Fast self-guided filter with decimated box filters

Dragomir El Mezeni, Lazar Saranovac
Department of electronics
University of Belgrade, School of Electrical Engineering
Belgrade, Serbia

Abstract—**Guided filter is very popular edge-aware filter which is already used in many commercial applications and is included into several widely spread image processing toolboxes. Although it's main computation is in evaluation of 4 box filters with complexity O(N) it can still be expensive if used for high resolution images. Also integral sums needed to achieve this complexity generate pressure on the memory side of the system since several full frame buffers are needed. Fast guided filter offers acceleration of $O(N/S^2)$ if input image is decimated by factor S and filter coefficients are calculated in this decimated domain. We continue on this trail and argue that with addition of simple pre-filtering stage quality of approximation measured in PSNR can be increased by 21dB in average. We also propose a way for efficient calculation of these pre-filters, called decimated box filters, without increasing algorithmic complexity.**

Keywords-guided filter; fast edge-aware filtering; high dynamic range imaging; decimation;

I. INTRODUCTION

Edge-aware filters have become one of the essential tools for image processing and computer vision over the last decade. They are used in a variety of applications such as image segmentation, de-noising, stylization, tone mapping, detail manipulation etc. The main goal of edge-aware filters is to selectively smooth the image by removing noise and small details while preserving significant edges. This property enables image de-noising which preserves sharpness of the original image. Also in applications that use image decomposition to base and detail layers usage of edge-aware filters prevent formation of halo artifacts.

There are many algorithms available with edge-aware properties and the most popular among them are bilateral filter [1], anisotropic diffusion [2], domain transform filter [3] and guided filter [4]. Bilateral filter have gained the most popularity over the years since it is easy for understanding and implementation, it is non-iterative and it was among the first approaches suggested for solving edge-aware filtering problem.

Bilateral filter belongs to the class of weighted average filters, where each output pixel is calculated as weighted average of all pixels inside the window centered on currently processed input pixel. In standard weighted average filters weights depend only on spatial distance from the central pixel, giving larger weights to the pixels that are close to the central pixel. If the input pixel is near the strong edge and if there are pixels from both sides of the edge inside averaging window, average output value will be in between of these two extremes producing blurry edge. To prevent this, besides spatial weight additional range weight is introduced and the final weight is a product of these two. Range weight depends on value difference from central pixel, giving lager weight to the pixels which value is similar to the input pixel value. Near the strong edge, all pixels on the other side of the edge, having very different value from the input pixel, will have range weight close to zero, thus not participating into calculation of output pixel value, providing edge preserving property.

Since the kernel of bilateral filter is spatially varying, bilateral filtering cannot be realized with linear convolution. Straightforward implementation of bilateral filter requires $O(NR^2)$ operations for entire image, where R is the size of the filter window and N is the number of pixels in the image. Complexity of bilateral filter rises very quickly with filter size increase making filters with large kernels highly inefficient. Since in many applications size of the filter is proportional to the input image size, and since image resolution is rapidly increasing with 16MPix images being common in today's consumer electronics, it is very important to enable efficient implementation of large edge-aware filters. This need has motivated many authors to search for the more efficient approach proposing solutions described in [5]-[9]. Approaches which exploit down-sampling [5][6] are very interesting since their efficiency rises with kernel size increase meaning that they are the most efficient for the large kernels.

In a search for more efficient edge-aware filtering guided filter is introduced by He et al in [4]. It is used for filtering an image by using constraints imposed by the gradients in another, guide, image. If the same image is used for filtering and for guidance than this filter is called self-guided and can be used in edge-aware filtering applications. It has better edge preserving properties than bilateral filter since it does not suffer from gradient reversals around strong edges. Its implementation is also very efficient and requires just 4 box filters and few scalar operations. Since box filters can be calculated in a constant time, independent of the filter size once the integral image is available, complexity of Guided filter per output pixel is O(N). However this approach requires calculation of 4 different integral images, requiring 4 full frame storage and even more since bit-depth needs to be significantly increased for integral value. For smaller kernels this can be prevented by calculating partial integral images, but for filters with large filter kernels problem of memory is still present. Although guided filter gained great popularity over the last few years and is even incorporated into official versions of Matlab and OpenCV there are not many papers dealing with its optimizations. There is a

