Cross-Based Local Multipoint Filtering

Jiangbo Lu[1], Keyang Shi[2]*, Dongbo Min[1], Liang Lin[2], and Minh N. Do[3]
[1]Advanced Digital Sciences Center, [2]Sun Yat-Sen University, [3]Univ. of Illinois at Urbana-Champaign

Abstract

This paper presents a cross-based framework of performing local multipoint filtering efficiently. We formulate the filtering process as a local multipoint regression problem, consisting of two main steps: 1) multipoint estimation, calculating the estimates for a set of points within a shape-adaptive local support, and 2) aggregation, fusing a number of multipoint estimates available for each point. Compared with the guided filter that applies the linear regression to all pixels covered by a fixed-sized square window non-adaptively, the proposed filtering framework is a more generalized form. Two specific filtering methods are instantiated from this framework, based on piecewise constant and piecewise linear modeling, respectively. Leveraging a cross-based local support representation and integration technique, the proposed filtering methods achieve theoretically strong results in an efficient manner, with the two main steps' complexity independent of the filtering kernel size. We demonstrate the strength of the proposed filters in various applications including stereo matching, depth map enhancement, edge-preserving smoothing, color image denoising, detail enhancement, and flash/no-flash denoising.

1. Introduction

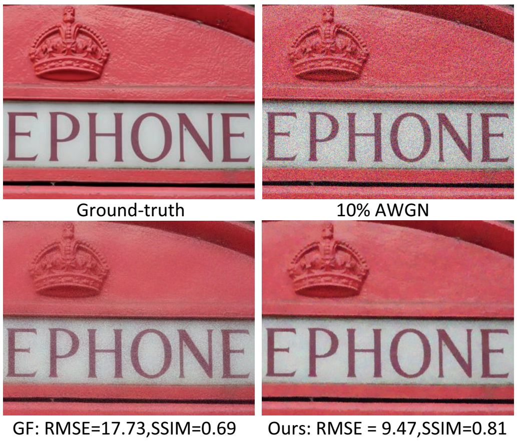
Edge-preserving or structure-preserving smoothing filtering is a desired property and a key component for many computer vision and graphics applications. Basically, the goal of edge-preserving smoothing filtering is to separate the main signal/image structures from the measurement noise or fine details, whereby the structure/edge should be well preserved and small fluctuations are smoothed out. From a general perspective, the signal to be filtered can take different forms such as an input color image with additive noise [8], or a cost slice/volume typically constructed in discrete labeling tasks (e.g., stereo [18]). Likewise, the structure used to guide the filtering process can also be defined in a broad way. For instance, the structure encoded in the input color image itself, or that in another guidance signal of high signal-to-noise ratio (SNR) aligned to the filter input. The latter case is also known as joint or cross filtering [12].

The bilateral filter (BF) [15] is arguably the most popular edge-preserving smoothing filter that is widely adopted in a variety of applications [11]. As a local, non-iterative filter, BF is intuitively simple. Using a range parameter σ_r and a spatial parameter σ_s, BF decides an adaptive weight for each point within a local square window. The final filtering output for the center pixel is simply computed as a weighted average of these neighboring pixels. Based on the taxonomy by Katkovnik *et al*. [7], BF is a local *pointwise* estimator in that it gives the estimate for a single point only i.e., the center pixel. BF uses a piecewise constant modeling where a zero-order local polynomial approximation is applied.

Recently, the guided filter (GF) [6] was proposed. Being computationally much faster than BF, GF has also demonstrated its unique advantage over BF in some applications such as detail enhancement and HDR compression. Moreover, when applied to cost volume filtering (e.g., stereo matching), GF has achieved state-of-the-art results among local stereo methods [14]. Though the authors did not make it explicit in [6], GF is essentially a local *multipoint* estimator according to [7]. In such a case, the estimates are calculated for all observation points used by the estimator. Since typically a number of such estimates are available for each point, they are aggregated (fused) together to compute the final estimate. This sort of redundant approximations with multiple estimates for each point is found to be drastically better than any of the windowed estimates [7]. Unlike BF, GF performs a first-order local linear modeling.

Inspired by the strong theoretical development in the image denoising field [7], we cast general-purpose edge-preserving smoothing filtering under a novel and broad framework of *local multipoint filtering*. Providing new theoretical understandings and extensions, the proposed framework comprises two major steps: 1) *multipoint estimation*, calculating the estimates for a set of points within a shape-adaptive local support, and 2) *aggregation*, fusing a number of multipoint estimates available for each point. Using spatial adaptivity to define local support regions and weighted averaging to fuse multiple estimates are two key generalizations in the proposed framework. Therefore, GF is a special instance of this framework in that fixed-sized square windows and simple averaging for multi-estimate fu-

*This work was done when Keyang Shi was an intern at ADSC.



Ground-truth

10% AWGN

GF: RMSE=17.73,SSIM=0.69

Ours: RMSE = 9.47,SSIM=0.81

Figure 1. GF [6] does not remove the noise near the edges well, while the proposed filtering method does a much better job. All the parameters in both algorithms have been fairly set.

sion are employed. Further, we propose two specific edge-preserving filters as novel instantiations from this generic framework, using a piecewise constant and piecewise linear approximation respectively to model the signal locally. They demonstrate better functional performance than GF and BF in a range of computer vision and graphics tasks. By leveraging and generalizing the cross-based local support decision and integration technique [19], the proposed filters perform the zero-order or first-order local polynomial modeling over pointwise-adaptive support regions efficiently. The two major steps' complexity is independent of the filtering kernel size, based on the integral image technique [4]. Also accommodating GF, our framework is versatile and can be flexibly configured to give the best variant.

2. Related Work and Motivation

Interested readers are referred to [11, 7, 6, 14] for a detailed review of edge-preserving smoothing filters and local multipoint filtering, covering both theory and applications. Here we focus on three most relevant filtering techniques.

