Gaussian Fourier Pyramid for Local Laplacian Filter

Yuto Sumiya, Tomoki Otsuka, Yoshihiro Maeda, Member, IEEE, and Norishige Fukushima, Member, IEEE

Abstract—Multi-scale processing is essential in image processing and computer graphics. Halos are a central issue in multi-scale processing. Several edge-preserving decompositions resolve halos, e.g., local Laplacian filtering (LLF), by extending the Laplacian pyramid to have an edge-preserving property. Its processing is costly; thus, an approximated acceleration of fast LLF was proposed to linearly interpolate multiple Laplacian pyramids. This paper further improves the accuracy by Fourier series expansion, named Fourier LLF. Our results showed that Fourier LLF has a higher accuracy for the same number of pyramids. Moreover, Fourier LLF exhibits parameter-adaptive property for content-adaptive filtering. The code is available at: https://norishigefukushima.github.io/GaussianFourierPyramid/

Index Terms—Local Laplacian filter, Laplacian pyramid, Fourier series expansion, scale-space

I. INTRODUCTION

MULTI-SCALE processing is an essential tool for detail manipulation of images and is used for various detail and contrast enhancement: high-dynamic-range imaging [1], detail enhancement [2], and contrast enhancement [3]. Multi-scale processing decomposes an input image into multiple layers using Gaussian and Laplacian pyramids [4], wavelet transform [5], and difference of Gaussians for scale-space analysis [6].

Unsharp masking is one of the simplest multi-scale methods; it consists of two layers: an input and a detail layer. The detail layer is a subtraction of the blurred image from the input image. This simple method produces large halos in steep edges; thus, nonlinear enhancement is used for coefficients to suppress large amplitudes.

The halo problem is resolved by edge-preserving multi-scale decomposition. Instead of linear filtering, bilateral filtering [7] is used to generate a base layer [1]. The bilateral filter is accelerated through fast Fourier transform (FFT) [1], and is further accelerated by state-of-the-art methods [8], [9], [10]. Next, iterative bilateral filtering is extended to decompose multiple layers [11]; however, it is difficult to determine the parameters in bilateral filtering, which still suffers from halo artifacts. Moreover, multi-scale decomposition is represented by other filters, such as least squares [2] and iterative local linear regression filtering [12], which is similar to guide image filtering [13]. Wavelet-based methods are also proposed for edge-preserving decomposition. Edge-avoiding wavelets [14] construct a basis according to the edge content of the images. Iterative bilateral filtering-based decomposition [11] is one of the A-Trous wavelet methods [15]. The method is extended to multilateral filtering for the computer graphics context [16]. These approaches generate detailed signals and then manipulate the signals through remap functions: linear-tone curve, gamma curve, and S-tone curve.

Local Laplacian filtering (LLF) [17] manipulates the contrast of the input signals using remap functions and then generates detailed signals. LLF locally enhances the image contrast and constructs a Laplacian pyramid for each pixel; thus, LLF can reduce halos with clearer images than the other multi-scale methods. Parameter adaptation for the remap function further improves the quality.

The per-pixel construction of the Laplacian pyramid is costly; thus, an accelerated method is proposed, called fast LLF [18]. Fast LLF reduces the number of Laplacian pyramids by selecting at finite sampled points. Then, the pyramids are linearly interpolated. However, when the number of sample points is insufficient, the approximation accuracy is low. Moreover, the parameter-adaptive version of fast LLF [19] is not an approximation of naive LLF since the precomputed pyramids are based on a fixed parameter.

For function approximations, Fourier series expansion is a better interpolation method than the linear one. Therefore, this study proposes an approximation for LLF using Fourier series expansion, named Fourier LLF. Fourier LLF generates Fourier pyramids, which include cosine and sine pyramids, and then product-sums the pyramids. Fourier LLF improves the accuracy and produces parameter-adaptive functionality. The generated Fourier pyramids are independent of the remap function; thus, we can change the function by switching only the coefficient for the pyramids. Once the pyramids are generated, we can change the parameter by \(O(1)\) operation.

The main contributions of this study are as follows:

- We formulate remap functions in LLF using Fourier series expansion, which has higher accuracy than that of the fast LLF [18].
- We verify that Fourier LLF exhibits parameter-adaptive property for the remap function.

