Journal of Digital Imaging

Fast Adaptive Unsharp Masking with Programmable Mediaprocessors

Unmin Bae, Vijay Shamdasani, Ravi Managuli, and Yongmin Kim

Unsharp masking is a widely used image-enhancement method in medical imaging. Hardware-based solutions can be developed to support high computational demand for unsharp masking, but they suffer from limited flexibility. Software solutions can easily incorporate new features and modify key parameters, such as filtering kernel size, but they have not been able to meet the fast computing requirement. Modern programmable mediaprocessors can meet both fast computing and flexibility requirements, which will benefit medical image computing. In this article, we present fast adaptive unsharp masking on two leading mediaprocessors or high-end digital signal processors, Hitachi/Equator Technologies MAP-CA and Texas Instruments TMS320C64x. For a 2k × 2k 16-bit image, our adaptive unsharp masking with a 201 × 201 boxcar kernel takes 225 ms on a 300-MHz MAP-CA and 74 ms on a 600-MHz TMS320C64x. This fast unsharp masking enables technologists and/or physicians to adjust parameters interactively for optimal quality assurance and image viewing.

KEY WORDS: Adaptive unsharp masking algorithm, programmable mediaprocessors, digital signal processors, fast computing-interactive unsharp masking, medical imaging

MEDICAL IMAGES typically have a wide dynamic range. For example, in a chest radiograph, lungs hardly attenuate X rays, whereas thoracic spine absorbs them strongly.[18] Clinicians are at times interested in subtle details, such as lung nodules and lesions in the thoracic spine. Unsharp masking, one of the image-enhancement techniques, sharpens the middle- to high-frequency components of an image while preserving low-frequency components. This leads to enhancing the details with weak contrast to make them more visible and at the same time maintaining the large range of intensities in the image. Before the advent of digital imaging, analog unsharp masking was performed on conventional radiographs.[1] With the development of digital acquisition systems in medical imaging, unsharp masking has become a standard image-enhancement method in many medical imaging modalities, e.g., computed radiography (CR), digital radiography (DR), and digital mammography.[12] For example, after an image is acquired by laser scanning a luminescence phosphor plate, various adjustments of gray scales and unsharp masking enhancements, depending on the examination type and other factors, are performed in CR, either automatically or under a technologist's control. Pisano et al.[14] studied radiologists' preference among eight different image-processing algorithms applied to digital mammograms and found that the mammograms processed with unsharp masking were significantly preferred in the diagnosis of masses. With the wide acceptance of the PACS (picture archiving and communications system) workstation, radiologists would increasingly want to enhance these images interactively when making a diagnosis.

Meeting the computational requirement for fast unsharp masking has been challenging. X-ray images, such as digital radiographs, typically have a resolution of more than 2k × 2k pixels at 12 bits per pixel.[2] Lowpass kernel size and other parameters are ideally different according to the

From the Image Computing Systems Laboratory, Departments of Electrical Engineering and Bioengineering, University of Washington, Seattle, WA, USA.

Correspondence to: Yongmin Kim, Department of Bioengineering, University of Washington, Box 352500, Seattle, WA 98195-2500; tel: 206-685-2271; fax: 206-221-6837; e-mail: ykim@u.washington.edu

Copyright © 2003 by SCAR (Society for Computer Applications in Radiology)

Online publication 20 October 2003
doi: 10.1007/s10278-003-1650-2

lesion type, e.g., a small kernel (2.7 mm) for pneumothorax lines and a large kernel (25 mm) for lung nodules.[15] When the kernel size cannot be changed, large kernels generally yield better results than small kernel.[3,15] Large-kernel convolution with large images is computationally intensive; e.g., convolving a 2k × 2k image with a 125 × 125 boxcar kernel (corresponding to 25 mm for a pixel size of 0.2 mm) alone requires about 66 billion arithmetic operations. In addition, more sophisticated unsharp masking is sometimes desired to support advanced algorithms that have been clinically proven to be efficacious in certain examination types—e.g., lumbar spine using Kodak's enhanced visualization processing (EVP) algorithm.[12]

Various approaches ranging from application- specific boards to programmable systems have been proposed and developed for computation of unsharp masking. A global unsharp masking system with a 15 × 15 Gaussian kernel for lowpass filtering has been implemented by Reza et al.[16] with Xilinx FPGA (field programmable gate array) to process 2 Mbytes/s of data, which corresponds to thirty 256 × 256 8-bit frames. Sivaswamy et al.[17] implemented quadratic-filter-based unsharp masking on an Altera EPLD (electrically programmable logic device) board, where a 512 × 512 16-bit image can be processed in 43 ms. Although FPGA/EPLD-based solutions offer some flexibility and speed, they have difficulty in incorporating a large change in lowpass kernel size and introducing significantly different algorithms because of limited computing resources and fixed I/O arrangements. In contrast, software approaches can provide flexibility in supporting diverse algorithms and dynamically changing to a different algorithm according to the examination and lesion types under consideration. However, traditional software solutions with general-purpose processors have suffered from long processing time; e.g., an adaptive unsharp masking algorithm on a 2k × 2k 12-bit image took 8.7 minutes on a VAX 8600 computer.[9] To achieve high performance with flexibility, a programmable processor with high computing power is desired. Mediaprocessors represent a new generation of programmable processors that are optimized for processing images and video with an excellent cost/performance ratio. Our goal is to develop software-based unsharp masking on these mediaprocessors with the target performance that is comparable to, if not better than that of the traditional hardwired approach.