Due to the edge-preserving smoothing property as well as its simplicity, BF [15] has been effectively employed in many applications [11]. However, because of the piecewise constant modeling used, BF generates the staircase effects in image smoothing operations [3]. Another known issue is the gradient reversal artifacts, caused by insufficient local support around transitional edges [6]. Computational efficiency is yet another challenge. As pointed out in [6], quantization-based fast implementations [13, 17] achieve satisfactory speed at the cost of quality degradation.

Cross-based local support decision and fast cost aggregation method were proposed in the context of stereo matching [19]. Based on a compact, pixel-wise varying local cross representation, the matching costs can be aggregated over a shape adaptive full support region using two orthogonal integration steps in O(1) time [4]. This method achieves the comparable disparity accuracy with the adaptive-weight method [18], but runs dozens of times faster. Later, this efficient and effective cost aggregation method has also been adopted as a fundamental building block in the top-ranking stereo method [9]. However, other than stereo, its applications to other problems have not been well explored yet. Also, the pointwise estimate based on hard weighting may not be fine enough for several graphics applications.

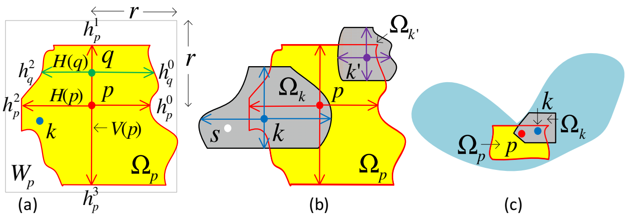


Figure 2. Cross-based local multipoint filtering. (a) Multipoint estimation for the local region Ω_p anchored at point p. (b) Aggregation of multiple estimates contributed by each $k \in \Omega_p$. (c) The proposed multi-estimate fusion approach deals with the concave structure better than using the single estimate from p alone [19].

As a special case of more general local multipoint estimators [7], GF [6] has shown its quality and speed advantages over BF. It has also been successfully applied to fast cost volume filtering [14]. However, GF is not without problems. First, the local linear modeling becomes very ineffective, when more than one models are present in most of the local windows covering the point to be filtered, as GF does not classify the local samples discriminatively into a few classes. Also, when the true signal is actually characterized by sharp transitions (e.g., depth discontinuities), GF results in undesired fuzzy object boundary in applications like depth map enhancement. From a perspective of statistics, it is clear that GF simply averages multiple estimates for the center pixel equally, without using any notion of linear regression quality or quality-of-fit for weighted fusion. Without spatial adaptivity, GF cannot be easily adapted to a piecewise constant model for the speed or quality purposes. Fig. 1 shows the deficiency of GF in denoising a color image, and we will discuss the reason and insights in Sect. 3.5.

3. Cross-Based Local Multipoint Filtering

3.1. Definition and Algorithm Overview

Before presenting the proposed filters, we first define the following notations. The signal/image to be filtered is denoted as Z, the guidance image as I, and the filter output image as Y. Note that I and Z can be identical, if the guidance image is the filter input itself. Let p be the pixel index of the estimation point, and k be an observation point or support pixel used by the estimator. As shown in Fig. 2(a), $k \in \Omega_p \subseteq W_p$, where Ω_p delineates an arbitrarily-shaped, connected local support region for anchor point p, based on some criteria (e.g. [19]). W_p is a square window of a radius

r, defining the maximum spatial extent as in GF [6].

The proposed cross-based local multipoint filtering (CLMF) framework consists of a few stages. First, for each pixel p, a set of four varying support arm lengths are decided on the guidance image I, i.e., $\{h_p^0, h_p^1, h_p^2, h_p^3\}$, the so-called cross skeleton in [19]. We will present an improved strategy for adaptive scale selection in Sect. 3.3. Once such a pixelwise cross skeleton is decided, a shape-adaptive full support region Ω_p is readily available as an area integral of multiple horizontal segments $H(q)$ spanned by pixel q [19]. Specifically, $\Omega_p = \bigcup_{q \in V(p)} H(q)$, where q is a support pixel located on the vertical segment $V(p)$ defined for pixel p, as shown in Fig. 2(a). After this preprocessing stage, CLMF performs two main filtering steps sequentially: 1) multipoint estimation, calculating the estimates Y_s^k for a set of points $s \in \Omega_k$ within a locally adaptive region anchored on point k, and 2) aggregation, fusing all the estimates $\{Y_p^k\}$ derived from each k-centered local region ($k \in \Omega_p$) to compute the final estimate (filter output) Y_p for each point p.

3.2. Generalization of Local Multipoint Filtering

Assuming a pixel-wise shape-adaptive support region Ω_p for each point p is given, we first present a generalized framework of local multipoint filtering here. Extending the local linear model adopted in GF [6], a local polynomial of order m is fit between the guidance I_s (independent variable) and the observations Z_s with $s \in \Omega_k$, for each anchor pixel k. We denote the fitted coefficient vector with $\mathbf{a}_k^m = [a_k^0, a_k^1, ..., a_k^m]$. In this paper, we only consider the cases of $m \leq 1$, so the zero-order ($m = 0$) or first-order polynomial model ($m = 1$) for each $s \in \Omega_k$ is given as

$$Y_s^k = \mathbf{a}_k^m \ \mathbf{I}_s^m = \begin{cases} a_k^0 & m = 0 \\ a_k^0 + a_k^1 I_s & m = 1 \ . \end{cases} \tag{1}$$

where $\mathbf{I}_s^m = [1, I_s, ..., I_s^m]^T$. Similar with GF, we use the method of least squares to fit the data, while enforcing that the model should be biased toward low-order polynomials to avoid over-fitting and gradient increase. Specifically, we minimize the following quadratic function:

$$E(\mathbf{a}_k^m) = \sum_{s \in \Omega_k} ((\mathbf{a}_k^m \ \mathbf{I}_s^m - Z_s)^2 + \epsilon \sum_{i \in [1,m]} (a_k^i)^2) \ . \tag{2}$$

