Fast $O(1)$ bilateral filtering using trigonometric range kernels

Kunal Narayan Chaudhury, Daniel Sage, and Michael Unser

Abstract

It is well-known that spatial averaging can be realized (in space or frequency domain) using algorithms whose complexity does not scale with the size or shape of the filter. These fast algorithms are generally referred to as constant-time or $O(1)$ algorithms in the image processing literature. Along with the spatial filter, the edge-preserving bilateral filter [1] involves an additional range kernel. This is used to restrict the averaging to those neighborhood pixels whose intensity are similar or close to that of the pixel of interest. The range kernel operates by acting on the pixel intensities. This makes the averaging process non-linear and computationally intensive, especially when the spatial filter is large. In this paper, we show how the $O(1)$ averaging algorithms can be leveraged for realizing the bilateral filter in constant-time, by using trigonometric range kernels. This is done by generalizing the idea in [2] of using polynomial kernels. The class of trigonometric kernels turns out to be sufficiently rich, allowing for the approximation of the standard Gaussian bilateral filter. The attractive feature of our approach is that, for a fixed number of terms, the quality of approximation achieved using trigonometric kernels is much superior to that obtained in [2] using polynomials.

Index Terms

Bilateral filter, Edge-preserving smoothing, Constant-time algorithm, $O(1)$ complexity, Raised cosines.

Correspondence: Kunal N. Chaudhury (kchaudhu@princeton.edu). Kunal N. Chaudhury is currently part of the Program in Applied and Computational Mathematics (PACM), Princeton University, Princeton, NJ 08544-1000, USA. Michael Unser and Daniel Sage are with the Biomedical Imaging Group, École Polytechnique Fédérale de Lausanne, Station-17, CH-1015 Lausanne, Switzerland. This work was supported by the Swiss National Science Foundation under grant 200020-109415.

Fast $O(1)$ bilateral filtering using trigonometric range kernels

I. INTRODUCTION

The bilateral filtering of an image $f(x)$ in the general setting is given by

$$\tilde{f}(x) = \eta^{-1} \int w(x,y) \; \phi(f(x),f(y)) \; f(y) \; dy$$

where

$$\eta = \int w(x,y) \; \phi(f(x),f(y)) \; dy.$$

In this formula, $w(x,y)$ measures the geometric proximity between the pixel of interest x and a nearby pixel y. Its role is to localize the averaging to a neighborhood of x. On the other hand, the function $\phi(u,v)$ measures the similarity between the intensity of the pixel of interest $f(x)$ and its neighbor $f(y)$. The normalizing factor η is used to preserve constants, and in particular the local mean.

In this paper, we consider the so-called *unbiased form* of the bilateral filter [1], where $w(x,y)$ is translation-invariant, that is, $w(x,y) = w(x-y)$, and where the range filter is symmetric and depends on the difference of intensity, $\phi(f(x),f(y)) = \phi(f(x)-f(y))$. In this case, the filter is given by

$$\tilde{f}(x) = \eta^{-1} \int_\Omega w(y)\phi(f(x-y)-f(x))f(x-y) \; dy \tag{1}$$

where

$$\eta = \int_\Omega w(y)\phi(f(x-y)-f(x)) \; dy. \tag{2}$$

We call $w(x)$ the *spatial kernel*, and $\phi(s)$ the *range kernel*. The local support Ω of the spatial kernel specifies the neighborhood over which the averaging takes place. A popular form of the bilateral filter is one where both $w(x)$ and $\phi(s)$ are Gaussian [1], [2], [3], [4].

The edge-preserving bilateral filter was originally introduced by Tomasi et al. in [1] as a simple, non-iterative alternative to anisotropic diffusion [5]. This was motivated by the observation that while standard spatial averaging performs well in regions with homogenous intensities, it tends to performs poorly in the vicinity of sharp transitions, such as edges. For the bilateral filter in (1), the difference $f(x-y)-f(x)$ is close to zero in homogenous regions, and hence $\phi(f(x-y)-f(x)) \approx 1$. In this case, (1) simply results in the averaging of pixels in the neighborhood of the pixel of interest. On the other hand, if the pixel of interest x is in the vicinity of an edge, $\phi(f(x-y)-f(x))$ is large when $x-y$ belongs to the same side of the edge as x, and is small when $x-y$ is on the

other side of the edge. As a result, the averaging is restricted to neighborhood pixels that are on the same side of the edge as the pixel of interest. This is the basic idea which allows one to perform smoothing while preserving edges at the same time. Since its inception, the bilateral filter has found widespread use in several image processing, computer graphics, and computer vision applications. This includes denoising [6], video abstraction [7], demosaicing [8], optical-flow estimation [9], and stereo matching [10], to name a few. More recently, the bilateral filter has been extended by Baudes et al. [3] to realize the popular non-local neighborhood filter, where the similarity between pixels is measured using patches centered around the pixels.

