A Non-Local Dual-Domain Approach
to Cartoon and Texture Decomposition:
Additional Material

Frédéric Sur
Université de Lorraine. LORIA UMR CNRS 7503, INRIA projet Magrit
Campus Scientifique, BP 239, 54506 Vandœuvre-lès-Nancy Cedex, France
November 30, 2018

This document is a companion paper to [6]. The reader is asked to zoom in on the figures, and is referred to [6] for explanations on the algorithm and experimental framework.

Matlab codes and data are available at the following URL:
https://members.loria.fr/FSur/software/NoLoDuDoCT/

Contents

1 Barbara 2
2 Barbara bis 11
3 Barbara ter 17
4 Texmos3 18
5 Manhattan 20
6 WTC 25
7 Broadway 30
8 Fingerprint 33
9 Kodim08 35
10 Mosaic 39
11 Mariner4 42
12 Mariner6 46
13 Florida 50
14 Our Heritage 55
15 Carry on 61
16 Calculation time 64
1 Barbara

Image credit: Alan Gersho’s lab at U.C. Santa Barbara.

Matlab command line:

\[
[\text{Ima}_C,\text{Ima}_T]=\text{CPT}(\text{double(Ima)}, [300,190;33,235;425,433;470,160;26,545]);
\]
Figure 1: **Barbara.** First row: original image. Second row: output of [2] (used as an input of the proposed algorithm). Third row: output of the proposed algorithm. Cartoon on the left and texture on the right.
Figure 2: Barbara. First row: output of [5]. Second row: output of [4] ($\sigma = 2$). Third row: output of [4] ($\sigma = 3$). Cartoons are on the left, textures on the right.
Figure 3: Barbara. Close-ups. First row: original image, proposed method, output of [1]. Second row: output of [5], output of [4] for $\sigma = 2$, and for $\sigma = 3$. 
Figure 4: Barbara. First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weights indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra.
Figure 5: Barbara. First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra.
Figure 6: Barbara. First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra.
Figure 7: Barbara. First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra.
Figure 8: Barbara. First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra.
2 Barbara bis

Matlab command line:

\[
\text{[Ima}_C, \text{Ima}_T]=\text{CPT( double(Ima), } [300,190;33,235;425,433;470,160;26,545], 32, 2, 1000 );
\]

The goal of this experiment is to illustrate the influence of parameter \( \beta \), which is deliberately set to 1000 so that the Gaussian weighting in Eq. (10) and (11) does not play any role. As we can see, texture components are not accurately retrieved, and edge components in the Fourier domain are also detected as texture, giving a blurry cartoon part.

Figure 9: Barbara bis. First row: original image. Second row: output of the proposed algorithm. Cartoon on the left and texture on the right. Compare with Figure 1.
Figure 10: Barbara bis. First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weights indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra. Compare with Figure 4.
Figure 11: Barbara bis. First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra. Compare with Figure 5.
Figure 12: **Barbara bis.** First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra. Compare with Figure 6.
Figure 13: **Barbara bis.** First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra. Compare with Figure 7.
Figure 14: Barbara bis. First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra. Compare with Figure 8.
3 Barbara ter

Matlab command line:

```
[ima_c,ima_t]=cpt( double(ima), [], 16);
[ima_c,ima_t]=cpt( double(ima), [], 64);
```

The goal of this experiment is to illustrate the influence of the size $L$ of the patches. We can see that small patches ($L = 16$) only capture high frequency texture components, and that large patches ($L = 64$) are likely to capture high frequency components that belong to edge instead of real textures.