II. PRELIMINARY

A. Gaussian Pyramid and Laplacian Pyramid

We introduce Gaussian and Laplacian pyramids, which are the bases of LLF. The Laplacian pyramid is used for
multi-scale processing and analysis of images for compression [20], texture synthesis [21], and harmonization [22]. The traditional Laplacian pyramid processing directly enhances the coefficients of the pyramid [23, 11, 24].

First, the Gaussian pyramid is introduced. Let \( I : S \rightarrow \mathcal{R} \) be a \( D \)-dimensional \( R \)-tone input grayscale image, where \( S \subset \mathbb{Z}^D \) and \( \mathcal{R} \subset \mathbb{R} \) denote the spatial and range domain (generally, \( D = 2 \) and \( R = 256 \)), respectively. The Gaussian pyramid is defined as the set of \( G_\ell[I] \subset \mathbb{R} \), where \( \ell \in \mathcal{L} = \{0,1,2, \ldots, \ell_{\text{max}}\} \subset \mathbb{Z} \). The lowest level of the pyramid is \( G_0[I] = I \), and its other level \( G_{\ell+1}[I] \) is the downsampled blurred image of \( G_\ell[I] \). This relationship can be described as follows:

\[
G_\ell[I] = (G_\sigma \ast G_{\ell-1}[I])_{\uparrow},
\]

where \( G_\sigma \ast \) is a Gaussian convolution with the standard deviation \( \sigma \). \( \downarrow \) is a downsampling operator, which halves the image size. The size of \( G_{\ell+1}[I] \) is the half-width and height of \( G_\ell[I] \). Usually, the Gaussian convolution is based on the binomial distribution for integer operations with five taps. The regarded standard deviation is approximately \( \sigma = 1 \).

The Laplacian pyramid is defined by the difference between the Gaussian pyramids of successive levels:

\[
L_\ell[I] = G_\ell[I] - (G_{\ell+1}[I])_{\uparrow},
\]

where \( \uparrow \) is an upsampling operator that doubles the image width and height using a smoothing kernel. Usually, the kernel is identical to \( G_\sigma \). The highest level of the Laplacian pyramid is \( L_{\ell_{\text{max}}}[I] = G_{\ell_{\text{max}}}[I] \).

B. Manipulation of the Laplacian Pyramid

The multi-scale detail enhancement with the Laplacian pyramid amplifies the pyramid coefficients except for the coarsest layer \( \sup \mathcal{L} \). For enhance functions, an \( S \)-tone-like function can suppress an overshoot, while the straightforward function is \( r(\cdot,0) = m\cdot \), where \( m \in \mathbb{R} \) is a constant value. If \( m = 1 \), the resulting image is the input image. The argument \( 0 \) in \( r(\cdot,0) \) is unnecessary in this case; it is used to match the latter remap function for LLF. Finally, the remapped signals are collapsed to obtain the output:

\[
O = L_{\ell_{\text{max}}}[I] + \sum_{\ell \in \mathcal{L}\setminus\{\ell_{\text{max}}\}} r(L_{\ell}[I],0),
\]

where \( \setminus \) indicates the excluding operator from the set.

C. Local Laplacian Filtering

The output pixel value of LLF \( O_p \) is defined by the Laplacian pyramid \( L[I] \) at \( p \) called the local Laplacian pyramid, where \( p = (x,y) \in S \) is the pixel position. Collapsing the pyramid generates the output image:

\[
O_p = \sum_{\ell \in \mathcal{L}} L_{\ell}[O_p].
\]

Let us consider the procedure for calculating \( L_{\ell}[O_p] \).

1. Repeat 2)–4) for all the pixels \( p \) and levels \( \ell \).
2. Build the Gaussian pyramid \( G_{\ell}[I] \) at level \( \ell \).
3. Apply a remap function discussed later and create a contrast-transformed image \( \hat{I} = r(I,G_{\ell}[I]) \).
4. Construct a Laplacian pyramid for \( \hat{I} \), and copy the pyramid coefficient at \( p \), \( L_{\ell}[\hat{I}] \), as an output Laplacian pyramid of \( L_{\ell}[O_p] \). The remapping value changes according to the level \( \ell \) at \( p \).
5. Collapse the pyramid using Eq. \( 7 \) to obtain \( O \).