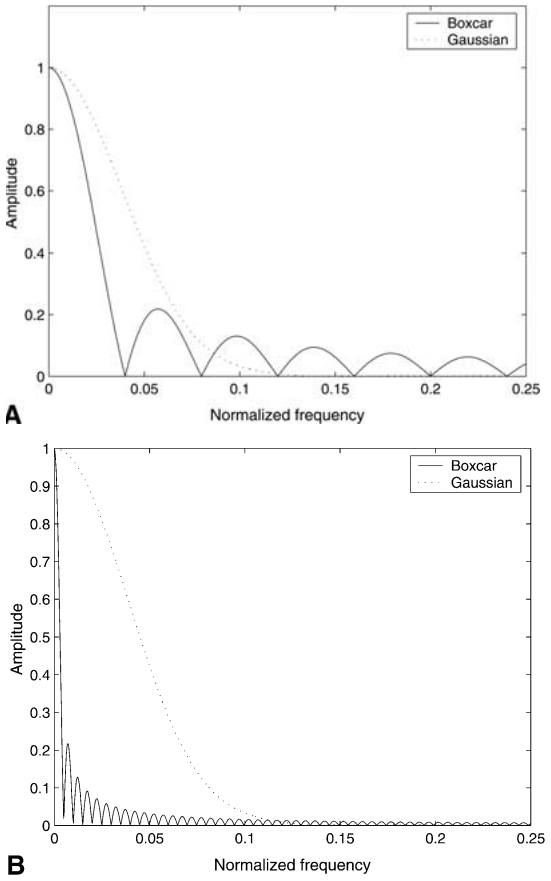

Fig 1. Transfer function of the boxcar and Gaussian filters. (a) With the same kernel size and (b) with a 201 × 201 boxcar kernel and a 25 × 25 Gaussian kernel.

ADAPTIVE UNSHARP MASKING

Unsharp masking is performed by subtracting a lowpass-filtered image ($f_{\mathrm{lpf}}(x, y)$) from an input image ($f_{\mathrm{in}}(x, y)$) to get a highpass-filtered image ($f_{\mathrm{hpf}}(x, y)$) and then adding the weighted highpass-filtered image to the input image.

$$\begin{aligned} f_{\mathrm{out}}(x,y) &= f_{\mathrm{in}}(x,y) + C(x,y)[f_{\mathrm{in}}(x,y) - f_{\mathrm{lpf}}(x,y)] \\ &= f_{\mathrm{in}}(x,y) + C(x,y) f_{\mathrm{hpf}}(x,y). \end{aligned} \tag{1}$$

The cutoff frequency of the lowpass filter determines the characteristics of highpass filtering. A boxcar kernel is preferred in lowpass

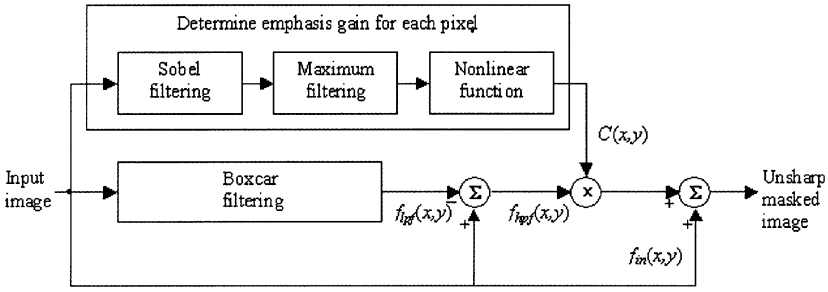

Fig 2. An adaptive unsharp masking algorithm.

filtering because it is computationally simple and its cutoff frequency can be made very low by increasing the kernel size, as shown in Figure 1. Even with the same kernel size, a boxcar kernel has a lower cutoff frequency than a Gaussian kernel, but it has a larger gain in the stop band (i.e., middle and high frequency) because of the side lobes, as evident in Figure 1a.[13] However, if we increase the kernel size as shown in Figure 1b, we can achieve a very low cutoff frequency with a boxcar kernel while maintaining a low gain in the stop band. The optimal boxcar kernel size depends on the lesion size. With a boxcar kernel smaller than a lesion, the lesion's edge as well as its noise is emphasized, whereas a boxcar kernel larger than a lesion leads to enhancement of the lesion without excessive noise.[8,15] Prokop et al.[15] evaluated the parameter setting for unsharp masking of CR chest images using a receiver operating characteristic (ROC) study. Unsharp masking with large boxcar kernels (e.g., 25 mm) was found superior to that with small kernels (e.g., 1.4 mm) in detecting pulmonary and mediastinal nodules. For fine-line structures, the results with the small kernels were better than with the large kernels. The pixel size of current commercial systems ranges from 0.1 mm to 0.2 mm for CR/DR (e.g., 0.2 mm in the General Electric DR system and 0.16 mm in the Canon DR system) and from 0.05 mm to 0.1 mm for digital mammography (e.g., 0.05 mm in the Fuji digital mammography system). We have tested the boxcar kernel sizes from 7×7 to 201×201, which corresponds to 1.40 mm and 40.20 mm for a pixel size of 0.2 mm in chest examinations and 0.35 mm to 10.05 mm for a pixel size of 0.05 mm in breast examinations.