ϵ is a regularization parameter to discourage the choices of large a_k^i ($i \geq 1$). When $m = 0$, the solution to (2) is given by

$$\mathbf{a}_k^0 \equiv a_k^0 = \bar{Z}_k = \frac{1}{|\Omega_k|} \sum_{s \in \Omega_k} Z_s \ , \tag{3}$$

where $|\Omega_k|$ is the number of pixels in Ω_k. When $m = 1$, the linear coefficients $\mathbf{a}_k^1 = [a_k^0, a_k^1]$ are given as follows,

$$a_k^1 = \frac{\frac{1}{|\Omega_k|} \sum_{s \in \Omega_k} I_s Z_s - \mu_k \bar{Z}_k}{\sigma_k^2 + \epsilon} \ , \tag{4}$$

$$a_k^0 = \bar{Z}_k - a_k^1 \mu_k \ , \tag{5}$$

where μ_k and σ_k^2 are the mean and variance of I in Ω_k. $\bar{Z}_k$ is the mean of Z in Ω_k as given in (3). To generalize (4,5) to the case of a color guidance image, the 3×3 covariance matrix Σ_k and color vectors (e.g., $\mathbf{I}_s$) are used as in GF [6].

In contrast to a pointwise estimator (e.g., BF) that gives the estimate for the center pixel k only, the multipoint estimator here is to calculate an estimate Y_s^k for all observation points used, i.e., $s \in \Omega_k$. For a given pixel p, as it is generally covered by multiple overlapping regions, it is involved in their respective linear regressions. It hence has a number of multipoint estimates $\{Y_p^k \mid p \in \Omega_k\}$. Different from GF [6] where these multipoint estimates are simply averaged together to compute the final estimate Y_p, the proposed CLMF framework takes into account of the fitting quality or the confidence of each multipoint estimate in the aggregation process. Specifically, the final estimate for each pixel is given as a weighted average of these multipoint estimates:

$$Y_p = \frac{\sum_{k:p \in \Omega_k} w_k Y_p^k}{\sum_{k:p \in \Omega_k} w_k} \ , \tag{6}$$

where w_k is the relative weight associated with each multipoint estimate Y_p^k. As Ω_k is intended to involve only the data points (inliers) that follow the similar signal structure or distribution in I [19], the outliers in the local square window W_k are rejected effectively. Therefore, the fitting error (residue) at every point $s \in \Omega_k$ can be considered as i.i.d. Gaussian noise, i.e., $Z_s = Y_s + \eta_s$, $\eta_s \sim N(0, \gamma^2)$. Normally, more data points (inliers) lead to a statistically more stable estimate of the true signal, as the random estimation error decreases. We hence set the value of w_k in (6) proportional to the number of sample points $|\Omega_k|$, though more sophisticated methods to decide w_k exist [7]. As a result, the aggregation of all multipoint estimates for pixel p is given by

$$Y_p = \frac{\sum_{k:p \in \Omega_k} |\Omega_k| Y_p^k}{\sum_{k:p \in \Omega_k} |\Omega_k|} \ . \tag{7}$$

For more regular computation and data access patterns, it is advantageous to transform the summations in (7) into the following p-pixel centric summations approximately:

$$Y_p \approx \frac{\sum_{k \in \Omega_p} |\Omega_k| Y_p^k}{\sum_{k \in \Omega_p} |\Omega_k|} \ . \tag{8}$$

Note that the above transformation may not lead to exactly the same final estimate as in (7). The reason is that for a more general way of defining $\Omega_{k/p}$, the mutual membership may not always hold, though most of time it does. More precisely, there could exist that $p \in \Omega_k$, but $k \notin \Omega_p$, or sometimes $k \in \Omega_p$, but $p \notin \Omega_k$ (e.g., k' in Fig. 2(b)). As discussed later, with this transformation, the multipoint estimate fusion process can be accelerated in the same way as the linear regression process for each support region [6]. For GF, as the mutual membership is decided symmetrically by requiring $||p - k||_\infty \leq r$, (8) gives the same result as (7).

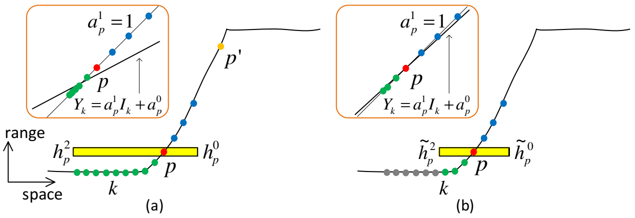


Figure 3. 1-D illustration for the reason and solution for gradient reversal artifacts. (a) Unbalanced arm lengths, making the linear regression biased to the green points. (b) Symmetric arm lengths.

3.3. Adaptive Scale Selection

The goal of adaptive scale selection is to decide for each direction an appropriate arm length, so they jointly delineate a shape-adaptive local support region [7]. Unlike the original cross-based method [19] where I_p is used firmly as the reference value for color similarity evaluation, we update it by the running average of the intensity of all the pixels covered by the current span h. This change makes the scale decision more robust against the measurement noise and the scales more extensible. Consider the right arm h_p^0 for pixel $p=(x_p, y_p)$. The proposed updating function is given as

$$\hat{I}_p^{(h)} = (1-\alpha)\hat{I}_p^{(h-1)} + \alpha I_{p+(h,0)} \ , \tag{9}$$

where $\hat{I}_p^{(0)} = I_p$. α is a parameter to control the updating rate. With an initial value set to 0, the optimal right arm length is decided as the largest right span $h^* \in [1, r]$ that satisfies the following requirements:

$$\begin{aligned} &\forall j \in [1, h^*], \delta(I_{p+(j,0)}; \hat{I}_p^{(j-1)}, \tau) = 1 \ , \\ &\text{and} \quad \delta(I_{p+(h^*+1,0)}; \hat{I}_p^{(h^*)}, \tau) = 0 \ . \end{aligned} \tag{10}$$

r is the preset maximum scale (window radius). As in [19], $\delta(I_s; I_t, \tau)$ measures the color similarity based on all color bands. If $\max_{c\in\{\mathrm{R,G,B}\}} |I_s^c - I_t^c| \le \tau$, then $\delta(I_s; I_t, \tau) = 1$, otherwise 0. Instead of deciding the scale for each color channel separately, this method decides the scale by using three color channels jointly. This addresses the issue when edges are not discriminable in any single color channel. Finally, we set the right arm length $h_p^0 = \max(h^*, 1)$, enforcing a minimum support scale for reliable regression.

