Design Principle and Static Coding Efficiency of STT-Coder: A Table-Driven Arithmetic Coder

Ikuro Ueno and Fumitaka Ono
Graduate School, Tokyo Polytechnic University, Kanagawa, Japan
Email: Ueno.Ikuro@df.MitsubishiElectric.co.jp
ono@image.t-kougei.ac.jp

***Abstract*—We have proposed the concept of simple and fast binary arithmetic coder, STT-coder, in which arithmetic operation can be executed by referring a state transition table just like the case of Huffman coding. In our previous study, STT-coder with 3-bit interval register was designed and evaluated. Its coding efficiency was found to be satisfactory despite of the introduction of the simplification of the process. Then we have been trying to extend the register to 6-bit length in order to improve the performance of higher MPS probability sources. In this paper, we introduce a coding parameter of STT-coder, which is called as offset, and optimize the value to maximize the average coding efficiency. We further tried to enlarge the expected interval size after the renormalization to improve the coding efficiency of higher MPS probability sources by sacrificing the accuracy of interval division ratio in low MPS probability sources, which turned out to improve the coding efficiency in total.**

***Index Terms*—image compression, arithmetic coding, entropy, coding efficiency**

I. Introduction

Arithmetic coding has flexible adaptability to various information sources, and provides high coding efficiency. Because of its advantage, it is extensively used in many image and video coding standards such as JPEG2000 [1] and MPEG-4 AVC [2]. On the other hand, the disadvantage of arithmetic coding is its complexity of operations. The studies on arithmetic coding have hitherto been mainly tuned for the simplification of probability interval calculation such as in range coder [3], Q-Coder [4] and MQ-Coder [1] and [5]. On the other hand, we have proposed an adaptive binary arithmetic coder, STT-coder, which can be realized by simple and fast arithmetic operations driven by state transition table [6]-[8].

In this paper, we report designing principle of STT-coder and its static coding efficiency, from two studying points. One is the optimization of a parameter of STT-coder, called as offset, which affects both of the state transition table size and the coding efficiency. The other is the trade-off between the accuracy of the interval division probability and the largeness of the valid interval width after the renormalization, in which, we will try to enlarge the interval size after renormalization by sacrificing the accuracy of probability interval of low MPS probability sources to improve the average coding performance.

II. Arithmetic Coder Driven by State Transition Table: STT-Coder

A. Structure of STT-Coder

STT-coder is a binary arithmetic coder dealing with binary symbols converted to MPS (More Probable Symbol) or LPS (Less Probable Symbol), being composed of two main blocks, probability estimation and probability interval division, as shown in Fig. 1. In this paper, we mainly focus on the probability interval division block in 6-bit STT-coder. The probability interval division procedure of STT-coder can be driven by a state transition table, which we call interval division table.

As the process of arithmetic coding is going forward, the valid interval becomes smaller and smaller. The conventional arithmetic coder will apply renormalization procedure whenever the interval becomes less than the half of maximum interval. However, we will consider whether next code bits are fixed or not, and proposed STT-coder will apply the renormalization only when next code bits are fixed. Therefore complicated operations like multiplication times control or carry-over bit control are not necessary. Therefore, the interval division table outputs a codeword (when the next code bits are fixed) or a succeeding probability interval (when the next code bit is not yet fixed) for each binary input symbol, according to the size of the current probability interval as shown in Fig. 1. In STT-coder, if the valid interval becomes smaller and smaller without fixing the next code bit, the valid interval cannot be renormalized despite of its size. In such a case, the coding efficiency will be degraded especially for high MPS probability sources because MPS/LPS subdivision ratio cannot follow its ideal balance. In order to minimize such effect, we will introduce a coding parameter called offset and examine the relation between the coding efficiency and the parameter value.

Manuscript received August 17, 2013; revised November 18, 2013.

doi: 10.12720/ijsps.1.2.177-182

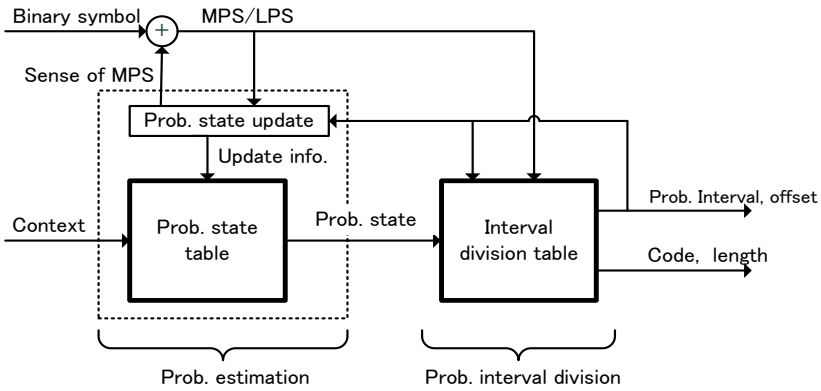

Figure 1. STT-coder block diagram.

