Fast Inpainting-based Compression: Combining Shepard Interpolation with Joint Inpainting and Prediction

Pascal Peter, Mathematical Image Analysis Group, Saarland University, Saarbrücken, Germany, peter@mia.uni-saarland.de

Advanced Grant
no. 741215

Motivation

Inpainting-based Compression:

- store sparse pixel mask, decode by interpolation

+ competitive quality on piece-wise smooth images

– sophisticated but slow inpainting and entropy coding

How far can we go with simple and fast ingredients?

Joint Inpainting and Prediction (JIP)

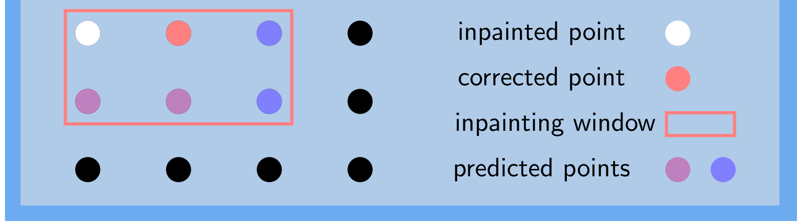
Shepard Interpolation: Reconstruct image $\boldsymbol{u}$ on domain Ω from known data points $\boldsymbol{f}$ on $K \subset \Omega$ by normalised convolution.

$$u_{i,j} = \frac{\sum_{(k,\ell)^\top \in K} w((k,\ell)^\top,(i,j)^\top) f_{k,\ell}}{\sum_{(k,\ell)^\top \in K} w((k,\ell)^\top,(i,j)^\top)}.$$

Choice of weights w: Gaussian kernel.

Combining Inpainting and Prediction:

- Iterate over known data points (in K).
- Predict known data points in inpainting window.
- Correct predictions with stored prediction errors.

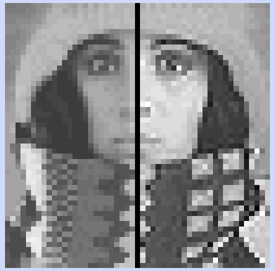


Compression with JIP

Step 1: Sample Quantisation

- uniform mapping to coarse quantised range $\{0, \dots, q-1\}$

Step 2: Fast Tonal Optimisation



- random walk over K
- try to improve grey value $u_{i,j}^{\text{old}}$
- find better (quantised) $u_{i,j}^{\text{new}}$
- update old inpainting $v_{i,j}$
- localise error computation to inpainting window $W_{i,j}$:

$$\sum_{(k,\ell)^\top \in W_{i,j}} \left(u_{k,\ell} - \frac{v_{k,\ell} + K_{k-i,\ell-j}(u_{i,j}^{\text{new}} - u_{i,j}^{\text{old}})}{w_{i,j}} \right)^2 .$$

Step 3: Joint inpainting and Prediction

Step 4: Finite State Entropy Encoding (Collet 2012)

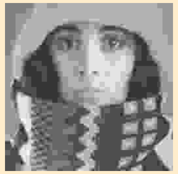
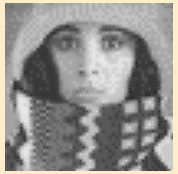
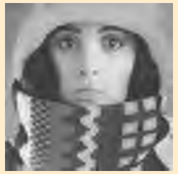
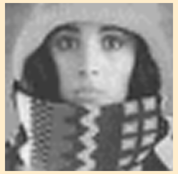
Decompression with JIP

Step 1: Entropy Decoding
Recover prediction errors with FSE.

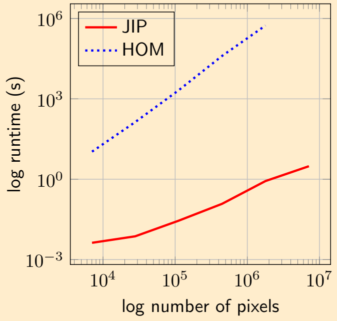
Step 2: JIP Reconstruction
Recover missing image data iteratively with joint inpainting and prediction.

Experiments

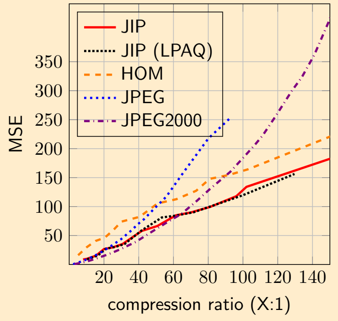
- better quality/runtime than homogeneous diffusion (HOM)
- Outperforms JPEG/JPEG200 for high compression ratios.

JPEG	HOM	JPEG 2000	JIP
			
MSE 160.57 SSIM 0.738	MSE 127.38 SSIM 0.751	MSE 109.906 SSIM 0.809	MSE 90.01 SSIM 0.827

Runtime Comparison



Quality Comparison



Conclusions and Future Work

- JIP: simple and fast alternative to classical inpainting codecs
- Next step: Optimise location of known data.