Capacity Control for Prediction Error Expansion based Audio Reversible Data Hiding

Alin Bobeica
Electrical Engineering Dept.
Valahia University of Targoviste
Targoviste, Romania
alin.bobeica@valahia.ro

Ioan Catalin Dragoi
Electrical Engineering Dept.
Valahia University of Targoviste
Targoviste, Romania
catalin.dragoi@valahia.ro

Ion Caciula
Electrical Engineering Dept.
Valahia University of Targoviste
Targoviste, Romania
ion.caciula@valahia.ro

Dinu Coltuc
Electrical Engineering Dept.
Valahia University of Targoviste
Targoviste, Romania
dinu.coltuc@valahia.ro

Felix Albu
Electrical Engineering Dept.
Valahia University of Targoviste
Targoviste, Romania
felix.albu@valahia.ro

Feiran Yang
Institute of Acoustics
Chinese Academy of Sciences
Beijing, China
feiran@mail.ioa.ac.cn

***Abstract*—This paper presents an efficient capacity control algorithm for prediction error expansion based audio reversible data hiding. Current state-of-the-art audio reversible data hiding schemes use a simple capacity control algorithm that was first developed for image reversible data hiding. The performance of this algorithm can be improved by using a simple two threshold based approach. The two threshold approach can be easily integrated into any prediction error expansion based framework. Experimental results are provided for two such frameworks.**

***Index Terms*—reversible data hiding, audio, prediction error expansion, capacity control**

I. Introduction

Digital reversible data hiding (RDH) aims to losslessly recover both the hidden data and the host signal. Up to recently, the research in RDH was mainly focused on embedding data into image files, but promising results were also obtained for audio and encrypted files.

The prediction error expansion (PEE) based RDH framework proposed by Thodi et. al. in [1] is the most commonly used framework for image RDH. For PEE RDH, the prediction error is computed as the difference between an original sample and a predicted value. This predicted value is computed based on neighboring samples situated in close proximity to the estimated sample. Space for the hidden data is created into each host sample by doubling its corresponding prediction error. A hidden data bit can now be stored in the least significant bit (LSB) position of the prediction error. The performance of the PEE based RDH framework can be improved by the use of more accurate predicted values. Doubling a smaller prediction error will produce a smaller distortion of the host sample. In order to further reduce the embedding distortion, Sachnev et a. al. proposed in [2] a two stage PEE based image RDH scheme using the rhombus average as a predictor. This approach was later adapted in [3] by Huo et. al into an audio RDH scheme.

This work was supported by UEFISCDI Romania, in the frame of PN-III-P4-IDPCE-2016-0339 and PN-III-P1-1.1-PD-2016-1666 Grants.

The RDH scheme of [3] processes the samples in two stages of embedding. First, the samples on odd positions are used to predict the samples on even position. The rhombus average of [2] is replaced by a simple average of two consecutive odd samples (past and future) around the estimated even sample. In the second stage, the modified even samples are used to predict the odd samples.

Wang et. al. in [4] proposed a linear prediction based audio RDH approach. A causal linear predictor is created using three consecutive samples. The coefficients of the predictor are computed using a differential evolution algorithm in order to minimize the embedding-distortions function. While the effectiveness of linear predictors was shown in other research field, the causal nature of the approach limits its overall performance.

Recently, Xiang et. al. proposed in [5] a two staged PEE audio RDH scheme which combines linear prediction with a non-causal prediction context. In other words, a host sample is predicted using a linear combination of past and future samples.

The PEE based approaches of [3]–[5] process the host signal in the time domain. Audio RDH schemes were also developed for the frequency domain, most notably [6], [7]. These approaches process the host signal in the integer DCT domain. While this domain is ideal for developing robust data hiding schemes, for RDH the PEE based approach offers superior distortion/bit-rate results.

