Physically Embodied Deep Image Optimisation

Daniela Mihai[*†] **Jonathon Hare**[*†]

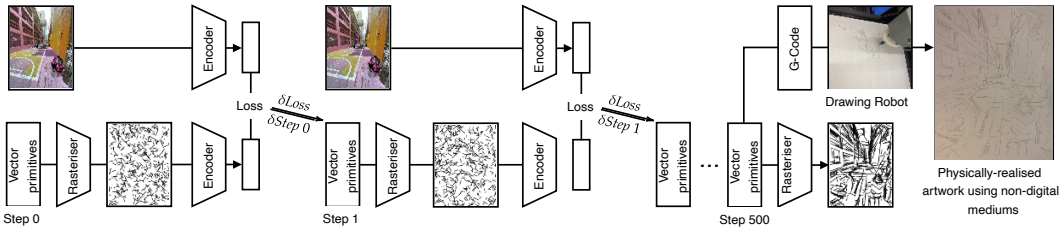


Figure 1: We create physical sketches by learning programs to control a drawing robot. A differentiable rasteriser is used to optimise sets of drawing strokes to match an input image, using deep networks to provide an encoding for which we can compute a loss. The optimised drawing primitives can then be converted in G-code commands which command a robot to draw the image using physical drawing instruments such as pens and pencils on a physical support medium.

1 Introduction

Recent progress in differentiable rasterisation of primitives (points, lines and curves) on a pixel raster allow us to build machines that turn digital raster images into continuous parametric representations and back into rasterised images again [8]. Using such an approach, we can model the physical act of drawing with a pen on paper.

This approach differs from other methods which use neural networks to generate painting-like images, such as style transfer [4] or GANs[2], which essentially map textures across pixel grids and do not model the physical process of drawing. Stroke based-rendering with some optimisation goal has been studied before the deep learning era [6]. In the field of robot art, Pix18 [7] excels at creating oil paintings with its own art subject, and with minimal human intervention.While recent advances to image generation have focused on reinforcement learning [3] or neural renderers [9, 10], the optimisation of brushstrokes against loss functions parameterised by deep networks, which can then be directly interpreted by a drawing robot has been missing from the field.

2 Optimisation of Stroke Primitive Parameters with Different Losses

With the machinery defined in Mihai and Hare [8] it is possible to define a complete system that takes the parameters describing stroke primitives and rasterises those strokes into an image. As shown in fig. 1, by introducing a (pretrained) neural network to extract features from both the complete rasterised image and a fixed target image, coupled with an appropriate loss function computed between those features, it becomes possible to compute gradients with respect to the parameters of the primitives that created the rasterised image. Minimising this loss will adapt the underlying primitives to "shapes" that maximise the feature similarity of the target image. If the rasterisation function is differentiable with respect to the stroke primitives, then the minimisation problem can be solved using gradient descent. Further, the optimised stroke parameters can be directly transformed

[*]Authors contributed equally

[†]Vision, Learning and Control Group, Electronics and Computer Science, University of Southampton, {adm1g15, jsh2}@ecs.soton.ac.uk

35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.

into instructions that control a robot (see appendix A) that can manipulate a drawing instrument like a pen or pencil over a support medium. This allows us to produce physical artefacts directly from the model.

In our experiments, we optimise the underlying primitive parameters by minimising either the MSE loss directly against the image pixels (the encoder network is the identity function) or a loss built upon features extracted from the intermediate layers of a deep convolutional network. For the latter loss which is inherently differentiable, features are extracted using a VGG16 network pretrained on StylisedImageNet dataset (SIN) [5] which has a shape bias, or the regular ImageNet dataset [1] which has been shown to be texture-biased.

3 Case Studies

We provide two case studies to demonstrate the effectiveness of our approach. Firstly we demonstrate the effects can be achieved by transforming photographs to line drawings by optimising against different intermediate representations of a deep network. Secondly, we demonstrate a variation of the idea of neural style transfer [4] where instead of generating raster images we directly create the underlying stroke information that allows an image to be drawn under different physical drawing constraints.