![Figure 15: Barbara ter. First row: output of the proposed algorithm, $L = 16$. Second row: output of the proposed algorithm, $L = 64$ Cartoon on the left and texture on the right. Compare with Figure 1 ($L = 32$).](image)
4 Texmos3

A synthetic periodic noise is added to three parts of the texmos3 image, available at the following url:
http://sipi.usc.edu/database/

Matlab command line:

```matlab
im_gt=double(imread('texmos3.s512.tiff'));
[x,y]=meshgrid(1:size(im_gt,2),1:size(im_gt,1));
PNoise=zeros(size(im_gt,1),size(im_gt,2));
PNoise(1:size(im_gt,1)/2,1:size(im_gt,2)/2) = ...
   60*sin(2*pi*128/size(im_gt,1)*x(1:size(im_gt,1)/2,1:size(im_gt,2)/2)).*...
   sin(2*pi*32/size(im_gt,2)*y(1:size(im_gt,1)/2,1:size(im_gt,2)/2));
PNoise(size(im_gt,1)/2+1:end,1:size(im_gt,2)/2) = ...
   60*sin(2*pi*128/size(im_gt,1)*x(size(im_gt,1)/2+1:end,1:size(im_gt,2)/2));
PNoise(1:size(im_gt,1)/2, size(im_gt,2)/2+1:end) = ...
   60*sin(2*pi*64*sqrt(2)/size(im_gt,2)*(x(1:size(im_gt,1)/2, size(im_gt,2)/2+1:end)+...
   y(1:size(im_gt,1)/2, size(im_gt,2)/2+1:end)))
PNoise=repmat(PNoise,1,1,3);
Ima=im_gt+PNoise;
figure, imshow(uint8(Ima));
[Ima_C,Ima_T]=CPT(Ima);
```
Figure 16: Texmos3. First row: original image. Second row: output of the proposed algorithm. Third row: output of [5]. Fourth row: output of [4] ($\sigma = 2$).
5 Manhattan

Image credit: By Hakilon - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=18111836

Matlab command line:

```matlab
[Ima_C,Ima_T]=CPT( double(Ima), [175,110;300,900] );
```
Figure 17: Manhattan. First row: original image. Second row: output of the proposed algorithm. Second row: output of [5]. Third row: output of [4] ($\sigma = 2$).
Figure 18: Manhattan. First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra.
Figure 19: Manhattan. First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra.
Figure 20: Manhattan. Close-ups. Original image, proposed method, output of [5], output of [4] for $(\sigma = 2)$. 


6 WTC

Matlab command line:

\[
[\text{Ima}_C, \text{Ima}_T] = \text{CPT}(\text{double}(\text{Ima}), [390, 116; 440, 340; 160, 540; 922, 580]);
\]

Figure 21: WTC. First row: original image, output of the proposed algorithm (cartoon and texture). Second row: output of [5], and output of [4], $\sigma = 2$ (cartoon and texture)
Figure 22: WTC. First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra.
Figure 23: WTC. First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra.
Figure 24: WTC. First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra.
Figure 25: WTC. Close-ups. Original image, proposed method, output of [5], output of [4] for ($\sigma = 2$).
Matlab command line:

\[ [\text{Ima}_C, \text{Ima}_T] = \text{CPT}( \text{double(Ima)}, [], 32, 3 ); \]
Figure 26: Broadway. First row: original image. Second row: output of the proposed algorithm. Second row: output of [5]. Third row: output of [4] ($\sigma = 3$).
Figure 27: Broadway. Close-ups. Original image, proposed method, output of [5], output of [4] for ($\sigma = 3$).
8 Fingerprint

Image credit: by Fazen, CC BY 2.0, https://www.flickr.com/photos/fazen/3778408/sizes/m/

Matlab command line:

[Ima_C,Ima_T]=CPT( double(Ima), [], 64, 4);
Figure 28: Fingerprint. First row: original image, output of the proposed algorithm (cartoon and texture). Second row: output of [5], and output of [4], $\sigma = 3$ (cartoon and texture).
9 Kodim08

Image credit: Kodak Image Suite.
http://r0k.us/graphics/kodak/

Matlab command line:

