Beyond Flatland:
Pre-training with a Strong 3D Inductive Bias

Shubhaankar Gupta[1], Thomas P. O'Connell[2], Bernhard Egger[2,3]

[1]Delhi Public School RK Puram
[2]Massachusetts Institute of Technology
[3]Friedrich-Alexander-University Erlangen-Nuremberg
guptashubhankar06@gmail.com

Abstract

Pre-training on large-scale databases consisting of natural images and then fine-tuning them to fit the application at hand, or transfer-learning, is a popular strategy in computer vision. However, *Kataoka et al., 2020* [1] introduced a technique to eliminate the need for natural images in supervised deep learning by proposing a novel synthetic, formula-based method to generate 2D fractals as training corpus. Using one synthetically generated fractal for each class, they achieved transfer learning results comparable to models pre-trained on natural images. In this project, we take inspiration from their work and build on this idea– using 3D procedural object renders. Since the image formation process in the natural world is based on its 3D structure, we expect pre-training with 3D mesh renders to provide an implicit bias leading to better generalization capabilities in a transfer learning setting and that invariances to 3D rotation and illumination are easier to be learned based on 3D data. Similar to the previous work, our training corpus will be fully synthetic and derived from simple procedural strategies; we will go beyond classic data augmentation and also vary illumination and pose which are controllable in our setting and study their effect on transfer learning capabilities in context to prior work. In addition, we will compare the 2D fractal and 3D procedural object networks to human and non-human primate brain data to learn more about the 2D vs. 3D nature of biological vision.

1 Introduction

Pre-training of artificial neural networks on available large-scale datasets offers a mechanism to reduce the need for excessive training data for the application at hand – simply by transferring features from large-scale databases to datasets covering restricted domains with limited data. This allows models with small quantities of data to still fair exceptionally well in objective performance metrics. Pre-training has become increasingly popular in the computer vision community owing to a substantial rise in the availability of large-scale datasets like ImageNet [2]. Since such natural image datasets have been widely accepted and lead to strong results with natural testing data, we might assume that images most closely resembling natural objects achieve maximum performance. However, since these natural images have to be manually labeled, their production is prone to errors and extremely labor-intensive. Recently, [1] countered the notion that pre-training can only be performed on natural images and proposed pre-training on 2D fractals. For most datasets, such pre-training led to performance comparable to models pre-trained on ImageNet and for some it even exceeds their performance.

We propose a novel technique for pre-training with a strong 3D inductive bias, which we plan to pursue this by replacing 2D fractals with procedurally generated 3D meshes rendered under various illumination conditions and poses. Rendering multiple images from a single mesh permits determining the type and magnitude of the variation to be implemented on a single object. It paves the way to generate massive numbers of images per class without manual labeling of images and the potential human error accompanying it. The dataset produced will then be used in a classification setting where the training objective is to identify the correct class for a fractal or 3D object independent of deformations or camera and illumination variation. We expect the 3D nature of our generated images to lead to a different inductive bias than the 2D fractals and expect networks pre-trained on 3D objects to generalize better to complex visual tasks, including object detection.

Intermediate features in convolutional neural networks (CNNs) optimized for visual tasks bear a notable resemblance to the hierarchy of neural activity along the primate ventral visual stream, which supports object recognition [3]. Since our network will be trained using 3D variations, a thought-provoking question arises: *Do models trained on 3D renders better match neural activity along the primate ventral stream than models trained with 2D images*? We will pursue experiments to explicitly study this question, comparing the relative linear fits between 2D/3D model features and brain activity from monkeys and humans. In addition, our 3D data generation process enables generation of 2D data (by removing all pose variation) and even enables the removal of 3D cues (by removing all illumination variation), allowing us to disentangle 2D and 3D properties of the primate visual system. Giving precedent to the computer vision aspect of the research which is to achieve better transfer learning performance compared to the 2D approach, we will also study whether an inductive bias towards 3D better predicts brain activity.

Since the natural image formation process is 3D, we hypothesize that our method will learn better low- and high-level features which better generalize for visual learning tasks and also match the brain data better than the 2D fractal work of [1]. We propose the following steps to pursue this study:

1. Generate synthetic object renders based on simple algorithms by attributing few meshes to each class and rendering them under various viewpoints and illumination conditions. We will use various combinations of datasets formed from 2D and 3D data.