The flow requires per-pixel and per-level construction of the pyramid; thus, its order is \( O(|\mathcal{L}| \cdot N \log |\mathcal{L}|) \), where \( N \) and \( |\mathcal{L}| \) are the number of pixels and levels, respectively. \( O(N \log |\mathcal{L}|) \) is the order of the pyramid building.

D. Fast Local Laplacian Filtering

The naive implementation accesses all layers of the pyramid for all pixels, while fast LLF [18] requires a limited number of pyramids for approximation. The algorithm of fast LLF is defined as follows:

1) Build the Gaussian pyramid \( G_{\ell}[I] \) \( \forall \ell \in \mathcal{L} \).
2) Sample the intensities \( k \in \mathcal{L} \subset \mathbb{R} \) at equal intervals \( \tau \).
3) Compute the remap images \( R_k = r(I,\tau k) \) for each \( k \) and build their Laplacian pyramids \( L_{\ell}[R_k] \).
4) For each pyramid and pixels \( (\ell \in \mathcal{L}, p \in S) \):
   a) Obtain the pixel value \( G_{\ell}[I]_p \) in the Gaussian pyramid.
   b) Find \( a(0 \leq a \leq 1) \) such that \( G_{\ell}[I]_p = \pi((1-a)k + a(k+1)), j = |G_{\ell}[I]_p|/\tau \).
   c) Compute the output pyramid by linearly interpolating the precomputed pyramids:
      \( L_{\ell}[O_p] = (1-a) L_{\ell}[R_k]_p + a L_{\ell}[R_{k+1}]_p \).
5) Collapse the pyramid using Eq. \( 7 \) to obtain \( O_p \).

The flow requires \(|\mathcal{L}| + 1 \) pyramids, where \(|\mathcal{L}| \) is the number of sampled intensities. The order is \( O((|\mathcal{L}| + 1) \cdot N \log |\mathcal{L}|) \).

E. Remap Function

The remap function for an intensity \( i \in \mathcal{R} \) is continuous and changes the input \( i \) around a reference value \( g \in \mathcal{R} \):

\[
r(i,g) = i - (i-g)f(i-g).
\]

LLF uses \( g = G_{\ell}[I]_p \), \( f \in \mathcal{R} \) is an even function usually and is specifically defined for each application. This study uses a Gaussian function with multiplication factor \( m \) denoted as \( w_r(i) \) for \( f \), which is defined as:

\[
w_r(i) = m \exp \left( \frac{i^2}{-2\sigma^2} \right).
\]

When \( m > 0 \), the function enhances images, while when \( m < 0 \), the function smooths images, as shown in Fig. [1]. The dashed lines indicate the \( y = x \) line, which is no remap function. The difference between the \( y = x \) line and the desired remap function indicates the degree of emphasis.

III. PROPOSED METHOD

A. Formulation

This study approximates the remap function using Fourier series expansion. The derivative function of the Gaussian function \( w_r \), in the remap function is:

\[
w'_r(i-g) = \sigma^{-2}(i-g)w_r(i-g)
\]

\[
\cdot (i-g)w_r(i-g) = \sigma^2 w_r(i-g).
\]

4. Construct a Laplacian pyramid for \( \hat{I} \), and copy the pyramid coefficient at \( p \), \( L_{\ell}[\hat{I}] \), as an output Laplacian pyramid of \( L_{\ell}[O_p] \). The remapping value changes according to the level \( \ell \) at \( p \).
5. Collapse the pyramid using Eq. \( 7 \) to obtain \( O \).
Substituting (7) into the remap function (5), the form is:
\[ r(i, g) = i - (i - g)w_r(i - g) = i - \sigma_r^2 w_r'(i - g). \] (8)

The Gaussian function in \( w_r(i) \) is an even function, which can be approximated by the number of finite cosine terms \( K \) of the Fourier series expansion [23]:
\[ w_r(i - g) \approx m(\alpha_0 + 2 \sum_{k=1}^{K} \alpha_k \cos(\omega_k(i - g))), \] (9)
where \( \alpha_k = \frac{\sigma \sqrt{2\pi}}{k} \exp\left(-\frac{1}{2}(\omega_k \sigma)^2\right) \) and \( \omega_k = \frac{2\pi}{T} k. \) \( T \) is the period. The derivative function of (9) is
\[ w_r'(i - g) \approx -2m \sum_{k=1}^{K} \alpha_k \sin(\omega_k(i - g))\omega_k. \] (10)