The direct implementation of (1) turns out to be rather computationally intensive for real time applications. Several efficient numerical schemes have been proposed in the past for implementing the filter in real time, even at video rates [11], [12], [13], [14]. These algorithms (with the exception of [11]), however, do not scale well with the size of the spatial kernel, and this limits their usage in high resolution applications. A significant advance was obtained when Porikli [2] proposed a constant-time implementation of the bilateral filter (for arbitrary spatial kernels) using polynomial range kernels. The $O(1)$ algorithm was also extended to include Gaussian $\phi(s)$ by locally approximating it using polynomials. More recently, Yang et al. [4] have proposed a $O(1)$ algorithm for arbitrary range and spatial kernels by extending the bilateral filtering method of Durand et al. [11]. Their algorithm is based on a piecewise-linear approximation of the bilateral filter obtained by quantizing $\phi(s)$.

In this paper, we extend the $O(1)$ algorithm of Porikli to provide an exact implementation of the bilateral filter, using trigonometric range kernels. Our main observation that trigonometric functions share a common property of polynomials which allows one to "linearize" the otherwise non-linear bilateral filter. The common property is that the translate of a polynomial (resp. trigonometric function) is again a polynomial (resp. trigonometric function), and importantly, of the same degree. By fixing $\phi(s)$ to be a trigonometric function, we show how this self-shiftable property can be used to (locally) linearize the bilateral filter. This is the crux of the idea that was used for deriving the $O(1)$ algorithm for polynomial $\phi(s)$ in [2].

II. CONSTANT-TIME BILATERAL FILTER

A. The main idea

It is the presence of the term $\phi(f(x-y)-f(x))$ in (1) that makes the filter non-linear. In the absence of this term, that is, when $\phi(s)$ is constant, the filter is simply given by the averaging

$$\overline{f}(x) = \int_{\Omega} w(y)f(x-y)\,dy, \qquad (3)$$

where we assume $w(x)$ to have a total mass of unity. It is well-known that (3) can be implemented in constant-time, irrespective of the size and shape of the filter, using the convolution-multiplication

property of the (fast) Fourier transform. The number of computations required per pixel, however, depends on the size of the image in this case [15]. On the other hand, it is known that (3) can be realized at the cost of a constant number of operations per pixel (independent of the size of the image and the filter) using recursive algorithms. These $O(1)$ recursive algorithms are based on specialized kernels, such as the box and the hat function [16], [17], [18], and the more general class of Gaussian-like box splines [19].

Our present idea is to leverage these fast averaging algorithms by expressing (1) in terms of (3), where the averaging is performed on the image and its simple pointwise transforms. Our observation is that we can do so if the range kernel is of the form

$$\phi(s) = \cos(\gamma s) \qquad (-T \leq s \leq T). \tag{4}$$

By plugging (4) into (1), we can write the integral as

$$\cos(\gamma f(\boldsymbol{x})) \int_{\Omega} w(\boldsymbol{y}) \cos(\gamma f(\boldsymbol{x} - \boldsymbol{y})) f(\boldsymbol{x} - \boldsymbol{y}) \, d\boldsymbol{y} + \sin(\gamma f(\boldsymbol{x})) \int_{\Omega} w(\boldsymbol{y}) \sin(\gamma f(\boldsymbol{x} - \boldsymbol{y})) f(\boldsymbol{x} - \boldsymbol{y}) \, d\boldsymbol{y}.$$

This is clearly seen to be the linear combination of two spatial averages, performed on the images $\cos(\gamma f(\boldsymbol{x})) f(\boldsymbol{x})$ and $\sin(\gamma f(\boldsymbol{x})) f(\boldsymbol{x})$. Similarly, we can write the integral in (2) as

$$\cos(\gamma f(\boldsymbol{x})) \int_{\Omega} w(\boldsymbol{y}) \cos(\gamma f(\boldsymbol{x} - \boldsymbol{y})) \, d\boldsymbol{y} + \sin(\gamma f(\boldsymbol{x})) \int_{\Omega} w(\boldsymbol{y}) \sin(\gamma f(\boldsymbol{x} - \boldsymbol{y})) \, d\boldsymbol{y}.$$

In this case, the averaging is on the images $\cos(\gamma f(\boldsymbol{x}))$ and $\sin(\gamma f(\boldsymbol{x}))$. This is the trick that allows us to express (1) in terms of linear convolution filters applied to pointwise transforms of the image.

Note that the domain of $\phi(s)$ is $[-T, T]$ in (4). We assume here (without loss of generality) that the dynamic range of the image is within $[0, T]$. The maximum of $|f(\boldsymbol{x}) - f(\boldsymbol{y})|$ over all $\boldsymbol{x}$ and $\boldsymbol{y}$ such that $\boldsymbol{x} - \boldsymbol{y} \in \Omega$ is within T in this case. Therefore, by letting $\gamma = \pi/2T$, we can guarantee the argument γs of the cosine function to be within the range $[-\pi/2, \pi/2]$. The crucial point here is that the cosine function is oscillating and can assume negative values over $(-\infty, \infty)$. However, its restriction over the half-period $[-\pi/2, \pi/2]$ has two essential properties of a range kernel—it is non-negative and has a bump shape (cf. the outermost curve in Figure 1). Note that, in practice, the bound on the local variations of intensity could be much lower than T.

