False contour reduction using neural networks and adaptive bi-directional smoothing

IEEE TRANSACTIONS ON CONSUMER ELECTRONICS
vol. 56, No. 2, 2010
Min-Ho Park, Ji Won Lee, Rae-Hong Park and Jae-Seung Kim

Presented by Shu Ran

School of Electrical Engineering and Computer Science
Kyungpook National Univ.
Abstract

- Proposed false contour reduction algorithm
  - Using neural networks and adaptive bi-directional smoothing
  - Consisting of two parts
    - False contour detection
      - Detecting false contour candidate pixels
        » Using directional contrast features
    - False contour reduction
      - NN processing
        » Reduced by pixelwise processing
      - Bi-directional smoothing
        » Applied to neighboring region of false contour
  - Computer simulations
    - Showing effectiveness of proposed method
Introduction

- Server kinds of artifacts
  - Larger display devices getting artifacts more noticeable
    - False contours and block artifacts
    - Charge coupled devices sensor noise and mosquito noise
  - Focusing on reduction of false contour
    - Observed over smooth region
      - Sky and water and single-colored object
        » Compressing or enhancing in image
    - Eyesore in large displays for high-quality images or videos
      - Need false contour reduction
Previous method for false contour reduction
- Using blue noise mask
  - Unstructured mosaic pattern
  - Requiring priori information
    - About cause of false contours and characteristics
- Applying spatiotemporal dither to lower bit-depth image
  - Performing requantization
  - Can be only applied to specific bit-depth
- Two-stage false contour reduction algorithm
  - False contour detection
    - Regarding candidate pixels as false contour
  - False contour reduction
    - Using one-dimensional adaptive-size directional smoothing filters
  - Not perfectly for false contour reduction
- Expanding false contour region using directional dilation
  - Reducing false contour using edge-preserving filtering
  - Producing blurring through bialteral filtering in texture region
    - Skin and lawn
Proposed method

- Using neural networks
  - Image enhancement
    - Restoring blurred edges
    - Reducing TV artifacts
  - Bi-directional filtering
    - Applied to neighboring region of false contour
False contour detection

- Gray level or color value at a pixel changing smoothly
  - Insufficient number of gray levels or color levels
    - Quantization and high image compression
    - Image enhancement and image processing
    - Object motion and illumination changing
  - Algorithm based on previous two-stage method
    - Removing smooth regions by requantization region
      - Remaining false contour and edge and textures
    - Separating false contours from edges or texture regions
      - Using four directional contrast features
• Four directional contrast features

\[
\begin{align*}
    f^h &= \frac{\sum_{l=0}^{L-1} \sum_{m=0}^{M-2} (I_f(l,m) - I_f(l,m+1))^2}{L(M-1)}, \\
    f^v &= \frac{\sum_{l=0}^{L-2} \sum_{m=0}^{M-1} (I_f(l,m) - I_f(l+1,m+1))^2}{(L-1)M}, \\
    f^d &= \frac{\sum_{l=0}^{L-1} \sum_{m=0}^{M-2} (I_f(l,m) - I_f(l-1,m+1))^2}{(L-1)(M-1)}, \\
    f^{ad} &= \frac{\sum_{l=0}^{L-2} \sum_{m=0}^{M-2} (I_f(l,m) - I_f(l+1,m+1))^2}{(L-1)(M-1)}
\end{align*}
\]

where \( I_f(l,m) \) represents the gray level at \((l,m)\) of the input image containing false contour and,
L x M denotes the mask size.

• Selecting maximum value among directional contrast features
  - Reduction of false contours
    » Using magnitude and direction information of feature
False contour reduction

- Intensity variation in smooth region with false contour
  - Larger than smooth region without false contour
    - Intensity changing more abruptly than at neighboring pixels
    - Small number of pixels detected as false contours
      » Unpleasant to eye
      » Degrade image quality
    - Dividing smooth region into more than two sub-regions

- Dealing false contour pixels and neighboring pixels
  - Pixelwise false contour reduction based on NN learning
    - Using a number of test images and their synthetic images
      » Containing false contours
  - Adaptive bi-directional smoothing
    - For false contour pixels and neighboring pixels
NN learning

- Performing NN learning for false contour reduction
  - Observing in smooth regions
    - Color or intensity varying smoothly
    - Changing with directionality
      » Defined as direction orthogonal to false contour
- Defining two features explaining intensity varies
  - Strength of intensity variation
    - Function of distance between false contour lines
      » Measured along line perpendicular to false contour lines
      » Short distance for pixel intensity changing abruptly
      » Long distance for pixel intensity varying slowly in region
  - Direction of intensity variation
    - Along line perpendicular to false contour line
## Proposed notations

