Evaluating Popular Non-Linear Image Processing Filters for their Use in Regularized Iterative CT

Wei Xu and Klaus Mueller, *Senior Member, IEEE*

***Abstract*– Iterative CT algorithms are becoming increasingly popular in recent years, and have been found useful when the projections are limited in number, irregularly spaced, or noisy, which are imaging scenarios often encountered in low-dose imaging and compressed sensing. One way to cope with the associated streak and noise artifacts in these settings is either to incorporate or to interleave a regularization objective into the iterative reconstruction framework. In this paper we explore possible techniques for the latter. We investigate a number of non-linear filters popular in the image processing literature for their suitability in iterative CT application, here OS-SIRT.**

I. Introduction and Related Works

In image processing and computer vision, edge-preserving image smoothing is widely used for visual appearance-preserving contrast reduction, image de-noising and multi-scale image decomposition. These methods are usually non-linear filters which reduce irregular artifacts or unwanted details while preserving edges. Recently, they have also been applied in medical imaging for image registration, segmentation and reconstruction.

Iterative methods are preferable when the recovery problem becomes ill-posed, for example, when the data are noisy, few-view or limited in angle. Experiments have shown that the OS-SIRT ordered subsets scheme can increase the performance in both quality and speed [5]. We have used OS-SIRT here to demonstrate the improvements that can be obtained with non-linear image processing filters forming the regularization mechanisms. OS-SIRT is a generalization of SIRT and SART, with SIRT having just one and SART having *M* subsets (with *M* being the number of projections). Its correction update is computed as:

$$v_j^{(k+1)} = v_j^{(k)} + \lambda \frac{\sum_{p_i \in OS_s} \frac{p_i - r_i}{\sum_{l=1}^{N} w_{il}}}{\sum_{i=1}^{N} w_{ij}} \qquad r_i = \sum_{l=1}^{N} w_{il} \cdot v_l^{(k)} \tag{1}$$

where OS_s is the subset *s* containing *M/s* projections and λ is the relaxation factor. In our regularized OS-SIRT, filtering is applied for each iteration as a final step after backprojecting all subsets. This removes artifacts at their early onset when the errors are just generated and thus steers the reconstruction towards more plausible and favorable solution regions. Just like the CT reconstruction all regularization computations are also fully GPU-accelerated, and so do not require any expensive texture upload/download operations between the CPU and GPU.

In [6] we introduced the use of the Bilateral Filter and compared it with the TVM filter in a computational context. In this current work, we expanded our exploration to the Non-Local Means and Trilateral filter and also focused primarily on quality issues.

Wei Xu and Klaus Mueller are with the Center for Visual Computing, Computer Science Department, Stony Brook University, NY 11794 USA. (e-mail: {wxu, mueller}@cs.sunysb.edu).

II. Methodology

A. Bilateral Filter

The bilateral filter (BF) [4] is an edge-preserving non-linear filter which averages similar and nearby pixels values. A fixed window area is usually used to achieve fast and accurate computation:

$$BF(x) = \frac{\sum_{t \in W} c(t) s(f(x), f(x+t)) \cdot f(x+t)}{\sum_{t \in W} c(t) s(f(x), f(x+t))} \tag{2}$$

Here, *W* is the window centered at *x*, *t* and *x* represent the spatial variables, *f* is the input image, and *c* and *s* are the measured closeness and pixel value similarity, respectively:

$$c(t) = e^{-\frac{\|t\|^2}{2 \cdot \sigma_d^2}} \qquad s(f(x), f(x+t)) = e^{-\frac{(f(x) - f(x+t))^2}{2 \cdot \sigma_r^2}} \tag{3}$$

where σ_d and σ_r control the amount of smoothing. The closeness function is a domain filter to average nearby pixels while the similarity function is a range filter used to exclude dissimilar pixels. Normalization forces the sum of pixel weights to 1. A detailed implementation of GPU-accelerated bilateral filtering can be found in [5].

