Neural Space-filling Curves

Hanyu Wang, Kamal Gupta, Larry Davis, Abhinav Shrivastava
University of Maryland, College Park

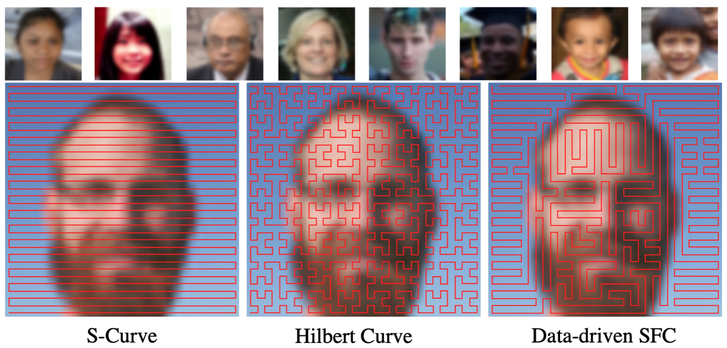
Project page: https://hywang66.github.io/publication/neuralsfc

Space-filling Curves

- A space-filling curve (SFC) is a continuous scan order that traverses all spatial locations in two or higher dimensional signals exactly once.
- Examples:
  - Peano Curve, Hilbert Curve, S-Curve, etc.
- Applications:
  - Compression, Security, Database, Parallel Computing, Generative Modeling, etc.

Context-based Space-filling Curves

- A context-based space filling curve is a kind of data-driven SFC that traverses an image in a spatially coherent order.
- Context-based space-filling curves are image-dependent, i.e., they are computed for specific image(s).
- Computed by the cover and merge algorithm. Weights are defined according to the difference of adjacent pixels.
- Limitations
  - The receptive field of edge weights is limited, so the resulting SFC does not take into account the long-range context in the image.
  - It only works for one image at a time, i.e., it cannot be applied to a set of images.
  - The context-based SFC obtained is closely tied to lag-2 autocorrelation.



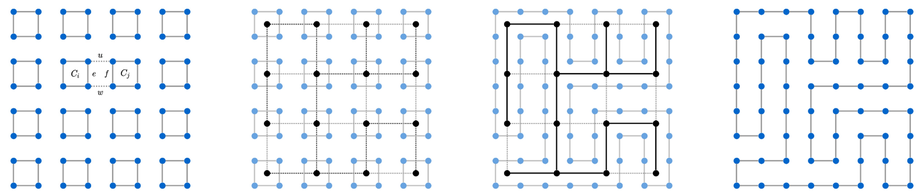
S-Curve | Hilbert Curve | Data-driven SFC

Contributions

- We propose a data-driven technique to find optimal SFCs for a set of images. We postulate that context-based SFCs are more suitable for linearizing a group of images (or a short video/gif), since the cost of storing the SFC itself can be amortized by the number of images.
- We devise a novel alternating minimization technique to train an SFC weights generator, which allows us to optimize for any given objective function, even when not differentiable.

Background: Cover and Merge Algorithm

Dafner et al. use “Cover and Merge” to discover an image-based context-based SFC.



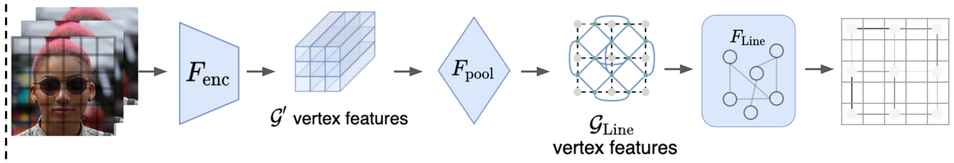
(a) Cover $\mathcal{G}$ (b) Dual $\mathcal{G}'$ (c) $\mathcal{T}$: MST of $\mathcal{G}'$ (d) Merged

(a)Given an image-grid graph G, construct dual graph G’ of disjoint 2x2 circuits.
(b)The dual graph G′ (black) built on the covering circuits G (blue).
(c)Based on the edge weights of G’, generate the Minimum Spanning Tree T (black solid lines) of G ′ (all black lines) and the Hamiltonian Circuit (blue).
(d)A Hamiltonian circuit merged from circuits. Cut it at any edge to form an SFC.

Neural SFC Model

Given a dataset of images, Neural SFC provides a data-driven way to compute SFC weights, resulting in an SFC for a set of images.

Weight Generator. We propose the Weight Generator, which is designed to take as input a single image I, and output the edge weights of input image’s dual graph G’.



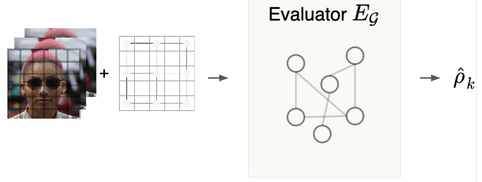
These weights are averaged over the batch during training, and are averaged over the entire image set at test time to induce the SFC for this set.