<table>
<thead>
<tr>
<th>Notations</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I$</td>
<td>original image</td>
</tr>
<tr>
<td>$I_f$</td>
<td>input image containing false contours</td>
</tr>
<tr>
<td>$I'_f$</td>
<td>intermediate reconstructed image (output of NN)</td>
</tr>
<tr>
<td>$\hat{I}_f$</td>
<td>false contour enhanced image (final result)</td>
</tr>
<tr>
<td>$B_f$</td>
<td>binary map denoting the pixel position of false contours</td>
</tr>
<tr>
<td>$B_{f,d}$</td>
<td>binary map denoting the position of the pixel to be filtered in the second step</td>
</tr>
<tr>
<td>$\theta_f$</td>
<td>false contour direction image</td>
</tr>
<tr>
<td>$f^h, f^d, f^v, f^{ad}$</td>
<td>directional contrast features</td>
</tr>
<tr>
<td>superscripts $h, d, v, ad$</td>
<td>horizontal, diagonal, vertical, and anti-diagonal directions</td>
</tr>
<tr>
<td>subscript $f$</td>
<td>false contour</td>
</tr>
<tr>
<td>$W$</td>
<td>weight matrix</td>
</tr>
<tr>
<td>$W^{(q)}$</td>
<td>weight matrix of $(q)^{th}$ layer</td>
</tr>
<tr>
<td>$w_j^{(q)}$</td>
<td>weight vector representing connections between $j^{th}$ node in $(q)^{th}$ layer and all nodes in $(q-1)^{th}$ layer</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Notations</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N^{(q)}$</td>
<td>number of nodes in $(q)^{th}$ layer</td>
</tr>
<tr>
<td>$L \times M$</td>
<td>mask size for detecting false contours</td>
</tr>
<tr>
<td>$o \times o$</td>
<td>mask size of input of NNs</td>
</tr>
<tr>
<td>$c$</td>
<td>node value vector (NN)</td>
</tr>
<tr>
<td>$t$</td>
<td>input data vector (NN)</td>
</tr>
<tr>
<td>$t'$</td>
<td>output data vector (NN)</td>
</tr>
<tr>
<td>$b^{(q)}$</td>
<td>bias vector in $(q)^{th}$ layer (NN)</td>
</tr>
<tr>
<td>superscript $(q)$</td>
<td>layer in NN</td>
</tr>
<tr>
<td>$X(Y)$</td>
<td>maximum expansion length along the $x$ ($y$) direction in the second step from the pixel detected as false contour</td>
</tr>
<tr>
<td>$r$</td>
<td>number of second-step filtering at a given pixel position</td>
</tr>
<tr>
<td>$O$</td>
<td>number of nodes in the input layer</td>
</tr>
<tr>
<td>$P$</td>
<td>number of nodes in the hidden layer</td>
</tr>
<tr>
<td>$Q$</td>
<td>number of nodes in the output layer</td>
</tr>
<tr>
<td>$\Delta$</td>
<td>update term in adaptive bi-directional smoothing</td>
</tr>
</tbody>
</table>
- Neighboring pixels of false contours
  - Showing tendency of spread direction
    - Using to obtain proper weighting in NN learning
- Applying $o \times o$ mask to false contour pixel
  - Showing pixel intensity
  - Tendency of neighboring pixels in mask
    - Describing how and to which direction false contours spread
- Implemented NNs consist of three layers
  - Input layer with $O$ nodes
    - $Q$ setting to 25
  - Hidden layer with $P$ nodes
    - $P$ setting to 20
  - Output layer with $Q$ nodes
    - $Q$ setting to 1
- Structure of NN used to obtain a weigh vector by learning

![Diagram of NN structure](image)

**Fig. 1.** Structure of NN used to obtain a weight vector
- Weight vector at node $j$ in $(q)^{th}$ layer

$$w^{(q)}_j = \begin{bmatrix} w^{(q)}_{j,1} & w^{(q)}_{j,2} & \cdots & w^{(q)}_{j,N^{(q-1)}} \end{bmatrix}^T$$

(2)

where $w^{(q)}_j$ is a weight vector representing connections between $j^{th}$ node in $(q)^{th}$ layer and all nodes in $(q-1)^{th}$ layer, $N^{(q)}$ represents the total number of nodes in $(q)^{th}$ layer

- Defining weight matrix of $(q)^{th}$ layer $W^{(q)}$

$$W^{(q)} = \begin{bmatrix} w^{(q)}_1 & w^{(q)}_2 & \cdots & w^{(q)}_{N^{(q-1)}} \end{bmatrix}$$

(3)