B. Non-Local Means

Non-Local means (NLM) [1] is a non-linear filter that replaces the pixel located at *x* with a mean of the pixels whose Gaussian neighborhood looks similar to the neighborhood of *x*:

$$NLM(x) = \frac{\sum_{y \in W} e^{-\frac{\sum_{t \in N} G_a(t) |f(x+t) - f(y+t)|^2}{h^2}} f(y)}{\sum_{y \in W} e^{-\frac{\sum_{t \in N} G_a(t) |f(x+t) - f(y+t)|^2}{h^2}}} \tag{4}$$

Here, *x*, *y* and *t* are spatial variables, *W* is the window centered at *x*, *N* is the neighborhood centered at *x* or *y*, G_a is a Gaussian kernel of zero-mean and standard deviation *a*, and *h* acts as a filtering parameter when increased, the weights to dissimilar pixels are increased to allow for more smoothing. In contrast to BF, NLM removes the spatial smoothing form but increases the dimension of the range filter. The comparison of the similarity of two neighborhoods, represented by the high dimensional vectors is applied. This modification brings more accuracy to the smoothing but also costs more time. The GPU acceleration of NLM is very similar to BF.

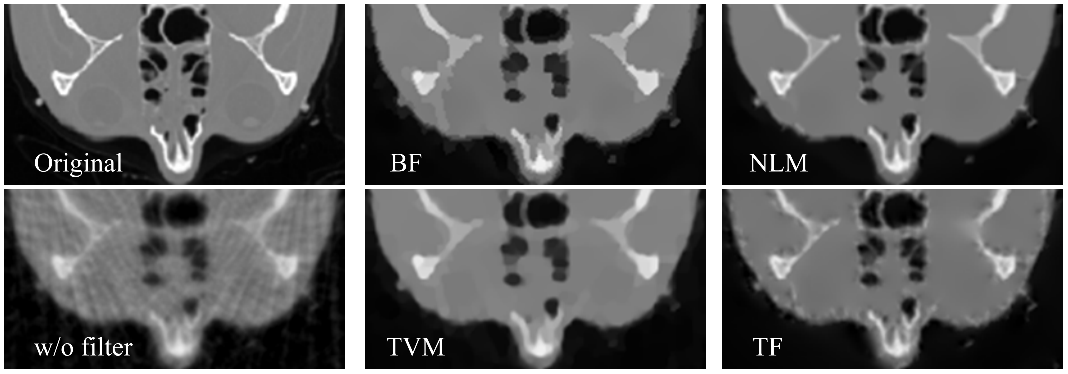

Figure 1: Reconstruction quality comparison among all filters.

C. *Trilateral Filter*

In order to accommodate the preservation of gradients, which the BF cannot do, the trilateral filter (TF) is introduced [3]. The TF is a BF that is tilted according to the smoothed gradient within an automatically generated adaptive neighborhood. With the gradient encoded a piecewise linear approximation is achieved, while an adaptive neighborhood to exclude outliers allows a sharp bound.

The implementation of TF includes three parts: generate the tilting vector, approximate the adaptive neighborhood and apply the filter. The tilting vector G_θ is essentially the bilaterally filtered gradient:

$$G_\theta(x) = \frac{\sum_{t\in W} c(t)s(\nabla f(x), \nabla f(x+t)) \cdot \nabla f(x+t)}{\sum_{t\in W} c(t)s(\nabla f(x), \nabla f(x+t))} \qquad (5)$$

where the discrete gradient ∇ is approximated by forward differencing. Then the tilted plane *P(x,t)* through *f(x)* and the new range differences $f_\Delta(x,t)$ are generated:

$$P(x,t) = f(x) + G_\theta \cdot t \quad f_\Delta(x,t) = f(x+t) - P(x,t) \qquad (6)$$

Finally, another BF to smooth the range differences is applied:

$$TF(x) = f(x) + \frac{\sum_{t\in W} c(t)s(f_\Delta(x,t))\, f_\theta(x,t) \cdot f_\Delta(x,t)}{\sum_{t\in W} c(t)s(f_\Delta(x,t))\, f_\theta(x,t)} \qquad (7)$$

where $f_\theta(x,t)$ is the adaptive neighborhood with similar gradient magnitude. The TF itself includes 7 internal parameters which could be computed automatically when given one parameter. We implemented all TF stages including parameter computation on the GPU by applying several passes.

