Improved Reversible Data Hiding in Encrypted Images Based on Reserving Room After Encryption and Pixel Prediction

Ioan Catalin Dragoi, Henri-George Coanda and Dinu Coltuc
Electrical Engineering Dept.
Valahia University of Targoviste, Romania
Email: {catalin.dragoi, henri.coanda, dinu.coltuc}@valahia.ro

***Abstract*—This paper proposes a new vacating room after encryption reversible data hiding scheme. Both joint and separate methods are presented. The most interesting features of the proposed scheme are the two staged embedding/decoding process and the group parity based data embedding for the separate method. Other new features are introduced as well. Compared with the state-of-the-art reserving room after encryption schemes, the proposed approach provides higher embedding bit-rates at lower distortion. Experimental results are provided.**

I. Introduction

Reversible data hiding (RDH) in encrypted images recently appeared as a promising research domain. As for RDH into clear images the correlation between image pixels is exploited (see [1], [2], [3]), but the encryption makes the domain more challenging.

Zhang introduced in [4] a RDH scheme for encrypted images based on dividing the encrypted image into blocks and embedding a bit in each block by flipping the 3 least significant bit values of half the pixels from the block. At the decoding stage, the correlation between the decrypted pixels of each block is used to detect which pixels had their bits flipped. The encrypted images were generated by an exclusive-or operation with pseudo-random bits. Hong et al. later improved this hiding scheme in [5] using better estimations and a side match technique. Ma et al. proposed in [6] a hiding scheme based on reserving room before encryption. As the name suggests, in [6] the image owner processes the image before encryption in order to create space for the data hider. This approach was further refined by multiple other papers, most notably [7]. Naturally, the embedding process is considerably more efficient then the one in [5], but the additional operations at encryption/decryption are inherently security risks and therefore schemes that do not have such drawbacks (like [5]) are still relevant. [4], [5] and other similar approaches, most notably [8] and [9], are now known as schemes based on reserving room after encryption. For a recent review of RDH in encrypted domain see [10].

This paper proposes a new vacating room after encryption RDH scheme. The main improvements of the proposed scheme are the use of a two staged embedding/decoding process, a pixel prediction based on the median context value and a group parity based embedding for the separate method. Compared with the state-of-the-art schemes like [8], higher embedding bit-rates can be obtained at lower distortion.

The outline of the paper is as follows. Both the joint and the separate versions of the reversible data hiding scheme of [8] are described in Section II. The proposed scheme, also with a joint and a separate version, is introduced in Section III. The experimental results are presented in Sections IV and the conclusions are drawn in Section V.

II. Related work

The reversible data hiding scheme for encrypted images introduced in [8] has two distinct versions: a joint method (where the hidden data extraction and the image restoration are both performed after the image was decrypted) and a separate method (where the hidden data can be decoded from the encrypted image, but the host image can only be restored after decryption).

Let C be the encrypted version of image I. Each bit plane of C, C_t, $(1 \leq t \leq 8)$ is computed as:

$$C_t = I_t \oplus r_t \tag{1}$$

where $\oplus$ is the exclusive-or operator and r is a standard pseudorandom bitstream sequence generated by an encryption key.

A. The joint method of [8]

This method embeds L bits in nL pixels of the encrypted image, $n \geq 1$. First nL pixels of C are selected based on a data hiding key. Once an encrypted pixel $C(i,j)$ is chosen, the pixels that form its prediction context (Figure 1.b) cannot be selected for data embedding. A data bit is embedded by flipping the t bit of n selected pixels:

$$C'_t(i,j) = \begin{cases} {\sim}C_t(i,j), & \text{if } b = 1, \\ C_t(i,j), & \text{if } b = 0. \end{cases} \tag{2}$$

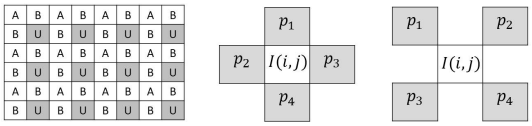
where $\sim$ is the not operator and $b \in \{0,1\}$ is the hidden bit.