B. General trigonometric kernels

The above idea can easily be extended to more general trigonometric functions of the form $\phi(s) = a_0 + a_1 \cos(\gamma s) + \cdots + a_N \cos(N\gamma s)$. This is most conveniently done by writing $\phi(s)$ in terms of complex exponentials, namely as

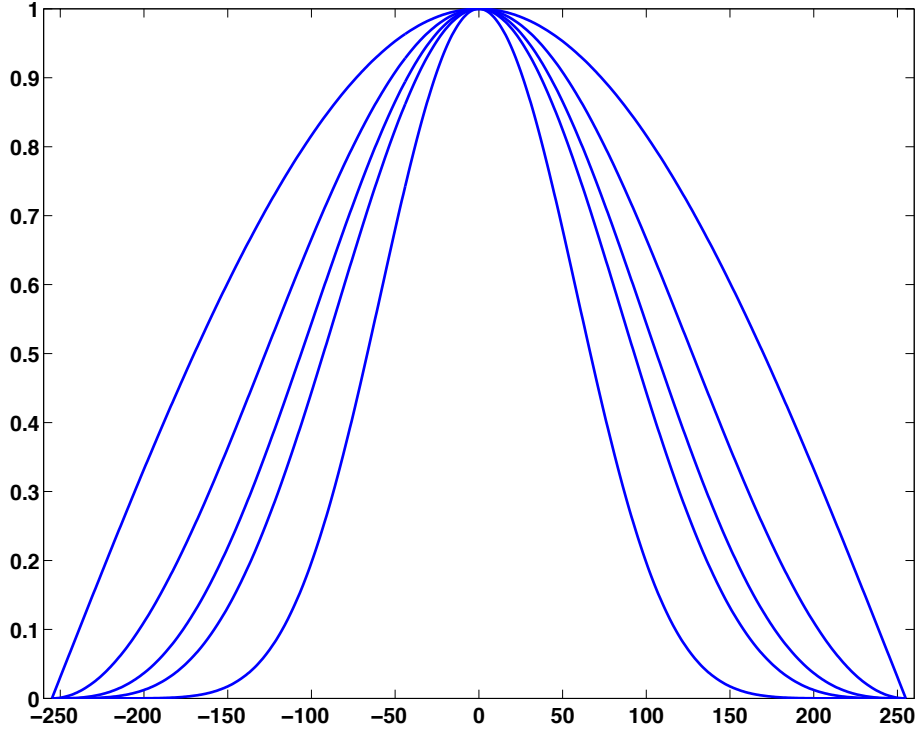$$\phi(s) = \sum_{|n| \leq N} c_n \exp(jn\gamma s). \tag{5}$$

Fig. 1. The family of raised cosines $g(s) = [\cos(\gamma s)]^N$ over the dynamic range $-T \leq s \leq T$ as N goes from 1 to 5 (outer to inner curves). We set $T = 255$ corresponding to the maximum dynamic range of a grayscale image, and $\gamma = \pi/2T$. They satisfy the two essential properties required to qualify as a valid range kernel of the bilateral filter—non-negativity and monotonicity (decay). Moreover, they have the remarkable property that they converge to a Gaussian (after appropriate normalization) as N gets large; see (7).

The coefficients c_n must be real and symmetric, since $\phi(s)$ is real and symmetric. Now, using the addition-multiplication property of exponentials, we can write

$$\phi(f(\boldsymbol{x} - \boldsymbol{y}) - f(\boldsymbol{x})) = \sum_{|n| \leq N} d_n(\boldsymbol{x}) \exp\left(jn\gamma f(\boldsymbol{x} - \boldsymbol{y})\right)$$

where $d_n(\boldsymbol{x}) = c_n \exp\left(-jn\gamma f(\boldsymbol{x})\right)$. Plugging this into (1), we immediately see that

$$\tilde{f}(\boldsymbol{x}) = \frac{\sum_{|n| \leq N} d_n(\boldsymbol{x}) \, \overline{g_n(\boldsymbol{x})}}{\sum_{|n| \leq N} d_n(\boldsymbol{x}) \, \overline{h_n(\boldsymbol{x})}} \qquad (6)$$

where $h_n(\boldsymbol{x}) = \exp\left(jn\gamma f(\boldsymbol{x})\right)$, and $g_n(\boldsymbol{x}) = f(\boldsymbol{x})h_n(\boldsymbol{x})$. We refer to $h_n(\boldsymbol{x})$ and $g_n(\boldsymbol{x})$ as the *auxiliary images*, and N as the *degree* of the kernel.

