Hierarchical Order Statistics Filtering for Fast Bi-Dimensional Empirical Mode Decomposition

Serkan Semiz, Anil Celebi, and Oguzhan Urhan

A hierarchical approach for fast bi-dimensional empirical mode decomposition (B-EMD) is proposed. The presented approach utilizes an efficient window size determination scheme that enables the multi-level computation of the order statistics filter (OSF). Our detailed experiments show that the proposed OSF computation approach allows a significantly faster computation of an EMD without degrading the decomposition accuracy.

Keywords: Empirical mode decomposition, hierarchical processing.

I. Introduction

Empirical mode decomposition (EMD) is a signal decomposition method that is more suitable for non-stationary and non-linear signals and is capable of composing signals more effectively compared to Wavelet and Fourier transforms [1]. EMD decomposes an input signal into so-called intrinsic mode functions (IMFs), which are directly related to the input data. EMD is a preferred method for various applications such as image and biomedical signal processing and seismic analysis [2]. It is iterative in nature, and therefore has a significantly higher computational complexity compared to Fourier and Wavelet transforms. For 1-D signals such as an electroencephalogram, the computational load is relatively small, and the required processing time is measured in minutes. The decomposition process might take several hours or even days for high-volume data such as multi/hyper-spectral images [3].

Currently, applications based on EMD are typically developed and tested on workstations with multi-core processing units and sufficient memory [2]. Several approaches were recently presented to obtain the EMD of input signals in hardware-assisted platforms for faster and possible real-time computations.

In [2], EMD is implemented on a hybrid system with a digital signal processor and field programmable gate array (FPGA) for 1-D signal decomposition. In [4], EMD is realized on an FPGA, but the algorithm is run on a processor using the C code. A real-time EMD hardware architecture is proposed in [5]. In this architecture, there are a constant number of interpolation iterations for each IMF, and a total of five IMFs are calculated. In [6], a hardware implementation of EMD is proposed, but no details on the architecture are given. In [7],

Manuscript received Nov. 17, 2014; revised Jan. 11, 2016; accepted Mar. 29, 2016.

Serkan Semiz (ssemiz@gmail.com), Anil Celebi (anilcelebi@gmail.com), and Oguzhan Urhan (corresponding author, urhano@gmail.com) are with the Electronics & Telecom Engineering Department, University of Kocaeli, Turkey.

© 2016 ETRI

order-statistics filtering is used to replace cubic spline interpolation for bi-dimensional EMD (B-EMD). Obtaining the IMFs by applying order-statistics filters with varying window sizes drastically reduces the computational load and makes the calculations easier to implement in hardware. This approach is called FABEMD and has received significant attention. In the present work, a hierarchical computation approach for B-EMD is presented to further speed up the FABEMD approach presented in [7]. To illustrate the real-time applicability of this approach, a hardware implementation of the FABEMD method for image fusion is presented in [8].

II. Proposed Approach

In the method proposed in [7], the local extremes of an image are initially located. Using these pixels, the window size of the OSF width applied in each iteration step is obtained. The smallest distance from each maximum pixel value to its neighbors is calculated. Because more than one maximum and/or minimum pixel might exist in the related window, a decision should be made based on the distance from the maximum and/or minimum pixel to the related pixel. As described in [7], there are four combinations that can be utilized for this decision:

- MAX_{min} - Closest maximum to the center
- MAX_{max} - Farthest maximum to the center
- MIN_{min} - Closest minimum to the center
- MIN_{max} - Farthest minimum to the center

Using the minimum of the MAX_{min} and MIN_{min} values as the window size is defined as *Type*-1, and using the maximum of MAX_{max} and MIN_{max} is defined as *Type*-4. To derive the upper and lower envelopes, order-statistics filtering and smoothing operations are applied to the windows in the image, the size of which are determined according to the above-mentioned approaches. The average of the upper and lower envelopes is subtracted from the image, and a standard deviation measure is investigated. A standard deviation smaller than a fixed threshold value indicates the successful extraction of the related IMF. Thus, the iterations continue with the residue data until the sum of the extreme is less than 3 to satisfy the monotonicity condition, as explained in [1]. Limiting the maximum number of iterations, and thus ignoring the standard deviation, is also proposed. This limit is called the maximum number of allowed iterations (MNAI). A flowchart of the proposed method is illustrated in Fig. 1. The novel contributions of the proposed approach to the FABEMD are highlighted is this figure.

