The Algorithm of Running Sample Sorting by List Merging for RS-Image L-Filtering

Oleg V. Tsymbal [a], *Vladimir V. Lukin* [b], *Jaakko T. Astola* [c], *Karen O. Egiazarian* [c]

[a] Kalmykov Center for Radiophysical Sensing of Earth NAS and NSA of Ukraine
12 Ak. Proskury Street, 61070, Kharkov, Ukraine,
Tel/fax + 38 0577 441012, E-mail: o_tsymbal@mail.ru

[b] Dept 504, National Aerospace University (Kharkov Aviation Institute)
17 Chkalova Street, 61070, Kharkov, Ukraine,
Tel/fax + 38 0577 441186, E-mail: lukin@xai.kharkov.ua

[c] Signal Processing Laboratory, Tampere University of Technology
P.O.Box-553, FIN-33101, Tampere, Finland, Tel. +358 3 365 3860,
Fax +358 3 365 3857, E-mails: karen@cs.tut.fi, jta@cs.tut.fi

Abstract

One-thread microprocessor system oriented algorithms of sample sorting for remote sensing (RS) image processing using L-filters are considered. The efficient algorithms of running sample sorting by list merging and hybrid list-histogram sorting are proposed. The efficiency is evaluated and compared to known standard and modified algorithms of sample sorting for filtering tasks in different noise situations that are typical for RS-data. It is shown that for the most frequently used apertures of sliding window the use of the proposed sorting algorithms is expedient.

1. Introduction

Most of computations that are required for forming the output value of order statistic filters (OSFs) or order statistic (OS) based local activity indicators deal with a sample sorting procedure. In general, many sorting algorithms are known [1]. Such algorithms as Hoare's "fast" sorting are commonly considered as the most efficient simple data sorting algorithms.

Nevertheless, OSFs applied to RS image processing have some peculiarities. First, sample size is rather small; in general, OSF sliding window apertures larger than 9x9 are used very seldom. Second, RS-image pixels in any of widely used raster formats of RS-data representation are commonly described by integer numbers. Moreover, one of the most widely used formats is the byte format of RS-data representation for which the image pixels have integer values within the range 0..255.

This representation format stipulates the efficiency of radix and, in particular, histogram sorting methods [2,3]. It also makes possible to easily obtain output values of rather simple OSFs (e.g., the weighted OS (WOS) filters like standard median, L_{pq} [4], etc.) using positive Boolean functions. Such rather wide filter class got the name of "stack filters" and it is the most suitable for parallel VLSI implementation [5].

The task considered in our case is constrained by condition of only one-thread (one-processor) execution of RS-image filtering. This is very a important condition since parallel/pipelined systolic filter (e.g., VLSI) implementation can be much rarely met in practice compared to one processor PC or one signal microprocessor realization.

There are also some another peculiarities of particular OSFs. There peculiarities are the amount of order statistics used for forming the OSF output, the location of these order statistics in the ordered sample, etc. This allows using more fast sorting algorithms referred to as partially sample sorting or OS extraction ones. In particular, the median search algorithms are known to have the asymptotic complexity estimation of $O(N\log_2\log_2 N)$ and $O(N^{2/3}\log_2 N)$ (where N is the sample size) [1], i.e. they have better performance compared to "fast" sorting that has the complexity $O(N\log_2 N)$. There is also the K-th OS searching algorithm having $O(N)$ performance estimation. There exist also OS extraction algorithms that exploit binary trees, Rao-Rao arrays etc. [5] for solving a wide spectrum of sorting-like tasks.

Nevertheless, there is a much larger set of tasks that require full (or near full) sample sorting. These are the standard L-filters, α-trimmed filters, complex WOS-filters (e.g. soft morphological filters) as well as the specialized filters (e.g., the impulse burst removing filter [6]). Note, that two latter filters require structural (spatial) information preservation.

In addition to already mentioned OSF algorithm peculiarities, there are also some principal peculiarities of sliding window filtering of RS-images. First of all, images are corrupted by rather intensive multiplicative, additive or mixed noise; impulse noise of salt'n'pepper type can be also present [6]. This leads to the situation when a data sample to be processed can contain the values within entire range of image representation (both extreme minimal and maximal values can be met simultaneously). This fact, as it will be shown later, reduces the efficiency of histogram type sorting algorithms.