When $m = 0$ or in other words, the zero-order polynomial model is used (named CLMF-0 hereinafter), we set $\alpha = 1/(h+1)$ in (9). This is consistent with the piecewise constant assumption made for the regression in (3). Different from [19] that firmly takes I_p as the reference value in (10), our method considers I_p only as a noisy measurement of the *ideal* unknown signal, which remains to be estimated.

When $m = 1$, the first-order polynomial model is used (named CLMF-1 hereinafter). We set α to a fixed value (e.g., 0.5) to allow fast spatial updating. This leads to a desired property of being able to extend the scale in gradient regions, so more pixels are involved for reliable regression.

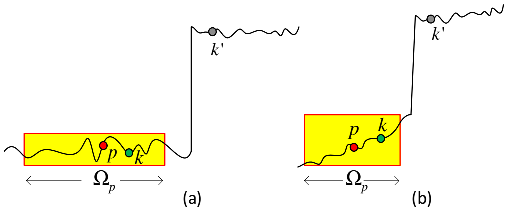


Figure 4. 1-D step edge linking two (a) piecewise constant regions or (b) piecewise linear regions. To filter the point p, CLMF-0 and CLMF-1 only use the multipoint estimates from points $k \in \Omega_p$.

However, if the CLMF-1 filter is directly applied to detail enhancement, we still noticed the gradient reversal artifacts as the BF usually has [6]. The origin of these artifacts can be explained by a 1-D signal that contains a ramp edge in Fig. 3. Given a pixel p (red) on the transitional edge, its left arm length is decided as h_p^2 (i.e., covering all the green points), while its right arm length is h_p^0 (i.e., covering all the blue points). It is clear that the numbers of the data points on the two sides of pixel p are very unbalanced, where there are much more points k such that $I_k \le I_p$ than the number of points satisfying $I_k \ge I_p$. If the linear regression in (4,5) is computed based on all these data points, the regression results will be more biased to the left side, as shown in the inset of Fig. 3(a). Thus, the final estimate Y_p at pixel p is much smaller than I_p, so boosting this difference ends up causing an upward hump in the enhanced signal at p. Similarly, a valley is created for the pixel p' (orange). This is known as the gradient reversal artifacts [6]. We also find that the above analysis is quite similar to the geometrical interpretation of the staircase effect by Buades *et al.* [3]. To address this issue caused by the asymmetric arm lengths while still keeping the advantage of the scale adaptation, this paper proposes to set the horizontal and vertical arm lengths symmetrically to the lower length, i.e., $\tilde{h}_p^0 = \tilde{h}_p^2 = \min(h_p^0, h_p^2)$, and $\tilde{h}_p^1 = \tilde{h}_p^3 = \min(h_p^1, h_p^3)$. The symmetric arm lengths use more balanced numbers of points from both sides of p for the linear regression, which tends to preserve the initial intensity I_p much better. This method effectively avoids the gradient reversal artifacts as shown later in Fig. 12.

3.4. Edge- and Gradient-Preserving Filtering

CLMF-0 and CLMF-1 have the edge-preserving smoothing property. This can be explained by an example of a 1-D step edge in Fig. 4. As presented in Sect. 3.1, for every pixel p, a shape-adaptive local region Ω_p, without straddling step edges, is first constructed. Local multipoint filtering is then performed based on Ω_p. As a result, the pixels (e.g., k') on the other side of the step edge are not (or rarely) used in the linear regression (or smoothing) process. In CLMF-0, τ controls the smoothing strength by deciding the neighboring pixels involved in (3). It has a similar effect as the range variance σ_r^2 in BF [15]. For CLMF-1, τ is set to classify the neighboring pixels and only the points within Ω_p are used

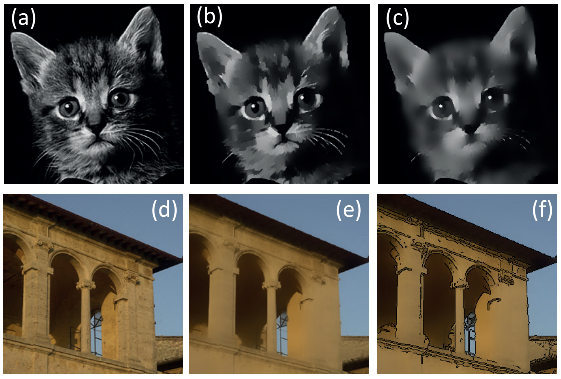


Figure 5. Edge-preserving smoothing results by CLMF-0 (top row, $r=8$) and CLMF-1 (bottom row). (a) Input gray-scale image. (b) $\tau=25/255$. (c) $\tau=50/255$. (d) Input color image. (e) $r=9$, $\tau=50/255$, $\epsilon=0.15^2$. (f) Edges on (e) for a stylized effect [16].

for the linear regression. As in GF, ϵ decides the smoothing level for these selected pixels. It is typically set based on the expected noise/detail level. The edge-preserving smoothing results of CLMF-0 and CLMF-1 are shown in Fig. 5.

