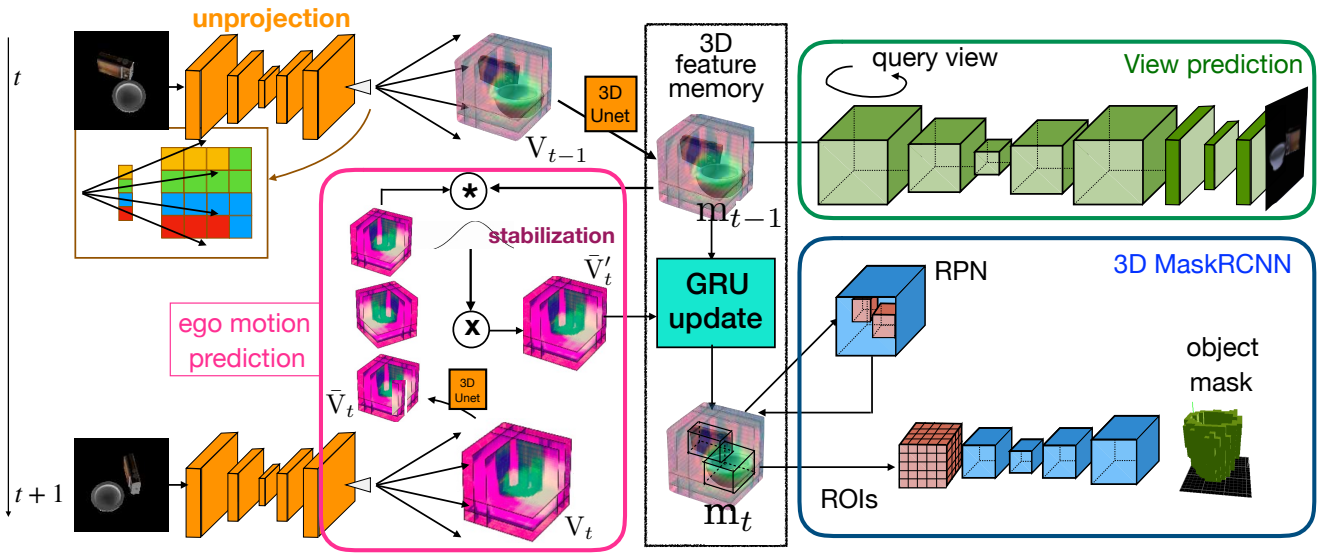

Figure 3. **GRNN architecture.** At each time step t, an RGB image I_t is the input to a 2D U-net. The resulting 2D deep feature maps are unprojected to 4D tensors V_t, which in turn are input to a 3D U-net (we do not show the optional combination with unprojected depthmaps for clarity). The resulting 3D deep feature maps $\bar{\mathrm{V}}$ are oriented to cancel the relative camera motion between the current viewpoint and the coordinate system of the 3D GRU memory state m_{t-1}, as estimated by an egomotion estimation module. The resulting oriented 3D deep feature maps $\bar{\mathrm{V}}_t$' update the 3D GRU memory state and output m_t. The updated state of the GRU module is then projected from specific viewpoints and decoded into a corresponding RGB image for view prediction, or fed into a 3D MaskRCNN to predict 3D object bounding boxes and object voxel occupancies.

2D convnet. This essentially makes the memory to always be *parallel to the ground plane*. The azimuth of the 3D feature memory is chosen to be the azimuth of the first view in the input frame sequence. We assume the camera does not translate, only rotates by varying two degrees of freedom, elevation and azimuth.

