A Fast Method for Computing Local Image Statistics with Rectangular Stencils

John Weatherwax

***Abstract*— In this paper we present a computationally fast method at computing a local images statistics using a moving window filter. The method we present can be much faster than the more conventional method of programming this algorithm, while being conceptually easy to understand and implement.**

I. Introduction

In image processing a very fundamental operation is that of spacial filtering [1], [2]. When *all* the coefficients of the stencil are one, this physically represents something very simple. The resulting filtered image is proportional to replacing each pixel in the original with its local average. In this statement, the local pixels to be averaged are specified by the filtering stencil. The filtered image and the local average image are equal when the former is divided by the number of elements in the stencil.

In this paper we present a computationally fast method at computing the locally averaged image. In addition, the computational method we present enables the efficient computation of higher order local statistics such as variance, skewness, kurtosis, etc. [3] using generalizations of the ideas presented here. For the computational method we present to be maximally efficient the stencils used must be rectangular or constructed from rectangular pieces. In addition, each stencil coefficient must be the same value. This implies that the stencil can have *no* zero elements. More general stencil types are possible but the efficiency of the method presented here is then lost. In figure 1 we present a representative sample of stencils the method described here will work efficiently for. To avoid descriptive complications the method will be described using the simplest stencil possible. Generalizations to more complicated shapes such as those shown in figure 1 involve slight modifications of the ideas presented here and will not be discussed.

The outline of this brief paper is as follows. In section II we present a description of the algorithm as it would be applied to a square stencil of odd dimension. We next comment on the algorithms computational complexity compared with that of the more naive implementation of the local filtering algorithm. Finally, in section III we conclude.

Manuscript received January 1, 2005; revised December 30, 2005. This work was sponsored by the U.S. Government under Air Force Contract F19628-00-C-0002. Options, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Government.

J. Weatherwax is with Lincoln Laboratory, Massachusetts Institute of Technology.

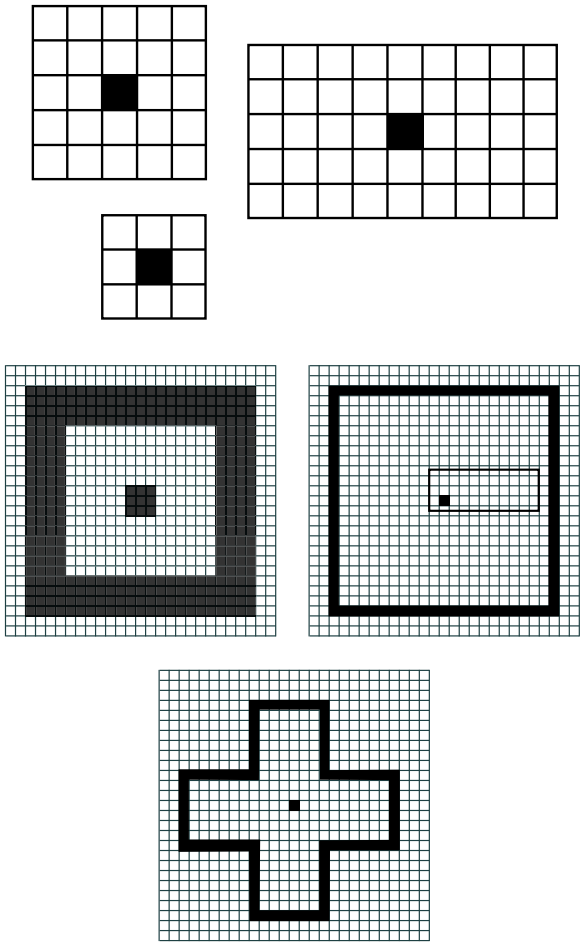

Fig. 1. A set of representative stencils to which the filtering method described in this paper maybe applied. In the top three stencils, it is imagined that each white box represents a neighbor of the center black box. The value placed in the black pixel in the filtered image represents the average over all pixels in the stencil (white and black). In the bottom three stencils, the average in the original image is taken over all black pixels and its value is placed in the center black pixel.

II. Algorithm Description

Assuming an input image I with the intensity at pixel (i, j) represented by $I_{i,j}$ and of dimensions $M \times N$, a kernel filtering operation is described mathematically [1], with the following equation

$$\bar{I}_{i,j} = \frac{\sum_{p=-d}^{d} \sum_{q=-d}^{d} W_{p,q}\, I_{i+p,j+q}}{\sum_{p=-d}^{d} \sum_{q=-d}^{d} W_{p,q}} . \quad (1)$$

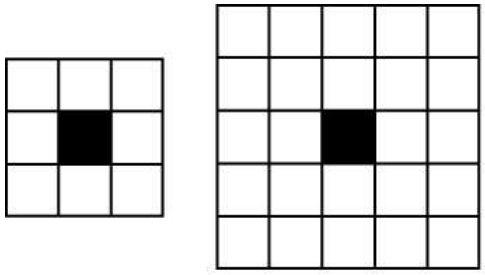

Fig. 2. Typical example square stencils of odd size used to describe the algorithm.

