

New Fast Transform and Recursive Computation Based Algorithms for Image Convolution, Resampling and Local Statistical Analysis

Leonid Bilevich

Under supervision of Prof. L. Yaroslavsky

1. Abstract

Convolution, resampling and local statistical analysis represent basic classes of operations commonly used in digital image processing. A crucial issue in their implementation is the computational complexity. Major means for reducing the computational complexity are fast transforms and recursive filters.

The objective of the work is to further extend existing fast transforms and recursive algorithms for computation of convolution, resampling and local statistical analysis of images.

The work represents:

- Shifted, scaled and rotated DFT and fast algorithm for image translation, scaling and rotation with discrete sinc-interpolation.
- Fast algorithms for convolution in the DCT domain that are virtually free from boundary effects characteristic to commonly used cyclic convolution in the DFT domain.
- A general framework for recursive computation of image local statistics in sliding window of virtually arbitrary shape with “per-pixel” computational complexity substantially lower than the window size
- recursive algorithms for computation of different local statistics (e.g., mean, other moments, histograms, ranks, median and other order statistics) over different non-rectangular and weighted windows.

2. An algorithm for image resampling with scaling and rotation

The new method of simultaneous scaling and rotation is presented. The method unifies two known “Chirp Transform” algorithms - the scaling algorithm and the rotation algorithm. The scaled and rotated image is obtained from the initial image in the following way:

$$\tilde{a}_{k,l} = \frac{1}{\sigma N} (\text{ChF}(k,l))^t$$

$$\bullet \text{PRUN} \left\{ \text{IFFT} \left[\text{FLIP_C} \left(\text{FFT} \left\{ \text{ZP} \left[\text{FFT}(a_{k,l}) \bullet (\text{ChF}(r,s))^* \right] \right\} \right) \right] \right\}$$

$$\bullet \text{FLIP_R} \left(\text{FFT} \left\{ \text{MR} \left[\text{ChF}(r,s) \right] \right\} \right) \right\},$$

where

$$\text{ChF}(k,l) = \exp \left\{ i \frac{\pi}{N} \left[(k^2 - l^2) \frac{\cos \theta}{\sigma} + 2kl \frac{\sin \theta}{\sigma} \right] \right\},$$

$(\text{ChF}(k,l))^t$ denotes transpose and $(\text{ChF}(k,l))^*$ denotes complex conjugate of the Chirp Function ChF .

The symbol \bullet denotes element-wise product of matrices. If $L = 2^{\lceil \log_2(N + \lceil \sigma N \rceil - 1) \rceil}$ is an extended width and height, then the **ZP**, **MR**, **FLIP_C**, **FLIP_R** and **PRUN** operators are defined as follows.

The Zero Padded signal is given by:

$$\mathbf{ZP} \{y_{r,s}\} = \begin{cases} y_{r,s} & \text{if } r = 0, \dots, N-1, s = 0, \dots, N-1, \\ 0 & \text{if } r = N, \dots, L-1, s = 0, \dots, N-1, \\ 0 & \text{if } r = 0, \dots, N-1, s = N, \dots, L-1, \\ 0 & \text{if } r = N, \dots, L-1, s = N, \dots, L-1, \end{cases}$$

$$= \begin{array}{c|cccccc} & 0 & \dots & N-1 & N & \dots & L-1 \\ \hline 0 & y_{r,s} & \dots & y_{r,s} & 0 & \dots & 0 \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ N-1 & y_{r,s} & \dots & y_{r,s} & 0 & \dots & 0 \\ \hline N & 0 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ L-1 & 0 & \dots & 0 & 0 & \dots & 0 \end{array}. \quad (2.1)$$

The Mirror Reflected kernel is given by:

$$\mathbf{MR} \{ChF(r, s)\} = \begin{cases} ChF(r, s) & \text{if } r = 0, \dots, [\sigma N] - 1, s = 0, \dots, L-1, \\ ChF(r-L, s) & \text{if } r = [\sigma N], \dots, L-1, s = 0, \dots, L-1. \end{cases}$$

$$= \begin{array}{c|ccc} & 0 & \dots & L-1 \\ \hline 0 & ChF(r, s) & \dots & ChF(r, s) \\ \vdots & \vdots & & \vdots \\ [\sigma N] - 1 & ChF(r, s) & \dots & ChF(r, s) \\ \hline [\sigma N] & ChF(r-L, s) & \dots & ChF(r-L, s) \\ \vdots & \vdots & & \vdots \\ L-1 & ChF(r-L, s) & \dots & ChF(r-L, s) \end{array}. \quad (2.2)$$