technical note [10] from the authors of the filter suggesting optimization trough down-sampling reducing its complexity for full image calculation from O(N) to O(N/S^2), where N is the number of pixels in the image and S is down-sampling factor. This optimization, while simple it is very significant since it impacts both aspects of implementation by reducing memory requirements and the number of operations needed.

Because of decimation step some information loss is introduced by this optimization and it needs to be quantified. Author in [10] provides only few example pictures showing that by visual inspection they provide almost indistinguishable results. This has motivated us to analyze further this optimization approach by quantifying the quality of the approximation and provide some guidelines for parameter selection. We have found that while following optimization approach described in [10] will produce visually good images for standard 8-bit inputs, much better results can be obtained by using simple box pre-filter before decimation. Increased quality of approximation is can be crucial when dealing with images with higher precision such as high dynamic range images.

II. GUIDED FILTER

Guided filter structure is depicted in Fig 1.

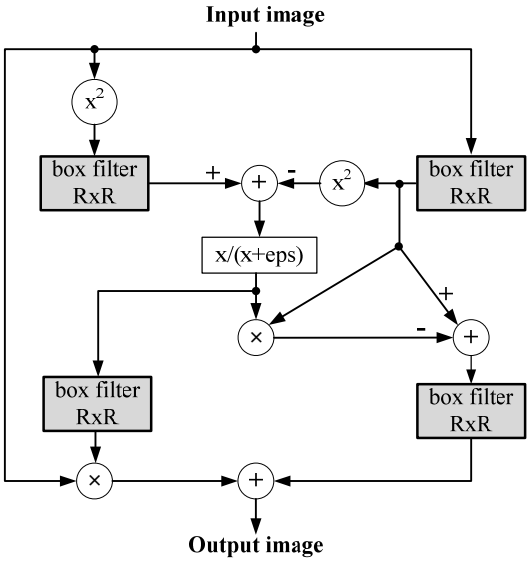

Figure 1. Guided filter algorithm structure

Most of the operations are point-wise additions and multiplications which can be efficiently implemented. There is one division operation which can be approximated using some linear or polynomial equivalent or by usage of look-up table with stored pre-calculated results. The most compute intensive operations are 4 box filters of size RxR. Box filters are separable and can be calculated in O(RN) time where N is the total number of pixels and R is the size of box filter. This poses a problem for filters with large kernels because calculation time can rise significantly. Also at least R input lines need to be fetched in order to start processing which poses significant memory requirement for large kernels and high resolution images. Box filters can be calculated in O(N) time if integral image is used. Memory requirements for storing 4 full integral images are more than 4N since higher precision needs to be used for integral value. Also multiple accesses to nonconsecutive memory locations are needed in order to fetch integral values needed for filter output calculations. Because of these restrictions guided filter is usually limited to relatively small filter kernels in applications with constrained memory and high performance requirements.

III. FAST GUIDED FILTER

In order to popularize usage of guided filter in scenarios where large kernels are needed, authors of guided filter suggest in [10] simple optimization depicted in Fig 2.