A watermarked version of I is obtained by decrypting each bit plane of C' using the encryption key:

$$I'_t = C'_t \oplus r_t \tag{3}$$

If the user has access to the data hiding key, the embedded data can be extracted. The nL watermarked pixels are first

ISBN 978-0-9928626-7-1



(a) (b) (c)

Fig. 1. The proposed pixel distribution (a), the prediction context used by set B and the scheme of [8] (b), the prediction context used by set A (c).

selected using the hiding key. For each of those pixels, $I''(i,j)$ is generated by flipping $I'_t(i,j)$. Then $\hat{I}(i,j)$, the predicted value of $I(i,j)$ (a weighted average on the prediction context), is used to determine b, the hidden bit inserted in each group of n pixels:

$$b = \begin{cases} 0, & \text{if } \sum_{k=1}^{n} \left| I'(i_k,j_k) - \hat{I}(i_k,j_k) \right| \leq \sum_{k=1}^{n} \left| I''(i_k,j_k) - \hat{I}(i_k,j_k) \right| \\ 1, & \text{if } \sum_{k=1}^{n} \left| I'(i_k,j_k) - \hat{I}(i_k,j_k) \right| > \sum_{k=1}^{n} \left| I''(i_k,j_k) - \hat{I}(i_k,j_k) \right| \end{cases} \tag{4}$$

where $|x| = \begin{cases} x, & \text{if } x \geq 0 \\ -x, & \text{if } x < 0 \end{cases}$.

If $b = 1$, all n pixels in the current group are replaced with their corresponding $I''(i,j)$ value, otherwise the pixels remain unchanged. Both n (the number of pixels in a group) and t (the bit plane used for data hiding) have a direct influence on the precision of equation (4). A larger value for n improves the precision of (4), but reduces the available space for the hidden data. Similarly, a larger value for t also improves the precision of (4), but the watermarking distortion increases.

B. The separate method of [8]

This method embeds L additional bits in L pixels (as opposed to nL pixels) selected based on the hiding key. The secret bit b is inserted directly in the host pixel $C(i,j)$ by substituting its t bit:

$$C'_t(i,j) = b \tag{5}$$

A user with access to the hiding key can decode the hidden data by reading the t bits of the L selected pixels. After (3) is used to decrypt $I'(i,j)$, the closest value to $\hat{I}(i,j)$ between $I'(i,j)$ and $I''(i,j)$ is selected as the original value of $I(i,j)$.

Note that all separate approaches can correctly decode the hidden data, but the restoring step remains affected by the possibility of errors. Also note that this approach needs a larger value for t in order to compensate for the embedding in a single pixel instead of a group of n. In [8] the author recommends $t \geq 7$ and adds a filtering stage after decryption in order to remove the distortions introduced by the watermark. This additional filtering step draws attention to the existence of the hidden data in the encrypted image and represents a serious security risk. The proposed scheme in Section III-B does not have this drawback.

III. Proposed scheme

Similarly to [8], the proposed scheme has two distinct versions: a joint method (watermark decoding and image restoration on the decrypted image) and a separate method (watermark decoding on the encrypted image, restoration on the decrypted image).

The encrypted image (generated with [1]) is split by both proposed methods into three distinct sets. These sets are presented in Figure 1.a. Only sets A and B (a total of $2/3$ of the image) are used for data hiding, set U is not modified by the embedding algorithm. Note that [8] can embed data in at most $1/2$ of the image (the other half is used for prediction).

The data hiding key is used to determine the order in which the pixels in set A and set B are processed. Set A is the first set to be embedded with the hidden data, the pixels in B are considered as possible hosts only after the capacity offered by A is completely exhausted. The pixels in each set are processed as groups of n pixels and a bit of data will be inserted in each group by modifying their t bit value.

A 32 bit identifier (used to distinguish between watermarked and non-watermarked images) together with the value of L (also 32 bits) are appended as a prefix to the hidden data. These 64 bits are the first to be embedded and extracted. Note that this approach assumes that the data hider uses fixed values for n, N and K based on t. A flexible approach is also possible by using the fixed values to insert the identifier, L, n, N, K and then switching to the indicated values for the remaining host pixels.

A. Proposed joint method

The proposed joint method uses error-correcting codes on the hidden data in order to reduce the number of decoding errors (at the cost of embedding capacity). More precisely Bose-Chaudhuri-Hocquenghem (BCH) codes [11] are used. The (N,K) BCH code adds to each K bits an additional $N-K$ bits (forming a group of N bits) in order to correct up to e errors. If in a group of N bits the number of errors exceeds e, the BCH decoding is compromised. After the BCH coding, $(64+L) \cdot n \cdot N/K$ host pixels are selected from sets A and B. Equation (2) is used to embed a data bit into each group of n pixels.