The matrix with Flipped Columns is given by:

$$\mathbf{FLIP_C} \{Y_{p,q}\} = \begin{array}{c|cccc} & 0 & 1 & \vdots & L-1 \\ \hline 0 & Y_{0,0}^{(L)} & Y_{0,L-1}^{(L)} & \vdots & Y_{0,1}^{(L)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ L-1 & Y_{L-1,0}^{(L)} & Y_{L-1,L-1}^{(L)} & \vdots & Y_{L-1,1}^{(L)} \end{array}. \quad (2.3)$$

The matrix with Flipped Rows is given by:

$$\mathbf{FLIP_R} \{V_{p,q}\} = \begin{array}{c|ccc} & 0 & \dots & L-1 \\ \hline 0 & V_{0,0}^{(L)} & \dots & V_{0,L-1}^{(L)} \\ 1 & V_{L-1,0}^{(L)} & \dots & V_{L-1,L-1}^{(L)} \\ \dots & \dots & \dots & \dots \\ L-1 & V_{1,0}^{(L)} & \dots & V_{1,L-1}^{(L)} \end{array}. \quad (2.4)$$

The Pruned matrix is given by:

$$\text{PRUN} \{g_{r,s}\} = g_{r,s} \quad \text{if } r = 0, \dots, \lceil \sigma N \rceil - 1, s = 0, \dots, \lceil \sigma N \rceil - 1$$

		0	...	$\lceil \sigma N \rceil - 1$	$\lceil \sigma N \rceil$...	$L - 1$	
=	0	$g_{r,s}$...	$g_{r,s}$	discard	...	discard	
	\vdots	\vdots		\vdots	\vdots		\vdots	
	$\lceil \sigma N \rceil - 1$	$g_{r,s}$...	$g_{r,s}$	discard	...	discard	
	$\lceil \sigma N \rceil$	discard	...	discard	discard	...	discard	
	\vdots	\vdots		\vdots	\vdots		\vdots	
	$L - 1$	discard	...	discard	discard	...	discard	

(2.5)

A flow chart of the scaling and rotation algorithm is shown in the Figure 1.