[Ima_C,Ima_T]=CPT( double(Ima), [90,420] );
Figure 29: Kodim08. First row: original image. Second row: output of the proposed algorithm (cartoon and texture). Third row: output of [5]. Fourth row: output of [4], $\sigma = 2$ (cartoon and texture).
Figure 30: Kodim08. First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra.
Figure 31: Kodim08. Close-ups. Original image, proposed method, output of [5], output of [4] for $\sigma = 2$. 
10 Mosaic

Image source: http://www.cse.cuhk.edu.hk/leojia/projects/rollguidance/

Matlab command line:

[Ima_C,Ima_T]=CPT( double(Ima), [400,40], 32, 2, 20, 20);
Figure 32: Mosaic. First row: original image, output of the proposed algorithm (cartoon and texture). Second row: output of [5] (cartoon and texture). Third row: output of [4], $\sigma = 2$ (cartoon and texture).
Figure 33: Mosaic. Close-ups. Original image, proposed method, output of [5], output of [4] for $\sigma = 2$. 
11 Mariner4

Image credit: NASA/JPL-Caltech
http://www.exploratorium.edu/mars/earlymissions.php

Matlab command line:

\[ [\text{Ima}_C,\text{Ima}_T] = \text{CPT}(\text{double}(\text{Ima}), [180,350], 32, 2, 10, 20); \]
Figure 34: Mariner4. First row: original image. Second row: output of the proposed algorithm. Third row: output of [5]. Fourth row: output of [4] ($\sigma = 2$). Cartoon on the left and texture on the right.
Figure 35: Mariner4. Close-ups. Original image, proposed method, output of [5], output of [4] for \((\sigma = 2)\).
Figure 36: Mariner4. First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra.
12 Mariner6

Image credit: NASA/JPL-Caltech
http://www.imageprocessingplace.com/

Matlab command line:

\[\text{[Ima}_\text{C}, \text{Ima}_\text{T}] = \text{CPT( double(Ima), [293,137; 100,347], 64, 1, 10, 20)};\]
Figure 37: Mariner6. First row: original image. Second row: output of the proposed algorithm. Third row: output of [5]. Fourth row: output of [4] ($\sigma = 2$). Cartoon on the left and texture on the right.
Figure 38: Mariner6. Close-ups. Original image, proposed method, output of [5], output of [4] for ($\sigma = 2$).
Figure 39: mariner6. First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra.
13  Florida

Image credit: NOAA photo library
http://www.imageprocessingplace.com/

Matlab command line:

[Ima_C,Ima_T]=CPT( double(Ima), [245,245; 288,416], 32, 3, 10, 20 );
Figure 40: Florida. First row: original image. Second row: output of the proposed algorithm. Third row: output of [5]. Fourth row: output of [4] ($\sigma = 2$). Cartoon on the left and texture on the right.
Figure 41: Florida. First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra.
Figure 42: Florida. First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra.
Figure 43: Florida. Close-ups. Original image, proposed method, output of [5], output of [4] for ($\sigma = 2$).
14 Our Heritage

Image credit: Norman Rockwell’s Our Heritage
http://muddycolors.blogspot.fr/2013/10/norman-rockwell-american-originals.html

Matlab command line:

\[ [\text{Ima}_C, \text{Ima}_T] = \text{CPT}( \text{double(Ima)}, [317, 179; 45, 843; 1105, 125], 32, 2, 30, 20 ); \]
Figure 44: Our Heritage. First row: original image, and output of the proposed algorithm. Second row: output of [5]. Third row: output of [4] ($\sigma = 3$). Cartoon on the left and texture on the right.
Figure 45: Our Heritage. First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra.
Figure 46: Our Heritage. First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra.
Figure 47: **Our Heritage.** First row: query patch (in red) and the 20 nearest neighbor patches in the sense of the Euclidean distance of the power spectra (in green, with weight indicated below). Second row, from left to right and top to bottom: power spectrum of the query patch, estimation of the expected power spectrum of the cartoon part, estimation of the standard deviation of the power spectrum of the cartoon part, probability distribution under the null hypothesis, significant coefficients in the power spectrum of the query patch (in yellow). The same logarithmic scale is used for displaying the power spectra.
Figure 48: Our Heritage. Close-ups. Original image, proposed method, output of [5], output of [4] for ($\sigma = 3$).
15 Carry on

