Photography Made Easy

Sylvain Paris, Adobe
After retouching
After retouching
Printing photos used to be hard
Now with computers, it is a lot easier.

But it is still hard... It does not have to be this way.
Many photos can become great after retouching.

I want to make it easy.
How to Make Photo Retouching Easier

- Some people know how to do it: photographers and artists
- There are plenty of examples of good photos available on the web
- Our strategy
  1. Develop algorithms that transfer the statistics of good photos.
  2. Learn how to retouch from examples of good photos
Photographic Style Transfer

SIGGRAPH 2006

Make this photo look like
Tonal Aspects of Look: Global Contrast

Ansel Adams

High global contrast

Low global contrast

Kenro Izu
Tonal Aspects of Look: Local Contrast

Variable amount of texture  Texture everywhere

Ansel Adams  Kenro Izu
Pipeline

Input Image

Decompose

Global contrast

Histogram transfer

Recombine

Soft focus

Toning

Grain

“Histogram transfer”

Local contrast

Result
Naïve Decomposition: Low vs. High Frequency

• Problem: introduce blur & halos

Low frequency
   *Global contrast*

High frequency
   *Local contrast*
Edge-Preserving Decomposition: Bilateral Filter

Bilateral filter output

Global contrast

Bilateral filter residual

Local contrast
Photoshop Demo
Local Laplacian Filters

A Better Decomposition

SIGGRAPH 2011
Background on Gaussian Pyramids

- Resolution halved at each level using Gaussian kernel
Background on Laplacian Pyramids

- Difference between adjacent Gaussian levels
Pros and Cons of Pyramids

😊 Useful for compression [Burt 83],
texture synthesis [Heeger 95],
harmonization [Sunkavalli 10]...

😊 Believed to be unsuitable for edge-aware processing
  – Use isotropic & spatially invariant kernels
  – But edges are anisotropic & well located
  – “Manipulating pyramids generate halos”
  ▶ We show otherwise.
Our Contributions

• **Edge-aware editing with Laplacian pyramids**
  – We use a classical multi-scale representation

• **Robustness: strong effects, no artifacts**
  – We achieve extreme enhancements where other methods fail
Our Strategy: Local Adaptation

1. Generate an image that looks good for a small neighborhood
   – It may look bad elsewhere

2. “Combine data from all neighborhoods”
Example: Local Contrast Increase

input
Example: Local Contrast Increase
Example: Local Contrast Increase

input -> output with local contrast increased
Simple Local S-shaped Curve
Only Local Result Matters

- Artifacts appear elsewhere
- Not a problem, we use only local data

The processed image only needs to look good locally

output with local contrast increased
Naïve Stitching

- Paste local results side by side?
  - Visible seams, artifacts...
  - Possible heuristics: vary patch size, smooth seams...
  - But \textit{ad hoc} and brittle
Our Multi-scale Approach

• We build the Laplacian pyramid of the output coefficient by coefficient

For each coefficient
1. Generate “locally good” image
2. Compute Laplacian pyramid of that image
3. Copy coefficient to output pyramid
Illustration

output
Laplacian pyramid

level 0

level 1

level 2

input image
Illustration

Output
Laplacian pyramid

level 0
level 1
level 2

input image
Illustration

Output
Laplacian pyramid

level 2

level 1

level 0

input image

“locally good” image
Illustration

Input image → “locally good” image → partial pyramid → Laplacian pyramid

Output:
- Level 2
- Level 1
- Level 0

“Locally good” image
Illustration

Output
Laplacian pyramid

Level 0

Level 1

Level 2

Input image

“Locally good” image

Partial pyramid

Copy
Illustration

Output
Laplacian pyramid

level 0
level 1
level 2

input image
Illustration

Input image

"locally good" image

Partial pyramid

Output

Laplacian pyramid

Level 0

Level 1

Level 2

Copy
Illustration

Output
Laplacian pyramid

level 2

level 1

level 0

Copy

input image

“locally good” image

partial pyramid
Possible Nonlinearities

• Detail manipulation
Possible Nonlinearities

- Detail manipulation
Possible Nonlinearities

- Dynamic range manipulation
Possible Nonlinearities

- Can be combined, e.g. tone map + boost details
Photoshop Demo
Back to Photo Style Transfer

Transactions on Graphics 2014
New Transfer Algorithm
old algorithm

new algorithm

a lot more details
higher quality
Takeaway message #1

Photo style transfer works.