One of the most important issues in the hardware implementation of fast B-EMD is the increasing window size for order-statistics filtering and smoothing with an increase in the IMF index. The window size (W_S), which is relatively small and easy to process for IMFs with smaller indexes, reaches sizes comparable to the image size ($M \times N$) itself. Under such a situation, the amount of data to be accessed and processed is $W_S^2 \times M \times N$. Using the nearest odd number with odd divisors instead of using the original window size required by the algorithm dramatically reduces the processing load, thus increasing the speed of the decomposition process. The window size that contains only odd divisors is referred to as the appropriate window size (W_{SA}).

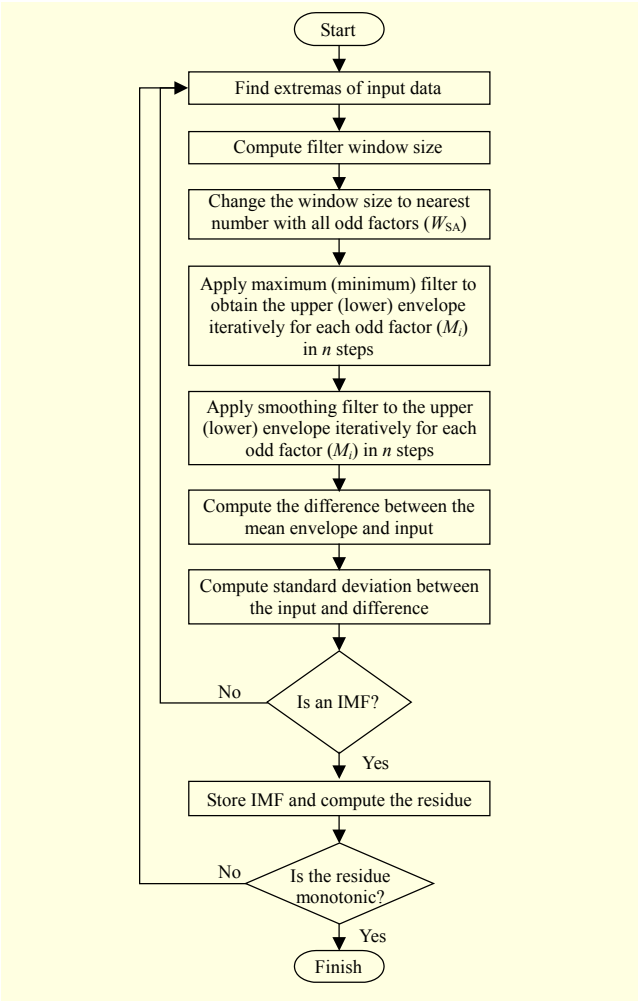

Fig. 1. Flowchart of the proposed method.

In the proposed method, the candidate window size is obtained as defined in [7]. The nearest W_{SA} to the window size candidate is detected, and its divisors are calculated. All of the divisors need to be odd numbers. The upper limit of the allowed divisors is defined as M_{max}. Choosing a small M_{max} increases the speed of the decomposition but decreases the window size values that are obtainable with a limited divisor set.

$$W_{SA} = \prod_{i=1}^{n} M_i . \quad (1)$$

ETRI Journal, Volume 38, Number 4, August 2016
http://dx.doi.org/10.4218/etrij.16.0114.1330

Table 1. First 36 appropriate window sizes (W_{SA}).

Order	Value	Order	Value	Order	Value	Order	Value
1	3	**10**	45	**19**	175	**28**	441
2	5	**11**	49	**20**	189	**29**	525
3	7	**12**	63	**21**	225	**30**	567
4	9	**13**	75	**22**	243	**31**	625
5	15	**14**	81	**23**	245	**32**	675
6	21	**15**	105	**24**	315	**33**	729
7	25	**16**	125	**25**	343	**34**	735
8	27	**17**	135	**26**	375	**35**	875
9	35	**18**	147	**27**	405	**36**	945

The first 36 window sizes that can be obtained through the proposed scheme are given in Table 1, where M_{max} is selected as 7. The W_{SA} values are more frequent in the lower region and less frequent as they increase, which is consistent with the nature of EMD.