In global or linear unsharp masking, the gain $C(x,y)$ multiplied to the highpass-filtered image ($f_{\mathrm{hpf}}(x, y)$) in Eq. (1) is kept constant over the entire image. Because X-ray photon noise is inversely proportional to the square root of the X-ray photon density, areas with low X-ray photon density have lower signal-to-noise ratios than other areas. Uniform emphasis of high-frequency components over an entire image leads to amplifying noise in low X-ray density areas, such as bones.[9] Hence, it is desirable for the enhancement gain to be limited in the low X-ray density areas. Figure 2 shows an adaptive unsharp masking algorithm, in which the emphasis gain, $C(x,y)$, is adaptively controlled based on local image characteristics. To detect low X-ray density areas, the neighborhood mean value can be considered, but the object of interest might be located in low-density areas, e.g., in the detection of lesions in thoracic or cervical spines. Thus, the low-density areas cannot be suppressed blindly. Local spatial activity information that is related to the presence of objects can be obtained by employing the Sobel operator.[5] The Sobel operator emphasizes both contours of objects and high-frequency isolated noise. The emphasis gain needs to be increased for local regions with object details and to be limited for noisy regions. Because anatomical objects tend to change the pixel values smoothly compared to spike-like noise, they do not produce very large gradients with the Sobel operator. The maximum gradient magnitude value in the 3×3 neighborhood can be used as a measure of whether isolated noise is present in the local region. As shown in Figure 3, the emphasis gain is linearly proportional to the maximum gradient magnitude to enhance object details until a threshold is reached. If the maximum gradient magnitude is above the threshold, the local region is likely to be noisy, and the emphasis gain is clipped. This

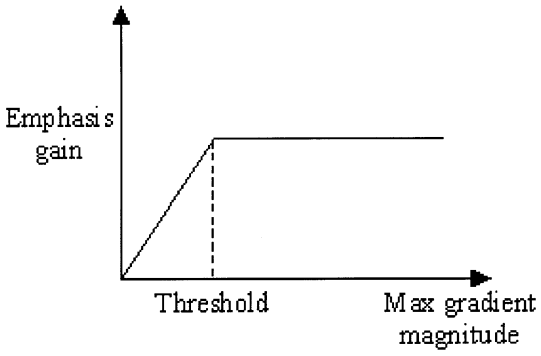

Fig 3. A nonlinear function to determine the emphasis gain.

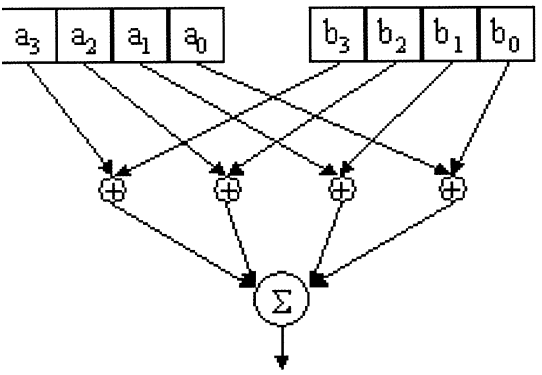

Fig 4. The *sum2* instruction, adding eight 16-bit data stored in two 64-bit registers on the MAP-CA.

threshold can be adjusted according to the body part and lesion type under consideration.

COMPUTING WITH MEDIAPROCESSORS

Mediaprocessors

Mediaprocessors achieve high performance by heavily utilizing several levels of parallelism. Instruction-level parallelism allows multiple operations to be initiated in a single clock cycle, whereas data-level parallelism allows a chosen operation to be executed on multiple smaller data partitions simultaneously. Examples include TTI TriMedia, Texas Instruments TMS320C64x, and Hitachi/Equator Technologies MAP-CA. These processors are accompanied by smart compilers for higher software development productivity. Thus, they can be programmed in C with intrinsics, which is a special C language extension to direct the compiler to use a certain assembly language instruction. Compared to programming in assembly language, using C with intrinsics significantly reduces the time and effort of mapping an algorithm on the processor.