B. Example of STT-Coder with 3-Bit Register

At first we will show an example of interval division of STT-coder with a quite short interval register size of 3-bit. To design the interval division table, we should consider three parameters, probability interval A_{ML}, MPS width A_M (or LPS width A_L ($=A_{ML}-A_M$)) and offset D, which will vary along the probability interval division for arithmetic coding as depicted in Fig. 2. In Fig. 3, all of the probability interval division patterns are shown. This was designed skillfully so that the interval division table will be kept quite small, yet it maintains the coding efficiency as high as possible. In Fig. 3, M and L denote MPS and LPS encoded with the corresponding interval width, respectively. Characters after a symbol are the codeword (not in parenthesis) and next probability interval index of the state transition (in parenthesis). After encoding a symbol with no probability interval index, the state returns to the initial condition, A_{ML}=8 and D=0.

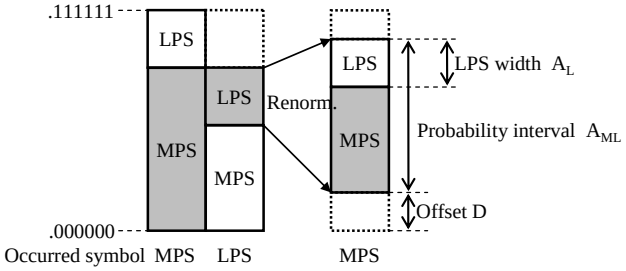

Figure 2. Probability interval division.

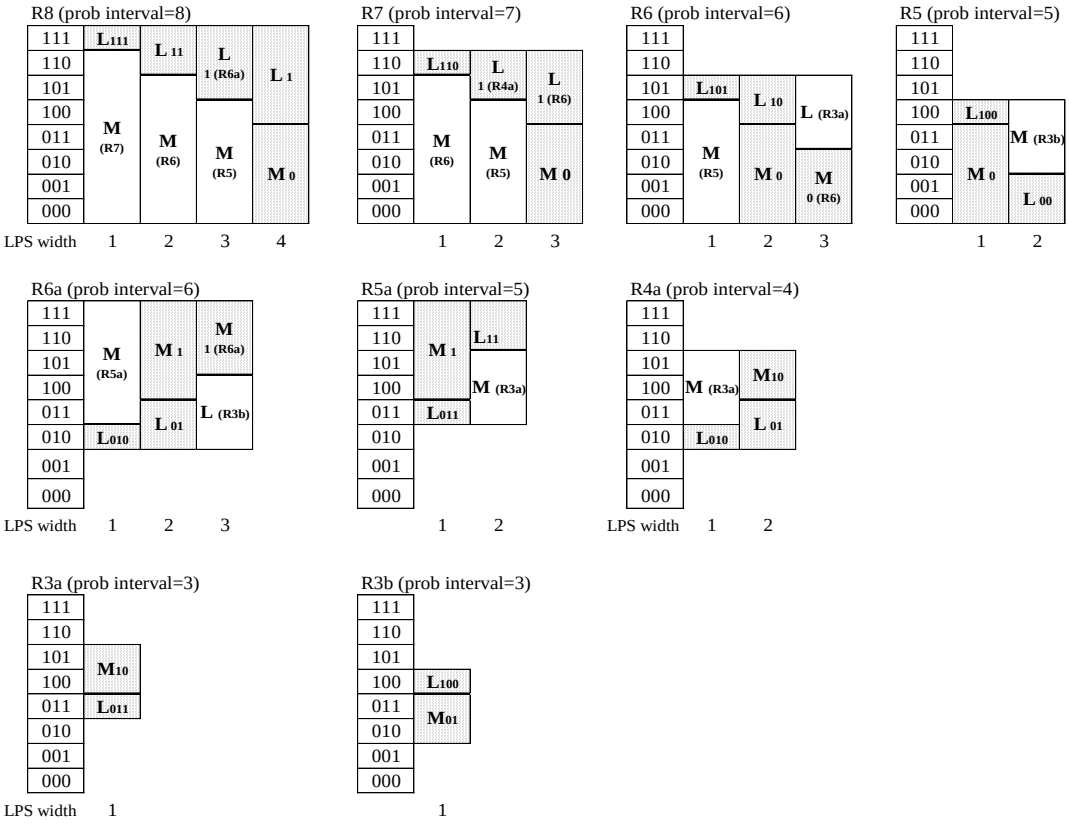

Figure 3. Interval division patterns of 3-bit STT-coder.