After calculating the nearest appropriate window size, factorization takes place, and W_{SA} is expressed as in (1), with n being the number of factors other than 1, and M_i being the factors.

Filtering is not conducted in a single step, but in as many steps as the number of factors (n) that exist. For each step, the pixel to be processed is kept at the center, and the data are read by jumping in the horizontal and vertical directions with the distances defined by the multiplication of the window sizes in the previous steps. In the first step, because there are no previous steps, the jump length is zero, and all of the data in the window are used. If we consider a case in which the window size is 45, the divisors are calculated as 3, 3, and 5. The number of steps required to execute the filtering is calculated as three. In the first step, a 3 × 3 window is moved onto each pixel of the image, and nine pixels are compared. This is illustrated in Fig. 2(a), where the center pixel is shown in black, and the others that are processed are shown in gray. The maximum and minimum values for each center pixel are written to the corresponding memory. This memory needs to have the same size as the original input image.

When the first step is finished, in each pixel of the buffer, the maximum and minimum values of the 3 × 3 windows centered at that specific pixel are obtained. In the second step, the data are read with three pixel jumps in both axes, and nine points are compared for each pixel, as shown in Fig. 2(b). Again, the maximum and minimum values are written to the corresponding memory locations.

Finally, in the last step, the data are read with nine pixel jumps, as shown in Fig. 3, and the maximum and minimum pixel values are obtained by comparing a total of 25 pixel values. In this way, the equivalent process for filtering the

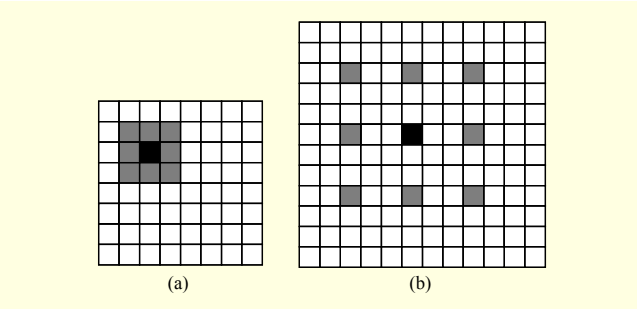

Fig. 2. Processed pixels in two steps for 45 × 45 windowing: (a) the processed pixels in the (a) first and (b) second steps.

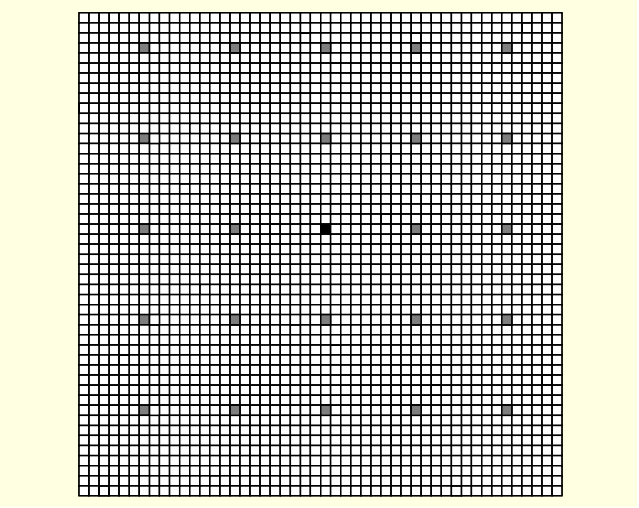

Fig. 3. Processed pixels in the third step for 45 × 45 windowing.

