SUPER-RESOLUTION IMAGE RECONSTRUCTION USING HIGH PERFORMANCE COMPUTING

K.Mathew, Dr. S.Shibu

Abstract-Super-resolution (SR) restoration aims to solve the problem: given a set of observed image(s), estimate an image at a higher resolution than is present in any of the individual images**.** In this paper we address the area of Super-Resolution (SR) imaging suggesting parallel computation in process of super resolution image reconstruction. This involvement outlines a method for extrapolating a signal beyond a limited number of known samples. The known signal samples are approximated by a set of basis functions, which are defined over an area covering known as well as unknown, samples. Minimizing a proper inaccuracy criterion and successively selecting the most dominant basis functions can obtain a non band-limited signal extrapolation. Extrapolation can fruitfully be used for finding a high resolution image using a low resolution image(s) and can defeat the inherent resolution constraint of the imaging system and perk up the performance of most digital image processing applications. This has driven the research area of super resolution imaging to develop more multifaceted algorithms that increase the resolution of images with enhanced conformity. One approach that can be used to increase the execution pace of these complex algorithms is to implement the algorithms on a parallel supercomputer. In this paper we concentrate on the parallel implementation of the SR algorithm on the MasPar MP-1, a massively parallel SIMD computer We present a parallel algorithm, which addresses this problem, and present consequences which illustrate real-time performance on 1024 x 1024 images.

Keywords—Super-Resolution(SR), Extrapolation, Processing Element (PE), Single-Instruction Multiple-Data (SIMD).

1. Introduction

Extending a signal beyond a limited number of known samples is commonly referred to as signal extrapolation. Extrapolation of the spectrum of an object beyond the diffraction limit of the imaging system is called *super-resolution.* Super-resolution aims reconstructing frequency components, which lie above the cutoff frequency of the imaging system. The spatial resolution that represents the number of pixels per unit area in an image is the principal factor in determining the quality of an image. In image and video communication, signal extrapolation is an important issue in various applications. The problem of concealing corrupted image data, for example, can be seen as an extrapolation of the surrounding available image data into the missing area. Approaches for signal extrapolation by spectral analysis are known from digital signal processing. The observed time-limited signal can be modeled as a multiplication of the unknown signal by a time-limited binary window function. In the frequency domain, the convolution of the unknown signal spectrum with the window spectrum leads to a blurred and spread spectrum of the observed signal. Our objective is to suggest a parallel algorithm to eliminate the influence of the known window spectrum by spectral analysis, thus extrapolating the signal beyond the known samples. Several iterative approaches are known for Fourier based spectral analysis. The spectrum of the observed windowed signal is first limited to the known band and then transformed to the time domain, yielding a signal, which is extrapolated beyond the known samples. Band limitation, however, also alters the samples within the time window. After replacing the altered samples by the known window-internal samples, the signal is transformed again into the frequency domain, where band limitation is enforced. The extrapolation is obtained by iterating this procedure. In spectral components are additionally eliminated which are below an adaptive threshold. However, because of the band limitation the signal decays rapidly in the time domain beyond the known samples in the extrapolated area.

Therefore the approach is not suitable for applications where the emphasis is placed on extrapolation into larger areas.

In recent years there has been a tremendous increase in the demand for digital imagery with high resolution. Applications include consumer electronics (Kodak's Photo-CD, HDTV, and Sega's CD-ROM video game), medical imaging, video-conferencing, scientific visualization, and multimedia. High Resolution (HR) means that pixel density within an image is high, and therefore an HR image can offer more details that may be critical in various applications. This has driven the research area of Superresolution (SR) imaging to develop algorithms and architectures that improves resolution of images with better fidelity. Unfortunately, these algorithms suffer from increased execution times. This has driven the research area of super resolution imaging to develop algorithms and architectures that increase the resolution of images with better fidelity. Unfortunately, these algorithms suffer from increased execution times.