2. Train and evaluate artificial neural networks using 2D and 3D datasets following the protocol of [1] and focus on the transfer learning capabilities on a set of image classification tasks and measure the performance of our models.

3. Evaluate brain predictability of the networks pre-trained based on those datasets to investigate if a 3D inductive bias produces networks that better resembles brain activity.

2 Related work

2.1 Rendering and procedural meshes

The generation of synthetic data is popular for various computer vision tasks e.g. [4, 5]. The goal of 3D rendering is usually to simulate the 3D world closely and accurately using high-quality assets (e.g. in [6]). The idea of domain randomization uses 3D assets, but starts to randomize texture and illumination [7] leading to non-realistic appearance. In contrast, our approach builds on easy to generate assets based on procedural strategies. Today, fractals are the most popular procedural structures for pre-training – most, however, are 2D. Procedural 3D meshes are commonly used in the entertainment industry e.g. to generate landscapes [8].

2.2 Pre-training datasets

Conventionally, pre-training of CNNs has been performed on large-scale natural image datasets [9–11]. As noted by [12], this practice of first training a network to perform image classification on large-scale datasets (i.e. pre-training) and then transferring the extracted features for a completely new work (i.e. fine-tuning) has become the rule of thumb for solving a wide range of computer vision problems. The development of denser and deeper supervised CNN architectures [13–16] has unquestionably impacted pre-training [13]. While such models function at the mere cost of more computational resources, they have substantially amplified performance metrics since they extract a huge number of perceptive features at various hierarchies and iterations.

Most approaches rely on large-scale real world datasets for pre-training. In contrast, [1] proposed to solely train on synthetically generated images which are derived from simple rules (fractals). Their method used one formula to generate one class, and used data augmentation to increase the number of images in the category. This approach performed very well in comparison to pre-training with natural-image datasets. Our method is different from it since we use 3D procedural and morphable models to generate different 3D objects as classes and render them under different viewpoints and illumination conditions to obtain multiple images per class.

2.3 Computational models for biological vision

CNN features explain appreciable variance in monkey and human neural activity, and the layer hierarchy in CNNs maps onto the ventral stream hierarchy, which underlies primate object recognition [3]. These general findings have been replicated across species (monkeys, humans), imaging modalities (electrophysiology, fMRI, MEG, EEG), and stimulus types (objects, scenes, faces) [17, 18]. Since we assume that the primate brain's neural tuning develops to capture the 3D structure of the world, we hypothesize that pre-training with the 3D dataset will produce CNN features that better explain activity in high level brain area than pre-training with 2D datasets like [1].

3 Methodology

Our approach explores different combinations of synthetic datasets. We use FractalDB [1] as a 2D baseline dataset (Fig. 1). Two 3D datasets will be built, namely ProcSynthDB (Fig. 2) and MorphSynthDB (Fig. 3). While the FractalDB uses a 2D fractal-based approach with 2D data augmentation for image generation, our proposed two synthetic 3D datasets build on three-dimensional procedural methods to generate 3D meshes per class and renders them under a variety of viewpoints and illumination conditions. A brief overview of the dataset comparisons can be found in Table 1 We will add results of standard ImageNet trained networks (AlexNet, VGG, ResNet-50) as baseline to our comparison.
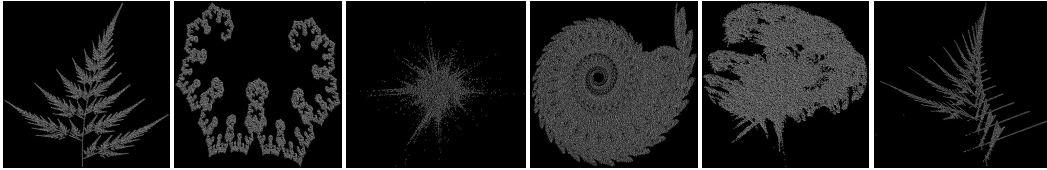

Figure 1: Fractal images from *Kataoka et al., 2020* [1]. This dataset is built using different fractal rules and image-based data augmentation. The dataset provides up to 10k different fractals (classes) and 1k images per class.

