Deep Bilateral Learning
for Real-Time Image Enhancement

M. Gharbi\textsuperscript{1}, J. Chen\textsuperscript{2}, J. T. Barron\textsuperscript{2}, S.W. Hasinoff\textsuperscript{2}, F. Durand\textsuperscript{1,3}
Speeding up high-res image operators

- High-resolution images: pressure on the pipeline
- Difficult to optimize for speed (mobile)
  - algorithm design, implementation
Scope: photographic enhancements

- No spatial warping
- No addition/deletion of content
Goal: reproduce a reference operator, faster

- Fast evaluation (especially on mobile)
- Handle high-frequency, edge-preserving effects
  - model many photographic transformations
Goal: reproduce a reference operator, faster

• Fast evaluation (especially on mobile)

• Handle high-frequency, edge-preserving effects
  – model many photographic transformations

• Treat operator as a black box
  – can be implicitly-defined
  – e.g. manually specified retouches, reverse engineering
Goal: reproduce a reference operator, faster

- Fast evaluation (especially on mobile)
- Handle high-frequency, edge-preserving effects
  - model many photographic transformations
- Treat operator as a black box
  - can be implicitly-defined
  - e.g. manually specified retouches, reverse engineering
- Capture relevant image semantics
  - content-dependent effects
Standard approaches to performance

• Algorithmic changes
  – Recursive filters [Huang 1979, Deriche 1993]
  – Convolution pyramids [Farbman 2011]
  – Permutohedral Lattice [Adams 2010]

• Systems: optimize the code
  – Halide [Ragan-Kelley 2012, Mallipudi 2016]
  – Darkroom [Hegarty 2014]
  – Compiling recursive filters [Chaurasia 2015]
Related work: operator approximation

- Joint bilateral upsampling [Kopf 2007]
- Guided filter [He 2013]
- Transform Recipes [Gharbi 2015]
- Bilateral Guided Upsampling [Chen 2016]
Related work: operator approximation

- Joint bilateral upsampling [Kopf 2007]
- Guided filter [He 2013]
- Transform Recipes [Gharbi 2015]
- Bilateral Guided Upsampling [Chen 2016]

➡ Still requires the operator at runtime!
Deep Learning: opportunities, challenges

• Rich semantic features, allows complex transformations
  – Colorization [Iizuka 2016]
  – Style transfer [Gatys 2015]
  – Image-to-image translation [Isola 2016]

• Costly: several seconds per megapixel on high-end desktop GPU
  – image-processing algorithms are mature and well-engineered
Our approach
We learn to reproduce the operator

high-res input

\[ f \]

high-res output
We learn to reproduce the operator

high-res input

fast neural network

high-res output
Fast, edge-aware neural network

- Fast: most computation at low-res
  - runs in real-time on mobile: 50 Hz @ 1080p

- Architecture preserves edges and high-frequencies
Fast, edge-aware neural network

- Fast: most computation at low-res
  - runs in real-time on mobile: 50 Hz @ 1080p

- Architecture preserves edges and high-frequencies

- Does not require the original operator at runtime
  - trained end-to-end
  - implicitly-defined effects (e.g. human retouches)
Heavy computation at low-res

- full-res input (e.g. 12 Mpix)
- low-res input 256x256
Heavy computation at low-res

full-res input (e.g. 12 Mpix)

low-res input 256x256

convolutional network
Heavy computation at low-res

- full-res input (e.g. 12 Mpix)
- low-res input 256x256
- intermediate low-res model

convolutional network
Heavy computation at low-res

full-res input (e.g. 12 Mpix)

low-res input 256x256

intermediate low-res model

convolutional network

apply

full-res output
Key ideas

• Predict local affine models, not pixels
  – focus on the image transformation
  – this also regularizes the network’s output
Key ideas

• Predict local affine models, not pixels
  – focus on the image transformation
  – this also regularizes the network’s output