At each time step t, we estimate the relative elevation and azimuth between the current frame's viewpoint and the feature memory. Note that we can alternatively predict the (absolute) elevation directly from each input view, without matching against the memory built thus far. For the azimuth, since we need to estimate the relative azimuth to the first view, such cross-view comparison is necessary. Specifically, the tensor $\bar{\mathrm{V}}_t$ is *rotated* by different azimuth and elevation angles and results in a stack of rotated feature tensors $\bar{\mathrm{V}}^{\mathrm{rot}} \in \mathbb{R}^{(L\cdot K)\times w\times h\times d\times c}$, where L, K are the total number of azimuths and elevation angles considered, respectively, after discretization. Similar to the bilinear interpolation used during unprojection, to fill in each feature voxel in a rotated tensor $\bar{\mathrm{V}}^{\mathrm{rot}}_{\cdot,i,j,k,:}$, we compute the 3D location (X, Y, Z) where it is rotated from and insert the bilinearly interpolated feature value from the original tensor $\bar{\mathrm{V}}_t$. We then compare each of the rotated feature maps with our current 3D feature memory $\mathrm{m}_{t-1} \in \mathbb{R}^{w\times h\times d\times c}$ using matrix inner products, to produce a probability distribution over azimuth and elevation pairs:

$$\bar{\rho}_t(r) = \mathrm{m}_{t-1} * \bar{\mathrm{V}}_{rot}(r, :, :, :, :), \quad r \in 1 \cdots L \cdot K$$
$$\rho_t = \mathrm{softmax}(\bar{\rho}_t),$$

where $*$ denotes matrix inner product. The resulting rotation $\bar{r}_t$ is obtained by a weighted average of azimuth and elevation angles where weights are in ρ_t. Finally, we orient the tensor $\bar{\mathrm{V}}_t$ to cancel the relative rotation $\bar{r}_t$ with respect to our 3D memory m_{t-1}, we denote the oriented tensor as $\bar{\mathrm{V}}'_t$.

Recurrent map update Once the feature tensor has been properly oriented, we feed $\bar{\mathrm{V}}'_t$ as input to a 3D convolutional Gated Recurrent Unit [4] layer, whose hidden state is the memory $\mathrm{m}_{t-1} \in \mathbb{R}^{w\times h\times d\times c}$, as shown in Figure 3. This state update outputs m_t. The hidden state is initialized to zero at the beginning of the frame sequence. For our view prediction experiments (Section 4) where we use a fixed number of views T, we found that averaging, namely $\mathrm{m}_T = \frac{1}{T}\sum_t \bar{\bar{V}}'_t$ works equally well to using the GRU update equations, while being much faster.

Projection and decoding Given a 3D feature memory m_t and a desired viewpoint q, we first rotate the 3D feature memory so that its depth axis is aligned with the query camera axis. We then generate for each depth value k a corresponding projected feature map $\mathrm{p}_k \in \mathbb{R}^{w\times h\times c}$. Specifi-

cally, for each depth value, the projected feature vector at a pixel location (x, y) is computed by first obtaining the 3D location it is projected from and then inserting bilinearly interpolated value from the corresponding slice of the 4D tensor m. In this way, we obtain d different projected maps, each of dimension $w \times h \times c$. Depth ranges from $D-1$ to $D+1$, where D is the distance to the center of the feature map, and are equally spaced.

Note that we do not attempt to determine visibility of features at this projection stage. The stack of projected maps is processed by 2D convolutional operations and is decoded using a residual convLSTM decoder, similar to the one proposed in [6], to an RGB image. We do not supervise visibility directly. The network implicitly learns to determine visibility and to choose appropriate depth slices from the stack of projected feature maps.

3.1. View prediction

Mobile agents have access to their egomotion, and can observe sensory outcomes of their motions and interactions. Training sensory representations to predict such outcomesis a useful form of supervision, free of human annotations, often termed *self-supervision* since the "labels" are provided by the embodied agent herself. Can spatial common sense, the notion of objects and scenes, geometry, visibility and occlusion relationships, emerge in a self-supervised way in a mobile agent that moves around and observes the world?