Objective Functions. We use two different objectives to optimize our model.

- **Autocorrelation.** 1D autocorrelation measures the internal local similarity of a 1D sequence. The smaller the 1D autocorrelation is, the better the SFC is.
- **Code Length.** This is the LZW sequence length. It’s inspired from the Lempel-Ziv Welch lossless compression algorithm.

Computing both objectives require us to first obtain a minimum spanning tree to infer the SFC, which is a non-differentiable operation.

Weight Evaluator. To overcome the problem of non-differentiability of SFC computation, we introduce the Weight Evaluator, which takes both the image and SFC weights as input and estimates the used objective value.



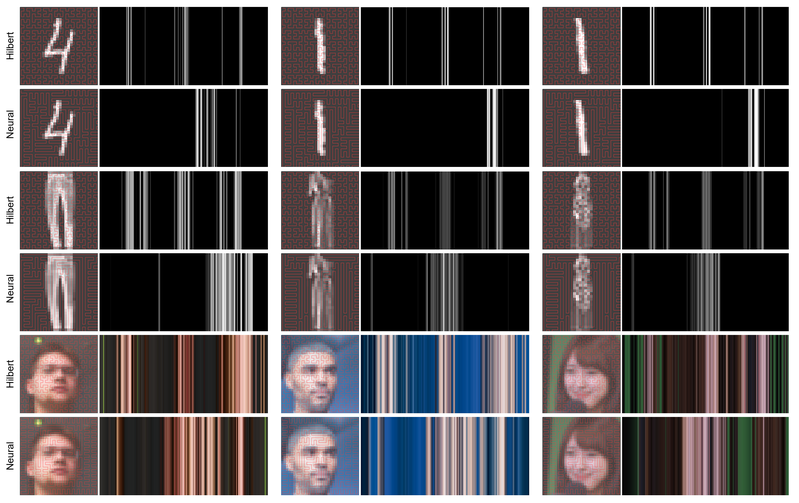
$$\Phi = -\rho_k \quad \text{or} \quad \Phi = L$$

$$\hat{\Phi} = \mathrm{E}_{\mathcal{G}}(\overline{\mathrm{W}}_{\mathcal{G}'}, I),$$

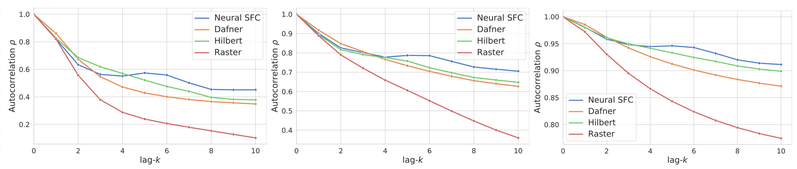
$$\mathcal{L}_{\mathrm{E}} = \mathbb{E}\left[\|\Phi - \hat{\Phi}\|\right],$$

Experiments

Qualitative Evaluation. Left: SFC (in red color) overlayed on the image. Right: Image flattened according to the SFC and visualized in 1-dimension.



Autocorrelation Curves. lag-k autocorrelation for MNIST, Fashion-MNIST and FFHQ datasets. While Dafner et al. provide higher autocorrelation for small lag, i.e., from k = 2 to k = 4, Neural SFCs outperform Dafner et al. for k > 4 in all the datasets.



(a) MNIST (b) Fashion-MNIST (c) FFHQ 32×32

Quantitative Comparison. Performance of lag-k autocorrelation and LZW Encoding length for different orders. ρ_k is lag-k autocorrelation.

Dataset	Method	$\rho_6 \uparrow$	$\rho_{10} \uparrow$	Size in bytes (Δ) $\downarrow$
MNIST	Raster	0.206	0.102	175.4
	Hilbert	0.475	0.378	182.7 (+7.3)
	Dafner [5]	0.401	0.348	-
	Neural SFC (Ours)	**0.558**	**0.451**	**171.1 (-4.3)**
FMNIST	Raster	0.552	0.360	425.8
	Hilbert	0.7 23	0.647	427.3(+1.5)
	Dafner [5]	0.704	0.627	-
	Neural SFC (Ours)	**0.786**	**0.705**	**412.4 (-13.4)**
FFHQ	Raster	0.824	0.775	688.0
	Hilbert	0.924	0.899	689.6(+1.6)
	Dafner [5]	0.901	0.871	-
	Neural SFC (Ours)	**0.943**	**0.911**	**678.3 (-9.7)**
TGIF	Raster	-	-	563.9
	Hilbert	-	-	567.0 (+3.1)
	Neural SFC (Ours)	-	-	**556.9 (-7.0)**