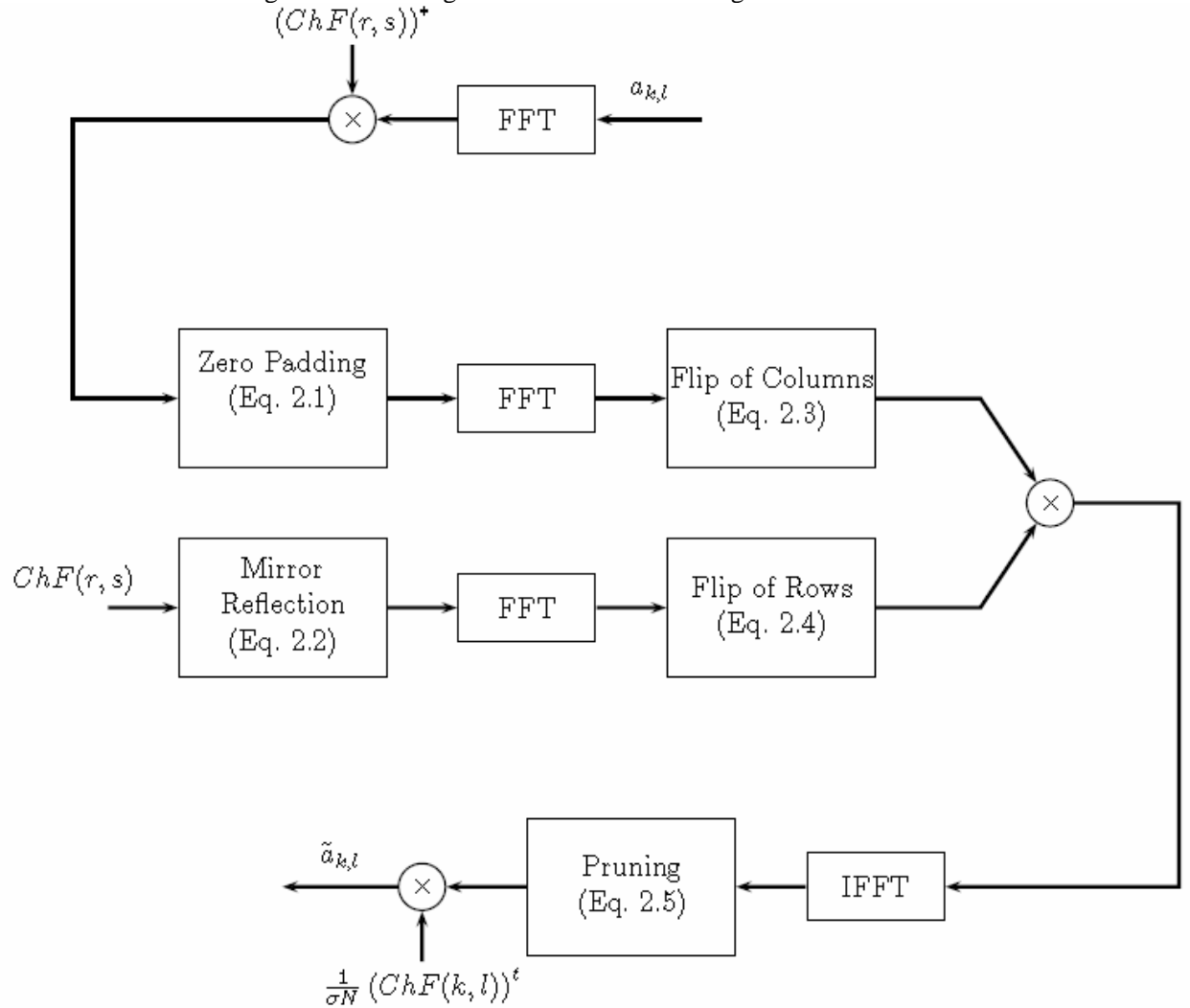


Figure 1: The flow chart of the scaling and rotation algorithm.

3. Fast DCT based algorithms for translation, convolution and scaling

Properties of DCT and DcST transforms are summarized and compared to the properties of the DFT in the Table 1. These properties were used to derive DCT based boundary effect free signal convolution and resampling algorithms. A flow chart of the DCT based convolution algorithm is shown in the Fig. 2. A flow chart of signal fractional translation through IDCT algorithm is shown in the Fig. 3.

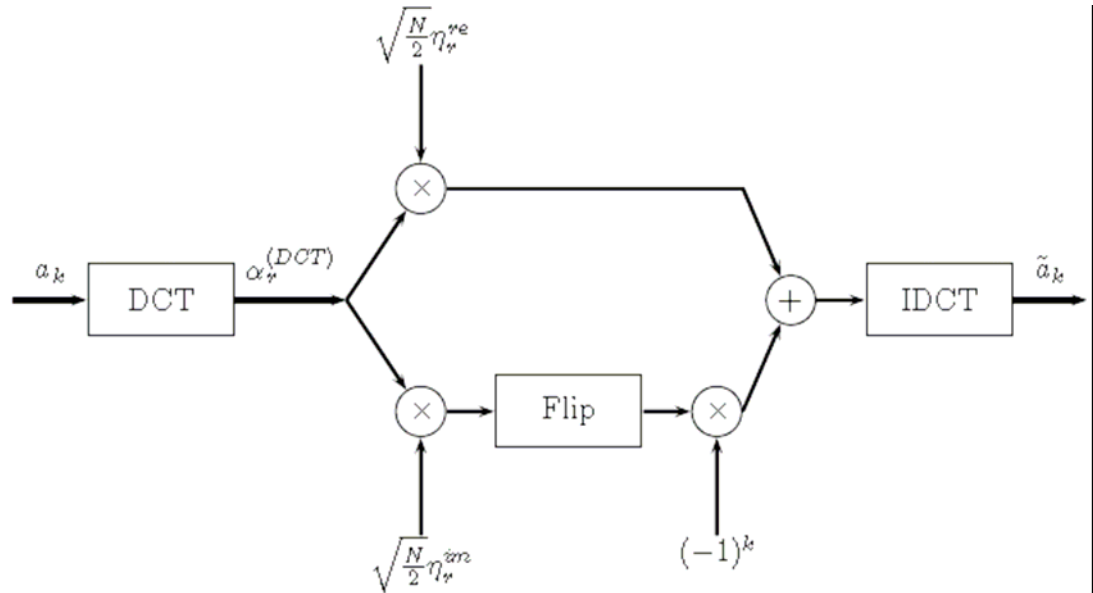


Figure 2: A flow chart of the DCT based convolution algorithm.