We train GRNNs to predict the image the agent would see from a novel viewpoint, given a short view sequence as input. Given the 3D feature memory and a query viewpoint, we orient the map to the query viewpoint, we project it to 2D and decode it to an RGB image, as described above. We train our view prediction using a standard cross-entropy pixel matching loss, where the pixel intensity has been squashed into the range $[0, 1]$. Our model is end-to-end differentiable. For view prediction, we did not use depth as input, nor did we use a 2D U-net to estimate it. We also did not set the memory to be parallel top the ground plane. We use only the RGB input and set the coordinate system of the memory to match that of the first camera view after unprojection, for a fair comparison with prior art. We show in Section 4 that GRNNs greatly outperform alternative geometry-unaware RNN architectures in view prediction and **strongly generalize** beyond the training set to novel scenes with different number of objects, appearances and arrangements. Training and implementation details are included in the supplementary file.

3.2. 3D object detection and segmentation

We train GRNNs in a supervised manner to predict 3D object bounding boxes and 3D object segmentation masks, using groundtruth 3D object boxes and 3D voxel segmentations from a simulator. We adapt MaskRCNN [11], a state-of-the-art object detector/segmentor, to have 3D input and output, instead of 2D. Specifically, we consider every grid location (X, Y, Z) in our 3D memory to be a candidate 3D box centroid. At each time step, the 3D feature memory m_t is fed to a 3D region proposal network to predict positive anchor centroids, as well as the corresponding adjustment for the box center location and the box dimensions, width, height and depth. Our 3D bounding box encoding is similar to the one proposed in VoxelNet [33]. We filter the proposed boxes using non-max suppression to reject highly overlapping ones. We train with a combination of classification and regression loss, following well established detector training schemes [21, 11]. The proposed 3D bounding boxes that have Intersection of Union (IoU) above a specific threshold with a corresponding groundtruth object box are denoted as Regions of Interest (ROIs) and are used to pool features from their interior to predict 3D object voxel occupancy, as well as a second refinement of the predicted 3D box location and dimensions.

Object permanence Even when an object is not visible in the current camera viewpoint, its features are present in the 3D feature memory, and our detector detects and segments it, as we show in the second column of Figure 6. In other words, object detections persist through occlusions and changes of the field of view caused by camera motion. Applying the detector on the latent 3D model of the scene as opposed to the 2D visual frame is beneficial. The latent 3D model follows the physical laws of 3D non-intersection and object permanence, while 2D visual observations do not.

4. Experiments

The term "spatial common sense" is broad and concerns the ability to perceive and understand properties and regularities regarding spatial arrangements and motion that are shared by ("common to") nearly all people. Such common sense includes the fact that objects have 3D shape as opposed to being floating 2D surfaces, the fact that scenes are comprised of objects, the 3D non-intersection principle, the fact that objects do not spontaneously disappear, and many others [7]. The model we propose in this work targets understanding of *static* scenes, that is, scenes that do not contain any independently moving objects, and that are viewed under a potentially moving observer. Thus, we restrict the term *spatial common sense* to refer to rules and regularities that can be perceived in static worlds. Our experiments aim to answer the following questions:

1. Do GRNNs learn spatial common sense?
2. Are geometric structural biases necessary for spatial common sense to emerge?
3. How well do GRNNs perform on egomotion estimation and 3D object detection?

4.1. View prediction

We consider the following simulation datasets:
i) *ShapeNet arrangement* from [3] that contains scenes with synthetic 3D object models from ShapeNet [2] arranged on a table surface. The objects in this dataset belong to four object categories, namely, cups, bowls, helmets and cameras. We follow the same train/test split of ShapeNet [2] so that object instances which appear in the training scenes do not appear in the test scenes. Each scene contains two objects, and each image is rendered from a viewing sphere which has 3×18 possible views with 3 camera elevations $(20°, 40°, 60°)$ and 18 azimuths $(0°, 20°, \ldots, 340°)$. There are 300 different scenes in the training set and 32 scenes with novel objects in the test set.
ii) *Shepard-metzler* shapes dataset from [6] that contains scenes with seven colored cubes stuck together in random arrangements. We use the train and test split of [6].
iii) *Rooms-ring-camera* dataset from [6] that contains rooms with random floor and wall colors, in which there are variable numbers of objects with different shapes and colors.