Image credit: Norman Rockwell’s Carry On
http://muddycolors.blogspot.fr/2013/10/norman-rockwell-american-origina.html

Matlab command line:

\[ [\text{Ima}_C, \text{Ima}_T] = \text{CPT}( \text{double(Ima)}, [], 32, 2, 40, 20 ); \]
Figure 49: Carry On. First row: original image. Second row: output of the proposed algorithm. Third row: output of [5]. Fourth row: output of [4] ($\sigma = 3$). Cartoon on the left and texture on the right.
Figure 50: Carry on. Close-ups. Original image, proposed method, output of [5], output of [4] for $(\sigma = 3)$. 
16 Calculation time

We give the calculation times of the experiments presented here. We use the implementation kindly provided by the authors of [5] and [3] (available here [3]). Experiments are run on a 12-core Intel Xeon E5-2650 v4 @ 2.20GHz CPU with 64 Gb memory.

| Image        | Size   | proposed method | BNN [5] | Direction filters [3, 4] |
|--------------|--------|-----------------|---------|-------------------------|
| Barbara (Fig. 1) | 567 × 787 | 35.5            | 332.1   | 6.5                     |
| Texmos (Fig. 16) | 512 × 512 | 17.8            | 172.3   | 3.4                     |
| WTC (Fig. 21)   | 1024 × 698 | 53.6            | 745.8   | 10.5                    |
| Manhattan (Fig. 17) | 575 × 1024 | 43.7            | 518.9   | 9.2                     |
| Broadway (Fig. 26) | 1024 × 698 | 68.3            | 1249.8  | 22.5                    |
| Fingerprint (Fig. 28) | 500 × 488 | 9.1             | 77.0    | 8.9                     |
| Kodim08 (Fig. 29) | 512 × 768 | 29.7            | 375.5   | 6.3                     |
| Mosaic (Fig. 32) | 600 × 450 | 19.5            | 110.1   | 4.1                     |
| Mariner4 (Fig. 34) | 500 × 450 | 25.1            | 139.7   | 5.1                     |
| Mariner6 (Fig. 37) | 461 × 471 | 15.1            | 168.2   | 1.2                     |
| Florida (Fig. 40) | 808 × 754 | 39.7            | 421.9   | 3.4                     |
| Our Heritage (Fig. 44) | 1600 × 1060 | 124.3          | 967.9   | 47.8                    |
| Carry On (Fig. 49) | 1362 × 1600 | 153.4          | 1356.6  | 59.4                    |

Table 1: From the left to the right: image name, size in pixels, and calculation times in seconds of the proposed method, of the block nuclear norm (BNN) based method [5], and of the directional filters of [3, 4].

References

[1] A. Buades, T. Le, J.-M. Morel, and L. Vese. Cartoon+texture image decomposition. Image Processing On Line, 1:1–8, 2011.

[2] A. Buades, T. M. Le, J.-M. Morel, and L. A. Vese. Fast cartoon + texture image filters. IEEE Transactions on Image Processing, 19(8):1978–1986, 2010.

[3] A. Buades and J.-L. Lisani. Directional filters for cartoon + texture image decomposition. Image Processing On Line, 6:75–88, 2016.

[4] A. Buades and J.L. Lisani. Directional filters for color cartoon+texture image and video decomposition. Journal of Mathematical Imaging and Vision, 55(1):125–135, 2016.

[5] S. Ono, T. Miyata, and I. Yamada. Cartoon-texture image decomposition using blockwise low-rank texture characterization. IEEE Transactions on Image Processing, 23(3):1128–1142, 2014.

[6] F. Sur. A non-local dual-domain approach to cartoon and texture decomposition. IEEE Transactions on Image Processing, 2019.