The MAP-CA has a VLIW (very long instruction word) fixed-point architecture optimized for image and video processing.[7] The processing core consists of two clusters that are capable of executing four different operations per clock cycle. It supports 64-bit partitioned operations, working on partitions of 8, 16, or 32 bits, which can provide performance improvements of approximately 8×, 4×, or 2×, respectively. As an example, the *sum2* instruction that accumulates eight 16-bit values is shown in Figure 4. The MAP-CA has an on-chip programmable direct memory access (DMA) controller, which can move the data between on-chip and external memory to prevent the processor from waiting for data to be available in the cache. Details of using the DMA controller can be found in Kim et al.[10]

Texas Instruments recently added TMS320C64x to its C6000 DSP family, targeting a wide array of applications ranging from digital communications to imaging and video.[19] The TMS320C64x has a VLIW core with eight 32-bit fixed-point functional units that are capable of executing eight partitioned operations per cycle. The TMS320C64x has two levels of on-chip memory, L1 and L2. The size of L2 memory is scalable up to 1 Mbyte. A programmable DMA unit handles data transfers between on-chip L2 and external memory.

We mapped the adaptive unsharp masking algorithm described above under Adaptive Unsharp Masking to the MAP-CA and the TMS320C64x. Our mapping techniques are not very specific to these mediaprocessors; thus they can be extended to other mediaprocessors as well.

Algorithm Mapping onto Mediaprocessors

Two-Dimensional Boxcar Convolution

Two-dimensional (2D) convolution on an $N \times N$ input image with an $M \times M$ boxcar kernel is

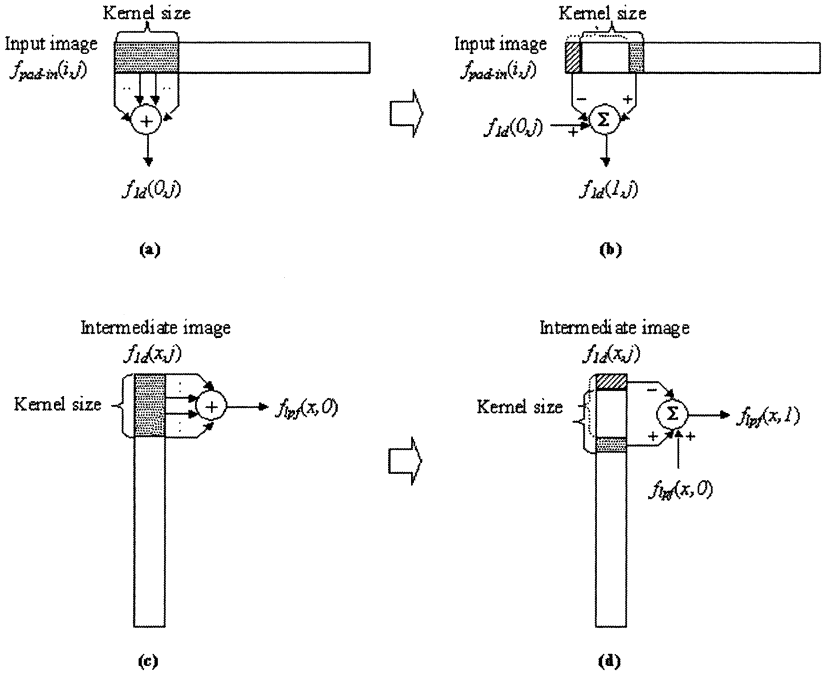

Fig 5. 2D boxcar convolution using moving average operations. (a) Computation of the initial row-wise pixel average for each row of the padded input image, (b) a typical row-wise moving average operation, (c) computation of the initial column-wise pixel average for each column of the intermediate image, and (d) a typical column-wise moving average operation.

given by

$$f_{\mathrm{lpf}}(x,y) = \frac{1}{M \times M} \sum_{j=y-(M-1)/2}^{y+(M-1)/2} \sum_{i=x-(M-1)/2}^{x+(M-1)/2} f_{\mathrm{in}}(i,j), \tag{2}$$

where M is an odd number. The filtered image shrinks proportionally to the kernel size. To keep the filtered image size the same as the input image size, we pad the input image with boundary image pixels before we begin performing boxcar filtering. Because a boxcar kernel is separable, 2D boxcar convolution can be performed by 1D convolution in the row direction, followed by another 1D convolution in the column direction. Thus, 2D boxcar convolution on an $(N+M-1) \times (N+M-1)$ padded input image, $f_{\mathrm{pad-in}}(i, j)$ becomes

$$\begin{aligned} f_{\mathrm{lpf}}(x,y) &= \frac{1}{M} \sum_{j=y-(M-1)/2}^{y+(M-1)/2} \left[\frac{1}{M} \sum_{i=x-(M-1)/2}^{x+(M-1)/2} f_{\mathrm{pad-in}}(i,j) \right] \\ &= \frac{1}{M} \sum_{j=y-(M-1)/2}^{y+(M-1)/2} f_{\mathrm{1d}}(x,j), \end{aligned} \tag{3}$$