We compare GRNNs against the recent "tower" architecture of Eslami et al. [6], a 2D network trained under a similar view prediction loss. At each time step, the tower architecture takes as input a 2D RGB image and performs a series of convolutions on it. The camera pose from which the image was taken is tiled along the width and height axes and then concatenated with the feature map after the third convolution. Finally, the feature maps from all views are combined via average pooling. Both our model and the baseline use the same autoregressive decoder network. For fairness of comparison, we use groundtruth egomotion rather than estimated egomotion in all view prediction experiments, and only RGB input (no depth input of depth estimation) for both our model and the tower baseline. In both the baseline and our model, we did not use any stochastic units for simplicity and speed of training. Adding stochastic units in both is part of our future work.

Test results from our model and baseline on test images of ShapeNet arrangements and Shepard-metzler datasets are shown in Figure 4. Reconstruction test error for the ShapeNet arrangement test set is shown in Table 1. GRNNs have a much lower reconstruction test error than the tower baseline. In Figure 4, in the first four rows, the distribution of the test scenes matches the training scene distribution. Our model outperforms the baseline in visual fidelity. In Figure 4, in the last four rows, the test scene distribution does not match the training one: we test our model and baseline on **scenes with four objects, while both models are trained on scenes with exactly two objects.** In this case, our model shows *strong generalization* and outperforms by a margin the geometry-unaware baseline of [6], the latter refuses to see more than two objects present. We argue the ability to spatially reason should not be affected by the number of objects present in the scene. Our results suggest that geometry-unaware models may be merely memorizing views with small interpolation capabilities, as opposed to learning to spatially reason.

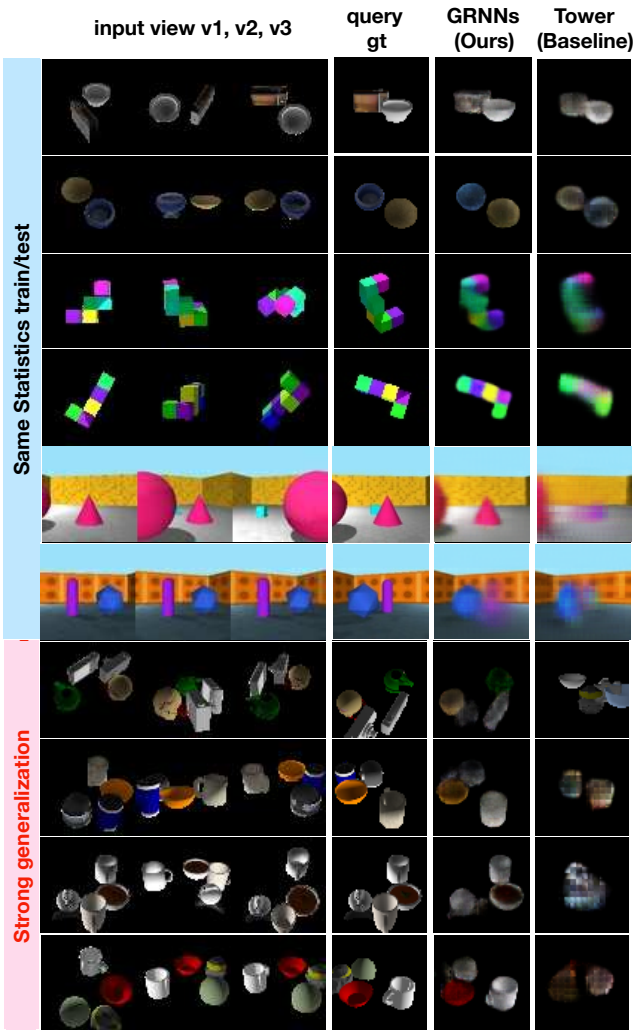

Figure 4. **View prediction results** for the proposed GRNNs and the tower model of Eslami et al. [6]. Columns from left to right show the three input views, the groundtruth image from the query viewpoint, the view predictions for GRNNs and for the tower baseline. The first two rows are from the ShapeNet arrangement test set of [3], the next two rows are from the Shepard-Metzler test set of [6], and the following two rows are from the *Rooms-ring-camera* dataset also from [6]. The last four rows show generalization to scenes with four objects from the ShapeNet arrangement dataset, while both models were trained only on scenes with two objects. GRNNs outperform the baseline by a large margin and *strongly* generalize under a varying number of objects.