In this paper we address the parallel implementation of the super resolution algorithm which is based on extrapolation on the MasPar MP-I, a massively parallel supercomputer. We chose to implement the algorithm on the MP-1 for several reasons. First, super resolution has become popular now a days and is used in a variety of applications. Finally, the MP-1 is fast enough to process image sequences in real-time, and thus can serve as both a prototype and production machine for the development of super resolution algorithms. We make known that the greatest difficulty lies not with the super resolution algorithm per se, but with the decomposition of input image into the smaller components which can be processed in parallel.

2. The Basis of Super-Resolution

Extrapolation of the spectrum of an object beyond the diffraction limit of the imaging system is called super-resolution [2]. Extrapolation means extending a signal outside a known interval. Extrapolation in the spatial coordinates could improve the spectral resolution of an image, whereas frequency domain extrapolation could improve the spatial resolution. Such problems arise in power spectrum estimation, resolution of closely spaced objects in radio astronomy, radar target detection and geological exploration and many more like these. A band limited signal f (x) can be determined completely from the knowledge of it over an arbitrary finite interval $[-\alpha, \alpha]$. This follows from the fact that a band limited function is an analytic function because its Taylor series

$$f(x+\Delta) = f(x) + \sum_{n=1}^{\infty} \frac{\Delta^n}{n!} \frac{f(x)}{dx^n} \qquad (1)$$

Is convergent for all x and Δ. By letting $x \in [-\alpha, \alpha]$ and $x + \Delta > \alpha$, equation 1.0 can be used to extrapolate f (x) anywhere outside the interval $[-\alpha, \alpha]$.

The forgoing ideas can also be applied to a space limited function i.e. f (x)= 0 for $|x| > \alpha$, whose Fourier transform is given over a finite frequency band. This means, theoretically, that a finite object imaged by diffraction limited system can be perfectly resolved by extrapolation in the Fourier domain. Extrapolation of the spectrum of an object beyond the diffraction limit of the imaging system is called super-resolution[2].

Super-resolution has been criticized as being impossible because it seeks to recover information that, presumably, has been irretrievably lost. The formation of an image is given by the equation:

$$g(x) = \int h(\xi - x) f(\xi) d\xi \qquad (2)$$

where g is the image, h is the point spread function (the inverse Fourier transform of the OTF) and f is the original object. From this equation we have the Fourier description:

$$G(u) = H(u)F(u) \qquad (3)$$

Equation (3) implies a division by zero to compute F from the quotient of G and H. Since H is zero beyond the diffraction limit cut-off, the reconstruction of any information about F beyond the cut-off is impossible. On this basis, the argument goes, super-resolution can be dismissed as either a theoretical or practical concept.

Consider the imaging in incoherent light of a compact object, i.e., an object that must be positive and wholly contained within some finite interval. The object has the properties:

$$f(x) > 0, x \in X$$
$$f(x) = 0, x \notin X \qquad (4)$$

where X is usually referred to as the region of support. The description of equation (4) can also be expressed more succinctly as:

$$f(x)rect(x/X) \quad (5)$$

where we have assumed the support X coincides with the standard definition of the rect function, with no loss of generality. We now divide the Fourier spectrum of f(x) into two parts: $F_b(u)$ is the portion of the spectrum below the diffraction limit cut-off, and $F_a(u)$ is the portion above the cut-off. From simple Fourier theorems and the multiplication by the rect function we have:

$$F(u) = [F_a(u) + F_b(u)] * \mathrm{sinc}(Xu) \quad (6)$$

The important characteristic of this equation is the convolution of the two portions of the spectrum of F with the sinc function, which is the Fourier transform of the rect function. Since the sinc function is infinite in extent, clearly there will be components of the spectrum portion above the diffraction limit cut-off that are introduced into the spectrum below the cut-off by the convolution with the sinc function. In other words, the nature of the compact object causes information about that object to be on hand in the region of the spectrum that is passed by the optical transfer function. Clearly, if we can find a way to use that information, then there is a basis to realize super-resolution.

