Brook+ SAMPLE

NLM Denoising

1 Introduction

Crystal clarity is an important quality of our pictures. Both a scientist at a space research organization and an Internet user streaming video on her desktop can benefit equally from an efficient system of noise removal. Naturally, denoising is a heavily investigated problem in image processing. A common method to remove noise from images is blurring or smoothing by convolving the image with a filter[1]. Though this is a simple method, it destroys fine texture and detail in the image because it is localized to small neighborhoods around each pixel. In this sample, we illustrate Non-Local Means (NLM) Image Denoising. It factors in global image information to perform denoising.

2 Non-Local-Means Image Denoising

Image denoising algorithms generally replace every pixel in the noisy image by some average of pixels in a spatial neighborhood of that pixel. However, pixels that are located in the same region may not necessarily be similar; for example, around edges. Averaging these can destroy the edge and other details. Non-local means that filtering generalizes the idea of a pixel's "neighborhood" to the set of pixels that have similar looking neighborhoods. These can be distributed across the entire image. In other words, the neighborhood now is specified by some "color distance," instead of a Euclidean distance.

The central idea of NLM denoising is to replace every pixel **x** by a weighted average of all the pixels in its generalized neighborhood. For a pixel **y** in this generalized neighborhood, the weight given to **y** is a function of how similar its Gaussian neighborhood is to the Gaussian neighborhood of **x**. This means pixels whose color distance from the pixel **x** is small have higher weight. The Euclidean distance of the pixel **y** is also a factor in computing the weight given to it. The motivation behind this is that pixels nearer to **x** are more likely to be similar to it.

The NLM filter is specified by the following formula [1].

$$NLu(\mathbf{x}) = \frac{1}{C(\mathbf{x})} \int_{\Omega} e^{-\frac{(G_\rho * |u(\mathbf{x}+\,) - u(\mathbf{y}+.)|^2)(0)}{h^2}} \; u(\mathbf{y}) \; d\mathbf{y}$$

Define ***u*** to be the input noisy image, and Ω a spatial neighborhood around pixel **x**, with **x** as its center; **G** is a Gaussian kernel with standard deviation ρ, and **C(x)** is the normalization factor. Then, the denoised value of pixel **x**, denoted by **NLu(x)**, can be obtained by computing the integral shown above, where **h** is a filtering parameter.

1. In a separate Brook+ sample, we demonstrate a simple image filtering algorithm.

3 NLM Image Denoising with Brook+

For the Brook+ implementation, we choose a 7 x 7 neighborhood to compute color distance and similarity. Also, we follow a block-partitioned implementation, where the image is broken up into blocks of size 8 x 8, and the search for similar pixels is restricted to inside each block. This is different from the original algorithm that searches the entire image. The program completes in two passes: the first pass computes the weights for each pixel; the second pass outputs the weighted average of the 8 x 8 window around every pixel. We discount pixels that have weights lower than a specified threshold. Figure 1 shows an image denoised with the NLM algorithm.

Figure 1 Left: Noisy Input Image; Middle: Denoised Image; Right: Reference Image

4 References

1. Buades, A., Coll, B., and Morel, J. 2005. A Non-Local Algorithm for Image Denoising. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cvpr'05) - Volume 2 -Volume 02* (June 20 - 26, 2005). CVPR. IEEE Computer Society, Washington, DC, 60-65. DOI= http://dx.doi.org/10.1109/CVPR.2005.38

2. Buades, A., Coll, B., and Morel, J. 2008. Nonlocal Image and Movie Denoising. *Int. J. Comput. Vision* 76, 2 (Feb. 2008), 123-139. DOI= http://dx.doi.org/10.1007/s11263-007-0052-1

Contact

Advanced Micro Devices, Inc.
One AMD Place
P.O. Box 3453
Sunnyvale, CA, 94088-3453
Phone: +1.408.749.4000

For Stream Computing:
URL: www.amd.com/stream
Questions: streamcomputing@amd.com
Developing: streamdeveloper@amd.com
Forum: www.amd.com/streamdevforum