In this expression, we have assumed a *square* stencil of size $2d+1$ along each side with $W_{p,q}$ representing the weights applied to each of the respective pixel values centered on and surrounding $I_{i,j}$. See figure 2 for two example stencils. This process is applied to *each* pixel in the input image and an new output image $\bar{I}$ is created. In the application of equation 1 over all pixels in the input image, boundary conditions will have to be specified. This can be done in a number of ways and as its specification is not a conceptually important part of this algorithm it will not be discussed further.

In this paper we will be concerned only with *rectangular* stencils with *all* unit coefficients. We note that the case of all stencil coefficients constant, but not equal to one, is simply a multiplicative scaling of the stencil with all ones and is thus a trivial modification. Stencils with zero components and non–constant coefficients can be handled by this method, although the resulting formulas are complicated and it is felt that few computational benefits would result in these cases. Least our method seem too specific, it is noted that the types of stencils described are ideal at computing local image statistics namely means and variances.

Under the assumption of unit coefficients equation 1 becomes

$$\bar{I}_{i,j} = \frac{1}{N_{\text{stencil}}} \sum_{p=-d}^{d} \sum_{q=-d}^{d} I_{i+p,j+q} \,. \tag{2}$$

Where $N_{\text{stencil}} = (2d+1)^2$ is the number of pixels in the stencil under consideration. Here equation 2 represents creation of new image $\bar{I}$, representing the *average* of the pixels in I specified by the processing stencil. We will for brevity sometimes call the image $\bar{I}$ the "mean" image.

We note that the ability to compute an image representing local averages, translates directly into the ability to compute an image representing local higher order moments. For example, the computation of the "variance" image is obtained by computing the mean image (using equation 2) of the following image

$$D = (I - \bar{I})^2 \,. \tag{3}$$

Here image subtraction is done point–wise. Higher order moments can be obtained with generalizations to these ideas.

The algorithm begins by first computing the cumulative summation along each row of the input image I. This is represented with the following equation

$$\tilde{I}_{i,j} = \sum_{q=1}^{j} I_{i,q} \,. \tag{4}$$

The next step is to perform a cumulative summation along the columns of the image $\tilde{I}$. This is given by the following equation

$$\hat{I}_{i,j} = \sum_{p=1}^{i} \tilde{I}_{p,j} \,. \tag{5}$$

Note that both these operations can be easily performed in most computational environments. For example in MATLAB$^{\text{TM}}$the *cumsum* command can be used to accomplish each transformation.

Taken together, the $\hat{I}$ image can be expressed directly in terms of the I image by the following equation

$$\hat{I}_{i,j} = \sum_{p=1}^{i} \sum_{q=1}^{j} I_{p,q} \,. \tag{6}$$

The mean image $\bar{I}$ is now computed from the image $\hat{I}$ with the *four* additions and *one* division. The equation expressing this is given by

$$\begin{aligned} N_{\text{stencil}} \bar{I}_{i,j} &= \hat{I}_{i+d,j+d} - \hat{I}_{i-d-1,j+d} \\ &- \hat{I}_{i+d,j-d-1} + \hat{I}_{i-d-1,j-d-1} \,. \end{aligned} \tag{7}$$

Here for equation formatting purposes we have not explicitly performed the division by N_{stencil}.

Each subtraction in equation 7 represents the removal of the cumulative summation of all points to the north–west of the pixel of interest. Similarly each addition represents the addition of the cumulative summation of all points to the north–west of the pixel of interest. Graphically this is represented in figure 3. In that figure, shading is used to represent the number of pixels that are influenced by each term in the above equation. Darker shading represents that a given pixel is influenced by more terms in equation 7. Specifically, the term $\hat{I}_{i+d,j+d}$ includes every pixel north–west of the lower right stencil corner of the stencil in the summation. The second term $-\hat{I}_{i-d-1,j+d}$ subtracts the sum of all pixels to the north–west of the upper right corner of the stencil. The third term $-\hat{I}_{i+d,j-d-1}$ subtracts the cumsum of all pixels to the north-west of the lower left corner of the stencil. At this point these three operations have resulted in double counting of the pixels to the north–west of the upper left corner of the stencil. The addition of the fourth term $\hat{I}_{i-d-1,j-d-1}$ corrects this deficiency by adding back the doubly subtracted terms. For the more algebraically inclined, a proof of this result will be given in the appendix. In the following subsection we discuss this algorithms complexity and compare it to the complexity of a more naive implementation of equation 2.