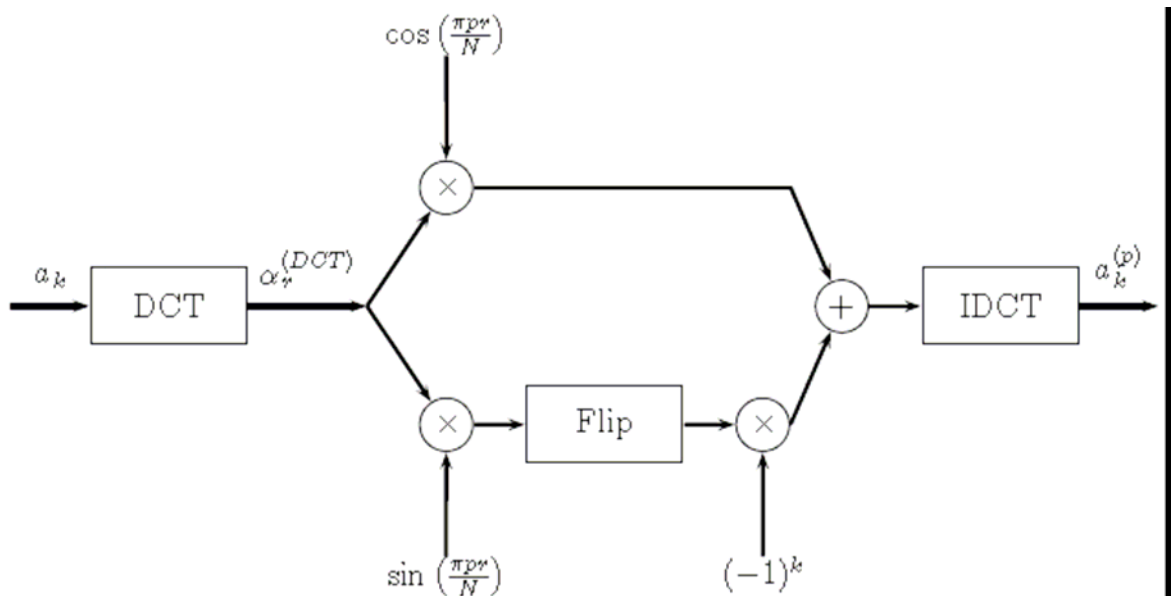


Figure 3: A flow chart of the DCT based signal fractional p -translation algorithm.

Table 1: Summary of properties and algorithms in DFT, DCT and DcST domains.