Takeaway message #2

Low-level aspects matter.
Learning from a Dataset
5000 reference photos adjusted by a pro
5000 reference photos adjusted by a pro

our result
Result

input photo

our version

photographer’s version
Result

input photo

our version

photographer’s version
Product Impact

• Transferred into Photoshop CS6 as “Auto” in Brightness & Contrast, Levels, Curves, and Auto Tone.

• Can also learn from users on the fly, transferred as Auto Smart Tone into Photoshop Elements 12.
Advantages of Gaussian Processes

• Can learn from a few hundred examples “only”
  – Hours of work by a professional → quality training data

• Revert to the mean when “unsure”
  – Does not ruin the image

• Easy to debug by looking at how training data are used
Back to Transferring from an Example
Previous Techniques Fail on Portraits

- Two-scale model not sufficient for skin texture
- Image-level statistics too coarse for portraits
Make this portrait look like this one
A Local and Multi-scale Model
A Local and Multi-scale Model

1. Construct Laplacian stacks for the input and the example
A Local and Multi-scale Model

1. Construct Laplacian stacks for the input and the example

Input

Target

2. Local match at each scale
A Local and Multi-scale Model

1. Construct Laplacian stacks for the input and the example

2. Local match at each scale

3. Collapse the matched stacks to create the output of this step

Input

Target

Output
Result (Plato)

input

output

example
Result (Kelly Castro)

input

output

example
Result (Martin Schoeller)

input

output

example
Input sequence with extreme facial expressions

Our style transfer result using the example in the gray box
Transferring Other Properties

Time-of-Day Hallucination

SIGGRAPH Asia 2013
Time-lapse Videos as Examples

- A database of 400+ time-lapse videos
- Shows how scenes change during the day
Our result
Our result (golden hour)
Deep Photo Style Transfer
A First Step Toward a Unified Approach

To appear at CVPR 2017
Motivation

• A Neural Algorithm of Artistic Style
  [Gatys et al. 2016]

• Several apps and websites
  • Prisma
  • DeepArt.io
The neural style algorithm works well for paintings.

What about photos?
So we tried...

Input

Model

Output of Neural Style Transfer
Two Problems to Solve

• Undesired distortion
  • Result still looks like a painting, not a photo

• Semantic mismatch
  • E.g., ground texture can appear on the sky
Background on Neural Style Transfer

1. Decompose the input and model images using a neural network
Background on Neural Style Transfer

1. Decompose the input and model images using a neural network

2. Impose some of the statistics of the model image
Background on Neural Style Transfer

1. Decompose the input and model images using a neural network

2. Impose some of the statistics of the model image

3. Reconstruct the output image
Background on Neural Style Transfer

1. Decompose the input photo using a neural network

2. Impose some of the statistics of the model image

3. Reconstruct the output image

Preventing distortion
Preventing Distortion

- We tried many options
  - Constraining the gradients
  - Multi-scale constraints akin to Portrait Style Transfer
  - Band-limiting the transformation
- All helped but none was 100% successful

- What worked was forcing the color transformation to be locally affine
  \[ [\text{Levin et al. 2006}] \]

\[
\begin{pmatrix}
  r_{out} \\
  g_{out} \\
  b_{out}
\end{pmatrix}
= A_{3 \times 3}
\begin{pmatrix}
  r_{in} \\
  g_{in} \\
  b_{in}
\end{pmatrix}
+ B_{3 \times 1}
\]
Output of Neural Style Transfer

Neural Style Transfer + Locally Affine Transformation
Background on Neural Style Transfer

1. Decompose the input photo using a neural network

2. Impose some of the statistics of the model image

3. Reconstruct the output image
Original Algorithm and Its Limitation

• Transfer the Gram matrix of the model to the input

\[ G_{ij}^{\ell} = \sum_k F_{ik}^{\ell} F_{jk}^{\ell} \]