A. Algorithmic Complexity

In this subsection we briefly discuss the algorithmic complexity of this algorithm for square stencils. A naive implementation of the algorithm represented by equation 2 would

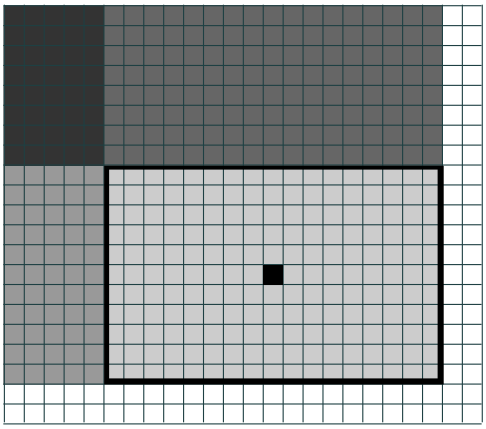

Fig. 3. Graphical representation of equation 7.

require one to perform $O((2d+1))^2)$ multiplications and summations for *each* pixel in the original image resulting in $O((2d+1)^2MN)$ computations in total. In the algorithm described in this paper both equations 4, and 5 require $O(NM)$ additions (only). The equation 7 requires $O(4NM)$ additions resulting in a total of $O(6NM)$ calculations. Thus this algorithm will represents a significant computational savings over the naive implementation when

$$6 \ll (2d+1)^2\,, \tag{8}$$

or

$$d \gg \frac{\sqrt{6}-1}{2} \approx 0.7247\,. \tag{9}$$

Thus, in fact for the 3×3 stencil shown in figure 2 this new algorithm is more efficient than the naive implementation. In addition the efficiency improves the larger the stencil size.

III. Conclusion

In this paper we have presented a computationally fast method at computing local pixel averages. This method has been shown to be sufficiently faster than the more obvious implementation of filter averaging. Since the computation of local higher order statistics are a generalization of local averaging the method presented here is valid also for quickly computing additional statistics such as variance. In addition it was shown that the larger the stencil size the greater the computational efficiency of this new method. It is hoped that this paper will enable the image processing community to quickly benefit from the improved speeds possible with this algorithm.

Appendix
Proof of Equation 7

Inserting equation 6 into the first line in equation 7, and performing the subtraction over the p index we get

$$\begin{aligned}\hat{I}_{i+d,j+d} - \hat{I}_{i-d-1,j+d} &= \\ \sum_{p=1}^{i+d}\sum_{q=1}^{j+d} I_{p,q} - \sum_{p=1}^{i-d-1}\sum_{q=1}^{j+d} I_{p,q} &= \\ \sum_{p=i-d}^{i+d}\sum_{q=1}^{j+d} I_{p,q}\,. \end{aligned} \tag{10}$$

Now substituting equation 6 into the (negative) of the second line of equation 7 and again performing the subtraction over the p index, we get

$$\begin{aligned}\hat{I}_{i+d,j-d-1} - \hat{I}_{i-d-1,j-d-1} &= \\ \sum_{p=1}^{i+d}\sum_{q=1}^{j-d-1} I_{p,q} - \sum_{p=1}^{i-d-1}\sum_{q=1}^{j-d-1} I_{p,q} &= \\ \sum_{p=i-d}^{i+d}\sum_{q=1}^{j-d-1} I_{p,q}\,. \end{aligned} \tag{11}$$

Now subtracting equation 11 from equation 10 and performing the subtraction over the q index, we get

$$\sum_{p=i-d}^{i+d}\sum_{q=1}^{j+d} I_{p,q} - \sum_{p=i-d}^{i+d}\sum_{q=1}^{j-d-1} I_{p,q} = \sum_{p=i-d}^{i+d}\sum_{q=j-d}^{j+d} I_{p,q}\,. \tag{12}$$

As this is the right hand side of equation 2 the proof is complete.

The author would like to thank Dr. John Kay for his helpful comments and suggestions while this research was carried out.

References

[1] J. C. Russ, *The image processing handbook (2nd ed.)*. CRC Press, Inc., 1995.
[2] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*. New Jersey, USA: Prentice Hall, 2001.
[3] M. R. Spiegel, J. Schiller, and R. A. Srinivasan, *Schaum's Outline: Probability and Statistics*. New York, USA: McGraw-Hill, 2000.

John L. Weatherwax received a B.S. with honors in mathematics and a S.B. with honors in physics from the University of Missouri–Columbia in 1996. In September 1996 he attended Massachusetts Institute of Technology, Cambridge Ma. with a National Science Foundation Fellowship. In 2001 he graduated from M.I.T. receiving a doctorate in mathematics in the area of non–linear hyperbolic systems. From 2001 on, he has been a member of the technical staff working at M.I.T. Lincoln Laboratory, Lexington Ma. Currently he is in the ballistic missile division working on discrimination algorithms.