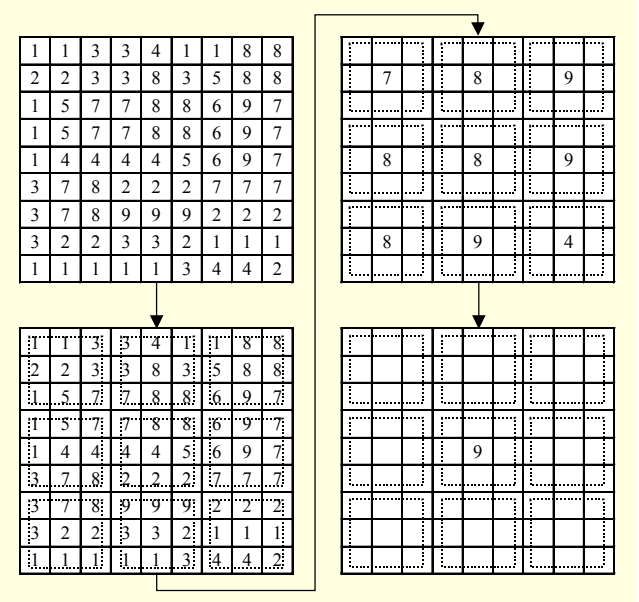

Fig. 4. Detailed example of the filtering process for a window size of 9 × 9.

Table 2. Process load for various window sizes.

Window size	Method in [7]	Proposed method	Ratio
3	9	9	1
27	729	27	27
45	2,025	43	47
105	11,025	83	133
525	275,625	108	2,552

image by moving a 45 × 45 window through all pixels is achieved. A zero padding method is utilized to apply this algorithm to the pixels close to the border of the image.

Using the hierarchical structure presented in this work, $3^2 + 3^2 + 5^2 = 43$ calculations are required for each pixel instead of $45 \times 45 = 2{,}025$ calculations. A numerical example of this procedure is also illustrated in Fig. 4 for a 9 × 9 windows size, where the maximum valued pixel is obtained using the proposed hierarchical approach. The processing load is now proportional to the sum of the squares of the size factors of the window, not to the square of the window size. The processing loads for an ordinary calculation in [7] and the method proposed in this work are given in Table 2. The benefit of the hierarchical approach is more obvious as the window size increases.

III. Experimental Results

In this section, the decomposition results of both the method proposed in [7] and the approach proposed herein, which were implemented using software, are compared. We initially generated a synthetic texture image (STI) by making use of the sinusoidal base functions (SBF) as in (2) with different frequency components,

$$\mathrm{SBF}(x,y) = \sin\left(2\pi \frac{x}{w} f_w\right) + \sin\left(2\pi \frac{y}{h} f_h\right), \qquad (2)$$

where x and y denote horizontal and vertical pixel positions, and w and h represent the image size. The frequency characteristic of the SBF is adjusted using f_w and f_h. We construct our STI image, as shown in Fig. 5(a), by combining three different SBFs, as illustrated in Figs. 5(b) through 5(d).

Tables 3 through 8 show various performance measures for the proposed approach and the conventional FABEMD presented in [7]. The processing times on a 2.3-GHz quad-core PC (Intel i7-4850HQ), using a non-optimized MATLAB™ implementation for the two methods with different parameters, are given in Table 3. Although the proposed method targets a hardware implementation, it also has a better performance for a

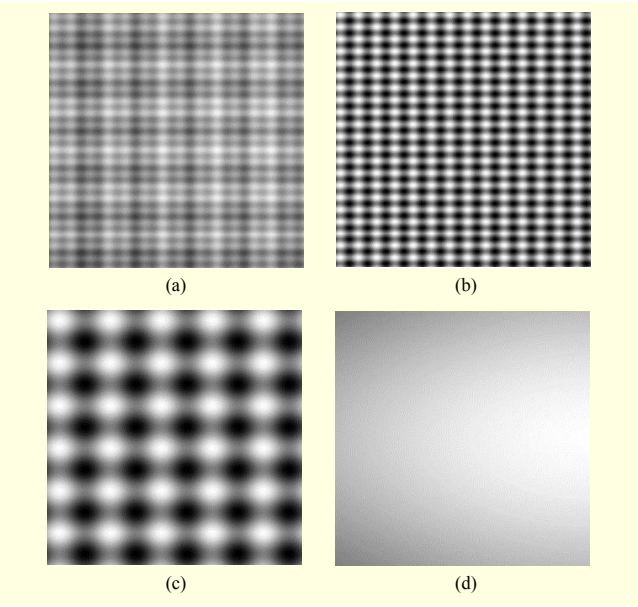

Fig. 5. STI image with its three components: (a) STI image, (b) Component-1, (c) Component-2, and (d) Component-3.

Table 3. Processing time and OI for STI.