Scene arithmetics The learnt representations of GRNNs are capable of scene arithmetics, as we show in Figure 5.

The ability to add and subtract individual objects from 3D scenes just by adding and subtracting their corresponding latent representations demonstrates that our model disentangles what from where. In other words, our model learns to store object-specific information in the regions of the memory which correspond to the spatial location of the corresponding object in the scene. Implementation details and more qualitative view prediction results are included in the supplementary file.

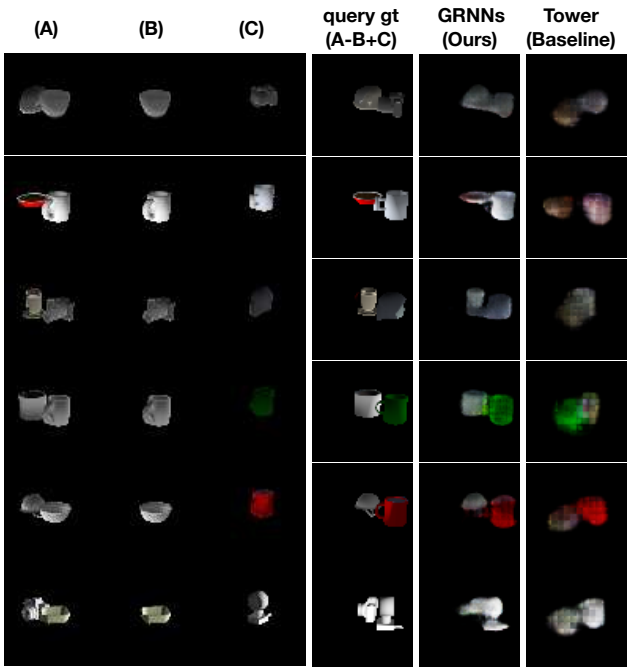

Figure 5. **Scene arithmetic** with GRNNs and the model of Eslami et al. [6] (tower). Each row is a separate "equation". We start with the representation of the scene in the leftmost column, then subtract (the representation of) the scene in the second column, and add the (representation of the) scene in the third column. We decode the resulting representation into an image. The groundtruth image is shown in the forth column. It is much more visually similar to the prediction of GRNNs than to the tower baseline.

	Tower (Baseline)	GRNNs (Ours)
ShapeNet	0.109 ± 0.029	$\mathbf{0.084} \pm 0.017$
Shepard-Metzler	0.081 ± 0.017	$\mathbf{0.073} \pm 0.014$

Table 1. **View prediction loss and the standard deviation** for the ShapeNet arrangement test set for two-object test scenes. Our model and baseline were trained on scenes that also contain two objects with different object instances.

4.2. Egomotion estimation

In this section, we quantify the error of our egomotion estimation component. We train our egomotion estimation module using groundtruth egomotion from a simulator, using the ShapeNet arrangement dataset. In Table 2, we show egomotion estimation error in elevation and azimuth angles. Our model improves its egomotion estimates with more views, since then a more complete feature memory is compared against each input unprojected tensor.

# views	one	two	three	avg.
GRNNs	$8.6/17.8$	$5.6/16.8$	$5.6/6.6$	$6.6/13.7$

Table 2. **Egomotion estimation error** of GRNNs in elevation and azimuth angles for the ShapeNet arrangement test set using different number of views. The error decreases with more views integrated in the memory.