The second important peculiarity is just the sliding window filtering approach: at each next step the OSF sliding window shifts usually one pixel aside. This means that the new obtained data sample contains up to 90% values already sorted at the previous step for the "old" sample. The use of this information lets speeding up the sorting algorithm. This approach has been put into basis of the so-called running sorting algorithms [2,3,5].

2. Known sorting algorithms and the proposed modifications

The standard histogram sorting algorithm [2,3] presumes forming the ordered sample as the result of full histogram look-through. Histogram is to be formed at the previous step as the result of one pass through the sample. The histogram look-through is stopped after full sorted array has been formed. The worst case would be the full histogram look-through (from the value 0 to 255). This algorithm is commonly considered as the basic non-running histogram sorting.

The amount of computations required for new histogram forming can be reduced from N to $2h$ (where h is sliding window vertical size) based on aforementioned peculiarity of sliding window filtering (one pixel window shifting). This can be done due to substituting the procedure of full histogram reforming by less computation consuming operations of deleting the values related to the excluding "old" column of sliding window and inserting the newly coming column values. Let us further call this variant of histogram sorting algorithm as "running histogram sorting".

For rather small sliding window apertures that determine the sample size the histogram of a sample as a rule does not reach both boundaries of pixel representation. Based on this assumption, one can expect the histogram sorting speeding up in the case of histogram one edge tracking. This is due to elimination of looking up the void histogram positions located in near boundary areas. Let us further refer such sorting algorithm as "edge tracking running histogram sorting".

Very often (especially for VLSI realization) the "radix" algorithms that presume the values to be sorted as numbers represented in positional numerical system appear to be efficient [1,3]. In our case, an 8-bit value can be represented as two 4-bit numbers, or four 2-bit numbers, or eight binary numbers. For sorting of such number sets, the corresponding histogram trees are used [1,3]. Principally this approach allows eliminating void histogram positions checking. In case of VLSI realization, in addition to possibility of parallel realization, this approach allows reducing its realization complexity due to simplifying of comparison blocks to binary logic cells in case of using binary histogram tree [5].

As in case of sorting methods discussed above, it is expedient to apply the modification for running case to the radix sorting algorithms. Also, the procedure of tracking the edges of "low" level histograms seems to be efficient for the case of 4-bit key radix histogram sorting.

3. The proposed list merge sorting

The first novel sorting algorithm proposed below is the running list-merging algorithm for sample sorting. It is intended for one-thread microprocessor systems or one microprocessor PCs implementation.

The proposed algorithm presumes saving the information processed before in specialized structures that allow to store information in a sorted manner. The structure organized on the basis of one-link list with additional fields that store information about elements locations (column and row in sliding window) have been decided as the best choice for this purpose. In practice, this list is realized as a fixed size linear array of "records" that store the element location information and number of next list element record ("pointer to next") in the same array in addition to element value. In fact, the record fields are the cells with the same numbers within the corresponding arrays. In such a case, the standard operation of allocating/releasing of dynamic memory are not needed. This structure is also convenient for merge sorting [1]. Besides, the amount of simple operations required in such a case does not really depend on dimension of data elements to be sorted and on service information that follows each pixel inside sliding window aperture.

The proposed algorithm of running sample sorting by list merging presumes storing the information in arranged order in structures of two types. The first one is the array of "short (vertical) lists". They represent (in the sorted order) the elements of vertical columns that are involved into the sliding window aperture during filtering an image in the corresponding row. Thus, the height of these columns (and short lists) is equal to the sliding window height (h) (see Fig.1).

The second structure is the "main list". It includes all the pixels within the sliding window. The initial forming of the short list array for the first row sliding window position takes place at preliminary stage of filtering. The second structure (main list) is formed as a result of merging the w short lists that correspond to columns of the sliding window for its initial position in each row. With each further filtering window shifting, the one "old" element is deleted form and one "new" is inserted (see Fig.1) to the short list that has just appeared in sliding window aperture. At the same time, the elements that correspond to "old" column (that leaves the sliding window) are deleted from the main list and the elements that correspond to "new" column (see Fig.1) are inserted to the main list as the result of merging of the main list with the corresponding short list. At each algorithm step after the described procedure, the main list represents the ordered structure allowing to deliver the values of any order statistic. This structure is ready to be used with different L-filters. For more complicated OSF algorithms like [6], the copying of the main list elements into linear array can be required.

