Edge-Optimized À-Trous Wavelets for Local Contrast Enhancement with Robust Denoising

Johannes Hanika, Holger Dammertz, Hendrik Lensch | September 17, 2011
Motivation: Edge-Aware Image Processing

- ongoing research:
- look transfer via bilateral filtering (Dae, Paris and Durand 2006)
Motivation: Edge-Aware Image Processing

- ongoing research:
- multi scale decomposition by solving a linear system (Farbman et al. 2008)
Motivation: Edge-Aware Image Processing

- ongoing research:
- colorization via edge-avoiding wavelets (Fattal 2009)
Motivation: Edge-Aware Image Processing

- ongoing research:
- local contrast via local histograms (Kass 2010)
Motivation: Edge-Aware Image Processing

- ongoing research:
- via domain transform (Gastal and Oliveira 2011)
Previous Work

- all based on multiscale decompositions:
  - iteratively applying a bilateral filter
    - lots of techniques to speed it up
    - still high memory footprint and/or low performance
  - high quality by solving a linear system
    - not meant to be high performance
  - fastest methods based on decimated wavelets (Fattal 2009)
Previous Work

- decimated wavelets fail to capture edges at all scales:
Previous Work

- decimated wavelets fail to capture edges at all scales:
Previous Work

- because coarse coefficients are sparse
Previous Work

- use à-trous wavelet

  \[
  \begin{array}{c}
  \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \\
  \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \\
  \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \quad \bullet \\
  \end{array}
  \]

  \[
  \begin{array}{c}
  i=0 \\
  i=1 \\
  i=2 \\
  \end{array}
  \]

- results in a full image (not decimated) per step
- ⇒ the transformation is *shift invariant*
Previous Work

- à-trous wavelet decomposition
  1. level $i = 0$ starts with the input signal $c_0(p)$
Previous Work

- à-trous wavelet decomposition
  1. level $i = 0$ starts with the input signal $c_0(p)$
  2. compute next base layer (convolution with holes)

\[
c_{i+1}(p) = \frac{1}{k} \sum_{q \in \Omega} h_i(q) \cdot c_i(p)
\]
Previous Work

- à-trous wavelet decomposition
  1. level $i = 0$ starts with the input signal $c_0(p)$
  2. compute next base layer (convolution with holes)

$$c_{i+1}(p) = \frac{1}{k} \sum_{q \in \Omega} h_i(q) \cdot c_i(p)$$

  3. compute next detail layer (difference)

$$d_i(p) = c_i(p) - c_{i+1}(p)$$
Previous Work

- à-trous wavelet decomposition

1. level $i = 0$ starts with the input signal $c_0(p)$
2. compute next base layer (convolution with holes)

$$c_{i+1}(p) = \frac{1}{k} \sum_{q \in \Omega} h_i(q) \cdot c_i(p)$$

3. compute next detail layer (difference)

$$d_i(p) = c_i(p) - c_{i+1}(p)$$

4. if $i < N : i := i + 1$; goto 2
Previous Work

- à-trous wavelet decomposition

1. level $i = 0$ starts with the input signal $c_0(p)$
2. compute next base layer (convolution with holes)

$$c_{i+1}(p) = \frac{1}{k} \sum_{q \in \Omega} h_i(q) \cdot c_i(p)$$

3. compute next detail layer (difference)

$$d_i(p) = c_i(p) - c_{i+1}(p)$$

4. if $i < N : i := i + 1$; goto 2
5. $\{d_0, d_1, ..., d_{N-1}, c_N\}$ is the wavelet transform of $c$. 
Previous Work

- à-trous wavelet decomposition edge-aware version

  1. level $i = 0$ starts with the input signal $c_0(p)$
  2. compute next base layer (convolution with holes)
     \[
     c_{i+1}(p) = \frac{1}{k} \sum_{q \in \Omega} h_i(q) \cdot c_i(p) \cdot w_\sigma(p, q)
     \]
  3. compute next detail layer (difference)
     \[
     d_i(p) = c_i(p) - c_{i+1}(p)
     \]
  4. if $i < N : i := i + 1$; goto 2
  5. \{ $d_0, d_1, \ldots, d_{N-1}, c_N$ \} is the wavelet transform of $c$. 
Previous Work

- à-trous wavelet decomposition edge-aware version

1. level \( i = 0 \) starts with the input signal \( c_0(p) \)
2. compute next base layer (convolution with holes)

\[
c_{i+1}(p) = \frac{1}{k} \sum_{q \in \Omega} h_i(q) \cdot c_i(p) \cdot w_{\sigma_r}(p, q)
\]

3. compute next detail layer (difference)

\[
d_i(p) = c_i(p) - c_{i+1}(p)
\]

4. if \( i < N \): \( i := i + 1 \); goto 2
5. \( \{d_0, d_1, ..., d_{N-1}, c_N\} \) is the wavelet transform of \( c \).

- synthesis: simply add up base and detail layers

\[
c = c_N + \sum_{i=0}^{N-1} d_i.
\]
Decomposition

- example coarse and detail layers

\[
\begin{array}{cccccc}
  c_4 & d_1 & d_2 & d_3 & d_4 \\
\end{array}
\]
Synthesis for Local Contrast

- add up boosted detail layers

\[ c = c_N + \sum_{i=N-1}^{0} \beta_i \cdot d_i. \]
Observation

- how to choose good edge weights $w_{\sigma_r}(p, q)$?
Observation

- how to choose good edge weights $w_{\sigma_r}(p, q)$?