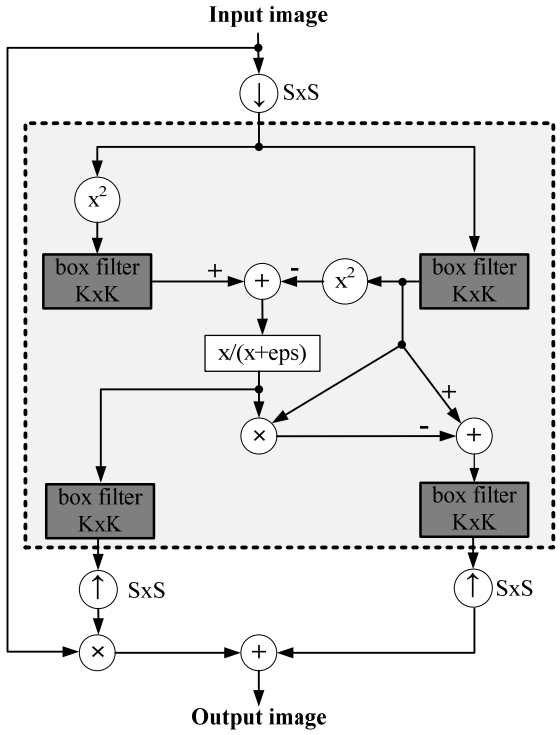

Figure 2. Fast guided filter algorithm structure as sugested in [10]

The main idea behind this optimization is that since averaging filters represent low-pass filters, signal can be decimated without a fear of aliasing. The basis of this idea is found in [5][6] where decimation both in spatial domain and range is used to accelerate bilateral filter. Proposed position of decimation in [10] is at the very beginning of the processing, and entire calculation is done in decimated domain. Filter coefficients are up-sampled, using bilinear interpolation, and combined with input samples to obtain output image. For producing effect equivalent to RxR guided filter input image can be decimated by factor SxS and all box filters can be scaled to size KxK where K=R/S. Benefits of this approach are following:

- Box filter support is reduced to K=R/S, making direct realization favorable over integral image if K is reduced to small kernel, thus relaxing memory requirements.

- Majority of operations and all 4 box filters are calculated on decimated image, reducing number of operations and required memory by factor S^2.

Complexity of the fast guided filter is reduced to $O(N/S^2)$ if integral sum is used or to $O(RN/S^3)$ where direct approach is used, having in mind box filter separability.

Authors in [10] didn't provide any numerical proof of the quality of theirs approximation. They show several example images where guided filter is used in different scenarios and show that two results for given images and parameters are indistinguishable.

IV. Optimization with decimated box filter

If we carefully examine solution proposed in [10], depicted in Fig. 2, it could be noticed that there is no low-pass filtering before decimation step. Since decimation implies lowering the sampling frequency it could lead to overlapping in frequency domain, thus producing aliasing. This can be prevented, or at least reduced by applying low-pass filter before decimation thus narrowing available spectral components and reducing aliasing. Structure of proposed fast guided with appropriate pre-filtering step is presented in Fig. 3.

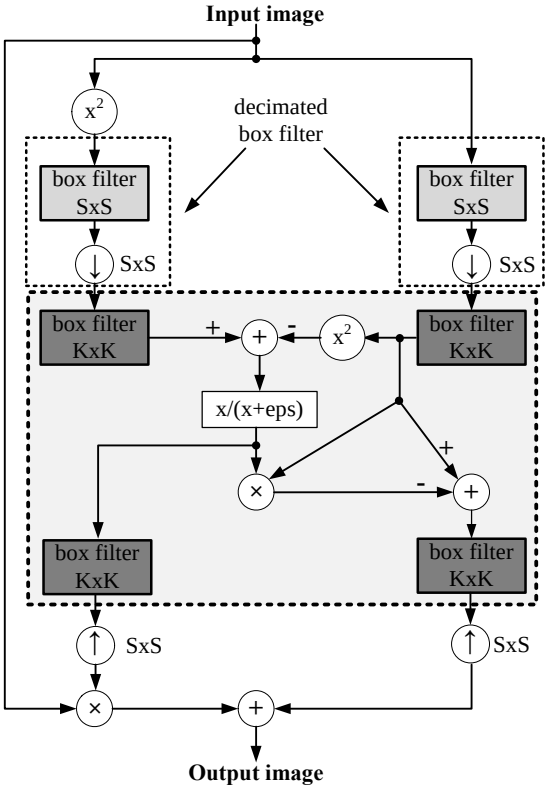

Figure 3. Fast guided filter using decimated box filters

Pre-filtering can be achieved with simple box filter of size SxS where S is down-sampling ratio. Altought it seems that output of pre-filters of size SxS must be calculated for full resolution input image, since it is immediately followed by decimation step it could be further optimized. We combine box filter and decimation of size SxS into single step called ***decimated box filter*** as outlined in Fig 3. with dashed box around them. Algorithm for efficient calculation of decimated box filter is illustrated in Fig. 4.