Finally the image formation process takes place, and from equation (3) we have the Fourier description of the image:

$$G(u) = H(u)\{[F_a(u) + F_b(u)] * \mathrm{sinc}(Xu)\} \quad (7)$$

It is this equation that describes the actual information available to a super-resolution algorithm for processing.

3. Parallel Architectures

Parallel architectures can be generally classified in several ways. A parallel processor in which each processing element (PE) executes the same instructions on its local memory is known as a single-instruction multiple-data (SIMD) machine [8,10]. A parallel processor in which each PE holds its own program instructions and data is known as a multiple-instruction multiple-data (MIMD) machine. Finally, the architectures can be classified by the communications networks used to interconnect the PES. The price paid for using parallel processing to increase execution speed is an increase in the complexity of developing the algorithm. Programming a parallel machine efficiently requires intimate knowledge of the hardware [10]. To offset this disadvantage the parallel algorithm designer can build a parallel algorithm from selected parallel algorithms and techniques, which have been found basic to almost all parallel computations [9, 10]. These techniques include partitioning, parallel reduction, parallel prefix computations, and pipelining.

Partitioning consists of (i) breaking up the problem into p independent sub problems of almost equal sizes, and then (ii) solving the p sub problems concurrently, where p is the number of processors available. As an example, the partitioning strategy for many window-based image processing tasks with image size n x n on a parallel machine of p processors with mesh interconnections is to divide the image into p sub images of size ($n/\sqrt{p}$) x ($n/\sqrt{p}$) and assign one sub image to each processor. The reduce operation lakes a binary (i.e. two operands) associative operator $\oplus$ with identity i, and a sequence [a_0, a_l , . . . , a_{n-l}] of n elements, and returns the value.

$$a_0 \oplus a_1 \oplus \ldots\ldots \oplus a_{n-1}$$

A prefix computation takes a binary associative operator $\oplus$, and an sequence of n elements [a_0, a_l , . . . , a_{n-l}], and returns the sequence [a_0 ,($a_0 \oplus a_{1)}$, …… ,($\oplus$ $a_{n-1}a_0 \oplus a_1 \oplus \ldots\ldots \oplus a_{n-1}$)].

A parallel prefix computation executes a prefix computation with time complexity O(1ogn) using a balanced binary tree, where n is the number of elements. Similarly, the parallel reduction operation is also O(1og n). Pipelining is the process of breaking up a task into a sequence of subtasks t_l , . . . , t_m , such that, once t_l is completed, the next task can begin and proceed at the same rate as the previous task.

A parallel architecture we suggest is the MasPar MP-1. The Super resolution reconstruction algorithm can be implemented on the MasPar MP-1, a SIMD computer [9]. A fully configured system with 16,384 processors can perform 30 GIPS (peak), with a representative instruction being a 32-bit integer addition. Physically, the computer is divided into two devices: a user interface or front end and the data parallel unit (DPU). The DPU consists of an array control unit (ACU), an array of at least 1024 (16,384 maximum) processing elements (PE), and PE communications mechanisms. The ACU

performs operations on data, which does not need to be distributed to the PE array, and controls the PE array by sending data and instructions to each PE simultaneously. A two-dimensional grid logically represents the PE array, with a maximum size of 128 x 128.Each individual PE is a 4-bit load/store arithmetic-processing element with dedicated registers and 16 kilobytes of RAM. One of the communications networks in the DPU is an eight-nearest neighbor network known as the Xnet. The Xnet is used for communicating information, which is local to a set of PES, or to a PE located in a straight line with the starting PE. The programming language for the MP-1 is a parallel variation of C known as MPL. There is a very efficient library of routines for most of the parallel techniques, including scan, which executes the parallel prefix and segmented parallel prefix computations, and reduce which executes a reduction for any of the associative operators.
Since the MP-1 is a SIMD machine, all of the processors must execute the same instruction at the same time. There are, however, parallel control structures, which allow a processor to become inactive, and not execute the instruction. (Similarly, processors, which execute the instruction, are termed active.) The processor's local memory can be modified whether the processor is active or inactive. The MP-1 has two routines, p-read and p-write, which allow efficient reading to the PE array and writing from the PE array. An important restriction on the functions, however, is that for any single parallel read or parallel write command the number of bytes input or output must be the same for all active PES.