where $0 \le x,y < N$ and, $-(M-1)/2 \le i, j < N+(M-1)/2$. The number of operations for each point in $f_{\mathrm{lpf}}(x, y)$ is reduced from $M^2 - 1$ additions in Eq. (2) to $2(M - 1)$ additions in Eq. (3). The number of operations in Eq. (3) can be further reduced by employing moving average operations in each 1D convolution. If we define the initial row-wise pixel average value over M pixels, $f_{\mathrm{1d}}(0, j)$, to be

$$f_{\mathrm{1d}}(0,j) = \frac{1}{M} \sum_{k=-(M-1)/2}^{(M-1)/2} f_{\mathrm{pad-in}}(k,j), \tag{4}$$

then we can compute $f_{\mathrm{1d}}(1, j)$ from $f_{\mathrm{1d}}(0, j)$ via

$$\begin{aligned} f_{\mathrm{1d}}(1,j) = f_{\mathrm{1d}}(0,j) &+ f_{\mathrm{pad-in}}\left(\frac{(M-1)}{2}+1, j\right) \\ &- f_{\mathrm{pad-in}}\left(-\frac{(M-1)}{2}, j\right), \end{aligned} \tag{5}$$

which is shown in Figure 5a and b. Once the initial pixel average, $f_{\mathrm{1d}}(0, j)$, is computed for every row, each subsequent moving average value is calculated with one addition and one subtraction by using the previous moving average value, as in Eq. (5). The *sum2* instruction of the MAP-CA (shown in Figure 4) or the inner-product instruction of the TMS320C64x is used in computing the initial pixel average for each row. This moving average method leads to

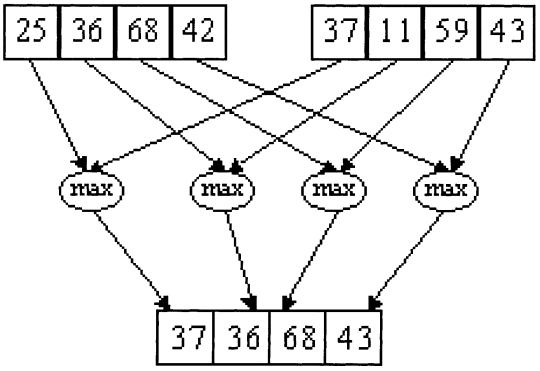

Fig 6. The *max* operation on four pairs of 16-bit data on the MAP-CA.

little change in computational cost as the boxcar kernel size increases. The only increase comes from the initial pixel average computation for every row of the image in Eq. (4). If the image size is much larger than the kernel size, even this dependency on the kernel size becomes negligible. After the row-wise convolution, we can repeat the moving average operations on each column of the intermediate image, $f_{1d}(x, j)$, as shown in Figure 5c and d.

Emphasis Gain Control

The main computing components of this module are to apply the Sobel operator to compute the gradient magnitude and to select the maximum value in a 3×3 neighborhood area. The kernel coefficients of the Sobel operator[5] have absolute values of only 0, 1, and 2. Partitioned addition and subtraction instructions are used to process multiple pixels at a time. Multiplications by 2 are computed using the partitioned multiplication instruction that can be executed as fast as the partitioned additions/subtractions. Local maximum computation requires several comparisons to rank-order nine pixels in the 3×3 neighborhood and select the largest value. However, the MAP-CA and the TMS320C64x have the partitioned *max* instruction that compares multiple pairs of pixels simultaneously and returns the maximum values as shown in Figure 6. Examples of C with intrinsics to support these partitioned operations and the corresponding generic C programs can be found online at http://icsl.ee.washington.edu/~verdant

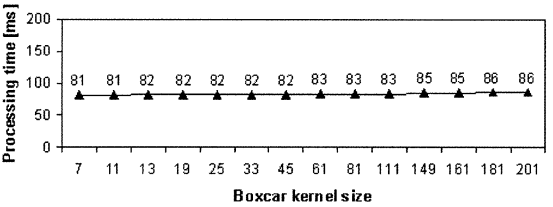

Fig 7. Performance of 2D boxcar filtering for a 2k × 2k 16-bit image on a 300-MHz MAP-CA.

RESULTS AND DISCUSSION

Figure 7 shows performance of 2D boxcar filtering for a $2k \times 2k$ 16-bit image with various kernel sizes on a 300-MHz MAP-CA. The processing time for 2D boxcar filtering is insensitive to the kernel size; e.g., a large increase in the kernel size from 7×7 to 201×201 results in a 6% increase in processing time from 81 ms to 86 ms. This is interesting considering that the processing time of generalized convolution increases quadratically as the kernel size increases. In case of brute-force $M \times M$ boxcar convolution, the total number of additions is $M^2 - 1$ for each point in the output image, which is still proportional to M^2. However, two levels of optimization improve the boxcar filtering performance substantially: (1) the separability property of 2D boxcar kernels reduces the number of operations to $2(M - 1)$ for every filtered output point and (2) the moving average method further reduces this into just two additions and two subtractions for each filtered output pixel. The kernel size affects only computation of the initial pixel average value in Eq. (4). For a large image, such as $2k \times 2k$, computation of the initial pixel average value is minimal compared to the subsequent moving average computation. As a result, Figure 7 shows only a small increase in processing time with the kernel size. Managuli et al.[11] reported generalized 2D convolution on the same MAP-CA mediaprocessor. As shown in Figure 8, the processing time of 2D generalized convolution rapidly increases as the kernel size increases; e.g., 2D convolution of a $2k \times 2k$ 16-bit image with a 3×3 generalized kernel takes 71 ms, whereas 25×25 generalized 2D convolution takes 859 ms. When a kernel is separable, 2D convolution can be computed by using two 1D