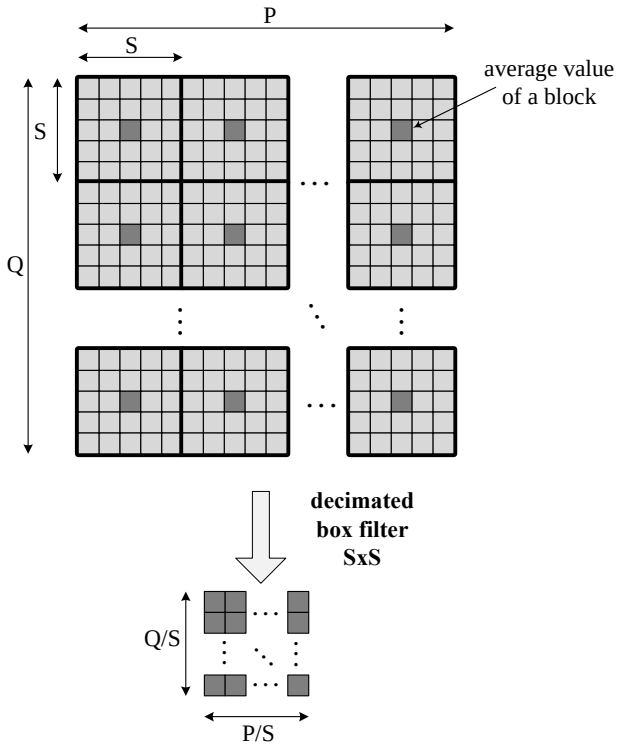

Figure 4. Calculation of decimated box filter

Since after decimation only small number of output pixels from the filtering stage, depicted darker in Fig. 4., will propagate further there is no need to caclulate full output of the filtering stage. Input image is instead divided into disjuncitve blocks of size SxS and mean value of each block will be propagated to the next stage as the output of decimated box filter. Since each input pixel is used for only one output pixel complexity of decimated box filter is O(N) and is independent of the filter size by nature, without need for integral image calculation. Also there is no need for any additional memory since decimated box filter can be calculated on-the-fly while reading input samples from memory.

V. Results and analysis

We have used several images with different proportion of edges and smooth regions for testing quality of two fast guided filters. All used images have resolution of 6MPix. Since image village contains lots of edges it is the best reference for testing approximation quality, and so this image is used for all the diagrams unless explicitly stated differently.

Figure 5. Images used for testing

Approximation quality is measured in terms of PSNR w.r.t. image processed with equivalent parameters using original guided filter as described in [4]. If an input image is processed with a fast guided filter using down-scaling ratio SxS and filter in decimated domain of size KxK then reference image was processed with original guided filter using filter of size RxR where R=KS. Input images were always padded before filtering. Two optimization approaches are compared: FGF – fast guided filter proposed in [10] and FGF-DBF – fast guided filter with decimated box filter proposed in this paper. We have tested three different aspects of these approximations. First we test how division of original kernel size R between S and K influence quality of approximations. Secondly we testes the influence of selectivity parameter epsilon to the approximation quality. And last we compare execution time of original guided filter implementation and two approximations.

A. Approximation quality for different values of K and S

We have tested PSNR value for three different images, over wide range of decimation factor S and for 3 different kernel sizes K in decimated domain. Equivalent filter produced with this parameter used as a reference is a filter of size R=KS. Same tests were executed using fast guided filter from [10] and proposed fast guided filter with decimated box filters. Results are presented in Fig. 6. – Fig. 8.