	DFT	DCT	DcST
Shift theorem	$\text{DFT}\{a_k^{(p)}\} = \text{DFT}\{a_k\} \exp\left[i\frac{2\pi pr}{N}\right]$	$\text{DCT}\{a_k^{(p)}\} = \text{DCT}\{a_k\} \cos\left[\frac{\pi pr}{N}\right] - [\text{DCT}\{(-1)^k a_k\}]_{N-r} \sin\left[\frac{\pi pr}{N}\right]$	$\text{DcST}\{a_k^{(p)}\} = \text{DcST}\{a_k\} \cos\left[\frac{\pi pr}{N}\right] + \text{DCT}\{a_k\} \sin\left[\frac{\pi pr}{N}\right]$
Transform of weighted signal	$\text{DFT}\{b_k\} = [\text{DFT}\{a_k\}]_{r+q}$	$\text{DCT}\{b_k\} = \frac{1}{2} \left\{ [\text{DCT}\{a_k\}]_{r-q} + [\text{DCT}\{a_k\}]_{r+q} \right\}$	$\text{DcST}\{b_k\} = \frac{1}{2} \left\{ [\text{DCT}\{a_k\}]_{r-q} - [\text{DCT}\{a_k\}]_{r+q} \right\}$
Transform of convolution	$\text{DFT}\{b_k\} = \sqrt{N} \text{DFT}\{a_k\} \text{DFT}\{h_k\}$	$\text{DCT}\{b_k\} = \sqrt{\frac{N}{2}} \times \left\{ \text{DCT}\{a_k\} \text{DCT}\{h_k\} - [\text{DCT}\{(-1)^k a_k\}]_{N-r} [\text{DCT}\{(-1)^k h_k\}]_{N-r} \right\}$	$\text{DcST}\{b_k\} = \sqrt{\frac{N}{2}} \times [\text{DCT}\{a_k\} \text{DcST}\{h_k\} + \text{DcST}\{a_k\} \text{DCT}\{h_k\}]$
Transform of correlation	$\text{DFT}\{b_k\} = \sqrt{N} [\text{DFT}\{a_k\}]^* \text{DFT}\{h_k\}$	$\text{DCT}\{b_k\} = \sqrt{\frac{N}{2}} \times \left\{ \text{DCT}\{a_k\} \text{DCT}\{h_k\} + [\text{DCT}\{(-1)^k a_k\}]_{N-r} [\text{DCT}\{(-1)^k h_k\}]_{N-r} \right\}$	$\text{DcST}\{b_k\} = \sqrt{\frac{N}{2}} \times [\text{DCT}\{a_k\} \text{DcST}\{h_k\} - \text{DcST}\{a_k\} \text{DCT}\{h_k\}]$
Product of transforms	$\text{DFT}\{b_k\} = \sqrt{N} \text{DFT}\{a_k\} \text{DFT}\{h_k\}$	$\text{DCT}\{b_k\} = \sqrt{2N} \times \text{DCT}\{a_k\} \text{DCT}\{h_k\}$	$\text{DcST}\{b_k\} = \sqrt{2N} \times \text{DCT}\{a_k\} \text{DcST}\{h_k\}$
Product of transforms	$\text{DFT}\{b_k\} = \sqrt{N} \text{DFT}\{a_k\} \text{DFT}\{h_k\}$	$\text{DCT}\{b_k\} = \sqrt{2N} \times [\text{DCT}\{(-1)^k a_k\}]_{N-r} [\text{DCT}\{(-1)^k h_k\}]_{N-r}$	$\text{DcST}\{b_k\} = -\sqrt{2N} \times \text{DcST}\{a_k\} \text{DCT}\{h_k\}$
Convolution	$\tilde{a}_k = \sqrt{N} \times \text{IDFT}\{\text{DFT}\{a_k\} \text{DFT}\{h_k\}\}$	$\tilde{a}_k = \sqrt{2N} [\text{IDCT}\{\alpha_r^{(DCT)} \eta_r^e\} + (-1)^k [\text{IDCT}\{\alpha_{N-r}^{(DCT)} \eta_{N-r}^e\}]]_k$	
Translation	$a_k^{(p)} = \text{IDFT}\{\alpha_r \exp(i\frac{2\pi pr}{N})\}$	$a_k^{(p)} = \text{IDCT}\{\alpha_r^{(DCT)} \cos(\frac{\pi pr}{N})\} + (-1)^k [\text{IDCT}\{\alpha_{N-r}^{(DCT)} \sin(\frac{\pi p(N-r)}{N})\}]_k$	
Scaling	$\sqrt{N} \text{IDFT}\{\text{ZP}[\text{DFT}\{a_k\}]\} = a_k^{(L)}$	$\sqrt{2N} \text{IDCT}\{\text{ZP}[\text{DCT}\{a_k\}]\}$	

4. Algorithms for image local statistical analysis in arbitrary windows

4.1 Local statistics

Computing image local statistics, such as local means, variance, general local moments, local order statistics, ranks, histograms, spectra, etc., in sliding window is frequently required in image processing. Generally, for arbitrarily shaped window of $WinSz$ pixels, the "per-pixel" computational complexity of this process is $O(WinSz)$ or even, for spectra, $O(\log WinSz)$. Even for moderate window sizes, this complexity might be formidably large, especially in real-time processing applications. Substantial reduction of the computational complexity is possible with the use of recursive computation methods.

We suggest unification and extension of the existing algorithms and present a general solution for recursive computing of local statistics in windows of virtually arbitrary shape.