	FABEMD method in [7]		Proposed method	
	MNAI = 1	MNAI = 5	MNAI = 1	MNAI = 5
Processing time (s)	84.98	362.78	13.32	54.18
OI	0.00178	0.00079	0.00175	0.00144

Table 4. Performance assessment for STI.

		FABEMD method in [7]		Proposed method	
		MNAI = 1	MNAI = 5	MNAI = 1	MNAI = 5
Window size	IMF1	27	25	27	25
	IMF2	95	95	105	81
	IMF3	229	143	N/A	135
Stopping point SD	IMF1	0.97537	0.03959	0.97543	0.03956
	IMF2	0.99124	0.00201	0.99079	0.01377
	IMF3	0.99676	0.14603	N/A	0.03963
Average IMF values	IMF1	–0.05952	–0.01985	–0.06358	–0.02967
	IMF2	–0.15212	–0.13609	–0.12324	0.01079
	IMF3	0.11870	0.15550	N/A	0.12744
	Residue	154.064	153.971	154.158	153.862
Sum of extremes in IMFs	IMF1	624	660	624	660
	IMF2	66	136	72	124
	IMF3	5	100	N/A	80
	Residue	1	2	2	2

ETRI Journal, Volume 38, Number 4, August 2016
http://dx.doi.org/10.4218/etrij.16.0114.1330

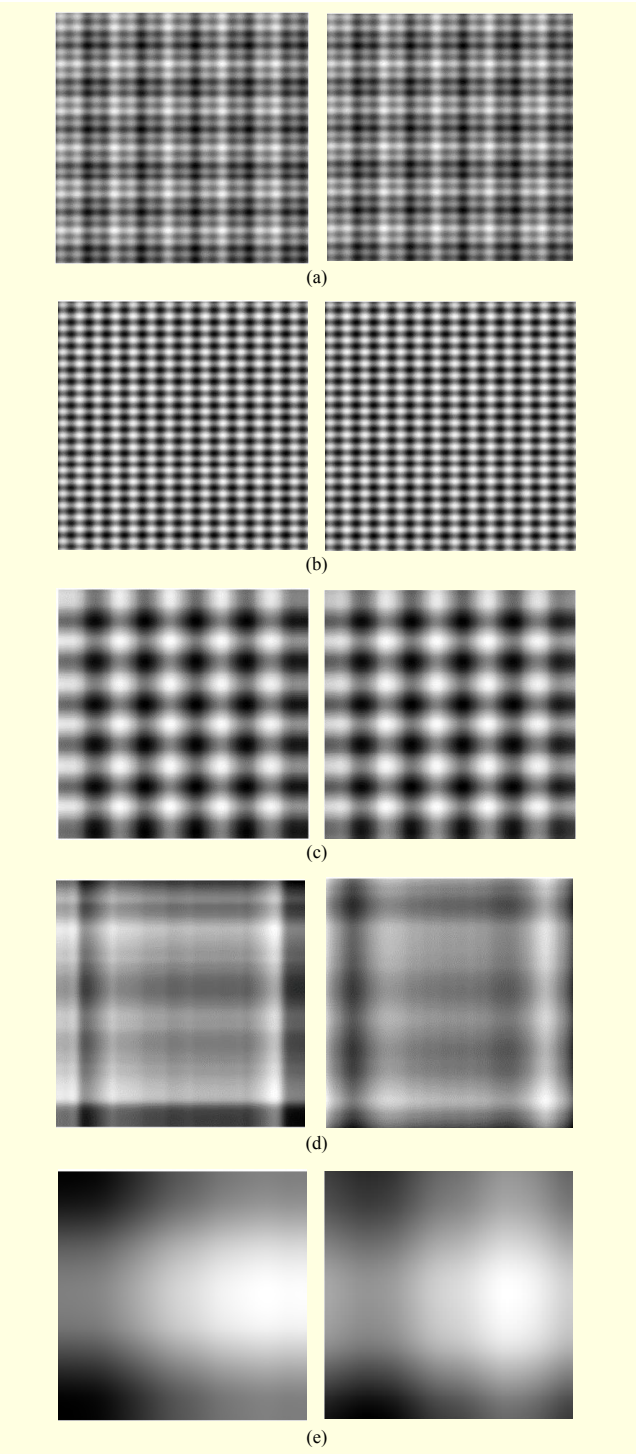

Fig. 6. Reconstructed STI image, IMFs, and residue using the FABEMD method (left column) and the proposed approach (right column): (a) reconstructed STI image, (b) IMF1, (c) IMF2, (d) IMF3, and (e) residue.

software implementation compared to the method in [7]. The orthogonality index (OI) shows the quality of the decomposition, and OI values of lower than 0.1 are acceptable, as described in [7]. As shown in Table 3, the proposed approach is able to produce acceptable IMFs with a speed-up factor of 6.