Current PEE based audio RDH schemes (like [3]–[5]) use the classic capacity control algorithm proposed in [1]: an embedding threshold is selected that provides the needed capacity, when this capacity is reached, the embedding process is considered complete and the remaining samples are kept unchanged. This capacity control algorithm was shown to be suboptimal, other approaches like [8] are more efficient. In this paper we present a new capacity control algorithm, derived from [8], that should outperform the classic approach of [1],

978-1-5386-4444-7/18/$31.00 ©2018 IEEE

the only approach that is currently used in PEE based audio RDH.

The outline of the paper is as follows. The basic framework of PEE based audio RDH is presented in Section II. The classic and proposed capacity control algorithms are discussed in Section III. Experimental results are provided in Section IV and the conclusions are drawn in Section V.

II. PEE BASED RDH

In this section, the PEE based RDH framework of [3] is discussed. This framework is also used by [5] and was derived from [2]. A hidden message is inserted into an host audio file using the algorithm presented in Section II-A. The message is extracted and the host is restored using the algorithm from Section II-B.

A. Embedding stage

The samples that form the audio file are first split into two distinct regions: R (reserved) and S (selected). Usually R contains the first 200 samples of the file. The LSB values of each sample in R are appended to the message. This reserved samples are later used to store the embedding parameters using LSB substitution.

Region S is further divided into two sets: cross and dot. The cross set contains samples with even numbered positions and is the first to be processed. The dot set contains samples with odd numbered positions and is processed after the cross set embedding process was concluded. The hidden data is divided into two equal halves, the first halve will be embedded into the cross set and the second into the dot set.

A prediction context is formed around each sample using neighboring samples. Based on this context, a predicted value is computed for each sample from the current set. The size and shape of the prediction context depends on the embedding scheme. Let x_i be the value of the sample found at the i position in the S region. For the RDH scheme of [3], the prediction context is formed from x_{i-1} and x_{i+1}, the corresponding predicted value is computes as:

$$\hat{x_i} = \left\lfloor \frac{x_{i-1} + x_{i+1}}{2} + \frac{1}{2} \right\rfloor \tag{1}$$

where $\lfloor a \rfloor$ represents the greatest integer less than or equal to a.

The RDH scheme of [5] uses a much more complex predictor. The prediction context is formed from up to 60 neighboring samples (the nearest 20 past samples belonging to the set that does not contain the current sample and the nearest 40 future samples). The prediction context is reordered based on the average difference between the neighbors and the current sample. Multiple linear predictors are then computed on the sorted context. The predicted values are determined with the linear predictor that provides on average the smallest absolute error (see [5]).

Next, for each sample from the current set, a prediction error is computed based on $\hat{x}_i$:

$$e_i = x_i - \hat{x}_i \tag{2}$$

A embedding threshold T is used to determine the samples that will serve directly as hosts for the hidden data. The selection of T is discussed in Section III.

If $e_i \geq -T$ and $e_i < T$, a hidden data bit is embedded into x_i by expanding its prediction error:

$$x_i' = x_i + e_i + b \tag{3}$$

where x_i' represents the new value of x_i, after embedding the hidden data bit $b \in \{0, 1\}$. The new prediction error for the current sample is:

$$e_i' = x_i' - \hat{x}_i = x_i + e_i + b - \hat{x}_i = 2e_i + b \tag{4}$$

The samples with $e_i < -T$ or $e_i \geq T$ are shifted by $\pm T$ in order to prevent their error values from overlapping with those that contain hidden data:

$$x_i' = \begin{cases} x_i + T & \text{if } e_i \geq T \\ x_i - T & \text{if } e_i < -T \end{cases} \tag{5}$$

The x_i sample values are limited to a spatial domain, namely $[-32768, 32767]$ for 16 bit audio files. The value of x_i' must be also limited to this domain, otherwise the sample is decoded incorrectly. This is known as the overflow/underflow problem and is solved in this paper using the flag-bit approach of [1]. A sample x_i is replaced with x_i' only if $x_i' \in [-32768, 32767]$. Otherwise its value remains unchanged and a the flag-bit "1" is embedded in the next available host sample (instead of a data bit) using equation (3). The flag-bit "0" is embedded in the next available host sample if $x_i' \notin [-32768+T, 32767-T]$ (in order to distinguish between these modified values and the values that were not modified because of overflow/underflow). If two or more flag bits need to be embedded, then the last flag bit has priority. If no host samples remain, the remaining flag-bits are stored in the reserved region R. Note that for most audio files there is no risk of overflow/underflow.