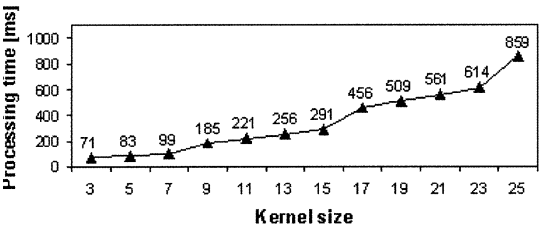

Fig 8. Performance of 2D generalized convolution for a 2k × 2k 16-bit image on a 300-MHz MAP-CA.

convolutions. Figure 9 shows the processing time of 2D convolution with different-size separable kernels,[11] which shows that separable convolution is advantageous when the kernel size is equal to or greater than 9 × 9. However, the overall processing time is still strongly dependent on the kernel size. For example, convolution with a 9 × 9 separable kernel takes 131 ms, whereas a 25 × 25 separable kernel takes 387 ms. In contrast, 2D boxcar filtering with 25 × 25 and 201 × 201 kernels takes only 82 ms and 86 ms, respectively. In other words, 201 × 201 boxcar convolution can be computed 4.5 times faster than 25 × 25 separable Gaussian convolution. At the same time, a 201 × 201 boxcar filter has a much lower cutoff frequency than a 25 × 25 Gaussian filter as shown in Figure 1b. This example illustrates the computational advantage of using boxcar kernels compared to using generalized kernels, especially for the case requiring large kernels/low cutoff frequencies. Table 1 lists the overall performance of adaptive unsharp masking with a 201 × 201 boxcar kernel for various image sizes. For a 2k × 2k 16-bit image, the 139-ms difference between Figure 7 and Table 1 (MAP-CA) is attributable to the remaining processing parts of adaptive unsharp masking—e.g., computing local image characteristics, applying a nonlinear function, multiplications, and subtractions/additions.

The large on-chip memory of the TMS320C64x is used to store intermediate results, such as results of 1D boxcar convolution, thus saving data transfer time between on-chip and external memory. On the TMS320C64x running at a high clock frequency of 600 MHz, 2D convolution with a 201 × 201 boxcar kernel takes only 28 ms for a 2k × 2k 16-bit image, which corresponds to 35 frames/s. The total processing time of adaptive unsharp masking with a 201 × 201 boxcar filter for a 2k × 2k 16-bit image is 74 ms, compared with 225 ms with the MAP-CA as in Table 1. For the same algorithm and image size, the floating-point C implementation takes 3.83 s on a 2.5-GHz Pentium 4. This performance is 52 and 17 times slower than that of TMS320C64x and MAP-CA, respectively.

Even though our unsharp masking algorithm is more complex than the global unsharp masking implemented by Reza et al.,[16] our data processing rates of 37 Mbytes/s on the MAP-CA and 113 Mbytes/s on the TMS320C64x are more than 18 times and 56 times faster than their processing rate of 2 Mbytes/s. As shown in Figure 9, convolution with a 15 × 15 Gaussian kernel, the main processing load of their global unsharp masking, can be performed in 183 ms for a 2k × 2k 16-bit image (46 Mbytes/s) on a 300-MHz MAP-CA. Our mediaprocessor-based unsharp masking easily allows large changes in kernel and image sizes. In contrast, FPGA/EPLD implementations are not designed for kernels and/or images much larger than the initially targeted kernel and image sizes (e.g., changing to digital mammograms), and would typically require major hardware modifications to accommodate large changes. Our

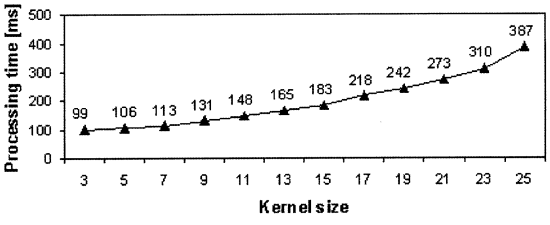

Fig 9. Performance of 2D convolution with a generalized separable kernel for a 2k × 2k 16-bit image on a 300-MHz MAP-CA.