Step by step, the proposed algorithm of running sample sorting by list merging for conditions when all preliminary operations are completed is the following:

– During one pass through the corresponding short list the following operations take place:

a) the decrement (in the corresponding field of each element) of the number of the row in which this element is located in the sliding window with respect to its upper border (see Fig.1); { h decrement operations }

b) the deletion in the list of the "oldest" element, i.e. the element with the position number with respect to the upper border of sliding window that became zero after decrement; { $h_{(max.)} \mid (h/2)_{(avrg.)}$ branch operations + 1 transfer }

c) the insertion into the list the "new" element with the value of h ($h = w = 5$ in Fig.1) in the field of row number with respect to the upper border of the sliding window; { $h_{(max.)} \mid (h/2)_{(avrg.)}$ branch operations + 2 transfers}

– During one pass through the main list the following operations are executed:

a) the operations that provide the merging of the main list with the renewed short list (the value in the field of column number with respect to the left border of the sliding window is set to w-1 ($h = w = 5$ for the case in Fig.1) for all inserted elements); {N comparison operations + $2h$ transfer operations}

b) processing (or copying to the random access structure) the values of arranged elements; {N transfer operations}

c) the decrement (in the corresponding field of each not new element of main list) the value of horizontal location of element with respect to the left border of the sliding window; {($N - h$) decrement operations}

d) the deletion (in the main list) the "oldest" elements, i.e. those ones whose value of column number with respect to the window left border became zero after decrement {N branch operations + h transfers}.

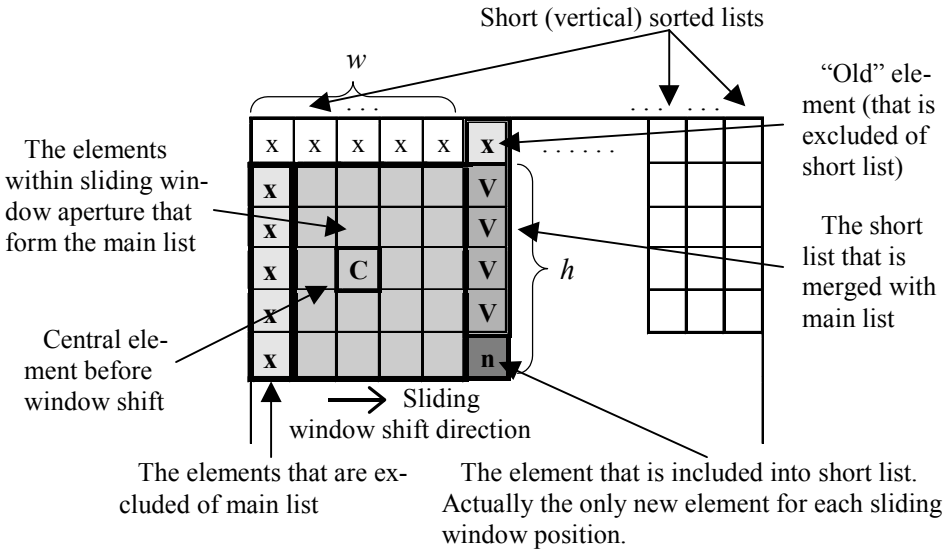

Fig.1. The principles of sorted lists forming in case of image processing sequence: left to right and top to bottom. Sliding window aperture is 5x5 (w=h=5).

4. Hybrid list-histogram sorting

To study the possibility of combining the positive features of list merging based and histogram sorting algorithms that do not require spatial-structural information preservation, some hybrid approaches have been designed and studied by us. The proposed hybrid algorithm worth mentioning here has got the name of "running list-histogram sample sorting".