After decryption, a user with the hiding key can extract the hidden data by determining the order in which the pixels were embedded and reforming the n pixel groups. Set A is decoded and restored first. For each pixel in A, the predicted value $I'(i,j)$ is computed as the median on its prediction context (Figure 1.c):

$$\hat{I}(i,j) = \left\lfloor \frac{p_{(2)} + p_{(3)}}{2} + \frac{1}{2} \right\rfloor \tag{6}$$

where $\lfloor x \rfloor$ represents the greatest integer less than or equal to x and $p_{(1)} \leq p_{(2)} \leq p_{(3)} \leq p_{(4)}$ are the sorted graylevel values of the pixels that form the prediction context. Note that all context pixels for set A belong to set U. $\hat{I}(i,j)$ is then used in equation (4) to determine the hidden data bit in the current group.

The value of t is set to 6 and the first $64 \cdot N/K$ bits are extracted. The BCH decoder is used to correct any possible decoding errors and if the watermark identifier is found, the algorithm proceeds to extract data from $L \cdot n \cdot N/K$ pixels.

ISBN 978-0-9928626-7-1 © EURASIP 2017

Otherwise t is decremented by 1 and the process is repeated. If the identifier was not found for $t = 3$, then the image did not contain hidden data. After the hidden bits in set A are extracted, the errors are corrected and the host pixels are restored based on the corrected bit values. After all the pixels in set A are restored, the additional bits introduced by the error correction code are removed.

The entire decoding process is repeated for set B, using the prediction context from Figure 1.b, consisting of original pixels from set U and restored pixels from set A.

B. Proposed separate method

The proposed separate method inserts L bits of data in $n(64 + L)$ pixels. As opposed to [8], both n and t are used to minimize the image restoration errors, at the same time allowing for a smaller value for t. As an additional requirement for this proposed approach, n must be an odd number. Also note that the above mentioned error correction codes cannot be used for this method, because the watermark extraction and the image recovery are done independently.

Instead of substituting the t bit of an encrypted pixel $C(i, j)$ with a hidden bit of data, the pixels are processed in groups of n. For each group, $\{C^1, C^2, ..., C^n\}$, a parity value is computed for the t bit plane:

$$s = C_t^1 \oplus C_t^2 \oplus ... \oplus C_t^n \tag{7}$$

Flipping the t bit values of all the pixels in the group will always flip the value of s if n is an odd number. This allows us to substitute the value of s with the hidden bit b by flipping the t plane bits of n pixels:

$$C'_t(i,j) = \begin{cases} {\sim}C_t(i,j), & \text{if } s \neq b, \\ C_t(i,j), & \text{if } s = b. \end{cases} \tag{8}$$

The data extraction stage is similar to the one in Section III-A: t is set to 6 and the first 64 bits are extracted from the parity values s of the first $64 \cdot n$ pixels ordered based on the hiding key. If the 32 bit watermark identifier is found, the algorithm proceeds to extract the remaining L bits. Otherwise t is decremented by 1 and the process is repeated until the identifier is found or no watermark is detected.

Because the pixels are processed as a group, they can be restored after the decryption stage with the help of equation (4). Note that in (4) b is no longer a hidden bit and is only used as an indicator to choose between $I'(i, j)$ and $I''(i, j)$.

IV. Experimental results

In this section, experimental results for the proposed RDH scheme for encrypted images are presented. Two sets of images are considered: eight classic graylevel 512×512 images extensively used in reversible watermarking (*Lena*, *Boat*, *Tiffany*, *Elaine*, *Lake*, *Mandrill*, *Jetplane* and *Barbara*) and the graylevel 768×512 versions of the 24 images of the Kodak set. The test images are presented in Figure 2. All test images are encrypted with (1) by using randomly generated encryption keys.

Fig. 2. The eight classic test images (first two rows) and the Kodak set.

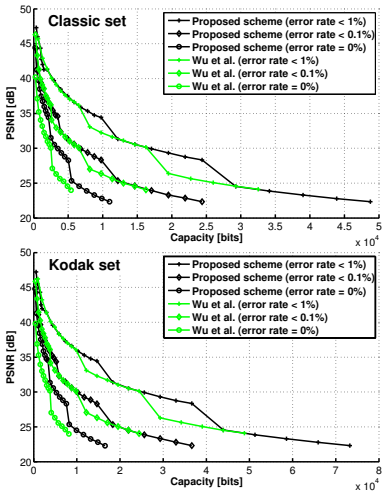


Fig. 3. Average capacity/PSNR results for three distinct error rates.

