Synthesising Abstract Deep Computer Vision Pretraining Data

Søren Winkel Holm, s183911

December 16, 2022

Contents

1 Introduction

In the field of computer vision, deep learning is often used to solve problems such as object detection and image segmentation. Two main problems are (1) the high cost of annotating the image by producing the label maps (such as this) (2) the high cost of training the deep learning models. Automatic generation of scenes with corresponding label maps might be usable even if the scenes are abstract. Such data could be used to pretrain the deep learning model and act as an inductive bias such that less training and less data is required when actually training on the real dataset. For this to work, the generated scenes should have objects with object having some assigned class which determines some visual characteristics of the object. The scenes do not have to have anything to do with the actural task but need to make the model learn som basics of object detection, classification and segmentation. An easy first solution would be having the class determine the colour of the object but it is much more realistic if the objects are textured with noise functions and the parameters of the procedurally generated noise were determined by the class.

2 Method

2.1 Synthetic Scene Rendering

Task 1: Noise Functions and The Simple Scene In the fragment shader, two noise functions were implemented, both accepting three-dimensional vector input.

1. Improved Perlin (IP) noise using a permutation polynomial as McEwan, Sheets, Richardson, and Gustavson [McE+12].

2. Sparse convolution (SC) noise as Frisvad and Wyvill [FW07].

The implementations were largely based on reference implementations [Fri16]: Only minor code changes and commenting were performed along with using the functions to texture the scene objects with a given noise scale.

Task 2: Random Scenes

Task 3: Object Classes

Task 4: The Full Scene With Label Maps

2.2 Deep Image Segmentation

Baseline Learning Task

Addition of Synthetic Pretraining Data

3 Results

4 Discussion

1. Powerful benchmark for different methods generalization towards different difficulties: Explainability

References

[Fri16] Jeppe Frisvad. *Exploring noise functions with WebGL*. Online. Accessed: 16/12 2022. 2016. URL: https://people.compute.dtu.dk/jerf/code/noise/.

[FW07] Jeppe Revall Frisvad and Geoff Wyvill. "Fast High-Quality Noise". In: *Proceedings of the 5th International Conference on Computer Graphics and Interactive Techniques in Australia and Southeast Asia*. GRAPHITE '07. Perth, Australia: Association for Computing Machinery, 2007, pp. 243–248. ISBN: 9781595939128. DOI: 10.1145/1321261.1321305. URL: https://doi.org/10.1145/1321261.1321305.

[McE+12] Ian McEwan, David Sheets, Mark Richardson, and Stefan Gustavson. "Efficient Computational Noise in GLSL". In: *Journal of Graphics Tools* 16.2 (2012), pp. 85–94. DOI: 10.1080/2151237X.2012.649621.