3.1 Parallel Algorithm

The parallel implementation of the SR algorithm is straightforward. For example, for a 1024 x 1024 image and a 128 x 128 array of PES, each PE can be assigned an 8 x 8 block of data. Since the Fourier and convolution steps are completely independent for each 8 x 8 block, perfect partitioning is achieved and the speedup over a single PE for these two steps is 16,384. The most difficult part of implementing in parallel the SR algorithm is not in the algorithm, but in realigning the data between the PES so that correct output operations can be performed with a minimum of number of communication steps.

After the Fourier step each PE contains an array of bytes whose number is dependent on the image data in the associated 8 x 8 block. The write primitives available for the MP-1 do not have the capability of simultaneously writing data, which is stored in the PES in arrays of varying length. In this paper we will be specifically examining efficiently writing the high-resolution image data from the PES to a parallel disk array, however the parallel output algorithm is generally applicable for any output device or channel. The most obvious solution might be to have each PE write its data one at a time in the correct sequential order. This is a prohibitively expensive solution since the MP-1 operates in SIMD mode; when one processor is writing the others must be inactive. The execution time using this technique for a 1024 x 1024 grayscale image is approximately. This is an order of magnitude larger than the time the entire SR algorithm takes to execute on a serial computer, e.g. a Sun SPARC2. The pipelining algorithm presented below is based on the following objective: while keeping the output bytes in sequential order, realign the data so that each PE either has the same number of bytes or has zero bytes of data. This allows the parallel write function on the MP-1 to efficiently move the data to disk. The algorithm is based on using a prefix sum computation to determine the effects of moving the data from one PE to a neighboring PE, coupled with a quotient/remainder operation used to determine how the bytes are to be transferred. For this algorithm a linear array is assumed to be embedded in the mesh interconnection network, i.e. the first PE of each row is connected to the last PE on the preceding row of the mesh. It can be shown that by aligning relative to blocks of maximum size, the data will only need to travel to the preceding PES and not need to travel to succeeding PES.

Algorithm

{Algorithm for data realignment for efficient parallel output using pipelining}
Input: Array of bytes stored on p PES.
Output: Array of bytes stored on a subset of PES where each PE, except for possibly the last, has the same number of bytes, 6.
Note: The bytes remain in the same sequential order.
WRITE- DATA- PIPEPLINE(L)
(Initialization)

```
1 Compute the maximum value, 6, of bytes contained in a single PE
2 Compute a prefix sum of the number of bytes in each PE
3 In parallel do
4 Find and mark PES with start of 6 byte blocks using a prefix sum quotient with divisor 6
5 Find extra bytes in marked PES using a prefix sum remainder with divisor 6
(Iteration)
6 In parallel do
7    for i = 1 to b
8        if have bytes
9            then send one byte to previous PE
10       if marked PE and received byte
11           then put byte at end of output array
12 if marked PE
13       then write 6 bytes of output array
```

4. Practical Considerations

Image compression experiments were performed on 1024 x 1024 8-bit grayscale and 24-bit RGB color images. The data rate was 0.742 bits/pixel for the grayscale image and 1.178 bits/pixel for the color image. The execution times for a Sun SPARC 2 was obtained using the UNIX time command, while the execution times for the MasPar MP-1 was obtained by the MPL function dpu Timer Elapsed(), which has an overhead of 0.000080 [Sec] per measurement. In each case the execution times given in this paper were averaged for ten runs of the algorithm.