Table 1. Processing Time (ms) of Adaptive Unsharp Masking with a 201 × 201 Boxcar Filter on a 300-MHz MAP-CA and on a 600-MHz TMS320C64x

Input Image Size (16 bits)	MAP-CA (300 MHz)	TMS320C64x (600 MHz)
512 × 512	16	6
1k × 512	31	11
1k × 1k	59	20
2k × 1k	114	38
2k × 2k	225	74

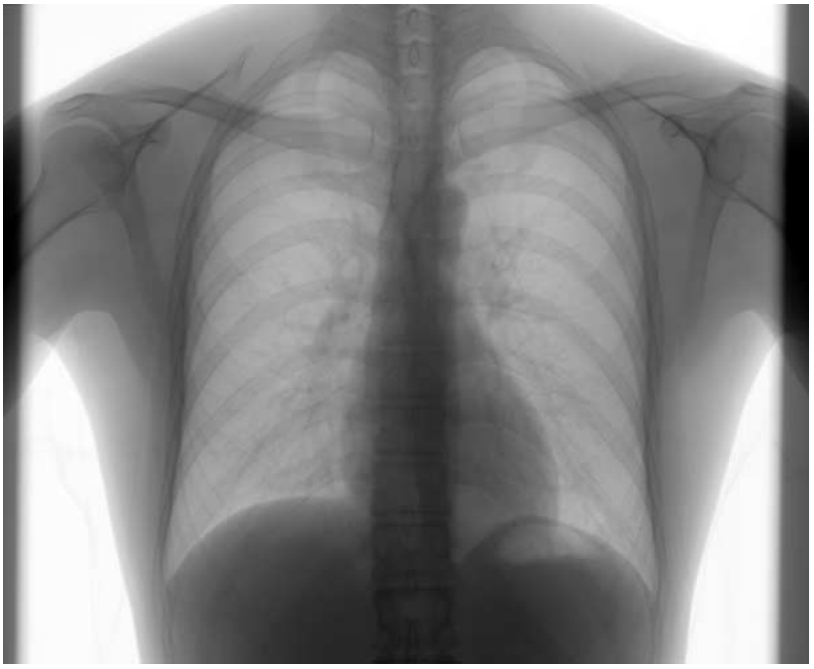

Fig 10. Original chest X-ray image.

adaptive unsharp masking performance of 6 ms on the TMS320C64x with a 201 × 201 boxcar kernel on a 512 × 512 16-bit image is more than 7 times faster than the 43 ms needed by 3 × 3 quadratic-filter-based nonlinear unsharp masking on an EPLD board for the same image size.[17] For a 2k × 2k 8-bit image, the EPLD implementation required 45.2 s because of the limited amount of memory on the board. That is much slower than our TMS320C64x performance of 74 ms. In addition, the same mediaprocessor-based hardware can perform other needed processing tasks in addition to fast unsharp masking. This multi-functionality obviates the need for additional hardware for the other tasks in medical imaging modalities—e.g., detector-related artifact correction, modulation transfer function (MTF) compensation, and gray-level adjustment. Depending on the modality, lesion type and radiologist's preference, other common and/or advanced image-enhancement methods, such as histogram-based, wavelet-based processing, and Kodak's EVP algorithm, could be supported easily. Using the same mediaprocessor, CR and DR images can be compressed quickly (e.g., using JPEG 2000[4]) for image transmission and archiving. Images can be decompressed on the fly before being displayed on the monitors of diagnostic and review workstations. Furthermore, radiologists can enhance the images with the enhancement method that they prefer using multiple presets and/or interactively, and computer-intensive computer-aided diagnosis algorithms could be performed by taking advantage of the high computation power of mediaprocessors.

Figure 10 shows an original DR chest X-ray image, and Figure 11 is the result of our adaptive unsharp masking. The edge details in the processed image are enhanced, and noise amplification is relatively suppressed. The effects of adaptive enhancement may not be evident in Figure 11; therefore, the original X-ray images and unsharp-masked images with different parameter sets are made available for comparison.[6]

Mediaprocessors have been developed primarily for the huge consumer electronics and telecommunications markets, where potential sales volume is enormous. The demand for high computing power at low cost has facilitated extensive advances in computer architecture and ease of programming over the last 10 years. Medical imaging systems based on these processors can directly benefit from the increased processor clock speed, faster memory technology, and lower cost.