4.3. 3D object detection and segmentation

We use the ShapeNet arrangement dataset, and the train/test scene split of [3]. We use mean Average Precision (mAP) to score the performance of our model and baselines for 3D object detection and 3D segmentation. Mean average precision measures the area under the precision-recall curve. We vary the cutoff threshold of Intersection over Union (IoU) to be 0.33, 0.5 and 0.75 between our predictions and the groundtruth 3D boxes and masks. We consider four ablations for our model: predicted egomotion (*pego*) versus groundtruth egomotion (*gtego*) used, and predicted depth (*pd*) versus groundtruth depth (*gtd*) used as input. We use suffixes to indicate the model we use.

We compare against the following 2D baseline model, which we call *2D-RNN*: we remove the unprojection, egomotion estimation and stabilization and projection operations from our model. The baseline takes as input an image and the corresponding depth map, feeds it to a 2D encoder-decoder network with skip connections to obtain a 2D feature tensor. The camera parameters for this view are concatenated as additional channels to the 2D feature tensor and altogether they are fed to another 2D encoder-decoder network to obtain the 2D feature tensor for a 2D GRU memory update. We then feed the 2D memory feature tensor to an additional 2D encoder-decoder network and reshape the channel dimension of its output into d feature vector of length 7 (one value for the anchor box prediction, six values for the 3D bounding boxes adjustments) to form a 4D tensor of size $w \times h \times d \times 7$ as prediction.

We show mean average precision for 3D object detection and 3D segmentation for our model and the baseline in Table 3, and visualize predicted 3D bounding boxes and segmentations from GRNNs (GRNN-gtego-gtd) in Figure 6. GRNNs significantly outperform the 2D-RNN. Groundtruth depth input significantly helps 3D segmentation. This suggests that inferring depth using a cost volume as in [15] would potentially help depth inference as opposed to relying on a per frame depthnet [5] that does not have access to multiple views to improve its predictions. Implementa-

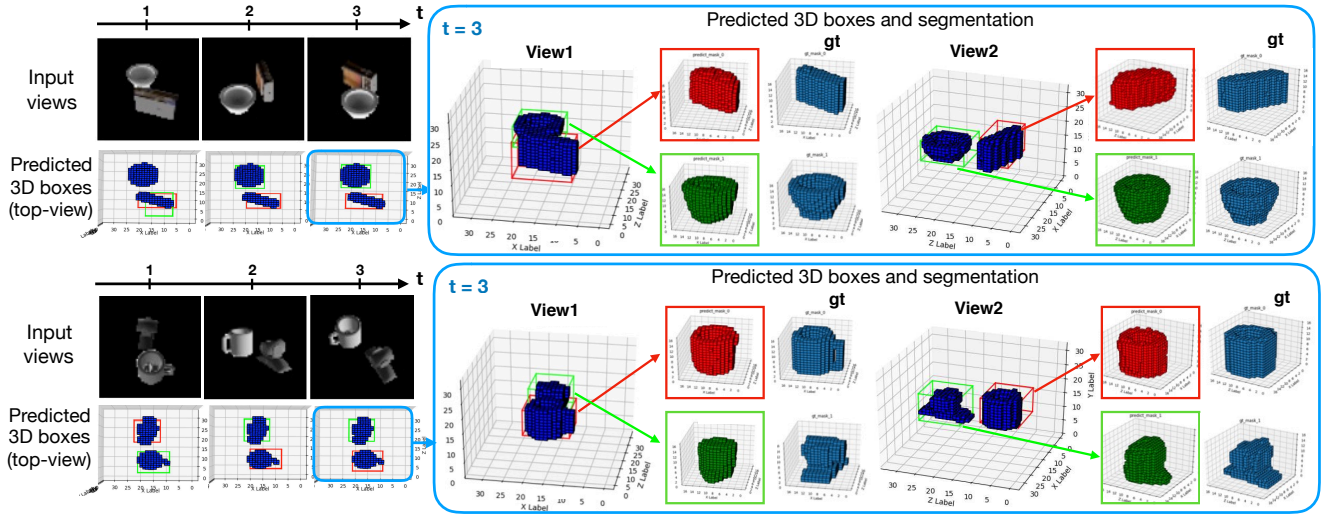

Figure 6. **3D object detection and segmentation** with GRNNs. In the first and second row on the left we show the input images over time, and their corresponding object detection results for a top view, respectively. Blue voxels denote groundtruth objects and the predicted bounding boxes are shown in red and green . On the right, we show segmentation results for the third time step, visualizing the results from two views. Predicted 3D boxes and their corresponding predicted masks are show in red and green, and we show in blue the corresponding groundtruth. Best seen in color.