Using (10) and the addition theorem of trigonometric functions, we can approximate the remap function (8) as follows:
\[ r(i, g) \approx i - m \sum_{k=1}^{K} \hat{\alpha}_k \left( \sin(\omega_k g) \cos(\omega_k i) - \cos(\omega_k g) \sin(\omega_k i) \right). \] (11)
where \( \hat{\alpha}_k = 2\sigma^2 \alpha_k \omega_k. \)

The period \( T \) for an \( R \)-tone image is usually determined such that the following equation becomes minimum [23]; the equation approximates the formula for the error between the Gaussian function and its approximation:
\[ \arg\min_T E_k(T) = \text{erfc}\left(\frac{T\sigma}{T}(2K+1)\right) + \text{erfc}\left(\frac{T-R}{\sigma}\right). \] (12)

The output local Laplacian pyramid is defined by
\[ L_p[I] = G_p[r(I, G_p[I])] - G_{p+1}[r(I, G_p[I])]. \] (13)

Recall that \( G_p[I] = L_p[I] \), \( L_p[I] \), and \( r(I, I_p) = I \); thus, the form is expressed as follows by substituting (11):
case: \( \ell = 0 \)
\[ L_0[I] \approx I - G_1[I] + m \sum_{k=1}^{K} \hat{\alpha}_k \left( \sin(\omega_k G_0[I])\hat{C}_{1,k} - \cos(\omega_k G_0[I])\hat{S}_{1,k} \right), \]
case: \( 1 \leq \ell \leq \ell_{\text{max}} - 1 \)
\[ L_\ell[I] \approx G_\ell[I] - m \sum_{k=1}^{K} \hat{\alpha}_k \left( \sin(\omega_k G_\ell[I])\hat{C}_{\ell,k} - \cos(\omega_k G_\ell[I])\hat{S}_{\ell,k} \right) - G_{\ell+1}[I] + m \sum_{k=1}^{K} \hat{\alpha}_k \left( \sin(\omega_k G_{\ell+1}[I])\hat{C}_{\ell+1,k} - \cos(\omega_k G_{\ell+1}[I])\hat{S}_{\ell+1,k} \right), \]
where \( \hat{C}_{\ell,k} = G_\ell[\cos(\omega_k I)] \) and \( \hat{S}_{\ell,k} = G_\ell[\sin(\omega_k I)]. \) (14)

B. Pixel-by-Pixel Enhancement

Changing the remap function for each pixel is essential for enhancement, such as the enhancement of flat regions to avoid noise signal boosting. The naïve implementation can locally change the function without any additional footprint; however, the computation itself is inefficient. The paper [19] also uses adaptive remap functions for fast LLF; however, the implementation is not an approximation of LLF. Fast LLF requires Laplacian pyramids for each remap function to interpolate the pyramid; thus, the multiple of the number of remap functions and pyramids is required for approximating LLF.

By contrast, the proposed method is independent of the remap functions for generating pyramids because the Gaussian Fourier pyramids of \( I \) do not depend on the remap function. Only the coefficients exhibit a dependency on the remap function; thus, we change the coefficient for the pixel-level parameter adaptation. Let \( \hat{\alpha}_{k,p} \) be the \( k \)-th coefficient for \( p \). The remap function can be switched by replacing \( \alpha_{k,p} \) pixel-by-pixel in (14) instead of \( \hat{\alpha}_k \). Similarly, we can also change the magnification factor of \( m \) to \( m_p \) for pixel-by-pixel adaptation. Therefore, we need \( 2K + 1 \) pyramids for this case, which are identical to the parameter-fixed case.