After the hidden data corresponding to the cross set was embedded, the embedding process is repeated for the dot set. The dot set samples are predicted using the modified values from the cross set (and future values from the dot set for [5]). e_i is computed with (2). Based on its e_i value, the current sample is either modified with (3) or (5). The embedding process concludes by storing the threshold T, the positions of the last modified samples from both sets and any remaining flag-bits in R by LSB substitution.

B. Decoding stage

The decoding process starts by reforming the R and S regions. The parameters that were stored in R are extracted by reading the corresponding LSBs. S is divided into the cross and dot sets. The samples are processed in reverse order from the embedding stage, starting with the last modified sample from the dot set. The samples in the cross set are processed after the samples in the dot set were completely restored.

If $x_i' \in [-32768+T, 32767-T]$, then the current (modified) sample was not at risk of overflow/underflow and did not require a flag-bit. The predicted value $\hat{x}_i$ is recomputed using

the same prediction context as at the embedding stage. The modified prediction error is then computed with:

$$e'_i = x'_i - \hat{x}_i \tag{6}$$

The current value contains a hidden bit if $e'_i \in [-2T, 2T)$. The bit is extracted by reading the LSB of the modified prediction error:

$$b = e'_i \bmod 2 \tag{7}$$

The host sample is then restored with:

$$x_i = x'_i - \lfloor e'_i/2 \rfloor - b \tag{8}$$

Samples with $e'_i < -2T$ or $e'_i \geq 2T$ do not contain a data bit, but were shifted with $\pm T$. Their values are restored with:

$$x_i = \begin{cases} x'_i - T & \text{if } e_i \geq 2T \\ x'_i + T & \text{if } e_i < -2T \end{cases} \tag{9}$$

If $x'_i \notin [-32768+T, 32767-T]$, then the last extracted hidden bit was a flag-bit. The current sample remains unchanged if the flag-bit is "1", otherwise x_i is restored with (8) or (9). In both cases, the flag-bit is then removed from the hidden data bitstream.

After the S region is completely restored, the original R region LSBs are removed from the hidden data bitstream. These values are used to restore the samples in R.

III. Capacity control for PEE

The data embedding/decoding scheme described in Section II requires an appropriate embedding threshold T in order to select the correct number of host samples, which in turn determines the provided capacity. T also controls the embedding distortions introduced by the PEE scheme, therefore the selection of T is crucial in obtaining optimal data hiding results.

A. The classic approach

Current state-of-art PEE based audio RDH schemes use the basic capacity control algorithm first introduced in [1]. Let N be the required number of host samples for the current set:

$$N = \frac{N_b}{2} + \frac{N_R}{2} + N_o \tag{10}$$

where N_b is the size of the hidden data, N_R is the number of stored LSBs from R and N_o is the estimated number of samples that are at risk of overflow/underflow (samples with $x_i \notin [-32768 + T, 32767 - T]$). Note that both in [3] and [5], the capacity control is only discussed in general terms, for simplicity we shall use the above equation.

The prediction error is determined with (2) for each pixel in the current set. These values are collected into a prediction error histogram. Let H be the prediction error histogram, where $H(e_i)$ is equal with the total number of samples in the current set with the prediction error e_i. The embedding threshold is selected as the smallest T value that fulfills:

$$\sum_{e_i=-T}^{T-1} H(e_i) \geq N \tag{11}$$

After all $\frac{N_b}{2}$ hidden data bits were inserted together with $\frac{N_R}{2}$ auxiliary data bits and no flag-bits remain, then the remaining pixels in the set are no longer processed and are kept unchanged (the position of the last modified sample is also recorded).