3.1 ProcSynthDB

For every class, we start off by creating a base mesh from simple building blocks (sphere, cylinder, cube etc.) and using various wireframes, smootheness, skin on them. Each base mesh should be different enough to visually classified as belonging to a separate class. Our first approach uses perspective to create alterations in renders by changing the 3D viewpoint as well as the illumination conditions it is subjected to. This enables to get multiple images per mesh with a strong 3D-based variation in appearance. In the rendering process, we also change the the size of the mesh twice across one single axis so that it produces natural-looking modifications. We describe the exact procedure to generate the meshes in pseudo-code in Algorithm 1 and examples of resulting shapes are found in Fig. 2.

3.2 MorphSynthDB

Our second 3D render dataset builds on ProcSynthDB and explores Gaussian process deformation models [19] for shape and texture variation [20]. We will start from the first 100 ProcSynthDB meshes and add random shape and color variation to them to obtain new shapes with additional variation in color. We use the implementation of [20] but with higher strength of the shape variation for the
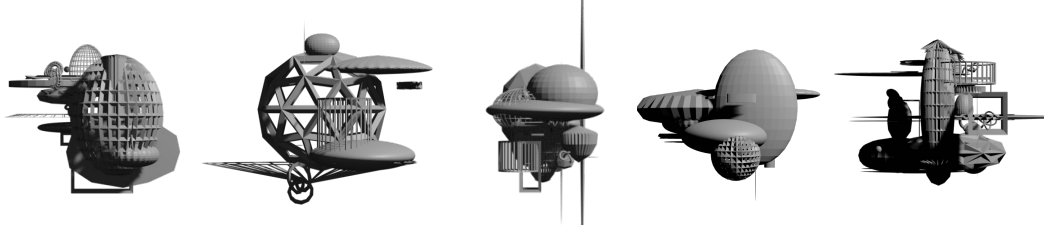
Figure 2: Example images of ProcSynthDB objects from different classes. In addition to the proceduraly created class, we change the viewpoint and illumination conditions.

Algorithm 1: Generation of ProcSynthDB Meshes

---

input : number of meshes n, $v = 3$, $w = 5$, $maxSize = 10$
output : n synthetic 3D meshes
$i \leftarrow 0$;
while $i < n$ **do**
    /* Add several primitives with random parameters and modifiers   */
    $p \leftarrow random(1, v)$;
    for $j \leftarrow 1$ **to** p **do**
        for $objectType \in cube, sphere, cone, cylinder, torus$ **do**
            for $l \leftarrow 1$ **to** *random(0,w)* **do**
                add $objecType$ with random parameters to the scene ;
                randomly modify mesh with wireframe or subdivide modifier or not ;
            end
        end
    end
    /* Quality Control to remove objects over the maximum size   */
    if $max(size_x, size_y, size_z) \leq maxSize$ **then**
        $i \leftarrow i + 1$ save mesh from all objects ;
    end
    delete all objects ;
end

---

shape Gaussian processes. In particular we changed the magnitude of the following shape and albedo parameters (original value in brackets) $b_s = 50(5), c_s = 300(3), b_a = 0.05(0.01), c_a = 0.2(0.01)$. Since this procedure is sensitive to large number of vertices we downsample the meshes before applying the transformations. Examples of resulting shapes are found in Fig. 3.
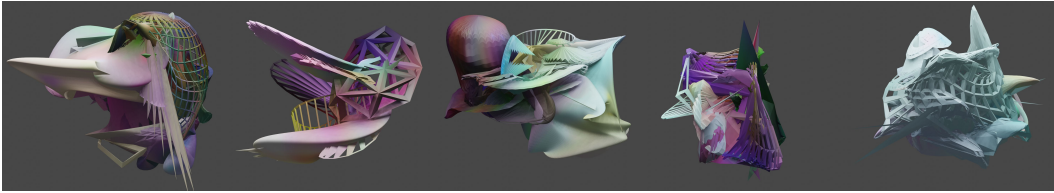

Figure 3: Example images of MorphSynthDB objects from different classes. In addition to shape variation based on Gaussian processes, this dataset also includes texture variation.

3.3 Pre-training and subsequent fine-tuning

We will use the ResNet-50 architecture [14] for pre-training our models across all dataset combinations. Our process will stay restricted to supervised multi-class classification. We will change the training phase of the model with our dataset leaving the fine-tuning step untampered, as proposed in [1]. For the sake of maintaining a uniform comparison, we will perform a hyperparameter search

4

Database	View Dimensionality	Colored	Illumination Variation
ProcSynthDB	3D	×	✓
MorphSynthDB	3D	✓	✓
FractalDB	2D	×	×

Table 1: Overview of database compositions

based on cross-validation for each set of network and data and will search around the values derived in [1].