The above analysis gives us the following $O(1)$ algorithm for the bilateral filter: We first set up the auxiliary images and the coefficients $d_n(\boldsymbol{x})$ from the input image. We then average each of the auxiliary images using a $O(1)$ algorithm (this can be done in parallel). The samples of the filtered image is then given by the simple sum and division in (6). In particular, for an image of size $M \times M$, we can compute the spatial averages for any arbitrary $w(\boldsymbol{x})$ at the cost of $O(M^2 \log_2 M)$ operations
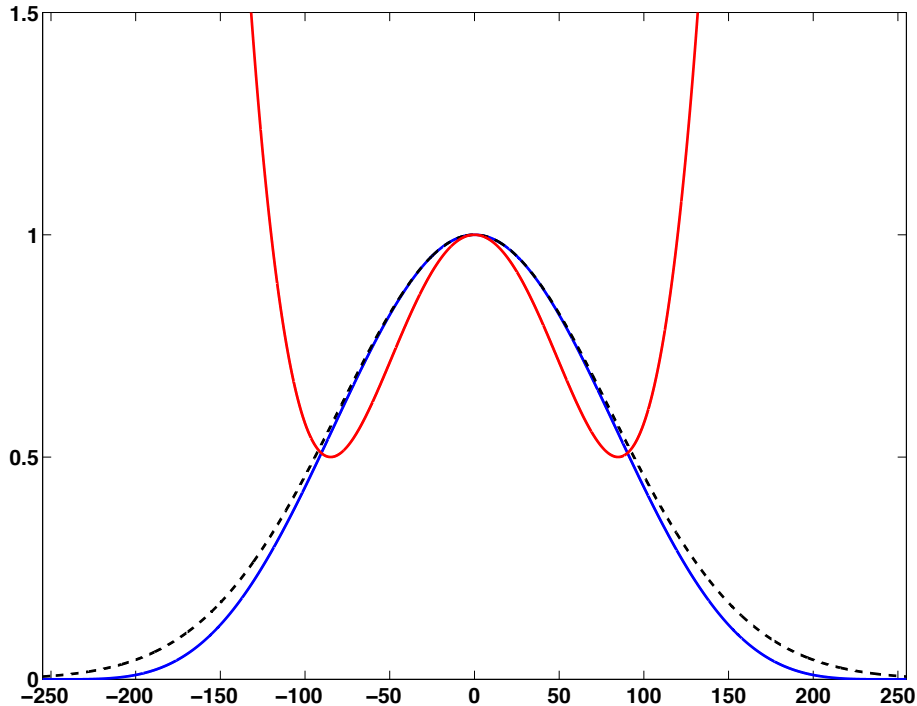
Fig. 2. Approximation of the Gaussian $\exp(-x^2/2\sigma^2)$ (dashed black curve) over the interval $[-255, 255]$ using the Taylor polynomial (solid red curve) and the raised cosine (solid blue curve). We set $\sigma = 80$, and use $N = 4$ for the raised cosine in (7). The raised cosine is of the form $a_0 + a_1 \cos(2\theta) + a_2 \cos(4\theta)$ in this case. We use a 3-term Taylor polynomial of the form $b_0 + b_1 x^2 + b_2 x^4$. It is clear that the raised cosine offers a much better approximation than its polynomial counterpart. In particular, note how the polynomial blows up beyond $|x| > 100$.

using the Fourier transform. As mentioned earlier, this can further be reduced to a total of $O(M^2)$ operations using specialized spatial kernels [16], [17], [19].

C. Raised cosines

We now address the fact that $\phi(s)$ must have some additional properties to qualify as a valid range kernel (besides being symmetric). Namely, $\phi(s)$ must be non-negative, and must be monotonic in that $\phi(s_1) \leq \phi(s_2)$ whenever $|s_1| > |s_2|$. In particular, it must have a peak at the origin. This ensures that large differences in intensity gets more penalized than small differences, and that (1) behaves purely as a spatial filter in a region having uniform intensity. Moreover, one must also have some control on the variance (effective width) of $\phi(s)$. We now address these design problems in order.

The properties of symmetry, non-negativity, and monotonicity are simultaneously enjoyed by the family of *raised cosines* of the form

$$\phi(s) = \big[\cos(\gamma s)\big]^N \qquad (-T \leq s \leq T).$$

Writing $\cos\theta = (e^{j\theta} + e^{-j\theta})/2$, and applying the binomial theorem, we see that

$$\phi(s) = \sum_{n=0}^{N} 2^{-N} \binom{N}{n} \exp\left(j(2n-N)\gamma s\right).$$

This expresses the raised cosines as in (5), though we have used a slightly different summation. Since $\phi(s)$ has a total of $(N+1)$ terms, this gives a total of $2(N+1)$ auxiliary images in (6). The central term $n = N/2$ is constant when N is even, and we have one less auxiliary image to process in this case.