This algorithm is based on the principles of aforementioned running list merging sorting with the following modifications. In short lists, the field of vertical position of an element in window aperture is replaced by the field of amount of equal valued elements. Accordingly, the procedure of decrement and comparing to zero of the field of an element vertical position for extraction and deletion of the "old" element is replaced by the procedure of "old" element search (see Fig.1). If the amount of elements with the values equal to the one that must be deleted is not unity, the deletion procedure is a trivial decrement of amount of elements with this value. The main list is organized as two-directional list of histogram. In practice, it is implemented as three linear arrays (0..255): the histogram, pointer to (array cell number of) the next and previous element. This allows direct deletion of "old" elements of main list without performing the decrement and checking location field of each list element due to using the "merging for deletion" of the excluded window column. It also permits to get rid of conventional memory management procedures.

5. Performance analysis

The analytic study of asymptotic complexity of the sorting algorithms has shown that, in general, all the considered algorithms have the complexity estimation of O(N) except the "fast" sorting which has O() estimation of $N(1+2\log_2 N)$ for a case of sample copying into linear array. But only the running list merging sample sorting has O() estimation that is almost not sensitive to the actual content of data to be sorted. Its O() estimation is $4N+4h$ (for array copying version). In opposite, the histogram (radix) like algorithms have information dependent estimations that, in general, can be represented as O($N + f(M)$), where $f(M)$ is some function of pixel representation dimension (M=255 in our case). $f()$ is determined by real information content of data to be sorted and actual sorting algorithm used. E.g., the complexities of simple histogram sorting and running histogram sorting with edge tracking can be estimated as $3N+h+E_{hi}$, $2N+3h+E_{hi}-E_{lo}$, respectively. Here E_{hi}, E_{lo} are the mean values of upper and lower histogram edges, respectively. Their actual values depend on image that is filtered but, most of all, on an image noise situation.

Taking into account the described features of sorting algorithms, the actual efficiency has been estimated for "Barbara" 512x512 image corrupted by three different types of noise typical for radar RS-images (see Fig.2). The algorithms were realized using MS VC++ 6.0 without code optimization.

As can be seen, the proposed new list merging and hybrid list-histogram sample sorting algorithms provide the best efficiency for all noise situations typical for radar RS images and for most frequently used OSF sliding window sizes. The proposed modifications for the standard histogram sorting is able to outperform slightly the newly proposed algorithms only for the case of large sliding windows (i.e., for more than 7x7) for not very noisy images.

6. Conclusions

The efficient running list merging algorithm of sample sorting and hybrid list-histogram algorithm are proposed for

L-filter type tasks. The requirement of preserving spatial-structural information for some tasks is taken into account.

Based on analytical complexity estimation and the real image processing, it is shown that the proposed algorithms provide the best efficiency for noise situations typical for radar RS images and for the most frequently used OSF sliding window sizes. Besides, the list merging algorithm also provides spatial-structural information preserving and, in general sense, is not sensitive to the dimension of elementary unit of data to be sorted.