Three primary factors are considered in evaluating the performance of a data hiding scheme based on reserving room after encryption: the embedding capacity, the error rate of the decoded bits and the distortions introduced by watermarking. The maximum capacity for the proposed joint method and [8] are presented in Table I (classic set) and Table II (Kodak set). Note that the bit plane t has no direct influence over the embedding capacity, but t does significantly influence the error rate and the watermarking distortions. The average error rate of the decoded bits on the entire test set are presented in Table III. As it can be seen from the three tables, the proposed scheme with no error correction offers a significant increase in capacity over [8], while maintaining similar error rates. Of course, the BCH codes can be used to reduce the error rate at the cost of capacity. This allows the data hider to embed data using a smaller value for t while maintaining the target error rate that one desires, which in turn will reduce the embedding distortions.

The embedding distortions are evaluated based on the peak signal-to-noise ratio (PSNR) between the original image and the watermarked decrypted version. Figure 3 shows the average capacity/PSNR results for the proposed scheme and the joint method of [8] for three distinct error rates on the two test sets. Note that both the joint and the separate proposed methods offer the same capacity/PSNR results, the separate

ISBN 978-0-9928626-7-1 © EURASIP 2017

TABLE I
MAXIMUM CAPACITY ON THE CLASSIC SET OBTAINED BY EMBEDDING ONE BIT OF DATA IN n PIXELS [BITS].

n	Wu et al. [8]	Proposed scheme							
		no coding	BCH (7,4)	BCH (15,7)	BCH (15,5)	BCH (31,21)	BCH (31,16)	BCH (31,11)	BCH (31,6)
5	26010	38715	21988	17907	12705	26118	19828	13538	7248
9	14450	21375	12084	9808	6920	14379	10884	7389	3894
13	10003	14704	8272	6693	4695	9843	7428	5013	2598
17	7650	11175	6252	5055	3525	7449	5604	3759	1914
22	6192	8988	5004	4026	2790	5958	4468	2978	1488
25	5202	7503	4156	3333	2295	4950	3700	2450	1200
29	4484	6426	3540	2829	1935	4236	3156	2076	996
33	3940	5610	3072	2451	1665	3690	2740	1790	840
37	3514	4971	2712	2157	1455	3249	2404	1559	714
41	3171	4456	2416	1912	1280	2913	2148	1383	618
45	2890	4035	2172	1716	1140	2619	1924	1229	534
49	2827	3940	2116	1674	1110	2556	1876	1196	516
53	2453	3379	1800	1408	920	2178	1588	998	408
57	2281	3121	1648	1296	840	1989	1444	899	354
61	2131	2896	1524	1191	765	1842	1332	822	312

TABLE II
MAXIMUM CAPACITY ON THE KODAK SET OBTAINED BY EMBEDDING ONE BIT OF DATA IN n PIXELS [BITS].

n	Wu et al. [8]	Proposed scheme							
		no coding	BCH (7,4)	BCH (15,7)	BCH (15,5)	BCH (31,21)	BCH (31,16)	BCH (31,11)	BCH (31,6)
5	39066	58299	33180	27042	19230	39390	29940	20490	11040
9	21703	32254	18300	14883	10545	21750	16500	11250	6000
13	15025	22237	12576	10207	7205	14946	11316	7686	4056
17	11490	16935	9544	7743	5445	11355	8580	5805	3030
22	9301	13651	7668	6210	4350	9150	6900	4650	2400
25	7813	11419	6396	5160	3600	7638	5748	3858	1968
29	6735	9802	5472	4411	3065	6525	4900	3275	1650
33	5919	8578	4768	3837	2655	5685	4260	2835	1410
37	5279	7618	4224	3382	2330	5055	3780	2505	1230
41	4764	6846	3780	3025	2075	4509	3364	2219	1074
45	4340	6210	3420	2731	1865	4110	3060	2010	960
49	4246	6069	3336	2668	1820	3984	2964	1944	924
53	3685	5227	2856	2269	1535	3417	2532	1647	762
57	3426	4839	2632	2094	1410	3165	2340	1515	690
61	3202	4503	2440	1933	1295	2934	2164	1394	624

method of [8] obtains a PSNR of around 32 dB, but without the filtering stage the PSNR is consistently bellow 20 dB. As can be seen from Figure 3 the proposed scheme outperforms [8] on the tested error rates. For an error rate smaller than 1%, the proposed scheme offers a capacity of 20000 bits on the classic set and 30000 bits on the Kodak set, while maintaining a PSNR of around 30 dB. The maximum capacity can be further increased to 48000 bits and 75900 bits, respectively, but at the cost of introducing visible watermarking distortion. On the tested sets, the proposed scheme also inserts around 4000 bits at a PSNR of 30 dB without any decoding errors.