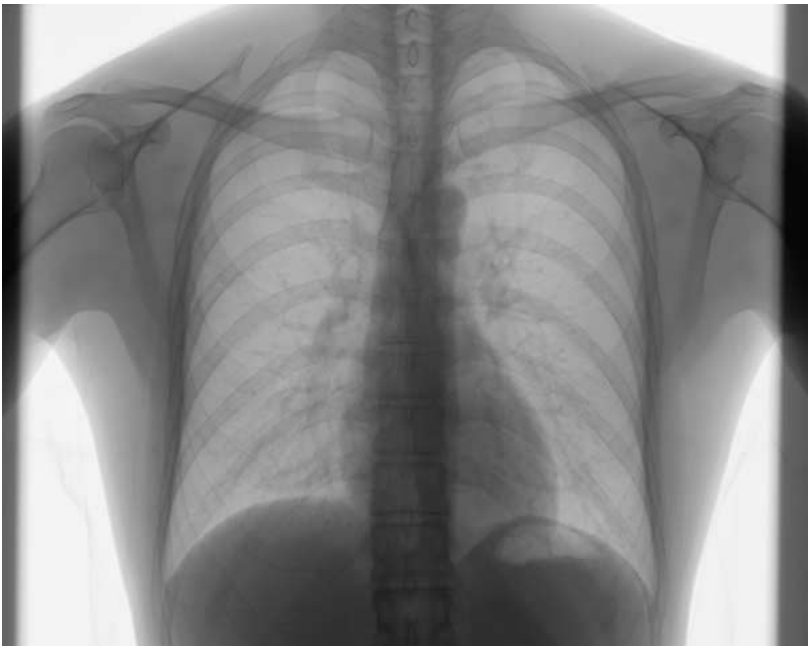

Fig 11. Enhanced chest X-ray image using adaptive unsharp masking.

CONCLUSION

An adaptive unsharp masking algorithm has been mapped onto high-performance mediaprocessors, the MAP-CA and the TMS320C64x. The processing time of adaptive unsharp masking with a 201 × 201 boxcar filter for a 2k × 2k 16-bit image is 224 ms and 74 ms, respectively, which is much faster than that of existing solutions. Two-dimensional boxcar filtering with a large kernel is performed as fast as with a small kernel, such as 7 × 7, whereas a kernel larger than 201 × 201 can be also supported efficiently. The performance of unsharp masking will directly benefit from the expected increase in the clock frequency of mediaprocessors. For example, using the recently introduced 400-MHz MAP[7] the processing time of adaptive unsharp masking can be reduced from 224 ms to about 170 ms. Similarly, with the future 1-GHz TMS320C64x, it can be completed in only about 45 ms. This high performance can facilitate interactive image enhancements by technologists and physicians in quality control and primary diagnosis/review sessions. The high performance, along with low cost, provides an excellent cost/performance ratio. In addition to its speed and low cost, mediaprocessor-based unsharp masking can easily accommodate significant changes—e.g., kernel and image sizes, more efficacious gain control algorithms in adaptive unsharp masking, significantly different image enhancement algorithms, or different and/or additional tasks in medical image modalities in the future.

REFERENCES

1. Armstrong II JD, Sorenson JA, Nelson JA, et al: Clinical evaluation of unsharp masking and slit scanning techniques in chest radiography. Radiology 147:351-356, 1983

2. Barry D, Cluff R, Duncan C, Kennedy J: High performance parallel image processing using SIMD technology. Proc SPIE 3658:344-351, 1999

3. Davies AG, Cowen AR, Parkin GJ, et al: Optimising the processing and presentation of PPCR images. Proc SPIE 2712:189-195, 1996

4. DICOM Standards Committee Working Group 4: Compression. Supplement 61:JPEG2000 Transfer Syntaxes. NEMA, Virginia, 2000

5. Gonzalez RC, Woods RE: Digital image processing, 2nd ed. Upper Saddle River, NJ: Prentice-Hall, 2002

6. http://icsl.ee.washington.edu/~verdant

7. http://www.equator.com

8. http://www.fujimed.com/ndt/cr_process3.html

9. Ji T-L, Sundareshan MK, Roehrig H: Adaptive image contrast enhancement based on human visual properties. IEEE Trans Med Imaging 13:573-586, 1994

10. Kim D, Managuli R, Kim Y: Data cache and direct memory access in programming mediaprocessors. IEEE Micro 21:33-42, 2001

11. Managuli R, York G, Kim D, et al: Mapping of 2D convolution on VLIW mediaprocessors for real-time performance. J Electronic Imaging 9:327-335, 2000

12. Metter RV, Foos D: Enhanced latitude for digital projection radiography. Proc SPIE 3658:468-478, 1999

13. Oppenheim AV, Schafer RW, Buck JR: Discrete-time signal processing. 2nd ed. Upper Saddle River, NJ: Prentice-Hall, 1999

14. Pisano ED, Cole EB, Major S, et al: Radiologists' preference for digital mammographic display. Radiology 216:820-830, 2000

15. Prokop M, Schaefer CM, Oestmann JW, Galanski M: Improved parameters for unsharp mask filtering of digital chest radiographs. Radiology 187:521-526, 1993

16. Reza A, Delva J, Turney R, Chapman K: Medical image enhancement using FPGA technology. Electronic Engineering 70:53-54, 56, 1998

17. Sivaswamy J, Salcic Z, Ling KL: A real-time implementation of nonlinear unsharp masking with FPLDs. Real-Time Imaging 7:195-202, 2001

18. Stahl M, Aach T, Dippel S: Digital radiography enhancement by nonlinear multiscale processing. Med Phys 27:56-65, 2000

19. Texas Instruments: TMS320C6000 CPU and Instruction Set Reference Guide. Literature Number: SPRU189F, 2000