D. Approximation of Gaussian kernels

Figure 1 shows the raised cosines of degree $N = 1$ to $N = 5$. It is seen that $\phi(s)$ become more Gaussian-like over the half-period $[-\pi, \pi]$ with the increase in N. The fact, however, is that $\phi(s)$ converges pointwise to zero at all points as N gets large, excepting for the node points $0, \pm\pi, \pm 2\pi, \ldots$. This problem can nevertheless be addressed by suitably scaling the raised coinse. The precise result is given by the following pointwise convergence:

$$\lim_{N \longrightarrow \infty} \left[\cos\left(\frac{\gamma s}{\sqrt{N}}\right)\right]^N = \exp\left(-\frac{\gamma^2 s^2}{2}\right). \tag{7}$$

Proof: Note that Taylor's theorem with remainder tells us that if $f(x)$ is sufficiently smooth, then $f(x) = \sum_{k=0}^{m-1} x^k f^{(k)}(0)/k! + x^m f^{(m)}(\theta)/m!$, where θ is some number between 0 and x. Applied to the cosine function, we have $\cos(x) = 1 - x^2/2 + x^4 \cos\theta/24$. In other words, $\cos(x) = 1 - x^2/2 + r(x)$, where $|r(x)| \lesssim x^4$ (we write $f(x) \lesssim g(x)$ to signify that $f(x) \leq C g(x)$ for some absolute constant C, where C is independent of x). Using this estimate, along with the binomial theorem, we can write

$$\left[\cos\left(\frac{\gamma s}{\sqrt{N}}\right)\right]^N = \left(1 - \frac{\gamma^2 s^2}{2N}\right)^N + \sum_{k=1}^{N} \binom{N}{k} r(s,N)^k \left(1 - \frac{\gamma^2 s^2}{2N}\right)^{N-k},$$

where $|r(s,N)| \lesssim s^4/N^2$. We are almost done since it is well-known that $(1 + x/N)^N$ approaches $\exp(x)$ as N gets large. To establish (7), all we need to show is that, for any fixed s, the residual terms can be made negligibly small simply by setting N large.

Now note that if $|s| \lesssim N^{1/2}$, then the magnitude of $(1 - \gamma^2 s^2/2N)$ is within unity, and, on the other hand, $s^4/N < 1$ when $|s| < N^{1/4}$. Thus, given any fixed s, we set N to be large enough so that s satisfies the above bounds. Then, following the trivial inequality $\binom{N}{k} < N^k$, we see that the modulus of the residual is

$$\lesssim \sum_{k=1}^{N} N^k \left(\frac{s^4}{N^2}\right)^k \lesssim N \left(\frac{s^4}{N}\right)^N \lesssim \frac{1}{N},$$

provided that $|s| < L_N = (N^{1-2/N})^{1/4}$. This can clearly be achieved by increasing N, since L_N is monotonic in N. ∎

We have seen that raised cosines of sufficiently large order provide arbitrarily close approximations of the Gaussian. The crucial feature about (7) is that the rate of convergence is much faster than that of Taylor polynomials, which were used to approximate the Gaussian range kernel in [2]. In particular, we can obtain an approximation comparable to that achieved using polynomials using fewer number of terms. This is important from the practical standpoint. In Figure 2, we consider the target Gaussian kernel $\exp(-s^2/2\sigma^2)$, where $\sigma = 80$. We approximate this using the raised cosine of degree 4, which has 3 terms. We also plot the polynomial corresponding to the 3-term Taylor expansion of the Gaussian, which is used in for approximating the Gaussian in [2]. It is clear that the approximation quality of the raised cosine is superior to that offered by a Taylor polynomial having equal number of terms. In particular, note that the Taylor approximation does not automatically offer the crucial monotonic property.

TABLE I

N_0 IS THE MINIMUM DEGREE OF THE RAISED COSINE REQUIRED TO APPROXIMATE A GAUSSIAN OF STANDARD DEVIATION σ ON THE INTERVAL $[-255, 255]$. THE ESTIMATE $\lceil (\gamma\sigma)^{-2} \rceil$ IS ALSO SHOWN.

σ	200	150	100	80	60	50	40
N_0	1	2	3	4	5	7	9
$\lceil (\gamma\sigma)^{-2} \rceil$	1	2	3	5	8	11	17

E. Control of the width of range kernel

The approximation in (7) also suggests a means of controlling the variance of the raised cosine, namely, by controlling the variance of the target Gaussian. The target Gaussian (with normalization) has a fixed variance of γ^{-2}. This can be increased simply by rescaling the argument of the cosine in (7) by some $\rho > 1$. In particular, for sufficiently large N,

$$\left[\cos\left(\frac{\gamma s}{\rho\sqrt{N}} \right) \right]^N \approx \exp\left(-\frac{s^2}{2\rho^2\gamma^{-2}} \right). \tag{8}$$

The variance of the target Gaussian (again with normalization) has now increased to $\rho^2\gamma^{-2}$. A fairly accurate estimate of the variance of the raised cosine is therefore $\sigma^2 \approx \rho^2\gamma^{-2}$. In particular, we can increase the variance simply by setting $\rho = \gamma\sigma$ for all $\sigma > \gamma^{-2}$, provided N is large enough.

