High-Resolution Image Inpainting using Multi-Scale Neural Patch Synthesis

{Chao "Harry" Yang, Xin Lu, Zhe Lin, Eli Schechtman, Oliver Wang and Hao Li}
*University of Southern California
†Adobe Research

Bad Things Happen

Imaging you would like to edit a photo after breaking up, or restore an old picture from damages, we designed a MULTI-SCALE DEEP LEARNING algorithm to help you! Our approach

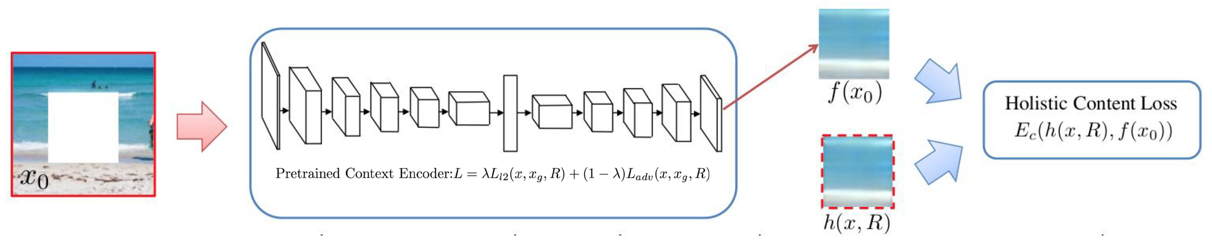
- proposed a joint optimization framework that can hallucinates missing image regions by modeling a global content constraint and local texture constraint with convolutional neural networks.
- further introduced a multi-scale neural patch synthesis algorithm for high-resolution image inpainting based on the joint optimization framework.

The Algorithm

High-Resolution Hole Filling with Multi-Scale Neural Patch Synthesis

Input: Image x,
the content network f,
the texture network t,
the number of scales N
1: Downsize x to 128×128.
2: Compute the initial content reference x^1. by giving x as input to f.
3: **for** $s \in [1, 2, \ldots, N]$:
4. Initialize $\tilde{x} = x^s$.
4: Update $\tilde{x}$ that minimizes the joint loss: $L = L_{content} + L_{texture} + L_{tv-smoothness}$.
5: Compute x^{s+1} by up-sampling $\tilde{x}$.
7: **end for**
8: Return $\tilde{x}^N$.

The Content Network

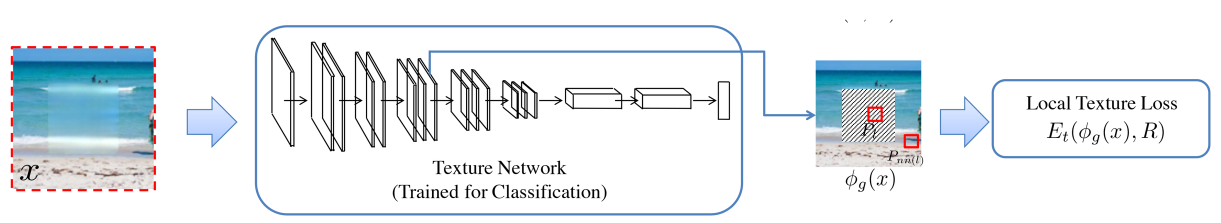


Context Encoder Predicts the Low-Res Content

The content constraint:

$$E_c(h(x,R), h(x_i,R)) = \| h(x,R) - h(x_i,R) \|_2^2$$

The Texture Network



Pre-trained VGG Optimizes the High-Res Texture

The texture constraint:

$$E_t(\phi_t(x), R) = \frac{1}{|R^\phi|} \sum_{i \in R^\phi} \| h(\phi_t(x), P_i) - h(\phi_t(x), P_{nn(i)}) \|_2^2$$