• Bilateral grid: costly computations at low resolution
  – neural network evaluation no longer a bottleneck
  – but preserve edges and high-frequency effects
Key ideas

• Predict local affine models, not pixels
  – focus on the image transformation
  – this also regularizes the network’s output

• Bilateral grid: costly computations at low resolution
  – neural network evaluation no longer a bottleneck
  – but preserve edges and high-frequency effects

• Optimize the entire pipeline for the high-res output
Idealized 1D example
Encode the transform at low resolution

image edge

pixel intensity

spatial dimension

input

target

output
Encode the transform at low resolution
Encode the transform at low resolution

coarse discrete grid, one linear model per cell
Encode the transform at low resolution

interpolate transforms:
preserves input edges and fine details
Encode the transform at low resolution

interpolate transforms: preserves input edges and fine details
What if we want to alter the edge?
Example: sharpening
Both sides are transformed identically!

\[ a_2 > 1 \]
Both sides are transformed identically!
We use a Bilateral Grid
Data-dependent lookup in a Bilateral Grid

coarse 2D grid, one linear model per cell

guided by intensity

input

target

output

$\begin{array}{cccc}
a_{10} & a_{11} & \cdots & \\
a_{00} & a_{01} & a_{02} & a_{03} \\
\end{array}$
Bilateral Grid: we can manipulate edges

brighten

darken

input

target

output
Bilateral Grid: we can manipulate edges
Back to 2D RGB images
We predict local affine transforms, not pixels

- Output = affine combination of input channels
- Can express many photographic operators
  - Matting [Levin 2008], Intrinsic Images [Bousseau 2009], Style Transfer [Shih 2013]
- Interpolated from a coarse 3D bilateral grid
  - Local: one transform per pixel
Slicing node to interpolate affine models

guidance map

3D bilateral grid
[Chen 2007]
Slicing node to interpolate affine models

3D bilateral grid
[Chen 2007]

guidance map
Slicing node to interpolate affine models

3D bilateral grid [Chen 2007]

Slicing node

guidance map

sliced affine coefficients (12 channels)
Model architecture
full-res input
256x256

low-res input

LOW-RES STREAM

low-level features

local features

global features

16x16x8 bilateral grid of affine models
full-res input

low-res input

256x256

low-level features

local features

global features

16x16x8 bilateral grid
of affine models

LOW-RES STREAM

HIGH-RES STREAM

Slice

sliced coefficients
full-res input

256x256

low-res input

slice

predict

guidance map

sliced coefficients

LOW-RES STREAM

HIGH-RES STREAM

16x16x8 bilateral grid of affine models

global features

local features

low-level features

sliced coefficients

bilateral grid of affine models

(\(a_{11}, a_{12}, a_{13}, a_{14}\))

(\(a_{21}, a_{22}, a_{23}, a_{24}\))

(\(a_{31}, a_{32}, a_{33}, a_{34}\))
full-res input

low-res input 256x256

low-level features

local features

global features

16x16x8 bilateral grid of affine models

LOW-RES STREAM

HIGH-RES STREAM

Slice

Predict

guidance map

sliced coefficients

Apply

full-res output
pointwise neural network

Predict

guidance map
sliced coefficients
Slice
Apply
full-res output

fast OpenGL shader
Training: optimize for the high-res!

- Minimize L2 loss on high-resolution output
  - ~4,000 high-res input/output pairs
  - ADAM [Kingma 2015]

- Faithful high-frequency effects
  - even though most of the model is at low-res
The slicing node preserves edges

- **Input**
  - Without slice (deconvolution network)
  - Ours (with slice)

- **Ground Truth**
Results
Evaluation

- Algorithmic operators

- Manual retouches (implicitly-defined operators)
  - 5 photographers from MIT5k [Bychkovsky 2011]
Evaluation

• Algorithmic operators

• Manual retouches (implicitly-defined operators)
  – 5 photographers from MIT5k [Bychkovsky 2011]