Bringing down the variance below γ^{-2}, on the other hand, is more subtle. This cannot be achieved simply by rescaling with $\rho < 1$ on account of the oscillatory nature of the cosine. For instance, setting $\rho < 1$ can cause $\phi(s)$ to become non-negative, or loose its monotonicity. The only way of doing so is by increasing the degree of the cosine (cf. Figure 1). In particular, N must be large enough so that the argument of $\cos(\cdot)$ is within $[-\pi/2, \pi/2]$ for all $-T \leq s \leq T$. This is the case if

$$N \geq \rho^{-2} \approx (\gamma\sigma)^{-2}.$$

In other words, to approximate a Gaussian having a small variance σ, N must at least be as large as $N_0 \approx (\gamma\sigma)^{-2}$. The bound is quite tight for large σ, but is loose when σ is small. We empirically determined N_0 for certain values of σ for the case $T = 255$, some of which are given in Table I. It turned out to be much lower than the estimate $(\gamma\sigma)^{-2}$ when σ is small. For a fixed setting of T (e.g., for grayscale images), this suggests the use of a lookup table for determining N_0 for small σ on-the-fly.

The above analysis leads us to an $O(1)$ algorithm for approximating the Gaussian bilateral filtering, where both the spatial and range filters are Gaussians. The steps are summarized in Algorithm 1.

Algorithm 1 Fast $O(1)$ bilateral filtering for the Gaussian kernel

---

Input: Image $f(x)$, dynamic range $[-T, T]$, σ_s^2 and σ_r^2 for the spatial and range filters.

1. Set $\gamma = \pi/2T$, and $\rho = \gamma\sigma_r$.

2. If $\sigma_r > \gamma^{-2}$, pick any large N. Else, set $N = (\gamma\sigma_r)^{-2}$, or use a look-up table to fix N.

3. For $0 \leq n \leq N$, set up the images $h_n(x) = \exp\left(j\gamma(2n - N)f(x)/\rho\sqrt{N}\right)$ and $g_n(x) = f(x)h_n(x)$, and the coefficients $d_n(x) = 2^{-N}\binom{N}{n}\exp\left(-j\gamma(2n - N)f(x)/\rho\sqrt{N}\right)$.

4. Use an $O(1)$ algorithm to filter $h_n(x)$ and $g_n(x)$ with a Gaussian of variance σ_s^2 to get $\overline{h_n}(x)$ and $\overline{g_n}(x)$.

5. Set $\tilde{f}(x)$ as the ratio of $\sum_{n=0}^{N} d_n(x)\overline{g_n}(x)$ and $\sum_{n=0}^{N} d_n(x)\overline{h_n}(x)$.

Return: Filtered image $\tilde{f}(x)$.

III. EXPERIMENTS

We implemented the proposed algorithm for Gaussian bilateral filtering in Java on a Mac OS X 2× Quad core 2.66 GHz machine, as an ImageJ plugin. We used multi-threading for computing the spatial averages of the auxiliary images in parallel. A recursive $O(1)$ algorithm was used for implementing the Gaussian filter in space domain [15]. The average times required for processing a 720×540 grayscale image using our algorithm are shown in Table II. We repeated the experiment for different variances of the Gaussian range kernel, and at different spatial variances. As seen from the table, the processing time is quite fast compared to a direct implementation of the bilateral filter, which requires considerable time depending on the size of the spatial filter. For instance, a direct implementation of the filter on a 512×512 image required 4 seconds for σ_s as low as 3 on our machine (using discretized Gaussians supported on $[-3\sigma, 3\sigma]^2$), and this climbed up to almost 10 seconds for $\sigma_s = 10$. As is seen from Table II, the processing time of our algorithm, however, suddenly shoots up for narrow Gaussians with $\sigma_r < 15$. This is due to the large N required to approximate the Gaussian in this regime (cf. Table I). We have figured out an approximation scheme

for further accelerating the processing for very small σ_r, without appreciably degarding the final output. Discussion of this method is however beyond the present scope of the paper.

We next tried a visual comparisonof the ouput of our algorithm with the algorithm in [2]. In Figure 3, we compare the outputs of the two algorithms with the direct implementation, on a natural grayscale image. As is clearly seen from the processed images, our result resembles the exact output very closely. The result obtained using the polynomial kernel, on the other hand, shows strange artifacts. The difference is also clear from the standard deviation of the error between the exact output and the approximations. We note, however, that the execution time of the polynomial method is slightly lower than that of our method, since it requires half the number of auxiliary images for a given degree.

We also tested our implementation of the Gaussian bilateral filter on color (RGB) images. We tried a naive processing, where each of the three color channels were processed independently. The results on a couple of images are shown in Figure 4. The Java source code can be downloaded from the web at http://bigwww.epfl.ch/algorithms/bilateral-filter.

TABLE II

THE TIME IN MILLISECONDS REQUIRED FOR PROCESSING A GRAYSCALE IMAGE OF SIZE 720×540 PIXELS USING OUR ALGORITHM. THE PROCESSING WAS DONE ON A MAC OS X, $2\times$ QUAD CORE 2.66 GHz MACHINE, USING MULTITHREADING.