In Table I, LPS width A_L (= A_{ML}-A_M) and corresponding probability interval index for each combination of A_M and A_{ML} values are summarized. Since the maximum width is 8 (=2^3), the ranges of the three parameters are limited as follows:

$$8/2 < A_{ML}+D <= 8 \quad (1)$$

$$A_{ML}/2 <= A_M < A_{ML} \quad (2)$$

The LPS interval is mostly placed at the upper side of the interval, but in some cases it is placed at the lower side to expect larger interval size after the renormalization to provide better coding performance. In Table I, gray cells indicate that the LPS interval is placed at the lower side, and cells with a diagonal line are not valid because those are beyond the limit of equation (1) and (2).

TABLE I. LPS WIDTH A_L FOR EVERY COMBINATION OF A_M AND A_{ML} (3-BIT STT-CODER).

(a) offset D=0

A_M \ A_{ML}	5	6	7	8
7				1 (R8)
6			1 (R7)	2 (R8)
5		1 (R6)	2 (R7)	3 (R8)
4	1 (R5)	2 (R6)	3 (R7)	4 (R8)
3	**2 (R5)**	3 (R6)		

(b) offset D=2

A_M \ A_{ML}	3	4	5	6
5				**1 (R6a)**
4			-	**2 (R6a)**
3		**1 (R4a)**	-	**3 (R6a)**
2	1 (R3b)	**2 (R4a)**		

(c) offset D=3

A_M \ A_{ML}	2	3	4	5
4				**1 (R5a)**
3			-	2 (R5a)
2		**1 (R3a)**	-	

C. Probability Estimations

The probability estimation system, if any, outputs another parameter, probability state. We map a range of MPS probability to corresponding probability state, and let P_{Mi} denote the best MPS probability to perform the highest efficiency for each probability state S_i. Note that we are dealing with static coding efficiency, assuming that the ideal probability state is always selected by the probability estimation system, in this paper. In order to execute an adaptive control of probability estimation to select appropriate coding parameters for unknown information sources, we will apply probability estimation system as shown in the left side of Fig. 1.

III. DESIGN PRINCIPLE OF STT-CODER

A. Design of STT-Coder

In the previous section, we showed the outline of STT-coder taking the case of quite short register size as a fundamental study. It was found that coding efficiency is satisfactory despite of the introduction of the simplification of the process. However, since the performance of the higher MPS probability sources is not good enough because of its register size, we extend the probability interval register size to 6 bits to prepare for higher MPS probability sources based on the study of the 3-bit interval register.

In this section, we will show the design principle of the probability interval table for the 6-bit STT-coder. To design the probability interval table, we need to decide an LPS width for every combination of offset D, probability interval A_{ML} and probability state S_i. For the 6-bit system, as the maximum value of A_{ML} is 64 (=2^6), these parameters range as follow:

$$64/2 < A_{ML}+D <= 64 \quad (3)$$

$$A_{ML}/2 <= A_M < A_{ML} \quad (4)$$

As was mentioned in the previous section, the probability interval A_{ML} takes any value between the range defined above, because LPS width A_L takes the minimum value 1 for every A_{ML} to express high MPS probabilities.

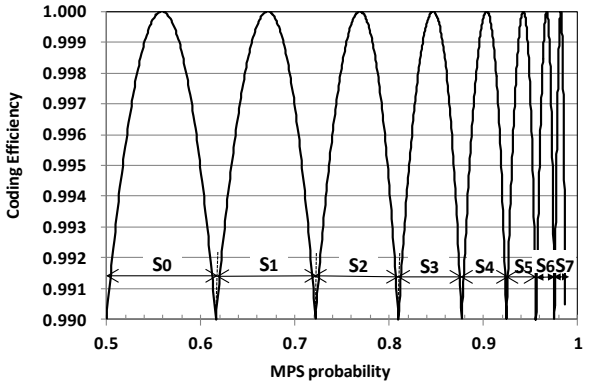

Figure 4. Ideal coding efficiency of the coding method.