B. The proposed approach

The proposed capacity control algorithm is derived from the image RDH scheme first proposed in [8] and further refined in [9]. As opposed to the classic approach (where the large majority of samples are modified using the selected threshold T), the proposed approach uses two distinct thresholds: T and $T-1$, where T was determined with (11).

Let D be the number of additional host samples that are required to hide a total of N bits if $T-1$ was used as the embedding threshold:

$$D = N - \sum_{e_i=-T+1}^{T-2} H(e_i) \tag{12}$$

where N was computed with (10) and $\sum_{e_i=-T+1}^{T-2} H(e_i)$ is the number of host samples provided by $T-1$.

The embedding process for the current set starts by using T as the embedding threshold. After a total of D bits were successfully embedded into samples with $e_i \in \{-T, T-1\}$, the embedding threshold becomes $T-1$ for the remaining samples in the set. In other words, the samples are modified using T until the required capacity that could not be provided by $T-1$ was inserted, then the remaining samples are modified with $T-1$.

The main improvements over [8] and [9] are the flag-bit based capacity estimation in (10) and the streamlined implementation model described in this section. Furthermore, both [8] and [9] were developed for image RDH and, as far as we know, the (inferior) classic capacity control algorithm is the only model currently used in audio RDH.

IV. Experimental results

In this section, the proposed capacity control algorithm is compared with its classic counterparts on the PEE audio RDH framework of [3] and the state-of-the-art scheme of [5]. The performance of the two approaches is evaluated on 70 standard audio files with a sampling rate of 44.1 kHz. The test set is available at [10] and contains: alignment signals (clips 1 and 2), artificial signals (3 to 7), single instruments (8 to 43), vocal (44 to 48), speech (49 to 54), solo instruments (55 to 60), vocal with orchestra (61 to 64), orchestra (65 to 68) and pop music (69 and 70).

Two metrics are used to evaluate the (reversible) distortion introduced into the host file by the embedding scheme, namely the signal-to-noise-ratio (SNR) and the objective difference grade (ODG). The ODG score between the original signal and its modified version was computed using the GstPEAQ software developed by Holters et. al. in [11], the basic mode of the program is used.

The two capacity control approaches are first compared on four of the test clips, the SNR/bit-rate results are shown in