\( \ell \)  layer index in the network
\( i, j \)  map index in the layer
\( k \)  pixel index in the map
Original Algorithm and Its Limitation

• Transfer the Gram matrix of the model to the input

\[ G_{ij}^\ell = \sum_k F_{ik}^\ell F_{jk}^\ell \]

\( G \) does not depend on the pixel position

\( \ell \) layer index in the network
\( i, j \) map index in the layer
\( k \) pixel index in the map
Original Algorithm and Its Limitation

• Transfer the Gram matrix of the model to the input

\[
G_{ij}^\ell = \sum_k F_{ik}^\ell F_{jk}^\ell
\]

*G does not depend on the pixel position*

• Property: the Gram matrix defines the vectors up to an isometry
  • “the neural responses can move but their distribution remains the same”
  • E.g., the G matrix implicitly encodes the “quantity of sky” in a photo.
Different Horizon Lines
Our Strategy to Ensure Consistent Matching

1. Run semantic segmentation on input and model [Chen et al. 2016]
   • Merge “visually equivalent” categories, e.g., sea and lake
   • Constrain the input labels to be a subset of the model labels

2. Transfer statistics within each category, e.g., sky to sky
Input

Model

Input segmentation

Model segmentation
Output of Neural Style Transfer + Locally Affine Transformation
Our result: Neural Style Transfer + Locally Affine Transformation + Semantic Matching
Our result: Neural Style Transfer + Locally Affine Transformation + Semantic Matching
Our result
Color Histogram Transfer
Input

Our result

Model

Portrait Style Transfer
Photorealism scores

- Histogram transfer (Pitié et al.): 3.14 ± 0.38
- Neural Style (Gatys et al.): 1.46 ± 0.30
- CNNMRF (Li et al.): 1.43 ± 0.37
- Our algorithm: 2.71 ± 0.20

Not photorealistic

Photorealistic
Histogram transfer (Pitié et al.)

Neural Style (Gatys et al.)

CNNMRF (Li et al.)

our algorithm

Photorealism scores

not photorealistic    photorealistic

1 2 3 4

Histogram transfer (Pitié et al.)

Statistics transfer (Reinhard et al.)

Photoshop Match Color

our algorithm

Style faithfulness preference

5.8%

4.6%

2.9%

86.8%

Disclaimer: specialized algorithms like to perform as well or better.
Required Steps for Productization

• Faster performance
  • Research prototype is too slow, on the order of minutes per photo
  • Feed-forward network, GPU, cloud...

• Smaller memory footprint
  • Currently based on VGG-19
  • Shallow network [Ustyuzhaninov et al., 2016]
What’s next?
Deep Photo Style Transfer is only a First Step

• Vast algorithm design space to explore

• Many questions to answer
  • Why a network trained for classification? Can we do better?
  • “Spillovers are obvious”, can we fix them without a full semantic analysis?
  • Where does the distortion come from?

Background on Neural Style Transfer

1. Decompose the input and model images using a neural network
2. Impose some of the statistics of the model image
3. Reconstruct the output image
Learning Creative Edits from Experts

- Challenge #1: experts cannot create 1 million+ photos for training a CNN
- Challenge #2: the range of effects is large
- Challenge #3: not all styles apply to all photos
Exploiting On-line Collections

• What can we do with 1 million or more photos?

• How do we find the “good ones” and exploit them?

• Working with sets instead of samples à la CycleGAN or Exposure is promising but still needs work
  [Zhu et al. 2017, Zhang et al. 2018]
A Great Photo from Every Shot
Better Photos, Easily
Photo Style Transfer

• A simple and powerful way to edit photos
• Challenges us to better understand images
  • What is style? What is content?
  • What makes an image look like it does?
  • How to preserve the image structure?
• What I did not show: videos, panoramas, weather, lighting, perception, performance...
Thanks!

Nothing would have been possible without my collaborators: Mathieu Aubry, Soonmin Bae, Kavita Bala, Connelly Barnes, Adrien Bousseau, Vladimir Bychkovsky, Eric Chan, Jiawen Chen, Jeff Chien, George Drettakis, Frédo Durand, Bill Freeman, Sam Hasinoff, Jan Kautz, Fujun Luan, Eli Shechtman, and Yichang Shih