Table 5. Performance assessment for Barbara.

		FABEMD method in [7]		Proposed method	
		MNAI = 1	MNAI = 5	MNAI = 1	MNAI = 5
Processing time (s)		73.87	1,827.99	39.94	129.59
OI		0.01254	0.00973	0.01130	0.01135
Window size	IMF1	13	11	15	9
	IMF2	59	43	63	45
	IMF3	189	145	225	125
	IMF4	N/A	379	343	315
	IMF5	N/A	511	N/A	441
Stopping point SD	IMF1	0.93547	0.02506	0.92990	0.04984
	IMF2	0.90022	0.02950	0.91184	0.12410
	IMF3	1.00809	0.02605	1.02029	0.31830
	IMF4	N/A	0.02355	0.96743	0.01761
	IMF5	N/A	0.04208	N/A	0.01988
Average IMF values	IMF1	–0.95647	–0.10575	–1.15332	–0.06846
	IMF2	2.54806	2.04984	2.23410	1.51551
	IMF3	–2.70188	0.39109	–3.48520	0.20772
	IMF4	N/A	–3.54991	1.52069	–2.82774
	IMF5	N/A	1.55820	N/A	2.05990
	Residue	118.503	117.049	118.276	116.506
Sum of extremes in IMFs	IMF1	26,058	32,153	26,058	32,365
	IMF2	339	921	291	1284
	IMF3	29	72	26	96
	IMF4	N/A	14	3	15
	IMF5	N/A	5	N/A	10
	Residue	2	0	1	1

Table 4 shows the window sizes for each IMF decomposed using both methods, the standard deviation (SD) of the stopping point, and sum of the extremes in IMFs. The window size parameter shows that the proposed approach is generally able to choose similar window sizes compared to FABEMD. The number of total extremes decreases for higher indexed IMFs because of the iterative smoothing process.

The SD at the stopping point is lower for a larger number of iterations as expected. Additionally, the global average values of the IMFs are near zero, which means that the proposed approach is able to obtain high-quality IMFs.

The number of IMFs found differs when the MNAI is changed, which is expected when, as in this case, the residue contains remaining information from the original signal. In general, as can be seen from Table 4, the outputs of the

Table 6. Performance assessment for Lena.

		FABEMD method in [7]		Proposed method	
		MNAI = 1	MNAI = 5	MNAI = 1	MNAI = 5
Processing time (s)		646.72	335.16	31.09	130.31
OI		0.00447	–0.00690	–0.00467	–0.00829
Window size	IMF1	13	11	15	9
	IMF2	81	47	63	45
	IMF3	309	145	315	135
	IMF4	463	331	N/A	343
Stopping point SD	IMF1	0.96043	0.02180	0.95502	0.03403
	IMF2	0.95728	0.04194	0.96375	0.04819
	IMF3	1.01947	0.02085	1.02097	0.03225
	IMF4	0.95264	0.03579	N/A	0.04573
Average IMF values	IMF1	–1.22240	–0.76991	–1.28543	–0.75042
	IMF2	–1.11398	0.16240	–0.76033	0.42737
	IMF3	–2.08625	–0.76791	–2.56863	–1.09274
	IMF4	2.86106	–3.20927	N/A	–3.80880
	Residue	125.612	128.637	128.665	129.275
Sum of extremes in IMFs	IMF1	24,882	34,893	24,882	35,717
	IMF2	374	1,301	310	1,698
	IMF3	12	70	14	74
	IMF4	5	14	N/A	11
	Residue	1	2	2	2

Table 7. Performance assessment for Elaine.