The range of the sources to be covered by a probability state is determined so that theoretical coding efficiency within the range would be larger than a predetermined minimum coding efficiency. Fig. 4 shows the coding efficiency for each probability state with the predetermined minimum coding efficiency of 0.99. After the set of probability states or set of P_{Mi} is determined, we chose LPS width A_L in given A_{ML}, for all probability states. The number of probability states is eight for 6-bit register case, and the number of offsets will be discussed later in following Section III.B.

It is ideal to set the MPS width A_M for the valid interval A_{ML} to be the closest value of [A_{ML} multiplied by P_{Mi}] as shown in Table II. Also it is necessary to satisfy that the next offset, after the happening of either symbol, shall be any of the allowed values. If the LPS interval can

be assigned to either of the upper or lower area in the probability interval, the assignment maximizing the minimum interval width value for the next symbols is selected. Table III shows a part of the Interval division table of 6-bit STT-coder. The total number of input bits is 12, and total number of output bits is 17.

TABLE II. LPS WIDTH OF EIGHT PROBABILITY STATES FOR EACH PROBABILITY INTERVAL (6-BIT RESISTER, OFFSET D=0) (UNDERLINED LPS WIDTH DENOTES THAT LPS INTERVAL IS ASSIGNED UNDER THE MPS INTERVAL.)

		Probability interval A_{ML}										Best MPS Prob. P_{Mi}
		64	63	62	61	60	…	…	35	34	33	
Probability state	S_0	28	28	28	28	28	…	…	16	16	16	0.559
	S_1	20	19	22	21	20	…	…	11	10	9	0.671
	S_2	16	15	14	15	14	…	…	7	6	9	0.769
	S_3	10	9	10	9	8	…	…	7	6	5	0.847
	S_4	6	7	6	6	6	…	…	3	2	5	0.904
	S_5	4	4	4	3	3	…	…	2	2	1	0.942
	S_6	2	2	2	2	2	…	…	1	1	1	0.967
	S_7	1	1	1	1	1	…	…	1	1	1	0.982

TABLE III. PROBABILITY INTERVAL DIVISION TABLE FOR 6-BIT STT-CODER.

Input				Output			
3bit	5bit	2bit	1bit	6bit	3bit	5bit	2bit
S_i	A_{ML}	D	MPS /LPS	code	Code length	A_{ML}	D
S_0	33	0	M	-	0	17	16
			L	00	2	64	0
	34		M	-	0	18	16
			L	00	2	64	0
	35		M	-	0	19	16
			L	00	2	64	0
	…		M	-	0	20	16
			…	…	…	…	…
…	…		…	…	…	…	…
	…		…	…	…	…	…
	…		…	…	…	…	…
			…	…	…	…	…
S_7	…		…	…	…	…	…
			…	…	…	…	…
	63		M	-	0	62	0
			L	111110	6	64	0
	64		M	-	0	63	0
			L	111111	6	64	0
S_0	17	16	M	-	0	9	24
			L	010	3	64	0
	18		M	-	0	10	24
			L	010	3	64	0
	…		…	…	…	…	…
			…	…	…	…	…
…	…		…	…	…	…	…
			…	…	…	…	…
…	…	28	…	…	…	…	…
			…	…	…	…	…
S_7	…		…	…	…	…	…
			…	…	…	…	…
	35		M	-	0	34	28
			L	111110	6	64	0
	36		M	-	0	35	28
			L	111111	6	64	0

B. *Restriction of Offsets*

Although the restriction of the allowed offset values contribute to simplify the coder, it restricts the liberty to choose best LPS width for a given probability state. Therefore we examined how the number of allowed offsets, which we denote by N, affects the coding efficiency.

First of all, we examined the coding efficiency for the case where only one offset (D=0) is allowed (N=1). Assuming the coding of a multi-context source, that is, the occurrence probability of a symbol depends on its context, we calculated the coding efficiency for each context of STT-coder. Note that we assume, in this paper, that an appropriate probability state is always selected according to the context when encoding a symbol. Also we assumed that MPS probability is uniformly distributed between 0.5 and 1 in supposed multi-context source.