We noticed that by introduction of decimated box filters quality of approximation rises significantly with average PSNR improvement of 21dB for image *statue* which contain moderate amount of edges. For images with more edges, such as image *village*, this improvement rises to 24dB, while for images with less edges, such as image *girl*, can fall down to 16dB. It is very interesting to notice that quality of approximation is mostly influenced by choice of parameter K while down-scale factor has little influence. This is important conclusion since it implies that for chosen filter size K in decimated domain, very large filters can be realized by increasing down-scale ratio S with almost no degradation in PSNR. Improvement is consistent over different images, producing somewhat lower PSNR for image village which was expected since this image is full of different edges.

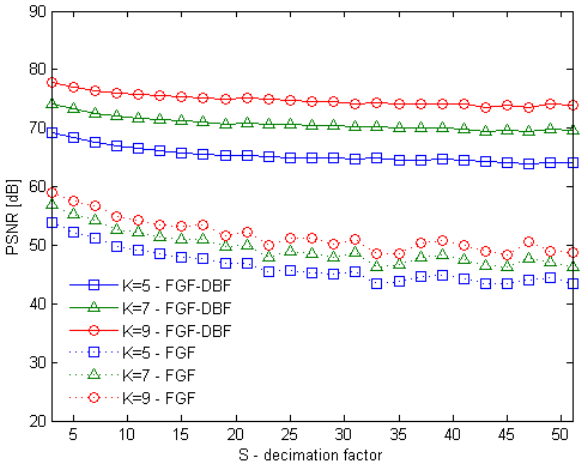

Figure 6. Approximation quality for image *statue*

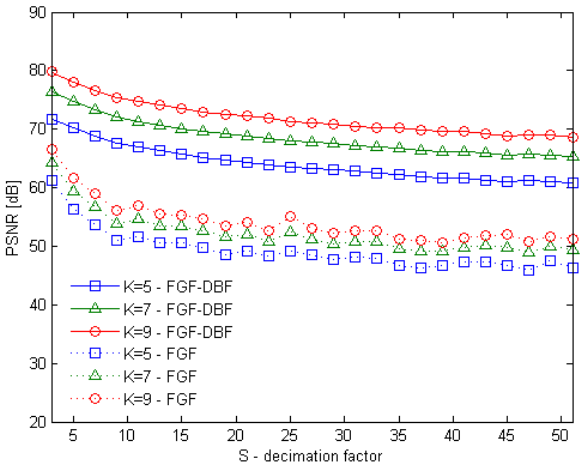

Figure 7. Approximation quality for image *girl*

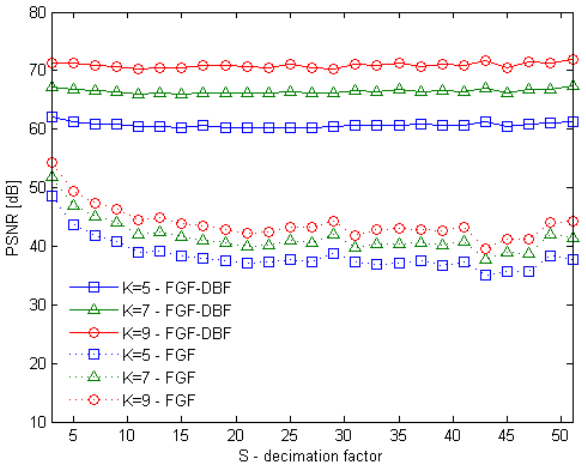

Figure 8. Approximation quality for image *village*

Since PSNR value is stable over a wide range of decimation factors for fixed filter size K, in the next test we varied these filter sizes while keeping decimation factor S fixed to 25. Result of that test is presented in Fig. 9. For better approximations lager decimated filter sizes should be used.