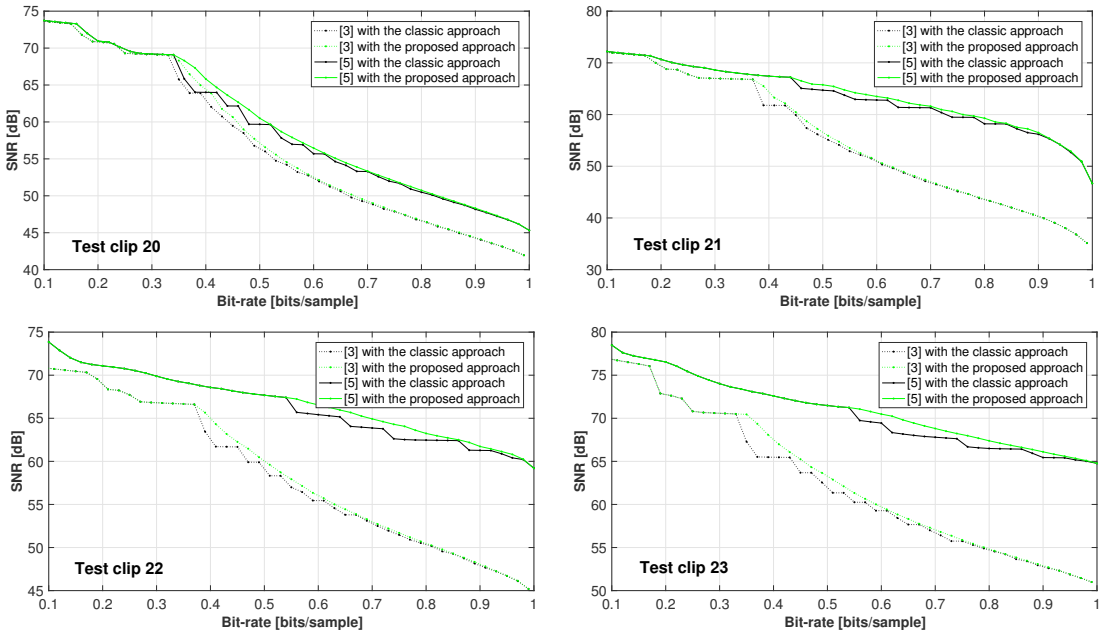

Fig. 1. SNR/Bit-rate results using the classic capacity control algorithm and the proposed approach

Figure 1 and the corresponding ODG/bit-rate results are shown in Figure 2. A clear gain in performance is observed on all four test clips. This gain is more pronounced when using the more recent audio RDH scheme of [5]. Note that when T has its smallest value (i.e. $T = 1$), the $T - 1$ threshold does not exist, therefore both approaches produce the same results (embed using T, otherwise keep the samples unchanged). This happens for the test clip 20 at bit-rates below 0.35 bits/sample.

The performance of the proposed scheme is also evaluated on the entire test set, the results are shown in Table I (SNR) and Table II (ODG). Using [3] with the proposed capacity control brings an average increase in performance of 0.31 dB (SNR) and 0.012 (ODG) for the fix bit-rate of 0.6 bits/sample. A larger improvement over the classic approach is observed on [5] with an average increase in performance of 0.51 dB (SNR) and 0.013 (ODG) for the same fixed bit-rate. As can be seen from Figure 1, 0.6 bits/sample is a good bit-rate for the proposed scheme with [5], but slightly smaller bit-rates produce the optimal results for the proposed scheme with [3] (because the performance graphs are offset by the weaker prediction of [3]).

V. Conclusions

A capacity control algorithm for prediction error expansion based audio RDH was introduced. The proposed approach is a refined version of the algorithms from [8] and [9], algorithms developed for image RDH. Compared with the classic capacity control used in audio RDH, the proposed approach offers superior distortion/bit-rate results. The structure of the algorithm is general and can be easily adapted into any PEE based audio RDH framework.

References

[1] D. M. Thodi, J. J. Rodriguez, "Expansion Embedding Techniques for Reversible Watermarking", *IEEE Trans. on Image Processing*, Vol. 16, No. 3, pp. 721–730, 2007.

[2] V. Sachnev, H. J. Kim, J. Nam, S. Suresh, Y. Q. Shi, "Reversible Watermarking Algorithm Using Sorting and Prediction", *IEEE Trans. on Circuits and Systems for Video Technology*, Vol. 19, No. 7, pp. 989–999, 2009.

[3] Y. Huo, S. Xiang, S. Liu ,X. Luo, Z. Bai. Reversible audio watermarking algorithm using non-causal prediction, *Wuhan University Journal of Natural Sciences*, Vol. 18, No. 5, pp. 455-460, 2013

[4] F. Wang, Z. Xie, Z. Chen. "High Capacity Reversible Watermarking for Audio by Histogram Shifting and Predicted Error Expansion", *The Scientific World Journal*, 2014.

[5] S. Xiang, Z. Li. "Reversible audio data hiding algorithm using noncausal prediction of alterable orders", *EURASIP Journal on Audio, Speech, and Music Processing*, 2017.