detection	2DRNN-gtego-gtd	GRNN-gtego-pd	GRNN-gtego-gtd	GRNN-pego-gtd	segmentation	2DRNN-gtego-gtd	GRNN-gtego-pd	GRNN-gtego-gtd	GRNN-pego-gtd
$\text{mAP}^d_{0.75}$	0.364	0.471	**0.816**	0.549	$\text{mAP}^m_{0.75}$	0.003	0.024	**0.058**	0.023
$\text{mAP}^d_{0.50}$	0.964	0.964	**0.998**	0.983	$\text{mAP}^m_{0.50}$	0.104	0.246	**0.338**	0.249
$\text{mAP}^d_{0.33}$	0.998	0.994	**0.999**	0.999	$\text{mAP}^m_{0.33}$	0.244	0.429	**0.485**	0.384

Table 3. Mean Average Precision (mAP) for 3D object detection and 3D segmentation for three different thresholds of Intersection over Union (IoU) (0.75,0.5,0.33) on ShapeNet arrangement test set of [3].

tion details and more qualitative results are included in the supplementary file.

5. Conclusion

We presented GRNNs, recurrent neural networks equipped with differentiable geometric operations to estimate egomotion and build 3D deep feature maps for visual scene understanding on mobile visual agents. GRNNs add a new dimension to the latent space of previous recurrent models and ensure a geometrically-consistent mapping between the latent state and the 3D world scene. We showed spatial common sense emerges in GRNNs when trained in a self-supervised manner for novel view prediction. They can predict object arrangements, visibility and occlusion relationships in scenes with novel number, appearance and configuration of objects. We also showed that view prediction as a loss does not suffice for spatial common sense to emerge, since 2-dimensional models of previous works fail to strongly generalize.

Thus far, GRNNs has been trained and tested on simulated scenes. Deploying our model on more realistic environmentsis a clear avenue for future work. We expect pretraining in simulated environments to help performance in the real world. Besides, one limitation of the current model is that it operates on static scenes. Extending the proposed architectures to dynamic scenes, scenes with independently moving objects in addition to camera motion, is another very useful direction of future work. Finally, exploiting the sparsity of our 4D tensors to save GPU memory is an important direction for scaling up our model to large scenes.

GRNNs pave the way for embodied agents that learn visual representations and mental models by observing and moving in the world: these agents learn autonomously and develop the reasoning capabilities of young toddlers as opposed to merely mapping pixels to labels using human supervision.

Acknowledgement

We would like to thank Xian Zhou for his help on training and testing the 3D MaskRCNN. This work is partly funded by a Google faculty award.

References

[1] P. Agrawal, J. Carreira, and J. Malik. Learning to see by moving. *CoRR*, abs/1505.01596, 2015.

[2] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. 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.

[3] R. Cheng, Z. Wang, and K. Fragkiadaki. Geometry-aware recurrent neural networks for active visual recognition. In *NIPS*, 2018.

[4] K. Cho, B. van Merrienboer, Ç. Gülçehre, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014.

[5] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2366–2374. Curran Associates, Inc., 2014.

[6] S. M. A. Eslami, D. Jimenez Rezende, F. Besse, F. Viola, A. S. Morcos, M. Garnelo, A. Ruderman, A. A. Rusu, I. Danihelka, K. Gregor, D. P. Reichert, L. Buesing, T. Weber, O. Vinyals, D. Rosenbaum, N. Rabinowitz, H. King, C. Hillier, M. Botvinick, D. Wierstra, K. Kavukcuoglu, and D. Hassabis. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, 2018.