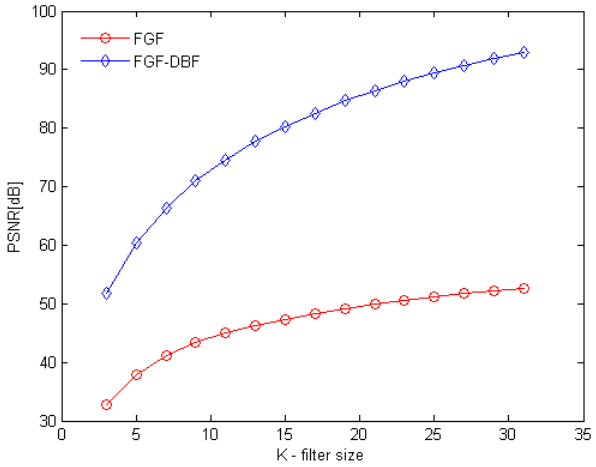

Figure 9. Approximation quality for fixed down-scaling factor S=25

B. *Influenace of selectivity parameter eps*

Parameter *eps* controls edge selectivity of guided filter. Since different values of this parameter are used for different applications it is important to evaluate how this parameter influence the quality of proposed guided filter approximation. For this test we used filter size K=7, while varying scale ratio and parameter *eps*, and results are shown in Fig. 10.

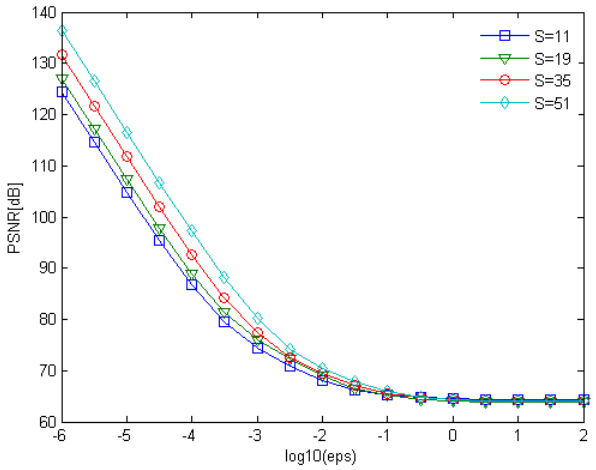

Figure 10. Inffluence of selectivity parameter eps to approximation quality for filter size K=7 over different down-scaling factors.

For large values of parameter *eps* guided filter converges to simple box filter losing all of its edge-aware properties, while for very small values of parameter *eps* it converges to identity, thus just copying input values to the output. As shown in Fig. 10. approximation quality rises for smaller values of parameter *eps* implying that this approximation is safe for use in a variety of edge-aware applications.

C. *Execution time*

Execution time is tested using simple Matlab scripts which implement described solutions. Since parameter K has most influence on approximation quality, we have tested its influence on execution time while keeping downscale ratio constant S=25. As shown in Fig. 11. both fast implementations have much lower execution time than original implementation.

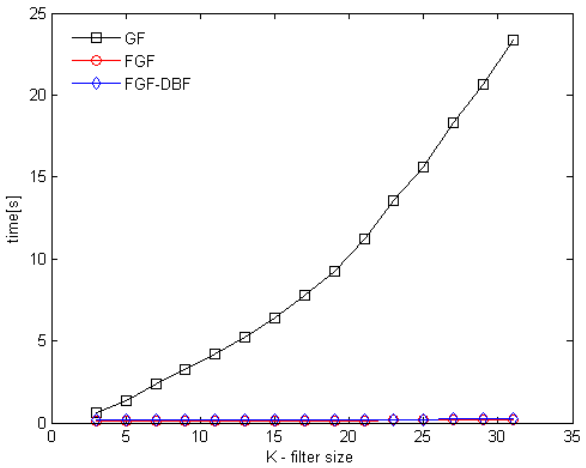

Figure 11. Execution time of different guiede filter implementations with fixed down-scaling factor S=25. Compared realizations are: GF – original guided filter [4], FGF – fast guided filter [10], FGF-DBF – fast guided filter with decimated box filters.