[6] R. Geiger, Y. Yokotani, G. Schuller. " Audio data hiding with high data rates based on intMDCT", *In Proc. IEEE Int. Conf. Acoust. and Speech, Signal Proc.*, Toulouse, France, 2006

[7] X. Huang , N. Ono, I. Echizen, A. Nishimura. "Reversible Audio Information Hiding Based on Integer DCT Coefficients with Adaptive Hiding" *Lecture Notes in Computer Science*, Vol 8389. Springer, Berlin, Heidelberg, 2013.

[8] I. Caciula, D. Coltuc. Capacity control of reversible watermarking by two-thresholds embedding, *IEEE International Workshop on Information Forensics and Security (WIFS2012), 2012.*

[9] I. Caciula, D. Coltuc. Capacity control of reversible watermarking by two-thresholds embedding: Further results, *International Symposium on Signals, Circuits and Systems (ISSCS2013)*, 2013.

[10] EBU Committee: sound quality assessment material recordings for subjective tests. Available: https://tech.ebu.ch/publications/sqamcd

[11] M. Holters, U. Zlzer. Gstpeaq an open source implementation of the peaq algorithm, *Proc. of the 18th Int. Conference on Digital Audio Effects (DAFx-15)*, 2015

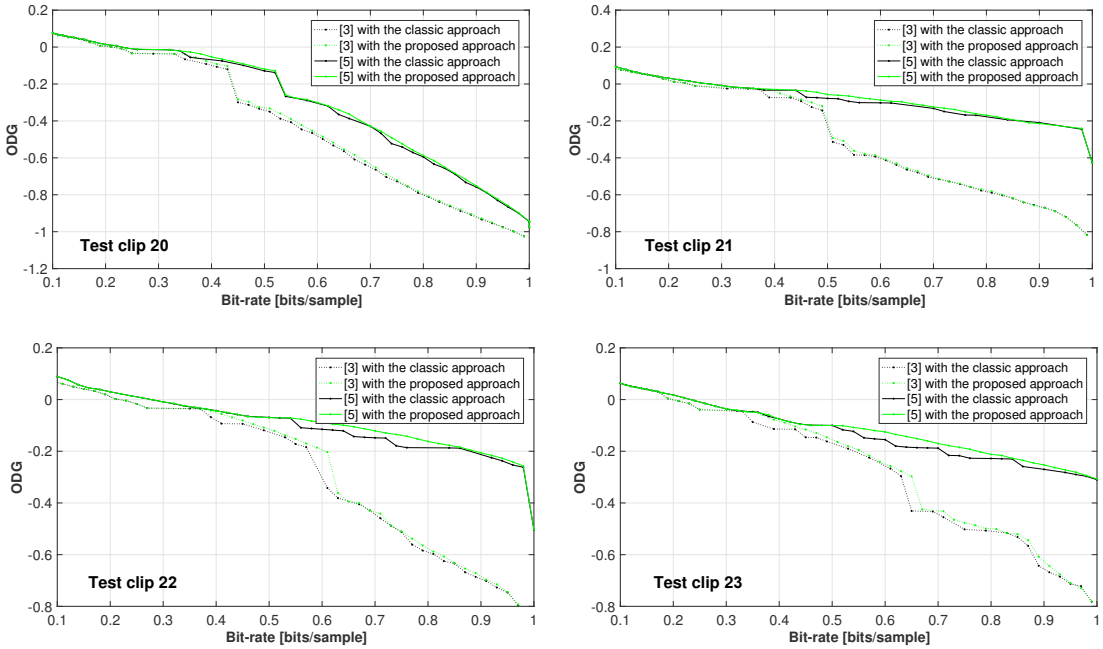

Fig. 2. ODG/Bit-rate results using the classic capacity control algorithm and the proposed approach

TABLE I
SNR RESULTS FOR A BIT-RATE OF 0.6 BITS/SAMPLE USING THE CLASSIC CAPACITY CONTROL ALGORITHM AND THE PROPOSED APPROACH [dB].