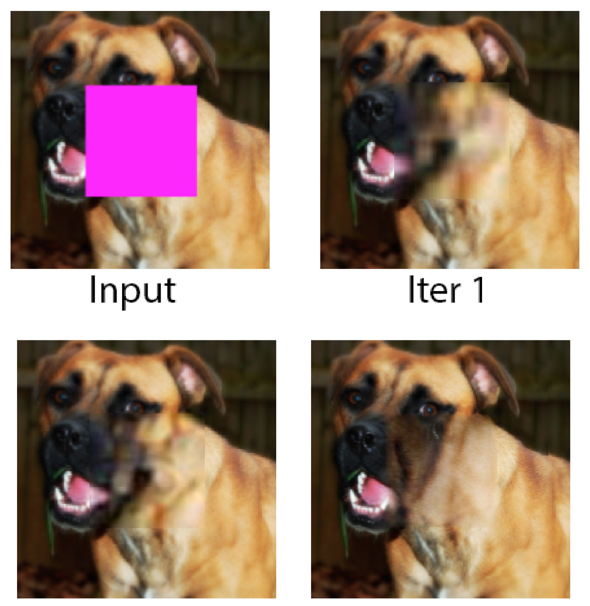
The Joint Loss Function

At each iteration, we minimize:

$$\tilde{x}_{i+1} = \arg\min_x E_c(h(x,R), h(x_i,R)) + \alpha E_t(\phi_t(x), R^\phi) + \beta \Upsilon(x)$$

Multi-scale Optimization

We optimize at three scales: 128, 256 and 512:



Changing the Texture Weight α

The weight α measures the contribution of the texture constraint relative to the content constraint. It is a trade off between the sharpness of the texture and coherence of the structure:

(a) Input image (b) $\alpha = 1e-6$ (c) $\alpha = 1e-5$ (d) $\alpha = 4e-5$

Dropping the Content Constraint

Without using the content term to guide the optimization, the structure of the inpainting results is completely incorrect, although they are visually sharp:

Input Output without CN Output with CN

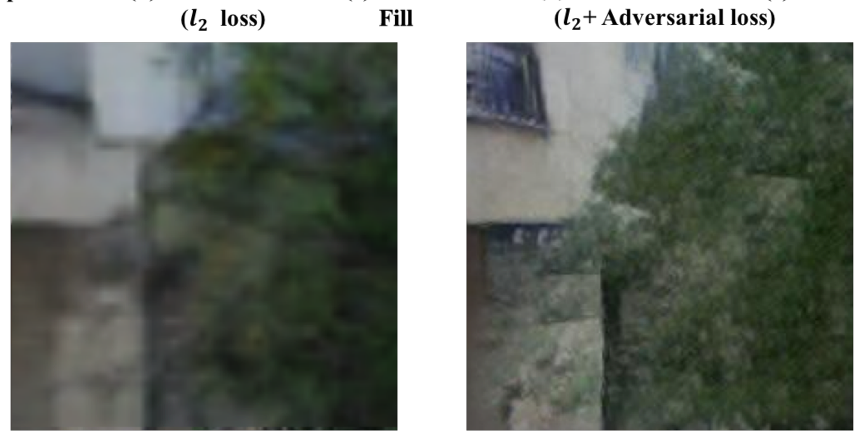
Dropping the Adversarial Loss

When the initial prediction is blurry (using ℓ_2 loss only), the final result becomes blurrier as well comparing with using the content network trained with both ℓ_2 and adversarial loss.

(a) CN with ℓ_2 loss (b) (a)'s final output (c) CN with ℓ_2 & Adv loss (d) (c)'s final output.