How does my deep network think an image should be sketched? We can use our technique to explore how different internal representations of pretrained encoder networks manifest themselves in terms of illustrations of photographs as shown in fig. 2. Interestingly this allows us to explore different artistic effects that arise, but also to qualitatively look at the variations between differing inductive biases and loss formulations.

Figure 2: Illustrations created from VGG-16 networks with SIN weights at a range of different depths (early layers left, later layers right). Each image was created using 1000 straight lines.

What if Edvard Munch's 'The Scream' was painted in the Pointillism technique? Or Cubism? Figure 3 showcases the results of optimising different primitives to fit a photo of the original "The Scream" by Edvard Munch [3]. We display results by optimising 2000 uniformly coloured points (figs. 3b and 3d) and 1000 coloured lines (figs. 3c and 3e) to fit the original image by minimising either an MSE or a perceptual loss (with SIN-pretrained VGG16). The contrast between the images with the same type of primitive parametrisation, but using a different loss, is striking. The perceptual loss captures the shape information rather well, while moving away from the colour or texture scheme of the original or the variant realised with MSE loss. The copies thus produced could be tagged as belonging to the Fauvism art movement. Changing the parametrisation of the drawings in the method described in section 2 gives us an idea of what the painting would've looked like if it were to have been drawn in a Pointillist style (figs. 3b and 3d), or a more abstract, Cubism-like style (figs. 3c and 3e), while at the same time, making it possible for the drawing to be physically produced on paper by a drawing agent. More examples can be found in appendix B.

(a) 'The Scream' (b) 2000 Points (c) 1000 Lines (d) 2000 Points (e) 1000 lines

Figure 3: Munch's 'The Scream'(fig. 3a) reduced to points and straight lines by gradient decent using LPIPS loss with SIN-pretrained VGG16 (figs. 3b and 3c) and MSE loss (figs. 3d and 3e).

[3]Photo taken from Wikipedia `https://en.wikipedia.org/wiki/The_Scream`

References

[1] 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. Ieee, 2009.

[2] Ahmed Elgammal, Bingchen Liu, Mohamed Elhoseiny, and Marian Mazzone. Can: Creative adversarial networks, generating" art" by learning about styles and deviating from style norms. *arXiv preprint arXiv:1706.07068*, 2017.

[3] Yaroslav Ganin, Tejas Kulkarni, Igor Babuschkin, SM Ali Eslami, and Oriol Vinyals. Synthesizing programs for images using reinforced adversarial learning. In *International Conference on Machine Learning*, pages 1666–1675. PMLR, 2018.

[4] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.

[5] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.

[6] Aaron Hertzmann. A survey of stroke-based rendering. Institute of Electrical and Electronics Engineers, 2003.

[7] Hod Lipson. Pix18. URL `http://www.pix18.com/`.

[8] Daniela Mihai and Jonathon Hare. Differentiable drawing and sketching. *arXiv preprint arXiv:2103.16194*, 2021.

[9] Reiichiro Nakano. Neural painters: A learned differentiable constraint for generating brushstroke paintings. *CoRR*, abs/1904.08410, 2019. URL `http://arxiv.org/abs/1904.08410`.

[10] Ningyuan Zheng, Yifan Jiang, and Dingjiang Huang. Strokenet: A neural painting environment. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=HJxwDiActX`.

A Our drawing robot

For producing our artwork we used a custom modified gantry-style robot to manipulate a pen or pencil over a paper support medium as illustrated in fig. 4. We used `vpype` (`https://github.com/abey79/vpype`) to optimise the stroke order produced by our network to minimise unnecessary movements between strokes, and `juicy-gcode` (`https://hackage.haskell.org/package/juicy-gcode`) to convert the strokes into gcode instructions that could be performed by the drawing robot.