		FABEMD method in [7]		Proposed method	
		MNAI = 1	MNAI = 5	MNAI = 1	MNAI = 5
Processing time (s)		21.51	62.99	4.82	17.15
OI		0.01357	0.01326	0.00801	0.01764
Window size	IMF1	17	17	15	15
	IMF2	63	53	63	35
	IMF3	255	139	245	125
	IMF4	N/A	255	N/A	245
Stopping point SD	IMF1	0.90666	0.04691	0.91302	0.02335
	IMF2	0.97276	0.04443	0.97302	0.04312
	IMF3	0.99221	0.03046	1.01004	0.01782
	IMF4	N/A	0.04042	N/A	0.02064
Average IMF values	IMF1	2.34344	2.40768	2.15539	1.75655
	IMF2	0.14837	0.44718	–0.08044	0.42704
	IMF3	–0.60441	2.97018	–1.85984	4.26003
	IMF4	N/A	–2.46432	N/A	–1.29750
	Residue	134.474	133.000	136.146	131.215
Sum of extremes in IMFs	IMF1	3,408	4,645	3,408	4,659
	IMF2	84	137	97	186
	IMF3	5	20	7	28
	IMF4	N/A	5	N/A	6
	Residue	1	1	1	2

proposed method are generally consistent with those from the method presented in [7].

Figure 6 shows the IMFs and the reconstructed image for the STI shown in Fig. 5 for both the proposed method and the FABEMD approach for the *Type*-4 and MNAI = 5 cases. As shown in the figure, the proposed approach provides similar IMFs and residue images compared to the method presented in [7].

The performance results for the Barbara [9], Lena, and Elaine [10] images are given in Tables 5, 6, and 7, respectively. As these tables indicate, the proposed approach is able to generate similar IMFs with acceptable OIs compared to the method in [7], which illustrates the effectiveness of the proposed approach. The experimental results show that the proposed approach is able to provide up to a 20-fold increase in speed compared to FABEMD in [7]. This improvement directly originates from the proposed hierarchical data processing scheme. Note that our implementation is in MATLAB, and it does not benefit from the presented memory organization. A proper hardware implementation of the proposed method will fully utilize the potential of the computational gain illustrated in Table 2. However, because the other portions of the original FABEMD are kept as is, the improvements shown in Table 2 cannot be achieved for the entire EMD process.



(a) (b)

Fig. 7. Original and blurred versions of the Lena image.

We also investigated the performance of the proposed method for blurred input images. For this purpose, we applied Gaussian-based low-pass filtering using a 15 × 15 kernel size and $\sigma = 10$. The original and blurred versions of the Lena

ETRI Journal, Volume 38, Number 4, August 2016
http://dx.doi.org/10.4218/etrij.16.0114.1330

Table 8. Performance assessment for Lena.

		FABEMD method in [7]		Proposed method	
		MNAI = 1	MNAI = 5	MNAI = 1	MNAI = 5
Processing time (s)		25.28	71.15	5.51	13.79
OI		0.00617	0.00135	0.00257	–0.01386
Window size	IMF1	61	33	63	25
	IMF2	151	109	147	75
	IMF3	227	203	225	175
	IMF4	N/A	255	N/A	N/A
Stopping point SD	IMF1	0.93457	0.01327	0.93667	0.02899
	IMF2	0.99527	0.01334	1.00082	0.03478
	IMF3	0.95134	0.03214	0.95955	0.02229
	IMF4	N/A	0.03093	N/A	N/A
Average IMF values	IMF1	–2.13244	–0.65484	–2.29025	–0.09023
	IMF2	–0.00752	–3.75675	–0.03344	–2.94190
	IMF3	2.85610	–0.36365	2.34053	–2.47467
	IMF4	N/A	3.45545	N/A	N/A
	Residue	123.394	124.430	124.395	129.617
Sum of extremes in IMFs	IMF1	65	944	65	931
	IMF2	10	48	18	53
	IMF3	3	8	5	11
	IMF4	N/A	5	N/A	N/A
	Residue	1	1	1	2

image are shown Fig. 7. It is clear from Table 8 that, compared to FABEMD, although the number of IMFs extracted differs for the MNAI = 5 case, the proposed approach is able to extract the IMFs with an acceptable quality in terms of the OI.

IV. Conclusion

In this work, a new approach for B-EMD computations targeting an efficient hardware implementation is proposed. The performance of the proposed approach was shown through detailed experiments. The results indicate that B-EMD can be computed significantly faster through the proposed approach than using the original method, without degrading the quality of the decomposition.