Test clip	Simple average [3]		Linear prediction [5]		Test clip	Simple average [3]		Linear prediction [5]	
	Classic	Proposed	Classic	Proposed		Classic	Proposed	Classic	Proposed
1	51.17	51.25	86.92	86.92	36	52.59	54.39	53.64	55.51
2	11.97	12.03	18.05	18.25	37	58.03	58.20	64.85	66.22
3	82.22	82.21	83.33	83.34	38	55.31	55.31	55.57	55.57
4	80.22	80.22	83.25	83.25	39	59.32	60.51	64.27	64.76
5	45.33	45.37	82.86	82.87	40	41.38	41.58	40.86	40.95
6	78.27	78.26	80.68	80.69	41	60.18	61.22	64.52	65.89
7	74.32	75.10	80.74	80.75	42	38.67	38.79	45.91	46.19
8	40.35	40.46	47.59	48.05	43	36.69	36.76	47.61	47.73
9	44.30	44.38	52.66	52.69	44	48.24	48.44	54.78	55.31
10	46.54	46.85	52.33	52.78	45	43.73	43.88	50.64	51.38
11	59.36	59.55	60.19	60.53	46	41.69	41.94	53.22	54.13
12	39.20	39.30	50.28	50.69	47	47.09	47.37	57.25	57.76
13	50.59	50.91	58.78	59.46	48	44.75	44.90	53.41	53.84
14	41.96	42.01	55.74	56.03	49	48.85	49.24	48.47	48.56
15	45.87	46.01	52.91	53.38	50	50	50.37	52.82	53.22
16	50.13	50.27	62.59	63.01	51	45.31	45.47	44.60	44.89
17	47.56	47.71	55.40	55.75	52	50.52	50.91	50.98	51.46
18	56.61	57.12	61.16	62.46	53	48.56	48.91	48.73	49.07
19	56.29	56.63	58.80	59.56	54	47.80	48.20	50.99	51.12
20	51.95	52.52	55.69	56.46	55	45.90	46.02	60.11	60.53
21	50.90	51.12	62.81	63.50	56	33.15	33.17	51.88	52.08
22	55.45	55.99	65.42	66.56	57	25.47	25.53	38.70	38.99
23	59.28	59.74	69.45	70.49	58	49.65	49.98	52.18	53.43
24	69.07	69.76	74.17	74.17	59	40.61	40.65	43.54	43.66
25	63.27	64.72	65.01	65.55	60	59.11	59.31	60.57	61.47
26	58.22	58.31	61.01	61.89	61	42.27	42.47	52.99	53.46
27	34.97	35.05	38.74	39.12	62	52.62	53.13	55.02	55.82
28	60.48	61.95	60.48	61.96	63	44.74	44.76	50.95	51.55
29	66.31	66.31	66.31	66.30	64	36.17	36.23	46	46.13
30	60.48	60.47	60.76	60.76	65	51.29	51.75	59.27	60.44
31	56.45	57.01	56.36	56.76	66	39.91	39.98	51.50	51.70
32	44.12	44.59	47.27	48.20	67	49.15	49.37	53.51	54.54
33	59.11	59.99	58.02	58.66	68	54.24	54.57	54.25	54.58
34	54.78	54.84	59.95	61.37	69	40.43	40.71	44.87	45.34
35	41.97	42.07	55.84	56.18	70	42.90	43.08	44.18	44.30
					Average	**50.22**	**50.53**	**56.63**	**57.14**

TABLE II

ODG RESULTS FOR A BIT-RATE OF 0.6 BITS/SAMPLE USING THE CLASSIC CAPACITY CONTROL ALGORITHM AND THE PROPOSED APPROACH.