Overall Comparison with Other Methods

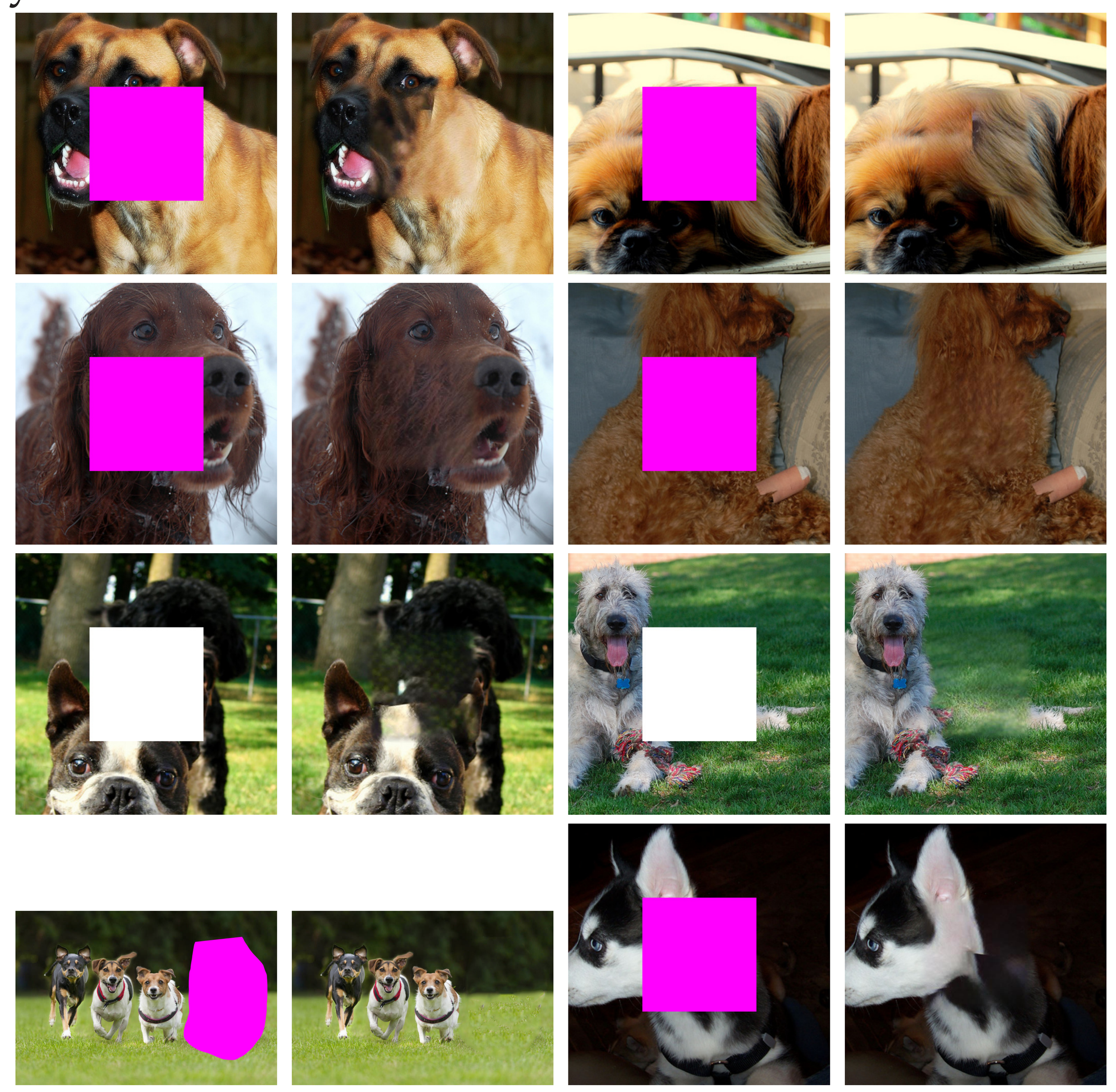
(a) Input hole (b) Context Encoder (ℓ_2 loss) (c) Content-Aware Fill (d) Context Encoder (ℓ_2+ Adversarial loss) (e) Our Method



(d) Zoom-in (e) Zoom-in

We Love Cats and Dogs

And we have collected so many dogs and cats for you:



And What Inpaints Paris Like Paris?

For Paper, Results, Code and More