$\sigma_r \rightarrow$	10	20	30	40	50	60	70	80	90	100
$\sigma_s = 10$	3604	452	195	120	74	61	49	34	32	27
$\sigma_s = 100$	3755	482	217	127	89	69	54	43	37	28

IV. DISCUSSION

We presented a general method of computing the bilateral filter in constant-time using trigonometric range kernels. Within this framework, we showed how feasible range kernels can be realized using the family of raised cosines. The highlights of our approach are the following:

• **Accuracy**. Our method is exact, at least for the family of raised cosines. It does not require the quantization of the range kernel, as is the case in [11], [4]. Moreover, note that the auxiliary images in (6) have the same dynamic range as the input image irrespective of the degree N. This is unlike the situation in [2], where the dynamic range of the auxiliary images grow exponentially with the N. This makes the computations susceptible to numerical errors for large N.

• **Speed**. Besides having $O(1)$ complexity, our algorithm can also be implemented in parallel. This allows us to further accelerate its speed.
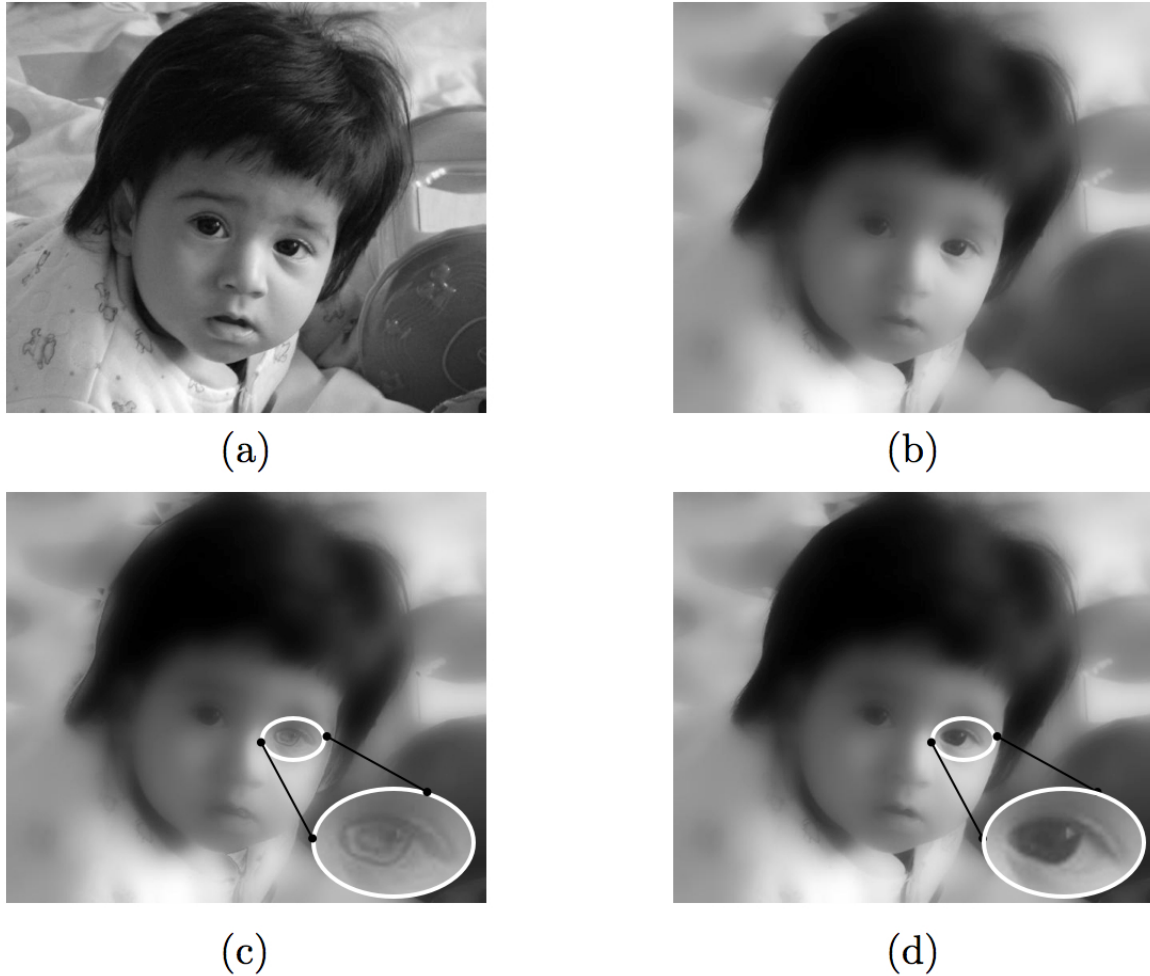
Fig. 3. Comparison of various implementations of the Gaussian bilateral filter on the grayscale image *Isha* of size 600×512. The filter settings are $\sigma_s = 15$ and $\sigma_r = 80$. (a) Original image; (b) Direct implementation of the bilateral filter; (c) Output obtained using polynomial kernel [2]; and (d) Output of our algorithm. Note the strange artifacts in (c), particularly around the right eye (see zoomed insets). This is on account of the distortion caused by the polynomial approximation shown in Figure 2. The standard deviation of the error between (b) and (c) is 6.5, while that between (b) and (d) is 1.2.