Not only an edge-preserving smoothing filter, CLMF-1 also has the desired property of preserving image gradients. When $m = 1$, (8) can be rewritten as: $Y_p = \bar{a}_p^1 I_p + \bar{a}_p^0$, where $\bar{a}_p^1 = \frac{\sum_{k\in\Omega_p} |\Omega_k| a_k^1}{\sum_{k\in\Omega_p} |\Omega_k|}$, and $\bar{a}_p^0 = \frac{\sum_{k\in\Omega_p} |\Omega_k| a_k^0}{\sum_{k\in\Omega_p} |\Omega_k|}$. Thanks to the symmetric arm length constraint enforced in Sect. 3.3, the derived linear coefficients $[a_k^1, a_k^0]$ become much more reliable and less biased. As the low-pass averaging output of these reliable coefficients, $\bar{a}^1$ should have much smaller gradients than that of I near strong edges. This means $\nabla Y \approx \bar{a}^1 \nabla I$, and the gradients in I is better preserved.

3.5. Limitations of Guided Filter

Unlike the proposed CLMF-1, GF makes a strong assumption that a single linear model is sufficient to model a local patch W_p centered at pixel p [6]. However, consider the case that $I = Z$, if the guidance image I has a high local variance in W_p, then the linear coefficients $[a_p^1, a_p^0]$ become very close to $[1, 0]$. This essentially means that Z_p just keeps its original measurement value without smoothing. Therefore, for the points close to a high-contrast edge, they do not undergo sufficient smoothing due to the single model assumption (recall Fig. 1). Let's consider such an example in Fig. 6(a). GF achieves strong smoothing for pixel p, only when it is contained in a (shifted) window without involving pixels from the object A. The upper-left corners of these "good" windows are denoted in orange, while their corresponding window centers k are marked in yellow (i.e., region U_p). In the end, GF simply aggregates all the multipoint estimates for p using unweighted averaging. This low-pass filtering uses far more un-smoothed values Z_p from the windows centered in the region of $W_p \backslash U_p$ than Y_p^k from those good windows with $k \in U_p$. Hence, this results in insufficient smoothing or denoising for p. In contrast, as shown in Fig. 6(b), CLMF-1 avoids the high-contrast edge thanks to its support scale adaptation, so the linear regression performance is not affected by the irrelevant points belonging to the object A. Furthermore, we use the weighted average to aggregate all the multipoint estimates from the windows centered at k, with $k \in \Omega_p$. This further ensures that more confident estimates (defined based on $|\Omega_k|$ in this paper) have a higher influence on the final filter output for p. While preserving high-contrast edges, CLMF-1 also smooths out noise or details in the regions nearby.

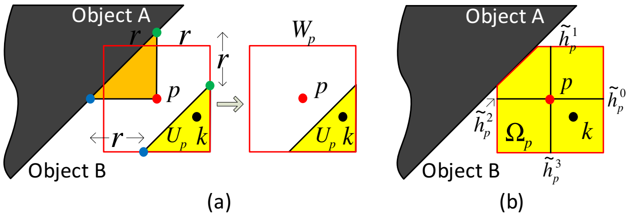


Figure 6. (a) GF does not smooth regions near high-contrast edges or high-variance regions. (b) CLMF-1. See the text for details.

Another issue with GF is that it may generate fuzzy object boundaries in the resulting filter output Y, if the true signal underlying the input signal Z actually has a sharp transition here. Such an artifact can be most visible in depth map enhancement (see Fig. 9), and it is known that the accuracy along depth discontinuities is very important for several applications. This artifact is due to the reason that GF performs the piecewise linear regression that involves all the pixels covered by a local window. So, it is particularly problematic in preserving sharp depth edges, if the color contrast across a depth edge is not high. Instead, using a low-order fitting in shape-adaptive support regions, CLMF-0 can significantly improve the recovery quality for sharp step edges in the input signal Z, without causing blurry boundaries.

3.6. O(1) Time Linear Regression and Aggregation

As detailed in [19], the integration of raw stereo matching cost or intensity over a shape-adaptive local region can be performed efficiently in O(1) time. This means that the time complexity is independent of the window radius r. This is actually thanks to the connectivity constraint made when constructing pixel-wise cross support skeletons. As a separable filter, the integration over a 2-D irregularly-shaped region (3) can be exactly and also efficiently computed by the integral image technique [4]. To normalize the integration result by the varying sample number covered by Ω_k in (3), one 2-D integration is also needed to compute $|\Omega_k|$ in O(1) time. As CLMF-0 augments [19] with a multipoint aggregation step as in (8), this step requires only two additional 2-D integrations in the same way. In total, only *four* O(1)-time 2-D integrations over cross-defined support regions are needed in CLMF-0, for both gray-scale

Table 1. Summary of the comparison between CLMF-0, CLMF-1 and other filters

Filter	Local modeling	Multipoint est.	Soft kernels	Edge/Grad. preserving	Explicit support	Speed
BF [15]	Piecewise constant	Pointwise	Y	Only edge	Without	Slow
CLF [19]	Piecewise constant	Pointwise	N	Only edge	With	Very fast
CLMF-0	Piecewise constant	Multipoint	Y	Only edge	With	Very fast
GF [6]	Piecewise linear	Multipoint	Y	Edge & gradient	Without	Fast
CLMF-1	Piecewise linear	Multipoint	Y	Edge & gradient	With	Fast