IV. EXPERIMENTAL RESULTS

We compared Fourier LLF with naïve LLF [17] and fast LLF [13]. The processing time was evaluated using two
This study proposes an approximation for LLF using Fourier series expansion, called Fourier LLF. The experimental results showed that our method achieved higher accuracy per pyramid building and better performance than those of the conventional method. We also showed that our method can approximate the adaptive filter. The limitation of the proposed method is that it cannot handle color image distances. Note that the conventional method is also for grayscale images. The aforementioned limitation can be solved by high-dimensional Gaussian kernel approximation methods [27, 28].
Supplemental material

We compared the Gaussian Fourier pyramid based LLF with two-layers enhancement methods and multi-layers enhancement methods. The two-layers methods are defined as follows:

\[ O = I + r(I - f*B, 0), \]

(15)

where, \( I \) and \( O \) are input and output images, respectively. \( r \) is the remap function of Eq. (8). \( f* \) is a filter function, and this paper used four filters: Gaussian filter, bilateral filter \[7\], domain transform filter \[29\], and guided image filter \[13\]. The two-layer with Gaussian filtering means unsharp masking. For multi-layers methods, we used three approaches: Laplacian pyramid enhancement of Eq. (3), multiscale bilateral filter (MSBF) \[11\], and edge-avoiding wavelet (EAW) \[14\]. For the experiments, we used enough order \( K \) (> 59dB). PSNR over 59dB means that the intensity difference between the ideal and approximated output is < 0.5, generating the same integer value images. Thus, the output of our LLF is the same as the original LLF \[17\].

Figures 7 and 8 show the effect of the multiple layers. The LLF with more layers enhances images strongly.

Figures 9 and 10 show the results of various enhancement methods. For two-layers cases, Gaussian filtering makes large halos, but the other methods and proposed method suppress the halos. The bilateral filtering generates edge-reversals, but the other suppress them. Both domain transform and guided image filters can suppress halos and edge-reversals; however, both filters cannot enhance much. The domain transform filter uses geodesic distance, which becomes large in complex texture regions; thus, the smoothing output in the textured region tends to keep the input image. The guided image filtering uses linear fitting, which tends to ignore minor wave signals; thus, only significant edges in the image are enhanced. For multi-layer cases, pyramid and MSBF enhancement are the multi-layers extensions of two-layers with Gaussian filtering and bilateral filtering; thus, the output and the problems are similar to the two-layer cases. The EAW is a wavelet extension for MSBF; thus, the problems are also similar to the MSBF.
Fig. 10. Visual comparison with various method for Beach image (3-layers, $\sigma_r = 30, m = 7, K = 10$).

REFERENCES

[1] F. Durand and J. Dorsey, “Fast bilateral filtering for the display of high-dynamic-range images,” ACM Transactions on Graphics, vol. 21, no. 3, pp. 257–266, 2002.

[2] Z. Farbman, R. Fattal, D. Lischinski, and R. Szeliski, “Edge-preserving decompositions for multi-scale tone and detail manipulation,” ACM Transactions on Graphics, vol. 27, no. 3, pp. 67, 2008.

[3] S. Dippel, M. Stahl, Rafael W., and Thomas Blaffert, “Multi-scale contrast enhancement for radiographies: Laplacian pyramid versus fast wavelet transform,” IEEE Transactions on Medical Imaging, vol. 21, no. 4, pp. 343–353, 2002.

[4] E. H. Adelson, C. H. Anderson, J. R. Bergen, P. J. Burt, and J. M. Ogden, “Pyramid methods in image processing,” RCA Engineer, vol. 29, no. 6, pp. 33–41, 1984.

[5] S. Mallat, A wavelet tour of signal processing, Elsevier, 1999.

[6] T. Lindeberg, Scale-space theory in computer vision, vol. 256, Springer Science & Business Media, 2013.

[7] C. Tomasi and R. Manduchi, “Bilateral filtering for gray and color images,” in Proc. IEEE International Conference on Computer Vision (ICCV), 1998, pp. 839–846.

[8] K. N. Chaudhury, “Acceleration of the shiftable o(1) algorithm for bilateral filtering and nonlocal means,” IEEE Transactions on Image Processing, vol. 22, pp. 1291–1300, 2013.

[9] K. Sugimoto, N. Fukushima, and S. Kamata, “200 fps constant-time bilateral filter using svd and tiling strategy,” in Proc. IEEE International Conference on Image Processing (ICIP), 2019.

[10] Y. Sumiya, N. Fukushima, K. Sugimoto, and S. Kamata, “Extending compressive bilateral filtering for arbitrary range kernel,” in Proc. IEEE International Conference on Image Processing (ICIP), 2020.