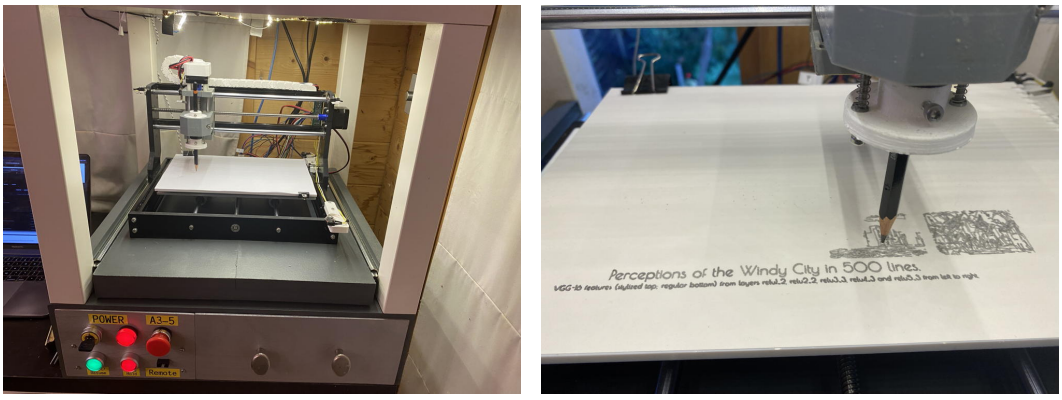
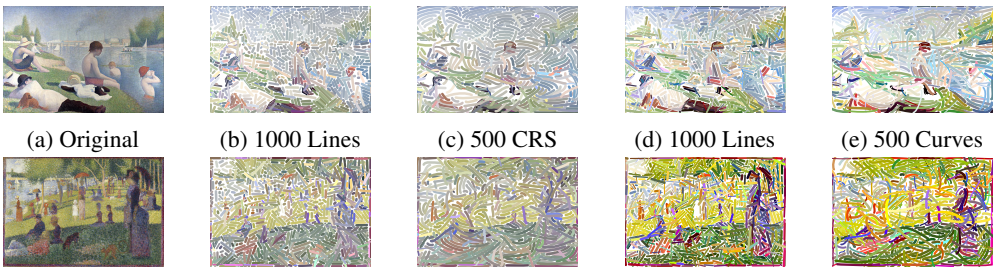


Figure 4: Photos of our drawing robot in action.

B More Examples of Image Optimisation

In this section, we provide more examples of our approach for image-based optimisation. Figure 5 shows results of optimising 1000 lines and 500 curves parameterised as Catmull-Rom splines against two of Seurat's paintings, 'Une Baignade at Asni'eres' and 'A Sunday Afternoon on the Island of La Grande Jatte'. This also shows the differences in optimising with a perceptual loss based on features extracted with a VGG network pretrained on SIN against a simple reconstruction loss.



(a) Original (b) 1000 Lines (c) 500 CRS (d) 1000 Lines (e) 500 Curves

(f) Original (g) 1000 Lines (h) 500 CRS (i) 1000 Lines (j) 500 Curves

Figure 5: Photographs of Seurat's 'Une Baignade, Asni'eres' (taken from `https://en.wikipedia.org/wiki/Bathers_at_Asnieres`) and 'A Sunday Afternoon on the Island of La Grande Jatte' (`https://en.wikipedia.org/wiki/A_Sunday_Afternoon_on_the_Island_of_La_Grande_Jatte`) reduced to straight lines or Catmull-Rom splines by optimising with an MSE loss (figs. 5b, 5c, 5g and 5h) or a perceptual loss using features extracted by a SIN-pretrained VGG16 (figs. 5d, 5e, 5i and 5j).

Similarly, fig. 6 shows a comparison between the image optimisation results using different primitive parametrisations by minimising a perceptual loss.

(a) Original (b) 500 Curves (c) 1000 Lines (d) 2000 Points

Figure 6: Van Gogh's 'The Starry Night' (taken from `https://en.wikipedia.org/wiki/The_Starry_Night`) reduced to curves, lines and points by optimisation using a perceptual loss with SIN-pretrained VGG16.