V. CONCLUSION

An original reserving room after encryption RDH scheme has been proposed. Both joint and separate RDH versions are investigated. The most interesting features are the use a two staged embedding and the group parity approach for the separate method. The experimental results obtained so far are very promissing.

ACKNOWLEDGMENT

This work was supported by UEFISCDI Romania, PNIII-P4-IDPCE-2016-0339 and PN-II-PTPCCA-2013-4-1762 Grants.

REFERENCES

[1] V. Sachnev, H. J. Kim, J. Nam, S. Suresh and Y. Q. Shi, "Reversible Watermarking Algorithm Using Sorting and Prediction", *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, pp. 989–999, 2009.

[2] I.-C Dragoi and D. Coltuc, "Local-Prediction-Based Difference Expansion Reversible Watermarking", *IEEE Trans. on Image Processing*, vol. 23, no. 4, pp. 1779–1790, 2014.

[3] X. Li, W. Zhang, X. Gui, and B. Yang, Efficient reversible data hiding based on multiple histograms modification, *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 9, pp. 20162027, 2015.

[4] X. Zhang, "Reversible data hiding in encrypted images", *IEEE Signal Process. Lett.*, vol. 18, pp. 255–258, 2011.

[5] W. Hong, T. Chen, and H.Wu, "An improved reversible data hiding in encrypted images using side match", *IEEE Signal Process. Lett.*, vol. 19, pp. 199–202, 2012.

[6] K. Ma, W. Zhang, X. Zhao, N. Yu, and F. Li, "Reversible Data Hiding in Encrypted Images by Reserving Room Before Encryption", *IEEE Trans. Inf. Forensics Security*, vol. 8, pp. 553–568, 2013.

[7] X. Cao, L. Du, X. Wei, D. Meng, and X. Guo, "High capacity reversible data hiding in encrypted images by patch-level sparse representation", *IEEE Trans. Cybernetics*, vol. 46, pp. 1132–1143, 2016.

[8] X. Wu and W. Sun, "High-capacity reversible data hiding in encrypted images by prediction error", *Signal Processing*, pp. 387–400, 2014.

[9] Y.-S. Kim, K. Kang and D.-W. Lim, "New Reversible Data Hiding Scheme for Encrypted Images using Lattices", *Appl. Math.*, vol. 9, pp. 2627–2636, 2015.

[10] Y.-Q. Shi, X. Li, X. Zhang, H.-T. Wu and B. Ma., "Reversible data hiding: advances in the past two decades", *IEEE Access*, vol. 4, pp. 3210-3237, 2016.

[11] R.C. Bose and D.K. Ray-Chaudhuri, "On A Class of Error Correcting Binary Group Codes", *Information and Control*, vol. 3, pp. 68–79, 1960.

TABLE III
AVERAGE ERROR RATE OF THE DECODED BITS INSERTED IN THE t BIT PLANE (ALL TEST IMAGES) [%].