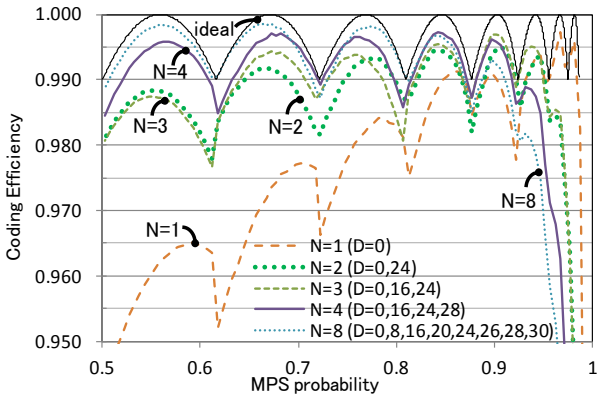

Figure 5. Coding efficiency for various offset number.

In Fig. 5, the coding efficiency of STT-coder having only one offset is denoted as N=1. It can be seen that coding efficiency for lower MPS probability is not so good, since allowed candidates of LPS width are rather limited. However, for higher MPS probability sources, the coding efficiency is quite high. It is because that the average probability interval width is large, and the LPS

width A_L=1 to assure high performance for high MPS probability sources can be used for all probability interval width A_{ML} in the case of offset D=0.

For the next step, we added one more offset D=24, which caused the highest improvement of coding efficiency among other possible offsets. Under these two offsets of D=0 and D=24, the candidates of LPS widths are increased and the coding efficiency for lower MPS probability was significantly improved. For higher MPS probability sources, the coding efficiency was decreased, since the expected probability interval width gets smaller on average especially for D=24.

With the same manner, the offsets of D=16 and D=28 were found to perform best for the third and fourth offset values respectively. Further addition of offsets is proved to cause a little improvement for lower MPS probability sources (see the coding efficiency for N=8 in Fig. 5), but some degradation for high MPS probability sources, and does not contribute to improve the total coding efficiency. Therefore we concluded that the optimal offset number N equals to four and the combination of four offset values are (D=0, 16, 24, 28) for the 6-bit STT-coder.

Table II is the partial data of LPS width of eight probability states for each probability interval in D=0. For different offset values, the possible number of the valid intervals are usually 32 same as the case of offset D=0, since the upper address of the valid intervals will take between 32 and 63. The different LPS width data table is necessary for each offset value.

C. *Trade-off of the Accurate Probability Interval Division and Larger Interval Providing Division*

In the experiments of the previous section, the addition of D=28 as the fourth offset is found to improve the coding efficiency for low MPS probability, but to degrade the coding efficiency for high MPS probability at the same time, as a larger offset value such as 28 generally work to reduce the average probability interval width. Therefore we examined the trade-off between the selection of the more accurate probability interval division with large offset and the division to provide larger interval size by causing the renormalization. That means, we tried to enlarge the interval size by causing the renormalization in sacrificing the coding efficiency of the low MPS probability source.

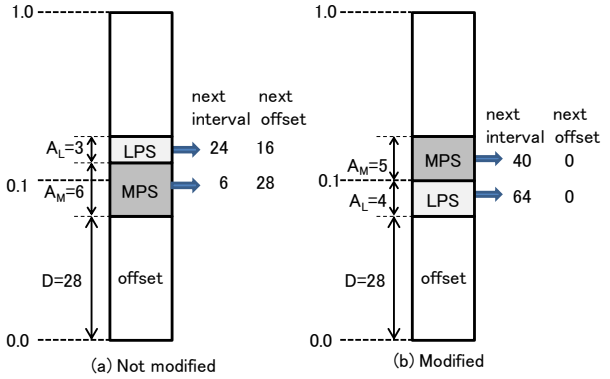

Figure 6. Modification of probability interval division.

In Fig. 6, an example of the modification is illustrated. In the case that probability interval A_{ML}=9, LPS width A_L=3 and offset D=28, the next values after renormalization will be A_{ML}=24 and D=16 if LPS happens, or A_{ML}=6 and D=28 if MPS happens, as shown in Fig. 6 (a). If we modify the interval division as in Fig. 6 (b), the next values after renormalization will be A_{ML}=64 and D=0 if LPS happens, or A_{ML}=40 and D=0 if MPS happens, which provides larger probability interval size and improves the average coding efficiency.

We applied this idea to all the cases of A_L=3 and 5, that is, if LPS width A_L=3 or 5 is selected for D=28 by the procedure described in Section III.B., the LPS width would be modified to A_L=4 and the LPS is assigned in the lower side of the interval. Also the case of A_L=8 for offset D=24 has the same effect. So we prohibited the probability interval division of A_L=7 or 9 for D=24, and changed it to A_L=8 and set the LPS to the lower side of the interval. Note that this modification of the interval division can be realized by changing the interval division table before encoding, without affecting the encoding procedure.