D. *Total Variation Minimization*

We selected a TVM implementation algorithm [2] to compare it with other filters. The solution u is the argument that minimizes the following energy functional:

$$\min_u \left\{ \frac{\|u(x) - f(x)\|^2}{2\lambda} + \sum_{x\in\Omega} |\nabla u(x)| \right\} \qquad (8)$$

where Ω is the image domain, x is the spatial variable, f is the input image and λ is a smoothing parameter. The TV of u is:

$$\sum_\Omega |\nabla u| = \sum_\Omega \sqrt{\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2} \qquad (9)$$

where x and y are the horizontal and vertical coordinates, respectively. The minimization is transformed according to its dual formulation, and a semi-implicit gradient descent algorithm is used to compute the nonlinear projection of f. The solution u is then obtained until convergence:

$$u = f - \lambda \operatorname{div} p^n \qquad p^{n+1} = \frac{p^n + \tau\nabla(\operatorname{div} p^n - f/\lambda)}{1 + \tau\left|\nabla(\operatorname{div} p^n - f/\lambda)\right|} \qquad (10)$$

Here, div is the divergence. In practice, when $\tau \le 1/4$ the algorithm converges.

III. Results and Conclusions

Our experiments were conducted on an NVIDIA GTX 280 GPU, programmed with GLSL and an Intel Core 2 Quad CPU @ 2.66GHz. We used the Visible Human's Human Head test image (size 256^2) to evaluate the performance of the different filters. We obtained 30 projections at uniform angular spacing of [-90^0, $+90^0$] in a parallel projection viewing geometry. OS-SIRT with 5 subsets was used as the reconstruction algorithm. The enlarged details of the reconstructed images with 30 iterations for all four regularization filters are shown in Fig. 1.The parameters to use for each filter were selected through exhaustive benchmark tests.

We observe that all filters successfully remove streaks while keeping salient features in different levels. BF and TVM perform very similarly in this case but they also remove small details and widen some structures, such as bones. TF and NLM form another similarly behaving pair. They successfully solve the problems encountered with BF and TVM. However, TF still includes some "randomly distributed" artifacts at locations where the corresponding filtered gradients still include artifacts. We conclude that the NLM performs best among the tested filters.

Having established the general strengths and weaknesses of these filters in terms of quality within an iterative reconstruction framework, our current focus is a more comprehensive exploration with a varied selection of datasets.

References

[1] A. Buades, B. Coll, and J. M. Morel, "A Review of Image Denoising Algorithms, with A New One," Multiscale Model. Simul., vol. 4, no. 2, pp. 490–530, 2005.

[2] A. Chambolle, "An Algorithm for Total Variation Minimizations and Applications," J. Math. Imaging Vis., vol. 20(1-2), pp. 89-97, 2004.

[3] P. Choudhury, J. Tumblin, "The Trilateral Filter for High Contrast Images and Meshes," EG Symp. on Rendering, pp. 186-196, 2003.

[4] C. Tomasi, R. Manduchi, "Bilateral Filtering for Gray and Color Images," Proc. Sixth Int'l Conf. Comp. Vision, pp. 839–846, Jan. 1998.

[5] F. Xu, W. Xu and Klaus Mueller, "On the Efficiency of Iterative Ordered Subset Reconstruction Algorithms for Acceleration on GPUs", Computer Methods and Programs in Biomedicine, 2010 (in press).

[6] W. Xu, K. Mueller, "A Performance-Driven Study of Regularization Methods for GPU-Accelerated Iterative CT," Proc. 10th Int'l Meeting on Fully 3D Image Recon. in Rad. and Nuc. Med, pp. 20-23, Sep. 2009.