[11] R. Fattal, M. Agrawala, and S. Rusinkiewicz, “Multiscale shape and detail enhancement from multi-light image collections,” ACM Transactions on Graphics, vol. 26, no. 3, pp. 51, 2007.

[12] B. Gu, W. Li, M. Zhu, and M. Wang, “Local edge-preserving multiscale decomposition for high dynamic range image tone mapping,” IEEE Transactions on Image Processing, vol. 22, no. 1, pp. 70–79, 2012.

[13] K. He, J. Shun, and X. Tang, “Guided image filtering,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 6, pp. 1397–1409, 2013.

[14] R. Fattal, “Edge-avoiding wavelets and their applications,” ACM Transactions on Graphics, vol. 28, no. 3, pp. 1–10, 2009.

[15] M. Holschneider, R. Kronland-Martinet, J. Morlet, and P. Tchamitchian, “A real-time algorithm for signal analysis with the help of the wavelet transform,” in Wavelets, pp. 286–297. Springer, 1990.

[16] H. Dammertz, D. Sewtz, J. Hanika, and H. P. A. Lensch, “Edge-Avoiding A-Trous Wavelet Transform for fast Global Illumination Filtering,” in Proc. ACM SIGGRAPH/EUROGRAPHICS Conference on High Performance Graphics, 2010.

[17] S. Paris, W.S. Hasinoff, and J. Kautz, “Local laplacian filters: Edge-aware image processing with a laplacian pyramid,” ACM Transactions on Graphics, vol. 30, no. 4, 2011.

[18] M. Aubry, S. Paris, J. Kautz, and F. Durand, “Fast local laplacian filters: Theory and applications,” ACM Transactions on Graphics, vol. 33, no. 5, 2014.

[19] Z. Qiang, L. He, Y. Chen, X. Chen, and D. Xu, “Adaptive fast local laplacian filters and its edge-aware application,” Multimedia Tools and Applications, vol. 78, pp. 5, 2019.

[20] P. J. Burt and E. H. Adelson, “The laplacian pyramid as a compact image code,” IEEE Transactions on Communications, vol. 31, no. 4, pp. 532–540, 1983.

[21] D. J. Heeger and J. R. Bergen, “Pyramid-based texture analysis/synthesis,” in Proc. Annual Conference on Computer Graphics and Interactive Techniques, 1995, pp. 229–238.

[22] K. Sunkavalli, M. K. Johnson, W. Matusik, and H. Pfister, “Multi-scale image harmonization,” ACM Transactions on Graphics, vol. 29, no. 4, pp. 1–10, 2010.

[23] P. Vuylsteke and E. P. Schoeters, “Multiscale image contrast amplification (musica),” in Proc. SPIE, Medical Imaging: Image Processing, 1994, pp. 551–560.

[24] Y. Li, L. Sharan, and E. H. Adelson, “Compressing and companding high dynamic range images with subband architectures,” ACM Transactions on Graphics, vol. 24, no. 3, pp. 836–844, 2005.

[25] K. Sugimoto and S. Kamata, “Compressive bilateral filtering,” IEEE Transactions on Image Processing, vol. 24, no. 11, pp. 3357–3369, 2015.

[26] Y. Maeda, N. Fukushima, and H. Matsuo, “Effective implementation of edge-preserving filtering on cpu microarchitectures,” Applied Sciences, vol. 8, no. 10, 2018.

[27] P. Nair and K. N. Chaudhury, “Fast high-dimensional kernel filtering,” IEEE Signal Processing Letters, vol. 26, pp. 377–381, 2019.

[28] T. Miyamura, N. Fukushima, M. Waqas, K. Sugimoto, and S. Kamata, “Image tiling for clustering to improve stability of constant-time color bilateral filtering,” in Proc. IEEE International Conference on Image Processing (ICIP), 2020.

[29] Eduardo S. L. Gastal and Manuel M. Oliveira, “Domain tools and applications,” in Proc. SPIE, Medical Imaging: Image Processing, 1994, pp. 551–560.

[30] F. Durand and J. Dorsey, “Fast bilateral filtering for the display of high-dynamic-range images,” ACM Transactions on Graphics, vol. 21, no. 3, pp. 257–266, 2002.