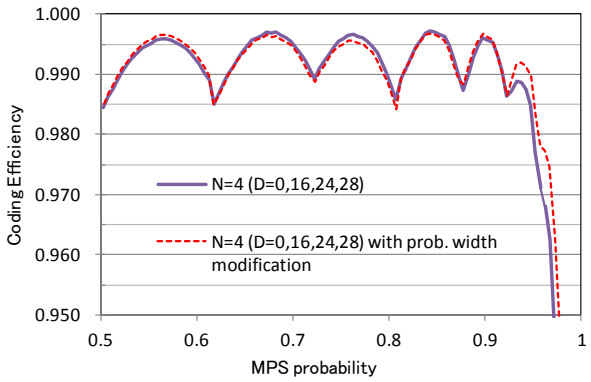

Figure 7. Coding efficiency.

In Fig. 7, we showed the coding performance of STT-coder using the above interval modification. Though slight degradation of coding efficiency can be found around the probability corresponding to the probability state S_2, at which the probability interval modification has a bad effect, its coding efficiency is improved especially for high MPS probability, and this modification is found to achieve better performance in total.

IV. Conclusion

In designing our proposed arithmetic coder, driven by a state transition table, we examined how its parameter, called offset, affects the coding efficiency.

It was found that the coding efficiency for low MPS probability gets better as the number of offsets N increases, though the efficiency for higher MPS probability gets worse, since the expected interval size becomes smaller. From the experiments, we concluded that the number of offset N equals to four, which are composed of the offset values of D=0, 16, 24, and 28, is the most efficient case for 6-bit STT-coder.

We also examined the trade-off between the accurate probability interval division and the less accurate but providing larger interval division by causing the renormalization. The trial to modify the division ratio of the symbols, especially for large offset cases, to make the interval size larger by causing renormalization was found to improve the coding efficiency in total, though the coding efficiency of the low MPS probability sources will be a little sacrificed.

We examined static coding efficiency in this paper, that is, ideal probability estimation of information source is applied. For the next step, we will introduce dynamic probability estimation method for more practical coder design.

REFERENCES

[1] JPEG 2000 Image Coding System, ISO/IEC 15444-1, 2000.
[2] Advanced Video Coding, ISO/IEC 14496-10, 2010.
[3] G. Nigel and N. Martin, "Range encoding: An algorithm for removing redundancy from a digitized message," presented at Video & Data Recording Conference, Southampton, UK, July 24–27 1979.
[4] W. B. Pennebaker, J. L. Mitchell, G. G. Langdon, Jr., and R. B. Arps, "An overview of the basic principles of Q-coder," *IBM J. Res. Develop.*, vol. 32, no. 6, pp. 717, Nov. 1988.
[5] ISO/IEC 14492 "Lossy/lossless coding of bi-level images," 2001.
[6] I. Ueno and F. Ono, "Dynamic coding efficiency of fast arithmetic coder using state transition table," in *Proc. Media Computing Conference*, R6-4, June 2012 (in Japanese).
[7] I. Ueno and F. Ono, "An adaptive binary arithmetic coder using a state transition table," presented at IEVC2012, Kuching, Malaysia, Nov. 21-24, 2012.
[8] I. Ueno and F. Ono, "Fast arithmetic coder driven by state transition ROM table," presented at IWAIT2013, Nagoya, Japan, Jan. 7-9 2013.

Ikuro Ueno Ikuro Ueno has received BE and ME from Tokyo University of Science, Chiba Japan, in 1989 and 1991 respectively. He has been working with Mitsubishi Electric Corp. from 1991 to now. He had been a visiting scholar at Rensselaer Polytechnic Institute, Troy U.S.A, from 2001 to 2002. He has been a Ph.D. candidate in Tokyo Polytechnic University, Kanagawa Japan, since 2011. He is a member of IEICE, ITE and IIEEJ. His research interests include image coding, image processing and entropy coding.

Fumitaka Ono Fumitaka Ono has received BE, ME and Ph. D, all from University of Tokyo, Tokyo, Japan, in 1971, 1973, and 1993 respectively. He has been working with Mitsubishi Electric Corp. from 1973 to 2000, and with Tokyo Polytechnic University from 2000 to now. Dr. Ono is Fellow of IEEE, Fellow of IEICE and Fellow of IIEEJ. He has received many awards including the Award from Ministry of Education and Science, and the Award from Ministry of Industry and Trade, for distinguished achievements on image coding and the contribution for standardization activities, respectively. His research interests include image coding, image processing and entropy coding.