• Oracle baselines (filter run at low-res, not a black box)
  – Transform Recipes [Gharbi 2015]
  – Bilateral Guided Upsampling [Chen 2016]

• CNN baselines
  • pix2pix [Isola 2016], dilated convolutions [Yu 2015]
Performance

• Real-time: under 20ms on mobile (Google Pixel 2016)
  – At full-HD resolution, 1920x1080

• “Oracle” baselines (which require the operator at runtime)
  – Bilateral Guided Upsampling [Chen 2016]: 17ms
  – Transform Recipes [Gharbi 2015]: ~1s

• CNN baselines: 100x to 1000x slower
Comparison to recent CNNs

pix2pix [Isola 2016], dilated convolutions [Yu 2015]

PSNR (dB, higher is better)

end-to-end running time (ms, lower is better)

- 16MP input
- 256x256 low-res
- 16x16x8 grid

Comparison to pix2pix and dilated convolutions.
We match/outperform the fidelity of methods that access the original filter at runtime!

<table>
<thead>
<tr>
<th>Method</th>
<th>PSNR (higher is better)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local Laplacian</td>
<td>30.3</td>
</tr>
<tr>
<td>HDR+</td>
<td>28.8</td>
</tr>
<tr>
<td>Face Brightening</td>
<td>33.7</td>
</tr>
<tr>
<td>Style Transfer</td>
<td>27.6</td>
</tr>
<tr>
<td>Photoshop Actions</td>
<td>35.7</td>
</tr>
</tbody>
</table>

- Ours
- Bilateral Guided Upsampling
- Transform Recipes
Non scale-invariant filters challenge other methods

input

BGU [Chen 2016]
17.7 dB

ours
32.1 dB

ground truth
(Local Laplacian)
Local Laplacian | 400 ms (desktop)
ours | 33.1 dB, 20 ms (desktop)
ours | 32.7 dB, 40 ms (mobile)
Our model captures semantic features

input

ground truth
(Face Brightening)

ours
Manually specified retouches

- MIT5K dataset [Bychkovsky 2011]
  - 5 photographers
  - 5,000 images
  - global color and tone corrections

<table>
<thead>
<tr>
<th></th>
<th>L<em>a</em>b*</th>
<th>L* only</th>
</tr>
</thead>
<tbody>
<tr>
<td>ours</td>
<td>7.8</td>
<td>5.5</td>
</tr>
<tr>
<td>[Yan 2016]</td>
<td>9.9</td>
<td>5.7</td>
</tr>
<tr>
<td>[Bychkovsky 2011]</td>
<td>—</td>
<td>5.8</td>
</tr>
<tr>
<td>[Hwang 2012]</td>
<td>15.01</td>
<td>—</td>
</tr>
</tbody>
</table>
Manually specified retouches

input

ground truth retouch

ours
Manually specified retouches

- **input**
- **ground truth retouch**
- **ours**
Failure cases

• Image colorization
  gray input: constant colors limited by grid resolution

• Portrait Matting
  affine at a much smaller scale than we handle

• Dehazing
  low-contrast, clipping
Screen-capture

on mobile
Conclusion

• End-to-end trainable image enhancement

• High-resolution, edge-aware, high-frequency effects
  – Affine transforms in bilateral space, new slicing operator
  – Optimize full-resolution loss

• Real-time image enhancement on mobile
  • most of the processing happens at low-res

Michaël GHARBI gharbi@mit.edu
code & data: www.mgharbi.com
Thank you!
Global features regularize local decisions

without global features
Global features regularize local decisions

with global features
Why learn the guidance map?

(linear) input
Why learn the guidance map?

ground truth
Why learn the guidance map?

without learned guide
Why learn the guidance map?

with
learned guide
Allowing more computation at full-res

input

ours standard

ground truth

ours, multi-scale features

63