5. Conclusion

In this paper, we have provided an introduction to the concepts and basis of the SR image reconstruction as well as a suggestion that use of parallel computation for the existing SR techniques would increase the performance. Experiment leads to the conclusion, borne-out in our many experiments, that super-resolution often may increase the real resolution of an image by only a few percent. There are lines of on-going research to mitigate and overcome these practical limitations. A large object can often be treated as a collection of smaller objects imposed on a larger outline. Thus, if we can successfully separate the smaller objects away from the larger outline and then subtract away the larger outline, the super-resolution of the smaller objects can be greatly increased. Each smaller object can be independently super resolved using independent Processing Element (PE) of parallel architecture. For *K* smaller objects of LR image with at least *K* PEs can improve the performance by factor *K*. The simplest approach to this is to treat small objects as being imposed on a background, and process smaller objects after having modeled and subtracted the background in the explicit formulation of the super-resolution algorithm. Recent results in applying such background techniques to images of satellites taken through ground-based telescopes have been encouraging.

TABLE I: EXECUTION TIMES

Operation	**Grayscale**	**Color**
Parse command line	Time [sec]	Time[sec]
Open and create	0.1145	0.1147
Initialize file reader	0.0638	0.2341
Initialize	0.0152	0.0724
Write header	0.1017	0.2200
Read data	0.0318	0.0720
Align data	0.0500	0.4127
Color convert/zero mean	0.0570	0.1011
Fourier transform	0.0030	0.0405
Convolution	0.0151	0.0451
Inverse Fourier transform	0.0066	0.0190
bit alignment	0.0407	0.0794
Byte stuffing	0.0056	0.0077
Output realignment	0.0089	0.0230
Write data	0.0083	0.0124
Write trailer/end	0.0263	0.0280
Total	0.0038	0.0038
Non-initialized total	0.5480	1.4769

6. References

1. S. Borman and R l Stevenson, "Spatial resolution enhancement of low resolution image sequences. A comprehensive review with directions for future research", Lab. Image and signal analysis, university of North dame, tech. Rep., 1998
2. Anil k Jain, " Fundamentals of Digital Image Processing", PHI,
3. J. W. Goodman, Introduction to Fourier Optics, McGraw-Hill, New York, Second Edition), 1997.
4. H. C. Andrews, B. R. Hunt, Digital Image Restoration, Prentice-Hall, Englewood Cliffs, NJ, 1976.
5. P. Sementilli, B. Hunt, M. Nadar, "Analysis of the limit to super-resolution in incoherent imaging," J Opt. Soc. Amer.-A, vol 10, pp. 2265-2276, 1993.
6. D. Sheppard, B. Hunt, M. Marcellin, "Iterative multiframe super-resolution algorithms for atmospheric turbulence degraded imagery," J. Opt. Soc. Amer.-A, vol. 15, No. 4, pp. 978-992, 1998.
7. P. Sementilli, M. Nadar, B. Hunt, "Empirical evaluation of a bound on image super-resolution performance," SPIE Proc. on Image Reconstruction and Restoration, vol 2302, San Diego, CA, 1994.
8. R. L. Stevenson, G . B. Adams, L. H. Jamieson, and E. J. Delp, "Parallel implementation for iterative image restoration algorithms on a parallel DSP machine," Jornal of VLSI Signal Processing, vol. 5, pp. 261-272, 1993.
9. L. H. Jamieson, E. J . Delp, C.-C. Wang, and J. Li, "A software environment for parallel computer vision," IEEE Computer, vol. 25, no. 2, pp. 73-77, February 1992.
10. J. Ji Jii , An Introduction to Parallel Algorithms. Reading, Massachusetts: Addison-Wesley PublishingCompany, 1992.
11. T. Blank, "The MasPar MP-1 architecture," Proceedings of the Thirty-fifth IEEE Computer Society International Conference, February 26-March 2 1990, San Francisco, California, pp. 20-24.

Authors

K.Mathew, Research Scholar, Karpagam University, Coimbatore, Tamilnadu, India-641 021
e-mail: mathewk07@gmail.com

Dr.S..Shibu, K.R Gouri Amma College Of Engineering, Valamangalam South.P.O., Thuravoor, Cherthala, Alappuzha – 688532
e-mail: shibu.s.1962@gmail.com