References

[1] N.E. Huang et al., "The Empirical Mode Decomposition and the Hilbert Spectrum for Nonlinear and Non-stationary Time Series Analysis," *Proc. Royal Soc. A., Math. Physics Eng. Sci.*, vol. 454, no. 1971, 1998, pp. 903–995.

[2] M.-H. Lee et al., "Hardware Implementation of EMD Using DSP and FPGA for Online Signal Processing," *IEEE Trans. Ind. Electron.*, vol. 58, no. 6, June 2011, pp. 2473–2481.

[3] I. Kose and A. Celebi, "On the FPGA Implementation of Empirical Mode Decomposition Algorithm Using FPGA," *Signal Process. Commun. Appl. Conf.*, Haspolat, Turkey, Apr, 24–26, 2013, pp. 1–4.

[4] L. Wang et al., "Hardware-Accelerated Implementation of EMD," *Int. Conf. Biomed. Eng. inform.*, Yantai, China, Oct. 16–18, 2010, pp. 912–915.

[5] L.F. Chiang et al., "On-Line Empirical Mode Decomposition Biomedical Microprocessor for Hilbert Huang Transform," *IEEE Biomed. Circuits Syst. Conf.*, San Diego, CA, USA, Nov. 10–12, 2011, pp.420–423.

[6] W.-C. Fanget al., "An Efficient and Accurate Empirical Mode Decomposition of the Technical Design and Methods for Biological Sound," *IEEE Biomed. Circuits Syst. Conf.*, Hsinchu, China, Nov. 28–30, 2012, pp. 320–323.

[7] S.M.A. Bhuiyan, R.R. Adhami, and J.F. Khan, "Fast and Adaptive Bidimensional Empirical Mode Decomposition Using Order Statistics Filter Based Envelope Estimation," *EURASIP J. Adv. Signal Process.*, vol. 2008, Jan. 2008, pp. 1–18.

[8] M. Wielgus et al., "Fast and Adaptive Bidimensional Empirical Mode Decomposition for the Real-Time Video Fusion," *Int. Conf. Inform. Fusion*, Singapore, July 9–12, 2012, pp. 649–654.

[9] CIV, *Test Images*, Accessed Jan. 11, 2016. http://www.io.csic.es/PagsPers/JPortilla/image-processing/bls-gsm/63-test-images

[10] USCViteribi, *SIPI Image Database*, Accessed Jan. 11, 2016. http://sipi.usc.edu/database/?volume=misc



Serkan Semiz received his BS and MS degrees in electronics and telecommunication engineering from Istanbul Technical University, Turkey in 2002 and 2005, respectively. He worked as a researcher at the Scientific and Technological Research Council of Turkey Ankara, Turkey, from 2006 to 2013, and has been working as a software engineer at Otokar in Sakarya, Turkey since 2013. He is currently a PhD candidate at the Electronics and Telecommunications Engineering Department at the University of Kocaeli, Turkey. His research interests include digital signal processing, image processing, video processing, and embedded designs.



Anıl Çelebi received his BS, MS, and PhD degrees in electronics and communication engineering from Kocaeli University, Turkey in 2002, 2005, and 2008, respectively. Since 2002, he has been with the Department of Electronics and Telecommunications Engineering, University of Kocaeli, where he is currently working as an assistant professor. He worked as a BK21 post-doctoral research fellow at the School of Electrical Engineering and Computer Science at Seoul National University, Rep. of Korea from April to July of 2009. His research interests include very large-scale integration design and implementation of analog/mixed signal systems, image processing, and video coding systems.



Oğuzhan Urhan received his BS, MS, and PhD degrees in electronics and telecommunication engineering from the University of Kocaeli, Turkey in 2001, 2003, and 2006, respectively. Since 2001, he has been at the Department of Electronics and Telecommunications Engineering, University of Kocaeli, where he is currently a full professor. He was a visiting professor at Chung-Ang University, Seoul, Rep. of Korea, from 2006 to 2007. He is currently the director of the Kocaeli University Laboratory of Embedded and Vision Systems, Turkey. His research interests include digital signals, image/video processing, and embedded systems.