The weighted local moment of order P of a signal $a_n^{(k)}$ in a window of N_w pixels in its k -th position is defined as follows:

$$M_{[P]}^{(k)} = \sum_{n=0}^{N_w-1} w_n [a_n^{(k)}]^P.$$

Common local statistical parameters (mean, variance, skew, kurtosis) are linear combinations of the local moments. These parameters are used for data statistical analysis, image smoothing, enhancement and feature extraction.

The weighted local histogram counts, for each gray level q , the weighted number of signal samples with this gray level. The value of weighted local histogram at gray level q is:

$$H^{(k)}\{q\} = \sum_{n=0}^{N_w-1} w_n \delta(q - a_n^{(k)}).$$

Local histograms and closely related local variational rows and local order statistics are used in rank filtering for signal/image denoising, smoothing, enhancement, extraction of object details and their boundaries.

Yet another representatives of local statistics are local spectra in different bases. Local signal spectrum $\alpha_r^{(k)}$ with respect to the basis $\psi_r(n)$ is defined as:

$$\alpha_r^{(k)\psi} = \sum_{n=0}^{N_w-1} w_n a_n^{(k)} \psi_r(n).$$

4.2 Parallelization

As a general way to recursive computing of local statistics in windows of arbitrary shape, including arbitrary weighted ones, we suggest parallelization of recursive computations by splitting a computational task into several independent sub-tasks that can be processed recursively. There are two different methods of parallelization:

- multiple windows method,
- expansion of the window functions over recursive bases.

In multiple windows method, a given scanning window is decomposed onto several non-overlapping or overlapping sub-windows, or "building-blocks," that allow recursive computations. The processing is performed in parallel on the sub-windows and then the results of sub-window computations are combined by addition with certain weight coefficients. If the sub-windows overlap, it will correspond to weighted windows with weight coefficient defined by the weight coefficients used in combining sub-window coefficient results. A simple example of composite window built from overlapping "building-blocks" is a combination of overlapping square and diamond. This window is an approximation of an octagon window, it is nearly isotropic and has "soft" edges.

The method of expansion of window over recursive bases allows to compute weighted local statistics in arbitrary shaped window with arbitrary weights assigned to each sample of the window.

Computing local moments in a weighted window can be regarded as a version of space invariant digital filtering of an input signal $[a_n^{(k)}]^P$ with a filter with point-spread function (PSF) w_n :

$$M_{[p]}^{(k)} = \sum_{n=0}^{N_w-1} w_n [a_n^{(k)}]^p.$$

According to this approach, filter with a given PSF is decomposed into a group of recursive sub-filters working in parallel. This decomposition is implemented through expansion of the window weight function $\{w_r\}$ over basis functions, say, $\{\psi_r(n)\}$ $r = 0, \dots, N_r - 1$, $N_r \leq N_w$, that allow recursive computation:

$$w_n = \sum_{r=0}^{N_r-1} \lambda_r \psi_r(n),$$

Then the filter output can be found as a weighted sum the input signal:

$$M_{[p]}^{(k)} = \sum_{r=0}^{N_r-1} \lambda_r \alpha_r(k),$$

of coefficients $\alpha_r(k)$ of signal series expansion over these basis functions:

$$\alpha_r(k) = \sum_{n=0}^{N_w-1} [a_{k-n}]^p \psi_r(n).$$

4.3 Parallel and recursive computation of local statistics in non-rectangular windows through combinations of multiple windows

Assume that a window \bar{W} outlines L “building-block” windows W_l ($l = 0, \dots, L - 1$):

$$\bar{W} = \bigcup_l W_l,$$

and a constant weight w_l is assigned to each “building-block” window W_l .

Parallel and recursive computation of local moments by means of combining results of computations in multiple recursive windows is based on the following theorem.

Theorem1: *The weighted sum of moments over any combination of windows is equal to the weighted moment over the outline (union) of the windows.*

Parallel and recursive computation of local histograms by means of combining results of computations in multiple recursive windows is based on the following theorem.

Theorem2: *The weighted sum of histograms over any combination of windows is equal to the weighted histogram over the outline (union) of the windows.*

Weighted local histograms can be used as bases for computing weighted local variational rows that are defined as cumulative sum of the weighted histogram and weighted local order statistics, such as weighted median, and their derivatives, such as inter-quantile distances and alike. Local minima and local maxima over composite windows can be found directly from recursively computed local minima and maxima over the window “building-blocks”.