A Related Works

Self-supervision has recently emerged as one of the most promising approaches to ease the need for supervision and yet maintain high performance. Self-supervision builds on the fact pretext tasks can be very useful for pre-training networks without the need for expensive manual annotations. With such pre-trained networks, only a modest amount of labelled data will be needed to fine-tune for a target task. Specifically, [11] shows that colorization can be a powerful pretext task as it serves as a cross-channel encoder. Deep features (e.g. ImageNet-trained VGG/ResNet features) are also demonstrated to be remarkably useful as a training loss for tasks including image synthesis and outperform all previous metrics by large margins in [12]. In [9], it is found out that pre-training on vision tasks (e.g. object detection) significantly improves generalization and sample efficiency for learning to manipulate objects. Therefore, directly transferring model parameters from vision networks to affordance prediction networks can result in successful zero-shot adaptation, where a robot can pick up certain objects with zero robotic experience. A comprehensive study [10] is proposed recently giving analysis of self-supervision. Specially, authors conclude that the weights of the early layers in a deep network contain low-level statistics of natural images, which can be learned decently through solely self-supervision or captured via synthetic transformations instead of using a large image dataset. Slightly more related are works that exploit the possibility of pre-training without natural images [3, 4]. Recently, authors of [3] generate image patterns and their category labels to construct FractalDB, a database without natural images, automatically by assigning fractals based on a natural law existing in the background knowledge of the real world. [4] further demonstrates the usefulness of FractalDB in pre-training Vision Transformers (ViTs). Beyond fractal noise, [2] provides a comprehensive study on how different noise types affect representation learning.

B Data Collection

B.1 Interaction Data Collection

For training, we choose ten Gibson environments—'Crandon,' 'Delton,' 'Goffs,' 'Oyens,' 'Placida,' 'Roane,' 'Springhill,' 'Sumas,' 'Superior,' and 'Woonsocket.' We create a 20k/10k training/validation set and a 50k/10k training/validation set from sampling 40/20 and 100/20 starting locations in each of the ten environments. We also create a small 1k/1k training/validation set from sampling 20 starting locations from 'Superior' and 20 starting locations from 'Crandon' respectively. Collectively, we have created three interaction dataset $\boldsymbol{D}_{2k}$, $\boldsymbol{D}_{30k}$ and $\boldsymbol{D}_{60k}$. For our random exploration strategy, we refer the readers to [5].

B.2 PTZ training Data Collection

First, we generate a training set from similar domains to the navigation experiments. Specifically, we sample 6500 photo-realistic home images sourced from 65 Gibson environments rendered by the Habitat-Sim simulator to form the training set and 2300 home images from 23 other Gibson environments to form the test set. Notice these images are generated i.i.d. without any action labels. We refer to this dataset as $\boldsymbol{D}_{habitat}$

Second, we generate Perlin noise and fractal noise using [7]. Perlin noise is generated from 2, 4, 8 periods and fractal noise is generated from 2, 4, 8 periods and 1-5 octaves. We generate 10k Perlin noise, 10k fractal noise, and 20k random shapes to form a 40k noise dataset $\boldsymbol{D}_{all\_noise}$. However, this particular composition of noise is rather wishful as we do not know yet which one is the best for PTZ encoder training. To uncover which noise is the best surrogate data source for natural home images, we also create a 40k $\boldsymbol{D}_{perlin}$, $\boldsymbol{D}_{fractal}$ and $\boldsymbol{D}_{shape}$, each containing only one kind of noise.

B.3 Noise Choice for PTZ Pre-training

We first tried training our PTZ encoder on Gaussian noise. The resulting poor performance suggests the particular choice of noise is critical. We hypothesize that patterned noise rather than high

frequency noise should be more useful as the encoder probably needs some visual cues to find relative transformations. To this end we include Perlin noise, which can be used to simulate cloud formations in the sky, and fractal noise, which can be found in nature [3], in the dataset to train the encoder. We further include random geometric shapes as they are found in man-made environments and can help the encoder learn edges and orientations. A sample of these three different kinds of random noise is shown in Fig 3. We follow the same procedure as before to sample random crops on these noise images. Using noise for pre-training completely removes the need to access an testing environment for data.