- Obtaining weight matrix

$$W = \{W^{(1)}, W^{(2)}\}$$

(4)
Eight false contour detections
  • Considered in proposed method
    – Assuming pixel intensity in dark gray region larger than bright

Fig. 2. Eight directions for false contours
Distinguish weight matrix of each NN in (4) by index $k$

$$W = \{ W^{(1)}(k), W^{(2)}(k) \}$$  \hspace{1cm} (5)

where $W^{(q)}(k)$ represent weights of $(q)^{th}$ layer

Direction $k$ is expressed as

$$1 \leq k = \left\lfloor \frac{\theta_f(l,m)}{45} \right\rfloor +1 \leq 8$$  \hspace{1cm} (6)

where floor operator $\lfloor z \rfloor$ gives largest integer less than or equal to a given value $z$

NNs eliminating artifacts only at false contour pixels
  • Helping increase accuracy of adaptive bi-directional smoothing
First step of false contour reduction

- Pixelwise false contour reduction using NNs
  - Using weights obtained by NN learning
    - Result at jth node of hidden layer
      \[ c_j = w_j^{(1)^T} t + b_j^{(1)} \]  
      where \( b_j^{(1)} \) represents a bias of \( j^{th} \) node in hidden layer
  
  - Obtaining newly computed pixel value
    \[ I'_f(l,m) = \begin{cases} 
      w^{(2)^T}(k)c + b^{(2)}, & \text{if } B_f(l,m) = 1 \\
      I_f(l,m), & \text{otherwise}
    \end{cases} \]  
    where \( b^{(2)} \) represents a bias in output layer and \( c \) signifies an output of the hidden layer

- More precise prediction for second step
Second step of false contour reduction

- Adaptive bi-directional smoothing
  - Better and natural false contour reduction
    - Considering detected points and neighboring pixels

**Fig. 3.** Adaptive bi-directional smoothing processes according to false contour directions (seven cases).
- Process of proposed adaptive bi-directional smoothing
  - **Step 1**
    - Detecting false contour pixel and selecting two directions
  - **Step 2**
    - Regions to be smoothed by bi-directional filter
      - Expanded along two directions
    - Edges or textures limited
      - Pixels to be smoothed denoted as $B_{f,d}(i,j) = 1$
- Pseudo code for second step
  - Explaining adaptive bi-directional smoothing filter working

```
Pseudo code (Second step of false contour reduction: adaptive bi-directional smoothing)
Input: Reconstructed image by the first step (output of NN $I'_f$)
    Binary map $B_{f,d}$ denoting the position of the pixel to be filtered in the second step
    Maximum expansion length $X$ (or $Y$) along the $x$ (or $y$) direction from the pixel detected as false contour
Local: Update term $\Delta$ used in adaptive bi-directional smoothing
Output: False contour enhanced image $\hat{I}_f$ (output of the second step: final result)
    (Initial pixel value of $\hat{I}_f$ is the same as $I'_f$)

FOR $x$ upto $X$
  FOR $y$ upto $Y$
    IF $B_{f,d} =$ TRUE THEN
      $\hat{I}_f \leftarrow \hat{I}_f + \Delta$
    ELSE
      $\hat{I}_f \leftarrow I'_f$
    ENDIF
  ENDFOR
ENDFOR
```

**Fig. 4.** Pseudo code for the second step of false contour reduction (adaptive bi-directional smoothing).
Experimental results and discussion

- Show effectiveness of proposed method
  - Using five test images by computer simulation
  - Comparing with previous method in various aspects
    - Result image
    - Edge maps
      - Detected by Sobel masks using same threshold
    - Peak signal-to-noise ratios
    - Structural similarity
Five test images used in experiments before requantization

- 8 bits/pixel

**Fig. 5.** Original images (8 bits/pixel). (a) Mouse (720 × 480), (b) Earring (480 × 480), (c) Sunset (720 × 480), (d) Circle (480 × 480), (e) Sun (720 × 480).
Input images after requantization
- 6 bits/pixel
- Effectively preserving high-frequency details

Fig. 6. Re-quantized input images (top, 6 bits/pixel) and binary false contour detection map (bottom). (a) Mouse, (b) Earring (c) Sunset, (d) Circle, (e) Sun.
- 1-D intensity profiles along highlighted line
  - Segment A marked in fig.5 (a)

**Fig. 7.** Comparison of results of each step of the proposed method (1-D intensity profile along a highlight segment A marked in Fig. 5(a), 50th row).
Illustrating performance comparison for Earring B region

(a) Original image (8 bits/pixel), (b) Input image (6 bits/pixel), (c) Daly and Feng’s method [3], (d) Lee et al.’s method [4], (e) Choi et al.’s method [5], (f) Proposed method.