[7] C. E. Freer, D. M. Roy, and J. B. Tenenbaum. Towards common-sense reasoning via conditional simulation: legacies of turing in artificial intelligence. *CoRR*, abs/1212.4799, 2012.

[8] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. *CoRR*, abs/1609.03677, 2016.

[9] D. Gordon, A. Kembhavi, M. Rastegari, J. Redmon, D. Fox, and A. Farhadi. IQA: visual question answering in interactive environments. *CoRR*, abs/1712.03316, 2017.

[10] S. Gupta, J. Davidson, S. Levine, R. Sukthankar, and J. Malik. Cognitive mapping and planning for visual navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2616–2625, 2017.

[11] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017.

[12] J. F. Henriques and A. Vedaldi. Mapnet: An allocentric spatial memory for mapping environments. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[13] D. Jayaraman and K. Grauman. Learning image representations equivariant to ego-motion. *CoRR*, abs/1505.02206, 2015.

[14] M. B. A. J. D. John McCormac, Ronald Clark and S. Leutenegger. Fusion++: Volumetric object-level slam. In *arXiv:1808.08378*. 2018.

[15] A. Kar, C. Häne, and J. Malik. Learning a multi-view stereo machine. *CoRR*, abs/1708.05375, 2017.

[16] C. Kerl, J. Sturm, and D. Cremers. Dense visual SLAM for RGB-D cameras. In *IROS*, 2013.

[17] M. Liang, B. Yang, S. Wang, and R. Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[18] D. Novotný, D. Larlus, and A. Vedaldi. Learning 3d object categories by looking around them. *CoRR*, abs/1705.03951, 2017.

[19] E. Parisotto and R. Salakhutdinov. Neural map: Structured memory for deep reinforcement learning. *CoRR*, abs/1702.08360, 2017.

[20] L. Regolin and G. Vallortigara. Perception of partly occluded objects by young chicks. *Perception & Psychophysics*, 57(7):971–976, 1995.

[21] S. Ren, K. He, R. B. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.

[22] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.

[23] T. Schöps, J. Engel, and D. Cremers. Semi-dense visual odometry for AR on a smartphone. In *ISMAR*, 2014.

[24] M. Tatarchenko, A. Dosovitskiy, and T. Brox. Single-view to multi-view: Reconstructing unseen views with a convolutional network. *CoRR*, abs/1511.06702, 2015.

[25] S. Tulsiani, T. Zhou, A. A. Efros, and J. Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. *CoRR*, abs/1704.06254, 2017.

[26] H. F. Tung, A. Harley, W. Seto, and K. Fragkiadaki. Adversarial inverse graphics networks: Learning 2d-to-3d lifting and image-to-image translation with unpaired supervision. *ICCV*, 2017.

[27] S. Vijayanarasimhan, S. Ricco, C. Schmid, R. Sukthankar, and K. Fragkiadaki. Sfm-net: Learning of structure and motion from video. *arXiv preprint arXiv:1704.07804*, 2017.

[28] J. Wu, Y. Wang, T. Xue, X. Sun, W. T. Freeman, and J. B. Tenenbaum. Marrnet: 3d shape reconstruction via 2.5d sketches. *CoRR*, abs/1711.03129, 2017.

[29] J. Wu, T. Xue, J. J. Lim, Y. Tian, J. B. Tenenbaum, A. Torralba, and W. T. Freeman. 3d interpreter networks for viewer-centered wireframe modeling. *International Journal of Computer Vision (IJCV)*, 2018.

[30] Y. Xiang and D. Fox. Da-rnn: Semantic mapping with data associated recurrent neural networks. In *Robotics: Science and Systems (RSS)*. 2017.

[31] B. Yang, M. Liang, and R. Urtasun. Hdnet: Exploiting hd maps for 3d object detection. In A. Billard, A. Dragan, J. Peters, and J. Morimoto, editors, *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 146–155. PMLR, 29–31 Oct 2018.

[32] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, 2017.

[33] Y. Zhou and O. Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. *IEEE Conference on Computer Vision and PatternRecognition (CVPR)*, 2018.