3.4 Comparing models to brain data

To assess the similarity between features in the procedural 2D and 3D models and brain activity, we will use linear regression to predict neural responses from monkeys and humans from model features [21]. The general procedure is as follows: 1.) model features are computed for the images used to measure neural responses, 2.) using 10-fold cross-validation, partial least squares regression will be used to learn a linear mapping from model features to neural activity, and 3.) using the held-out data, the overall fit between a given model and neural responses will be quantified as the correlation between the predicted and true neural responses [3]. This final correlation between true and predicted responses constitutes the *neural predictivity* of a given model for the targeted brain region. We will also perform Representational Similarity Analysis (RSA) to measure the similarity between our computational model and brain activity.

4 Experimental protocol

4.1 Combinations of the pre-training datasets

For our experiments, we pre-train our models using the two 3D render databases we produced i.e. the ProcSynthDB and MorphSynthDB; and the third 2D baseline FractalDB. We will carry out our experiments on varied combinations achieved by incorporating these three datasets by utilizing six permutations of our them, as shown in Table 2.

Dataset combinations for pre-training	No. classes from each subdataset
FractalDB	1000
ProcSynthDB	1000
MorphSynthDB	1000
ProcSynthDB + FractalDB	500
ProcSynthDB + MorphSynthDB	500
ProcSynthDB + MorphSynthDB + FractalDB	333

Table 2: Combinations of the three datasets to be used during pre-training. We aim at a total of 1000 classes for each combination and will mix them uniformly. For each class we will generate 1000 images following the protocol of FractalDB [1].

[1] re-iterated the well-established fact that the performance achieved by pre-training models is in proportion to an increase in the number of classes and the number of objects or images per class. Therefore, for the sake of uniform recordings and to maintain a category-instance ration, we use 1000 category × 1000 instances per category for all datasets and combinations.

4.2 Transfer learning

We will pre-train ResNet-50 [14] using all dataset combinations defined in Table 2. Then, we will fine-tune the models on secondary tasks (natural image datasets, Table 3) to assess their transfer learning performance. For the transfer learning experiments we will closely follow the protocol of [1]. Better transfer performance after pre-training on any of our 3D object datasets relative to the 2D fractal datasets would support our hypothesis that pre-training with a strong 3D inductive bias will more closely capture the statistics of natural images. In addition to this quantitative evaluation we will also visualize the learned features of the resulting models similar as in [1] and compare them to features derived from ImageNet trained models.

Test datasets	CIFAR 10	CIFAR 100	ImageNet 1K	Places 365	Pascal VOC 2012	Omniglot

Table 3: We will use these datasets for our transfer learning evaluation of the models trained with the datasets in Table 2. Those testing datasets are also used in the FractalDB work [1].

4.3 Comparing models to brain data

The monkey neural responses [22] were measured for a set of 2560 images of rendered objects with multi-array electrophysiology from areas V4 (an intermediate visual area) and IT (a late-stage ventral stream area that represents high-level object properties). For this data, we predict that 2D models will better predict V4 responses, while 3D models will better predict IT responses.

The human neural responses [23] were measured for a large set of 10k+ natural images per subject with functional Magnetic Resonance Imaging (fMRI). For these analyses, we will target early (V1, V2), mid (V3, V4), and late (LO, PfS) stages of the ventral stream. We predict that the 2D models will best predict responses in early- and mid-level regions, while the 3D models will best predict responses in late regions.

4.4 Ablation study

One of the key concerns of our method could be that the observed differences could be caused by the different techniques to generate the datasets. To overcome this limitation we will generate a 3D based dataset that has all the 3D variation removed - it will not contain illumination and pose variation and will therefore be a 2D dataset based on 3D data. For this ablation study we will ablate the best performing 3D dataset and build such a 2D variant and name it FlatWorldDB.

5 Conclusion

We propose a strategy to perform supervised classification using synthetic images rendered from 3D meshes. We will investigate the influence of 3D vs. 2D inductive bias via varying the set of fully synthetic databases. Additionally, we will compare the effects of the data-generation methods on a CNN's performance after transfer learning, on brain predictivity, and their similarity to each other. The goal of this research is not to outperform ImageNet trained models but to measure relative differences resulting from 2D vs. 3D pre-training.