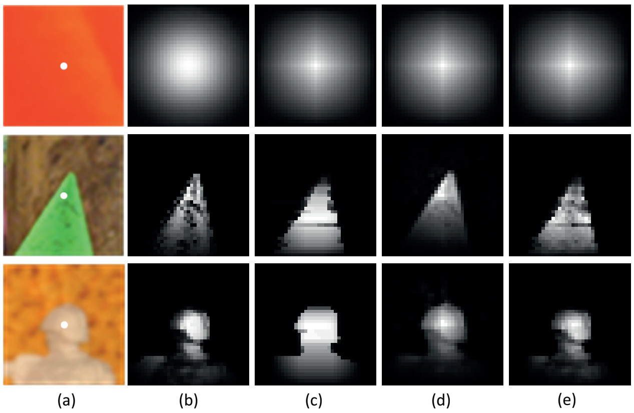


Figure 7. Filter kernels for (a) different image patches computed by (b) BF ($\sigma_s = 9, \sigma_r = 0.1$), (c) CLMF-0 ($\tau = 30/255$), (d) GF ($\epsilon = 0.1^2$), (e) CLMF-1($\tau = 30/255, \epsilon = 0.1^2$), all with $r = 9$.

and color guidance images. In comparison, for a gray-scale guidance image, both GF and CLMF-1 need to compute the second-order moments, thus requiring *seven* and *eight* O(1)-time 2-D integrations, respectively. For RGB guidance images, far more numbers of O(1)-time 2-D integrations are needed in addition to intensive vector arithmetic involved for each pixel, so they run noticeably slower than CLMF-0.

3.7. Connection with Other Filters

Table 1 summarizes the comparison between the proposed CLMF filters with other popular filters. Based on the fundamental assumption made about local patches, these filters can be categorized into two main classes: piecewise constant local modeling and piecewise linear local modeling. For the filters using piecewise constant modeling, they usually cannot preserve the image gradient information well. Without an aggregation step that adaptively fuses multiple estimates as adopted in CLMF-0 and CLMF-1, the original cross-based local filtering (CLF) [19] does not achieve a soft filter kernel of spatially varying support weights. This is due to the hard decision of support scales and absence of a soft range kernel. One advantage associated with all cross-based filters is that an explicit support region Ω_p mostly involving confident inliers is defined for every pixel p. As shown later, this region can be effectively reused in some applications such as refining stereo matching results. Representing such a locally adaptive support region is also memory efficient, as only four bytes per pixel are needed to store the four arm lengths [19]. Compared with GF, the proposed CLMF framework is a more generalized form. It performs either the linear regression or local averaging over a shape-adaptive support region, rather than within a fixed-sized square window. Besides the functional strength, CLMF-0 and CLMF-1 gain the computational efficiency similarly from the integral images technique as in GF [6]. However, GF does not support piecewise constant local modeling natively. Fig. 7 shows the filter kernels computed for different patches all extracted from real images. CLMF-1 defines the weights quite similar to GF. Even just using a hard scale decision and hard weighting in the first step, CLMF-0 achieves soft weights after the aggregation step, which are visually better than BF at certain locations.

Table 2. Middlebury stereo evaluation results (as of Dec. 2011)

Method	Rank	Avg. error %	Avg. time
CLMF-1	12	5.13	3.9 sec
CLMF-0	15	5.24	1.0 sec
CostFilter (GF) [14]	17	5.55	3.7 sec
P-LinearS (GF) [5]	21	5.68	33 sec
AdaptWeight (BF) [18]	51	6.67	60 sec
Var.Cross (CLF) [19]	64	7.60	0.9 sec

4. Applications and Experimental Results

This section presents various computer vision and graphics applications of the proposed CLMF-0 and CLMF-1.

4.1. Stereo Matching

The accuracy of local stereo methods is highly dependent on the cost aggregation schemes used. As prior methods that employ BF [18] and GF [14] for cost aggregation, CLMF can also be applied to filter the cost volume while preserving edges. First, the pixel-wise raw matching cost is defined by the sum of an absolute color difference and the Hamming distance of two census transforms [9]. To allow the cross arms to extend in regions with very similar color patterns, we also adopt a larger window radius R than r to include more pixels for textureless regions. Similar to [9], a much stricter color similarity threshold τ_s is enforced in (10), when the current arm length $h > r$. Then, the cost volume is filtered by CLMF-0/1 using both input images symmetrically, followed by a simple Winner-Takes-All strategy and occlusion detection and filling as presented in [14]. Unlike [14] using a weighted median filter for postprocessing, where the weights are given by a costly BF, our method

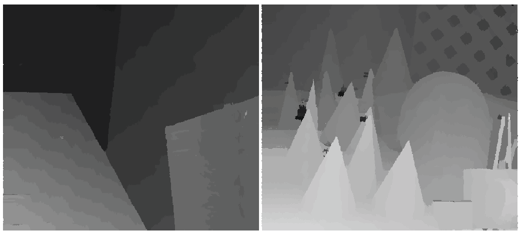


Figure 8. Disparity maps for *Venus* and *Cones* using CLMF-0.

Table 3. Evaluation of different cost aggregation methods

Method	Avg. error %
a. Single pointwise est., $m = 0$[19]	8.29
b. Simple avg. of multipoint est., $m = 0$	6.86
c. Weighted avg. of multipoint est., $m = 0$	6.47
d. Weighted avg. of multipoint est., $m = 1$	6.30
e. GF-based aggr. w/o postprocessing [14]	8.85

reuses the explicit local support region Ω_p (created already for each point p) for efficient mode filtering as in [19].