**Fig. 8.** Performance comparison of false contour reduction algorithms (Earring region B). (a) Original image (8 bits/pixel), (b) Input image (6 bits/pixel), (c) Daly and Feng’s method [3], (d) Lee et al.’s method [4], (e) Choi et al.’s method [5], (f) Proposed method.
Comparison of performance using Sobel edge masks

- Fig. 9. Performance comparison of the performance of false contour reduction algorithms using Sobel edge masks (Earring, threshold =15). (a) Original image (8 bits/pixel), (b) Input image (6 bits/pixel), (c) Daly and Feng’s method [3], (d) Lee et al.’s method [4], (e) Choi et al.’s method [5], (f) Proposed algorithm.
– Performance comparison

Fig. 10. Performance comparison of false contour reduction algorithms (Sunset region C). (a) Original image (8 bits/pixel), (b) Input image (6 bits/pixel), (c) Daly and Feng’s method [3], (d) Lee et al.’s method [4], (e) Choi et al.’s method [5], (f) Proposed method.
– PSNR comparison of false contour reduction algorithm

**Table 2.** PSNR COMPARISON OF FALSE CONTOUR REDUCTION ALGORITHMS

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Mouse</td>
<td>46.36</td>
<td>45.50</td>
<td>46.97</td>
<td>47.62</td>
<td>48.98</td>
</tr>
<tr>
<td>Earring</td>
<td>46.42</td>
<td>43.35</td>
<td>46.68</td>
<td>46.10</td>
<td>47.06</td>
</tr>
<tr>
<td>Sunset</td>
<td>46.58</td>
<td>44.27</td>
<td>46.79</td>
<td>48.10</td>
<td>47.50</td>
</tr>
<tr>
<td>Circle</td>
<td>46.66</td>
<td>49.99</td>
<td>47.48</td>
<td>49.73</td>
<td>49.85</td>
</tr>
<tr>
<td>Sun</td>
<td>47.33</td>
<td>46.03</td>
<td>47.92</td>
<td>49.50</td>
<td>48.38</td>
</tr>
</tbody>
</table>
SSIM comparison of false contour reduction algorithms

**Table 3.** SSIM COMPARISON OF FALSE CONTOUR REDUCTION ALGORITHMS

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Mouse</td>
<td>0.9680</td>
<td>0.9600</td>
<td>0.9537</td>
<td>0.9810</td>
<td></td>
<td>0.9757</td>
</tr>
<tr>
<td>Earring</td>
<td>0.9217</td>
<td>0.9564</td>
<td>0.9533</td>
<td>0.9725</td>
<td></td>
<td>0.9881</td>
</tr>
<tr>
<td>Sunset</td>
<td>0.9692</td>
<td>0.9646</td>
<td>0.9617</td>
<td>0.9787</td>
<td></td>
<td>0.9732</td>
</tr>
<tr>
<td>Circle</td>
<td>0.9698</td>
<td>0.9927</td>
<td>0.9619</td>
<td>0.9897</td>
<td></td>
<td>0.9850</td>
</tr>
<tr>
<td>Sun</td>
<td>0.9670</td>
<td>0.9653</td>
<td>0.9566</td>
<td>0.9788</td>
<td></td>
<td>0.9727</td>
</tr>
</tbody>
</table>
Comparison of computation time

**Table 4. SSIM COMPARISON OF FALSE CONTOUR REDUCTION ALGORITHMS**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Mouse</td>
<td>56,013</td>
<td>0.35</td>
<td>1.00</td>
<td>1.04</td>
<td>0.83</td>
</tr>
<tr>
<td>Earring</td>
<td>66,380</td>
<td>0.35</td>
<td>1.00</td>
<td>0.99</td>
<td>0.84</td>
</tr>
<tr>
<td>Sunset</td>
<td>46,526</td>
<td>0.31</td>
<td>1.00</td>
<td>1.05</td>
<td>0.74</td>
</tr>
<tr>
<td>Circle</td>
<td>724,333</td>
<td>0.46</td>
<td>1.00</td>
<td>1.02</td>
<td>0.71</td>
</tr>
<tr>
<td>Sun</td>
<td>40,711</td>
<td>0.42</td>
<td>1.00</td>
<td>1.04</td>
<td>0.86</td>
</tr>
</tbody>
</table>
Conclusions

- Proposing effective false contour reduction algorithm
  - False contour reduction part
    - NN processing
      - Reducing false contour by pixelwise processing
    - Bi-directional smoothing
      - Applied to neighboring region of false contour
  - Experimental results
    - Efficiently reducing false contours
      - Both real and synthetic images