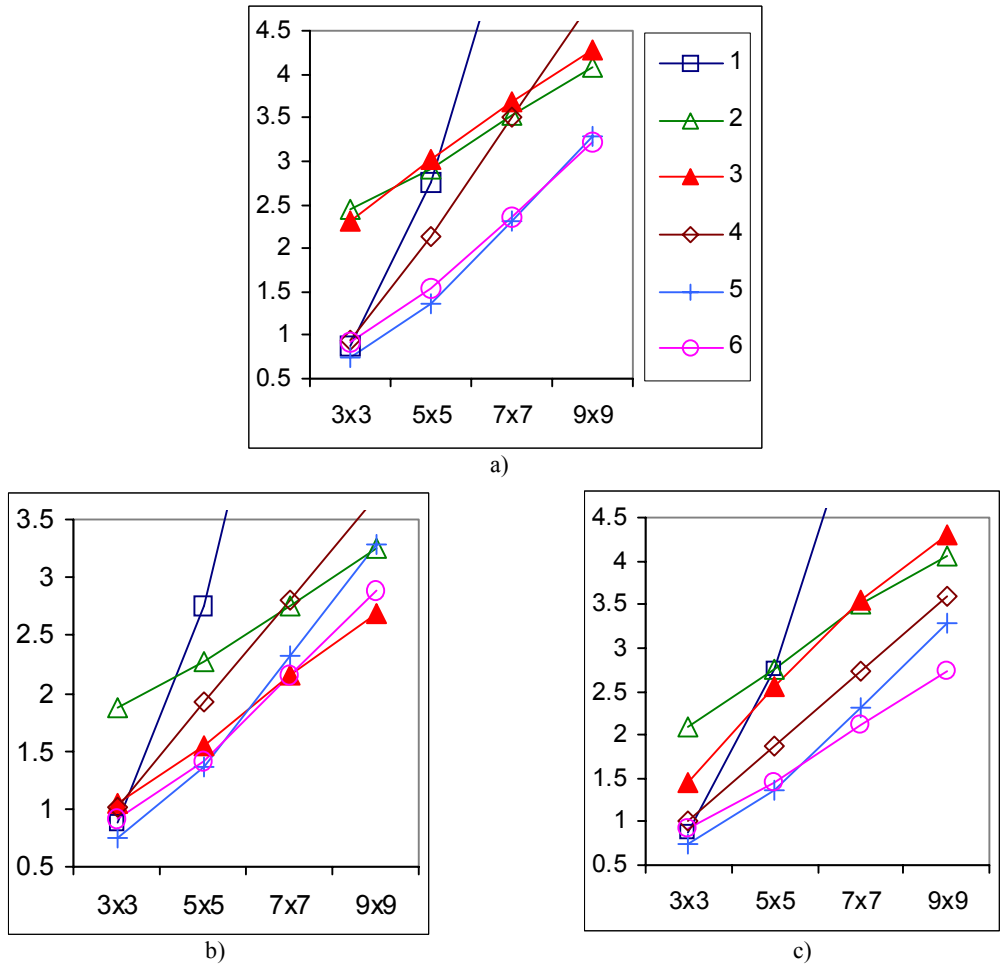

Fig.2. The diagrams of real time consumption for sample sorting algorithms: 1- "fast" (Hoare's), 2- running histogram, 3- running histogram with edge tracking, 4- running histogram tree (radix key – 4-bits) with sub-histogram edge tracking, 5- running list merging (with copying into a linear array), 6- hybrid list-histogram. Time spent is estimated in seconds for Pentium-166 for processing the image "Barbara" 512x512 corrupted by: a) Rayleigh multiplicative noise (noise variance σ_μ^2=0.273), b) Gaussian multiplicative noise with σ_μ^2=0.012, c) mixture of Gaussian multiplicative noise with σ_μ^2=0.005 and impulsive "salt'n'pepper" noise with probability of impulses P_{imp}=0.05.

7. References

[1] Knuth D.E. *The Art of Computer Programming: Sorting and Searching*, Reading, - MA: Addison-Wesley, 1973, 492 p.

[2] Huang T.S., Yang G.J., Tang G.J. "A Fast Two-dimensional Median Filtering Algorithm", *IEEE Trans. On Acoustics, Speech, and Signal Processing*, 1979, Vol. ASPP-27, P. 13-18.

[3] Yli-Harja O., Astola J., Neuvo Y. "Generalization of the Radix-Method of the Median to Order Statistic, Weighted Median and Weighted Order Statistic", *SPIE Symp., Visual Communication and Im. Proc., III,* Cambridge (Mass.), 1988, P. 69-75.

[4] Lukin V.V., Melnik V.P., Pogrebniak A.B., Zelensky A.A., Astola J.T., Saarinen K.P. "Digital adaptive robust algorithms for radar image filtering", *Journal of Electronic Imaging, 1996, Vol.5(3),P.410-421.*

[5] Agaian S., Astola J., Egiazarian K. *Binary polynomial and nonlinear digital filters*, New York (USA): Marcel Dekker Inc., 1995, 394 p.

[6] O.V. Tsymbal, V.V. Lukin, P.T. Koivisto, V.P. Melnik, "Removal of Impulse Bursts in Satellite Images", *Proceed. of Second IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS'2003, Ukraine, Lviv*, 2003, pp.324-329.