References

[1] Hirokatsu Kataoka, Kazushige Okayasu, Asato Matsumoto, Eisuke Yamagata, Ryosuke Yamada, Nakamasa Inoue, Akio Nakamura, and Yutaka Satoh. Pre-training without natural images. *Asian Conference on Computer Vision (ACCV)*, 2020.

[2] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

[3] Daniel LK Yamins, Ha Hong, Charles F Cadieu, Ethan A Solomon, Darren Seibert, and James J DiCarlo. Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the national academy of sciences*, 111(23):8619–8624, 2014.

[4] Chuang Gan, Jeremy Schwartz, Seth Alter, Martin Schrimpf, James Traer, Julian De Freitas, Jonas Kubilius, Abhishek Bhandwaldar, Nick Haber, Megumi Sano, et al. Threedworld: A platform for interactive multi-modal physical simulation. *arXiv preprint arXiv:2007.04954*, 2020.

[5] Adam Kortylewski, Andreas Schneider, Thomas Gerig, Bernhard Egger, Andreas Morel-Forster, and Thomas Vetter. Training deep face recognition systems with synthetic data. *arXiv preprint arXiv:1802.05891*, 2018.

[6] Erroll Wood, Tadas Baltrusaitis, Charlie Hewitt, Sebastian Dziadzio, Thomas J Cashman, and Jamie Shotton. Fake it till you make it: Face analysis in the wild using synthetic data alone. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3681–3691, 2021.

[7] Jonathan Tremblay, Aayush Prakash, David Acuna, Mark Brophy, Varun Jampani, Cem Anil, Thang To, Eric Cameracci, Shaad Boochoon, and Stan Birchfield. Training deep networks with synthetic data: Bridging the reality gap by domain randomization. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 969–977, 2018.

[8] Jacob Olsen. Realtime procedural terrain generation. 2004.

[9] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). URL `http://www.cs.toronto.edu/~kriz/cifar.html`.

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

[11] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[12] Minyoung Huh, Pulkit Agrawal, and Alexei A Efros. What makes imagenet good for transfer learning? *arXiv preprint arXiv:1608.08614*, 2016.

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, page 1097–1105, Red Hook, NY, USA, 2012. Curran Associates Inc.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[15] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[16] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[17] Daniel LK Yamins and James J DiCarlo. Using goal-driven deep learning models to understand sensory cortex. *Nature neuroscience*, 19(3):356–365, 2016.

[18] Blake A Richards, Timothy P Lillicrap, Philippe Beaudoin, Yoshua Bengio, Rafal Bogacz, Amelia Christensen, Claudia Clopath, Rui Ponte Costa, Archy de Berker, Surya Ganguli, et al. A deep learning framework for neuroscience. *Nature neuroscience*, 22(11):1761–1770, 2019.

[19] Marcel Lüthi, Thomas Gerig, Christoph Jud, and Thomas Vetter. Gaussian process morphable models. *IEEE transactions on pattern analysis and machine intelligence*, 40(8):1860–1873, 2017.

[20] Skylar Sutherland, Bernhard Egger, and Joshua Tenenbaum. Building 3d morphable models from a single scan. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2514–2524, 2021.

[21] Kendrick N Kay, Thomas Naselaris, Ryan J Prenger, and Jack L Gallant. Identifying natural images from human brain activity. *Nature*, 452(7185):352–355, 2008.

[22] Najib J. Majaj, Ha Hong, Ethan A. Solomon, and James J. DiCarlo. Simple learned weighted sums of inferior temporal neuronal firing rates accurately predict human core object recognition performance. *Journal of Neuroscience*, 35(39):13402–13418, 2015. ISSN 0270-6474. doi: 10.1523/JNEUROSCI.5181-14.2015. URL `https://www.jneurosci.org/content/35/39/13402`.

[23] Emily J. Allen, Ghislain St-Yves, Yihan Wu, Jesse L. Breedlove, Logan T. Dowdle, Brad Caron, Franco Pestilli, Ian Charest, J. Benjamin Hutchinson, Thomas Naselaris, and Kendrick Kay. A massive 7t fmri dataset to bridge cognitive and computational neuroscience. *bioRxiv*, 2021. doi: 10.1101/2021.02.22.432340. URL `https://www.biorxiv.org/content/early/2021/02/22/2021.02.22.432340`.