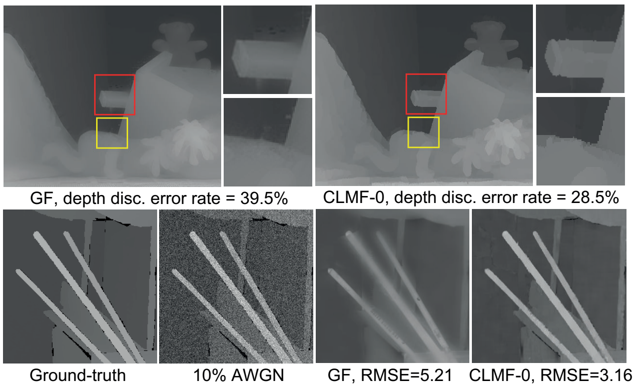
We evaluated the proposed method based on the Middlebury stereo benchmark [1]. The parameters are set constant across all datasets: $r = 5$, $\tau = 20$, $R = 13$, $\tau_s = 6$. Table 2 lists the results of the most relevant local stereo methods. It shows that our CLMF methods are the best-performing local stereo methods, even better than [14] that used to be the best. Moreover, CLMF-0 achieves an average rank of 5.3 and ranks 3rd out of over 110 stereo methods for the challenging *Cones* scene. Fig. 8 shows the disparity maps estimated by CLMF-0, which are piecewise smooth with depth discontinuities well preserved. Turning off all the postprocessing in our stereo method and that in CostFilter [14] (based on its public Matlab code), we have compared the performance of different cost aggregation schemes in Table 3. It justifies the strength of each algorithmic improvement in the CLMF framework as well as the advantage over the GF-based aggregation. Table 2 also reports the overall CPU runtime of CLMF methods and the competitors[1] for *Tsukuba*. CLMF-0 stands out as the most cost-effective one.

4.2. Depth Map Enhancement

Given a low-resolution and/or noisy depth map plus a registered high-resolution, noise-free color image, the resolution or quality of the depth map can be enhanced by joint filtering with the color image as the guidance. Table 4 compares five different filters when applied to upsample a low-resolution depth map by a scaling factor of 8. Compared with BF and GF, CLMF-0 significantly improves the depth map accuracy, particularly for challenging depth discontinuities. This is due to the reason revealed in Sect. 3.5. For the same reason, CLMF-1 does not perform so well like CLMF-0, though it is better than GF. In addition, CLMF-0 achieves the results close to those of a state-of-the-art method [10], but it runs much faster. Fig. 9 shows the visual results of depth upsampling and depth denoising. CLMF-0 yields piecewise smooth depth maps with much cleaner and sharper depth edges than GF visually and quantitatively.

Table 4. Quantitative evaluation for depth map upsampling in *disc.* and *all* regions (All the parameters have been fairly configured.)

Error rate %		BF	GF	[10]	CLMF-1	CLMF-0
Tsuk.	Disc.	40.3	49.6	25.1	39.1	**19.7**
	All	8.94	11.3	6.10	8.75	**4.30**
Venus	Disc.	15.0	25.6	**6.14**	12.2	9.98
	All	1.48	2.60	**0.63**	1.14	0.93
Teddy	Disc.	30.8	39.5	**26.1**	31.0	28.5
	All	11.6	15.5	**10.1**	12.0	11.6
Cones	Disc.	27.3	35.2	**23.7**	26.5	25.3
	All	13.1	17.0	**11.8**	12.9	13.1



GF, depth disc. error rate = 39.5%   CLMF-0, depth disc. error rate = 28.5%

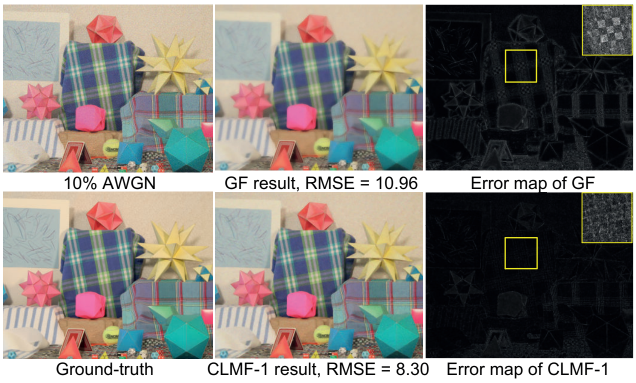
Ground-truth   10% AWGN   GF, RMSE=5.21   CLMF-0, RMSE=3.16

Figure 9. Depth upsampling results (top row, 8× upscaling in Table 4) and depth denoising results (bottom row) by GF ($\epsilon = 0.05^2$) and CLMF-0 ($\tau = 10/255$), both with $r = 9$ and the best settings.

4.3. Single Color Image Denoising

Like BF used for noise reduction [8], GF and CLMF-1 can also be applied to this task. Compared with more sophisticated methods e.g., non-local means [2], local filters do not achieve the best denoising results. However, they typically have advantages of fast speed and easy implementation [11]. Fig. 10 compares the color denoising results of CLMF-1 and GF, where the best parameter settings have been used. Compared to GF, CLMF-1 removes the additive noise adequately, achieving much better signal recovery for high-variance regions. Fig. 11 studies the effects of the parameters when varied to denoise the same image. Increasing σ_r or ϵ after the best setting, both BF and GF overly smooth the image (increased RMSE), but CLMF-1 performs consistently better thanks to τ used to select the data points for regression. We have also fixed ϵ to 0.2^2 and varied τ settings for CLMF-1. When τ is set between $20/255$ and $50/255$, quite similar denoising results are obtained. If τ is set even larger, CLMF-1 behaves increasingly close to GF.

[1] We measured our careful C++ implementation of CostFilter [14] on the same PC, but excluding its costly weighted median filtering runtime.


10% AWGN | GF result, RMSE = 10.96 | Error map of GF

Ground-truth | CLMF-1 result, RMSE = 8.30 | Error map of CLMF-1

Figure 10. Color image denoising by GF ($\epsilon = 0.2^2$) and CLMF-1 ($\tau = 30/255$, $\epsilon = 0.2^2$) with $r = 9$. Intensity scaled by 3 for insets. For 20% AWGN, RMSE is 17.25 for GF and 14.95 for CLMF-1.