Test clip	**Simple average [3]**		**Linear prediction [5]**		**Test clip**	**Simple average [3]**		**Linear prediction [5]**	
	Classic	**Proposed**	**Classic**	**Proposed**		**Classic**	**Proposed**	**Classic**	**Proposed**
1	-1.521	-1.527	-0.057	-0.057	36	-0.110	-0.101	-0.118	-0.099
2	-0.876	-0.871	-0.278	-0.269	37	-0.280	-0.270	-0.117	-0.093
3	-0.135	-0.139	-0.126	-0.126	38	-0.116	-0.116	-0.115	-0.114
4	-0.103	-0.104	-0.104	-0.106	39	-0.207	-0.190	-0.113	-0.108
5	-3.264	-3.263	-0.091	-0.093	40	-0.436	-0.424	-0.455	-0.450
6	-0.143	-0.144	-0.137	-0.137	41	-0.377	-0.217	-0.148	-0.132
7	-0.160	-0.137	-0.097	-0.097	42	-0.641	-0.633	-0.360	-0.353
8	-0.716	-0.710	-0.367	-0.361	43	-0.851	-0.849	-0.404	-0.398
9	-0.660	-0.653	-0.320	-0.317	44	-0.605	-0.591	-0.292	-0.273
10	-0.563	-0.541	-0.280	-0.272	45	-0.536	-0.527	-0.274	-0.252
11	-0.397	-0.382	-0.299	-0.286	46	-0.761	-0.745	-0.283	-0.218
12	-0.742	-0.728	-0.420	-0.395	47	-0.470	-0.460	-0.140	-0.108
13	-0.537	-0.518	-0.132	-0.121	48	-0.665	-0.656	-0.281	-0.269
14	-0.868	-0.867	-0.444	-0.427	49	-0.462	-0.445	-0.351	-0.344
15	-0.705	-0.695	-0.528	-0.503	50	-0.409	-0.396	-0.194	-0.185
16	-0.608	-0.602	-0.256	-0.251	51	-0.546	-0.541	-0.398	-0.379
17	-0.672	-0.666	-0.425	-0.418	52	-0.337	-0.329	-0.238	-0.223
18	-0.442	-0.415	-0.291	-0.277	53	-0.387	-0.378	-0.255	-0.241
19	-0.406	-0.393	-0.161	-0.147	54	-0.379	-0.366	-0.230	-0.225
20	-0.499	-0.472	-0.320	-0.305	55	-0.609	-0.604	-0.133	-0.124
21	-0.403	-0.403	-0.103	-0.086	56	-1.092	-1.091	-0.099	-0.098
22	-0.345	-0.194	-0.115	-0.090	57	-0.982	-0.979	-0.109	-0.101
23	-0.266	-0.252	-0.154	-0.125	58	-0.402	-0.384	-0.080	-0.068
24	-0.174	-0.160	-0.113	-0.114	59	-0.577	-0.570	-0.317	-0.313
25	-0.191	-0.166	-0.149	-0.133	60	-0.157	-0.145	-0.119	-0.103
26	-0.112	-0.110	-0.100	-0.083	61	-0.703	-0.693	-0.239	-0.223
27	-0.521	-0.520	-0.373	-0.359	62	-0.383	-0.370	-0.141	-0.133
28	-0.137	-0.132	-0.139	-0.120	63	-0.629	-0.628	-0.285	-0.271
29	-0.109	-0.109	-0.095	-0.096	64	-0.893	-0.893	-0.356	-0.352
30	-0.105	-0.105	-0.105	-0.106	65	-0.396	-0.390	-0.224	-0.216
31	-0.107	-0.107	-0.107	-0.102	66	-0.565	-0.561	-0.204	-0.204
32	-0.420	-0.405	-0.222	-0.196	67	-0.431	-0.424	-0.320	-0.168
33	-0.155	-0.157	-0.146	-0.154	68	-0.359	-0.361	-0.204	-0.198
34	-0.210	-0.204	-0.133	-0.109	69	-0.672	-0.660	-0.264	-0.253
35	-1.046	-1.044	-0.214	-0.204	70	-0.606	-0.600	-0.306	-0.302
					Average	**-0.5193**	**-0.5069**	**-0.2224**	**-0.209**