- too strong edge weights: gradient reversals
Observation

- how to choose good edge weights $w_{\sigma_r}(p, q)$?

- too strong edge weights: gradient reversals
- too soft edge weights: halos
Observation

- how to choose good edge weights $w_{\sigma_r}(p, q)$?

- too strong edge weights: gradient reversals
- too soft edge weights: halos
- Kass and Solomon (2010) do explicit diffusion on coarse buffer as post
Decomposition is fast!

- ⇒ optimization by synthesis to acquire $\sigma_r$ per pixel!
- stay in wavelet framework
Edge-Optimized Decomposition

- at each scale, do several decompositions using $\sigma_j^r$, $j = 0, 1, ...$
Edge-Optimized Decomposition

- at each scale, do several decompositions using $\sigma_j^r$, $j = 0, 1, ...$
- compute error measure $e_j$
  \[
e_j = d_{i,j}^2 + \lambda \cdot \|\nabla c_{i,j}\|
  \]
- prefer low energy in details $d$ and smooth base layer $c$
**Edge-Optimized Decomposition**

- at each scale, do several decompositions using $\sigma^j_r, \ j = 0, 1, \ldots$
- compute error measure $e_j$

$$e_j = d_{i,j}^2 + \lambda \cdot \| \nabla c_{i,j} \|$$

- prefer low energy in details $d$ and smooth base layer $c$
- choose per-pixel edge weight

$$\sigma^k_r(p) : k = \arg\min_j \{ e_j \}$$

- details how to make noisy estimates of $\nabla c$ stable in the paper
Edge-Optimized Decomposition

- error images $e_j$

- choice of $\sigma_r$ and input image
Decomposition Quality

input

edge-avoiding

colored output for visualization as (Farbman 08) bilateral

(Farbman 08) WLS
edge-optimized
Decomposition Quality

input

edge-avoiding

colored output for visualization as (Farbman 08)

(Farbman 08) WLS

text: edge-optimized
comparable quality, orders of magnitude faster

bilateral
Synthesis with Denoising

- synthesis after local contrast boost also boosts noise!
Synthesis with Denoising

- synthesis after local contrast boost also boosts noise!
- wavelet framework $\Rightarrow$ can use robust noise variance estimate and BayesShrink threshold
  \[
  d'_i = \max\left\{0, |d_i| - T\right\} \cdot \text{sign}(d_i)
  \]
  and $c_{i-1} = c_i + \beta \cdot d'_i$
- details in the paper
Denoising Quality

input 5% noise

à-trous PSNR 32.5

EAW PSNR 39.1

EOW PSNR 39.8
Denoising Quality

input 10% noise

à-trous PSNR 26.3

EAW PSNR 34.6

EOW PSNR 35.9
Denoising Quality

input 40% noise

à-trous PSNR 26.5

EAW PSNR 15.0

EOW PSNR 19.6
Performance (CPU)

<table>
<thead>
<tr>
<th>algorithm</th>
<th>wallclock</th>
</tr>
</thead>
<tbody>
<tr>
<td>EAW (Fattal 09)</td>
<td>0.088s</td>
</tr>
<tr>
<td>(core i7, $\alpha = 1$)</td>
<td></td>
</tr>
<tr>
<td>EAW (Fattal 09)</td>
<td>0.296s</td>
</tr>
<tr>
<td>(core i7, $\alpha = 0.8$)</td>
<td></td>
</tr>
<tr>
<td>this paper</td>
<td>0.197s</td>
</tr>
<tr>
<td>(core i7)</td>
<td></td>
</tr>
</tbody>
</table>

- 1 megapixel, 3 scales, 4 channels per pixel Lab data
- core i7 CPU: 8 threads on 4 cores
- (Fattal 09) with $\alpha = 1$ removes expensive exponentiation
## Performance (GPU)

<table>
<thead>
<tr>
<th>ms</th>
<th>number of $\sigma_r$ tested</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>1 scale</td>
<td>19</td>
</tr>
<tr>
<td>2 scales</td>
<td>27</td>
</tr>
<tr>
<td>3 scales</td>
<td>35</td>
</tr>
<tr>
<td>4 scales</td>
<td>42</td>
</tr>
<tr>
<td>5 scales</td>
<td>55</td>
</tr>
</tbody>
</table>

- edge-optimized wavelet transform on a GTX480 for a one megapixel image
- numbers are in milliseconds
Results (Local Contrast)

▷ (video)
Limitations

- high contrast, axis aligned changes (in hdr images) can lead to aliasing:

![Image]

- transparently reduced by our optimization (both via $d^2$ and smoothness term)
- technique to further ameliorate that in the paper
- not the world’s best denoising technique, but helps suppress noise enhancement during local contrast step
Summary

- edge avoiding à-trous wavelets are useful!
- they can be fast (suitable for video processing)
- and achieve high-quality coarse/detail decompositions
  - avoid gradient reversals
  - avoid halos
  - better match the assumptions of BayesShrink denoising
- parameter free, if you want it
- super simple to implement
Thank you for listening!

- some of the code is available at http://darktable.sf.net (hardcore SSE optimized + OpenCL)
- thanks to Edouard Gomez and Rostyslav Pidgorny for the fast SSE version!