t	n	**Wu et al. [8]**	**Proposed scheme**							
			no coding	**BCH (7,4)**	**BCH (15,7)**	**BCH (15,5)**	**BCH (31,21)**	**BCH (31,16)**	**BCH (31,11)**	**BCH (31,6)**
3	5	19.746	18.968	31.842	36.185	40.6	29.099	34.377	41.244	52.729
	9	11.985	11.459	16.978	18.463	18.886	16.415	18.327	19.542	22.615
	13	8.274	7.9947	11	11.614	11.336	10.745	11.519	11.369	12.82
	17	6.1128	5.9338	7.7662	8.0261	7.5812	7.6798	8.0222	7.7532	8.4848
	21	4.7622	4.6299	5.8053	5.8132	5.5543	5.8583	5.8776	5.6179	5.9998
	25	3.8158	3.7848	4.5309	4.5692	4.3503	4.5912	4.5648	4.6571	4.8608
	29	3.0802	3.1252	3.6236	3.6757	3.1752	3.7081	3.6739	3.5205	3.4662
	33	2.6513	2.5868	3.0113	3.0248	2.7547	3.0085	3.1645	2.7962	2.9261
	37	2.2726	2.2331	2.5998	2.6478	2.2517	2.5756	2.7172	2.5252	1.9699
	41	1.8339	1.9536	2.0629	2.2633	1.8871	2.1796	2.3898	1.9431	1.5465
	45	1.6667	1.6931	1.8604	2.0041	1.4553	1.9262	1.8741	1.7779	1.0118
	49	1.4574	1.5214	1.6369	1.6669	1.366	1.7777	1.6779	1.3614	0.96461
	53	1.2868	1.3277	1.33	1.4738	1.1296	1.5514	1.4956	1.2048	0.78375
	57	1.096	1.2235	1.3044	1.2451	0.82625	1.3981	1.3646	0.9286	0.57749
	61	1.05	1.102	1.1386	1.0898	0.75231	1.3346	1.313	0.88308	0.75053
4	5	9.6593	8.9878	12.018	12.512	11.569	12.305	12.638	11.916	11.868
	9	4.1017	3.7507	4.2354	4.0948	3.4751	4.3145	4.1266	3.5996	3.1437
	13	2.2457	2.061	2.0968	1.9763	1.39	2.1942	2.1679	1.6225	0.89574
	17	1.4154	1.3097	1.253	1.1426	0.68459	1.3141	1.2017	0.67198	0.23177
	21	0.93534	0.89304	0.77812	0.62806	0.29321	0.85013	0.68942	0.26224	0.066445
	25	0.65259	0.62871	0.45008	0.35132	0.17922	0.60081	0.38438	0.13795	0.025814
	29	0.45159	0.50631	0.33073	0.21528	0.070264	0.36465	0.26147	0.074337	0
	33	0.3327	0.35953	0.18112	0.14529	0.063346	0.23246	0.10939	0	0
	37	0.23919	0.28058	0.17031	0.073216	0.016667	0.14037	0.059073	0.034611	0
	41	0.18571	0.22617	0.094373	0.035298	0	0.071878	0.046908	0	0
	45	0.14829	0.17726	0.053108	0.016777	0	0.054368	0.02068	0.0091285	0
	49	0.10681	0.13853	0.034085	0.013515	0	0.053327	0	0	0
	53	0.086596	0.11181	0.034575	0	0	0.053284	0	0	0
	57	0.044823	0.10244	0.026493	0	0	0.031896	0	0	0
	61	0.058604	0.064371	0.0037681	0.012892	0	0	0.009273	0	0
5	5	3.479	3.104	2.7369	2.4532	1.6252	3.0538	2.6343	1.6962	0.89949
	9	0.83273	0.77156	0.42941	0.31203	0.14618	0.54429	0.35215	0.077159	0.011062
	13	0.29571	0.29965	0.12971	0.065236	0.018923	0.12905	0.04358	0.0074788	0
	17	0.12152	0.13717	0.049519	0.0030366	0	0.03002	0.0048275	0	0
	21	0.056349	0.06998	0.011076	0.0014352	0	0.0032511	0	0	0
	25	0.023821	0.036386	0.0056205	0	0	0.0027505	0	0	0
	29	0.013226	0.020706	0	0	0	0	0	0	0
	33	0.0040977	0.0099789	0	0	0	0	0	0	0
	37	0.00057604	0.0050218	0	0	0	0	0	0	0
	41	0.00096124	0.0058149	0	0	0	0	0	0	0
	45	0.0017615	0.00098301	0	0	0	0	0	0	0
	49	0.000768	0.00053851	0	0	0	0	0	0	0
	53	0	0.0009058	0	0	0	0	0	0	0
	57	0	0.002249	0	0	0	0	0	0	0
	61	0	0.00068202	0	0	0	0	0	0	0
6	5	0.78397	0.68696	0.29609	0.17812	0.051234	0.32722	0.16552	0.037294	0.012159
	9	0.075746	0.081664	0.0094983	0.0034543	0.0027973	0.0088271	0.0011773	0	0
	13	0.010181	0.017533	0.00068691	0	0	0	0	0	0
	17	0.0030748	0.0029545	0	0	0	0	0	0	0
	21	0.00080061	0.0013145	0	0	0	0	0	0	0
	25	0	0	0	0	0	0	0	0	0
	29	0	0	0	0	0	0	0	0	0
	33	0	0	0	0	0	0	0	0	0
	37	0	0	0	0	0	0	0	0	0
	41	0	0	0	0	0	0	0	0	0
	45	0	0	0	0	0	0	0	0	0
	49	0	0	0	0	0	0	0	0	0
	53	0	0	0	0	0	0	0	0	0
	57	0	0	0	0	0	0	0	0	0
	61	0	0	0	0	0	0	0	0	0

ISBN 978-0-9928626-7-1 © EURASIP 2017