● **Approximation property**. Trigonometric functions yield better (local) approximation of Gaussians than polynomials. In particular, we showed that by using a particular class of raised cosines, we can obtain much better approximations of the Gaussian range kernel than that offered by the Taylor polynomials in [2]. The final output is artifact-free and resembles the true output very closely. The only flip side of our approach (this is also the case with [2], as noted in [4]) is that a large number of terms are required to approximate very narrow Gaussians over large intervals.

● **Space-variant extension**. The spatial kernel in (1) can be changed from point-to-point within the image to control the amount of smoothing (particularly in homogenous regions), while the

Fig. 4. Results on the color images *Greekdome* and *Tulip*, using our implementation of the Gaussian bilateral filter. The original image is on the left, and the processed image is on the right. In either case, the red, green, and blue channels were processed independently. We used $\sigma_s = 10$ and $\sigma_r = 20$ for *Greekdome*, and $\sigma_s = 20$ and $\sigma_r = 60$ for *Tulip*. (Images courtsey of Sylvain Paris and Frédo Durand).

range kernel is kept fixed. Thanks to (6), this can be done simply by computing the space-variant averages of each auxiliary image. The good news is that this can also be realized for a $M \times M$ image at the cost of $O(M^2)$ operations, using particular spatial kernels. This includes the two-dimensional box and hat filter [16], [17], and the more general class of Gaussian-like box splines in [19].

V. ACKNOWLEDGEMENT

REFERENCES

[1] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," *IEEE International Conference on Computer Vision*, pp. 839–846, 1998.

[2] F. Porikli, "Constant time O(1) bilateral filtering," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2008.

[3] A. Buades, B. Coll, and J.M. Morel, "A review of image denoising algorithms, with a new one," *Multiscale Modeling and Simulation*, vol. 4, pp. 490–530, 2005.

[4] Q. Yang, K.-H. Tan, and N. Ahuja, "Real-time O(1) bilateral filtering," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 557–564, 2009.

[5] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 7, pp. 629–639, 1990.

[6] E.P. Bennett, J.L. Mason, and L. McMillan, "Multispectral bilateral video fusion," *IEEE Transactions on Image Processing*, vol. 16, pp. 1185–1194, 2007.

[7] H. Winnemöller, S. C. Olsen, and B. Gooch, "Real-time video abstraction," *ACM Siggraph*, pp. 1221–1226, 2006.

[8] R. Ramanath and W. E. Snyder, "Adaptive demosaicking," *Journal of Electronic Imaging*, vol. 12, pp. 633–642, 2003.

[9] J. Xiao, H. Cheng, H. Sawhney, C. Rao, and M. Isnardi, "Bilateral filtering-based optical flow estimation with occlusion detection," *European Conference on Computer Vision*, pp. 211–224, 2006.

[10] Q. Yang, L. Wang, R. Yang, H. Stewenius, and D. Nister, "Stereo matching with color-weighted correlation, hierarchical belief propagation and occlusion handling," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 31, pp. 492–504, 2009.

[11] F. Durand and J. Dorsey, "Fast bilateral filtering for the display of high-dynamic-range images," *ACM Siggraph*, vol. 21, pp. 257–266, 2002.

[12] B. Weiss, "Fast median and bilateral filtering," *ACM Siggraph*, vol. 25, pp. 519–526, 2006.

[13] T.Q. Pham and L.J. van Vliet, "Separable bilateral filtering for fast video preprocessing," *IEEE International Conference on Multimedia and Expo*, pp. 1–4, 2005.

[14] S. Paris and F. Durand, "A fast approximation of the bilateral filter using a signal processing approach," *European Conference on Computer Vision*, pp. 568–580, 2006.

[15] I. Young, J. Gerbrands, and L. van Vliet, *Fundamentals of Image Processing*, Delft PH Publications, 1995.

[16] P.S. Heckbert, "Filtering by repeated integration," *International Confernece on Computer Graphics and Interactive Techniques*, vol. 20, no. 4, pp. 315–321, 1986.

[17] F. C. Crow, "Summed-area tables for texture mapping," *ACM Siggraph*, vol. 18, pp. 207–212, 1984.

[18] I.T. Young and L.J. van Vliet, "Recursive implementation of the Gaussian filter," *Signal Processing*, vol. 44, no. 2, pp. 139–151, 1995.

[19] K.N. Chaudhury, A. Muñoz-Barrutia, and M. Unser, "Fast space-variant elliptical filtering using box splines," *IEEE Transactions on Image Processing*, vol. 19, pp. 2290–2306, 2010.