To better examine execution time trend for fast guided filter implementations we plot only them in Fig. 12. We can notice that although proposed FGF-DBF is somewhat slower than FGF because of pre-filter operations, slope of those diagrams are almost identical meaning that those two approaches are the same in terms of scalability.

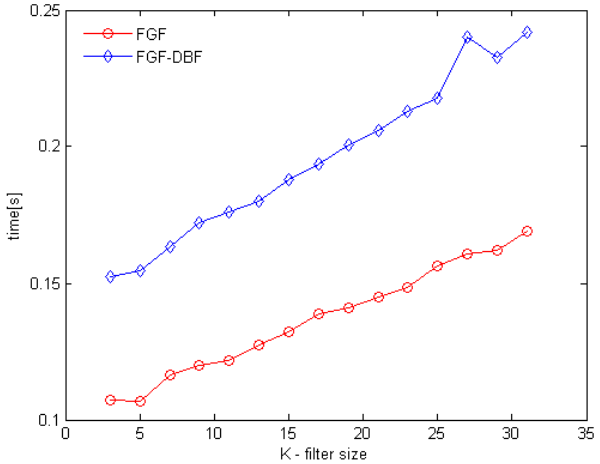

Figure 12. Execution time of two fast guided filter implementations with fixed down-scaling factor S=25. Compared realizations are: FGF – fast guided filter [10], FGF-DBF – fast guided filter with decimated box filters.

VI. Conclusion

We have shown that introduction of pre-filtering stage before decimation can significantly improve quality of the fast guided filter approximation. Our results show that PSNR value is changing a little with decimation factor increase and that it mostly depends on the size of the filters used in decimation domain. Since box filter used for pre-filtering step is immediately followed by decimation it can be efficiently calculated in constant time without need for integral sums, making complexity of proposed approach equivalent to the original fast guided filter.

These findings enable efficient implementation, in terms of both memory and computation, of guided filters with large kernels opening possibilities for they usage in embedded applications with limited resources and real-time requirements. Increased precision of approximation enables its usage even for filtering high dynamic range images.

References

[1] Tomasi, Carlo, and Roberto Manduchi. "Bilateral filtering for gray and color images." Computer Vision, 1998. Sixth International Conference on. IEEE, 1998.

[2] Perona, Pietro, and Jitendra Malik. "Scale-space and edge detection using anisotropic diffusion." Pattern Analysis and Machine Intelligence, IEEE Transactions on 12.7 (1990): 629-639.

[3] Gastal, Eduardo SL, and Manuel M. Oliveira. "Domain transform for edge-aware image and video processing." ACM Transactions on Graphics (TOG). Vol. 30. No. 4. ACM, 2011.

[4] He, Kaiming, Jian Sun, and Xiaoou Tang. "Guided image filtering." Computer Vision–ECCV 2010. Springer Berlin Heidelberg, 2010. 1-14.

[5] Durand, Frédo, and Julie Dorsey. "Fast bilateral filtering for the display of high-dynamic-range images." ACM transactions on graphics (TOG). Vol. 21. No. 3. ACM, 2002.

[6] Paris, Sylvain, and Frédo Durand. "A fast approximation of the bilateral filter using a signal processing approach." International journal of computer vision 81.1 (2009): 24-52.

[7] Pham, Tuan Q., and Lucas J. Van Vliet. "Separable bilateral filtering for fast video preprocessing." Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on. IEEE, 2005.

[8] Yang, Qingxiong, Kar-Han Tan, and Narendra Ahuja. "Real-time O (1) bilateral filtering." Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on. IEEE, 2009.

[9] Weiss, Ben. "Fast median and bilateral filtering." Acm Transactions on Graphics (TOG). Vol. 25. No. 3. ACM, 2006.

[10] He, Kaiming, and Jian Sun. "Fast guided filter." arXiv preprint arXiv:1505.00996 (2015), unpublished.