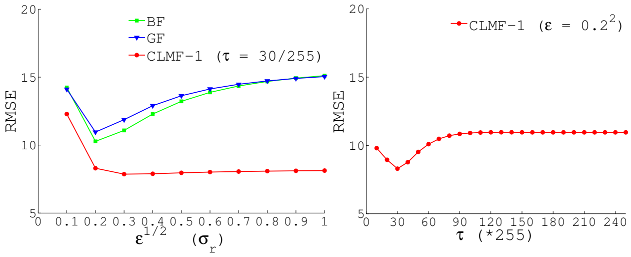


Figure 11. Color denoising performance evaluation. (a) Varying ϵ in GF and CLMF-1, or σ_r in BF. (b) Varying τ in CLMF-1.

4.4. Graphics Applications

Besides the application to image/video abstraction [16] in Fig. 5, CLMF-1 can also be used for several graphics applications. Next, we present the results for detail enhancement and flash/no-flash denoising in Fig. 12. As a gradient-preserving smooth filter, CLMF-1 achieves visually similar results as GF [6]. Both of them do not have unwanted gradient reversal artifacts in the resulting images that BF has.

5. Discussion and Future Work

This paper proposed a generic framework of performing cross-based local multipoint filtering efficiently. CLMF-0 and CLMF-1 find very competitive applications into many computer vision and graphics tasks. On the theoretical side, the current framework can be further extended to accommodate nonlocal algorithms [7]. Exploring other approaches to decide adaptive scales directionally is interesting. Our recent study has led to a O(1)-time cross construction method that greatly reduces the computational overhead. This will be reported elsewhere. Zhang *et al.* [20] showed that the cross-based technique is very friendly for GPUs, so we also plan to map our filters onto GPUs for significant speedup.

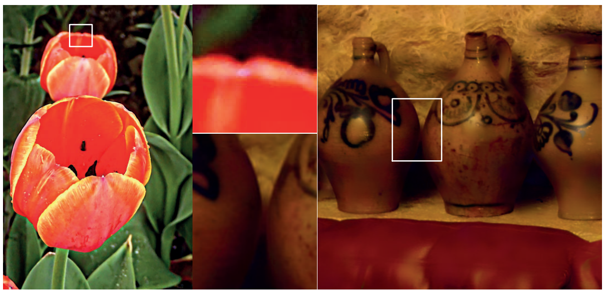


Figure 12. Detail enhancement ($r = 16$, $\tau = 60/255$, $\epsilon = 0.1^2$) and flash/no-flash denoising results ($r = 8$, $\tau = 40/255$, $\epsilon = 0.02^2$) by CLMF-1. See [6] for details and comparison with BF and GF.

References

[1] http://vision.middlebury.edu/stereo/. 7

[2] A. Buades, B. Coll, and J.-M. Morel. A non-local algorithm for image denoising. In *Proc. of CVPR*, 2005. 7

[3] A. Buades, B. Coll, and J.-M. Morel. The staircasing effect in neighborhood filters and its solution. *TIP*, 2006. 2, 4

[4] F. Crow. Summed-area tables for texture mapping. In *SIGGRAGH*, 1984. 2, 5

[5] L. De-Maeztu, S. Mattoccia, A. Villanueva, and R. Cabeza. Linear stereo matching. In *Proc. of ICCV*, 2011. 6

[6] K. He, J. Sun, and X. Tang. Guided image filtering. In *Proc. of ECCV*, 2010. 1, 2, 3, 4, 5, 6, 8

[7] V. Katkovnik, A. Foi, K. Egiazarian, and J. Astola. From local kernel to nonlocal multiple-model image denoising. *Int. Journal of Computer Vision*, 86(1):1–32, 2010. 1, 2, 3, 4, 8

[8] C. Liu, W. T. Freeman, R. Szeliski, and S. B. Kang. Noise estimation from a single image. In *CVPR*, 2006. 1, 7

[9] X. Mei, X. Sun, M. Zhou, S. Jiao, H. Wang, and X. Zhang. On building an accurate stereo matching system on graphics hardware. In *Proc. of GPUCV*, 2011. 2, 6

[10] D. Min, J. Lu, and M. N. Do. Depth video enhancement based on weighted mode filtering. *IEEE TIP*, Mar 2012. 7

[11] S. Paris, P. Kornprobst, J. Tumblin, and F. Durand. Bilateral filtering: Theory and applications. *Foundations and Trends in Comp. Graphics and Vision*, 4(1):1–73, 2008. 1, 2, 7

[12] G. Petschnigg, M. Agrawala, H. Hoppe, R. Szeliski, M. Cohen, and K. Toyama. Digital photography with flash and no-flash image pairs. In *SIGGRAGH*, 2004. 1

[13] F. Porikli. Constant time O(1) bilateral filtering. In *Proc. of CVPR*, 2008. 2

[14] C. Rhemann, A. Hosni, M. Bleyer, C. Rother, and M. Gelautz. Fast cost-volume filtering for visual correspondence and beyond. In *Proc. of CVPR*, 2011. 1, 2, 6, 7

[15] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Proc. of ICCV*, 1998. 1, 2, 4, 6

[16] H. Winnemoller, S. C. Olsen, and B. Gooch. Real-time video abstraction. In *ACM SIGGRAGH*, 2006. 5, 8

[17] Q. Yang, K. H. Tan, and N. Ahuja. Real-time O(1) bilateral filtering. In *Proc. of CVPR*, 2009. 2

[18] K. Yoon and I. Kweon. Adaptive support-weight approach for correspondence search. *IEEE PAMI*, 2006. 1, 2, 6

[19] K. Zhang, J. Lu, and G. Lafruit. Cross-based local stereo matching using orthogonal integral images. *IEEE Trans. CSVT*, 19(7):1073–1079, July 2009. 2, 3, 4, 5, 6, 7

[20] K. Zhang, J. Lu, Q. Yang, G. Lafruit, R. Lauwereins, and L. V. Gool. Real-time and accurate stereo: A scalable approach with bitwise fast voting on CUDA. *IEEE Trans. CSVT*, 21(7):867–878, July 2011. 8