Regular Paper

A Low-Energy Application Specific Instruction-Set Processor towards a Low-Computational Lossless Compression Method for Stimuli Position Data of Artificial Vision Systems

Tomoki Sugiura[1,a)] Masaharu Imai[1,2,b)] Jaehoon Yu[1,c)] Yoshinori Takeuchi[1,2,d)]

Received: May 26, 2016, Accepted: November 1, 2016

Abstract: This paper proposes a novel data compression method for artificial vision systems and its low-energy implementation in order to reduce energy consumption in a wireless communication subsystem. The artificial vision systems are one of the methods for realizing visual prosthesis by controlling stimulus to visual nerves, and they consist of an inner stimulating unit and an outer image processing unit. The outer unit transmits information regarding stimulation to the inner unit via wireless communication, which occupies a large portion of the whole energy consumption. Reducing traffic in wireless communication is important to prevent damage caused by extra heat dissipation of the inner unit, which leads to excess energy consumption. The proposed compression method marks a higher compression ratio than the conventional compression methods by taking advantage of the analyses of stimuli position data, which is dominant in traffic. The proposed method is implemented as an application-domain specific instruction-set processor to achieve both configurability of stimulation control and compression efficiency. The evaluation results show that the proposed implementation reduces energy consumption by about 87% and 62% in the compression and decompression process, respectively. These results indicate that the proposed method can expect to reduce energy consumption in a wireless communication receiver dramatically.

Keywords: application-domain specific instruction-set processor, lossless data compression, exponential Golomb coding, entropy coding, cortical visual prosthesis, artificial vision system, low-energy consumption, implantable embedded system

1. Introduction

About 285 million people suffer from visual disorder in the world and the number of visually handicapped people increases because of glaucoma, age-related macular degeneration, or diabetic retinopathy [1]. For one of method for visual prosthesis, artificial vision systems have been researched. The artificial vision systems provide their user with pseudo vision, by means of a phenomenon that stimulations to the visual nerves evoke bright spots in vision (see Ref. [2]). As implantable embedded systems, medical demands for artificial vision systems are safety for users, long-time use, and minimizing the gap between vision and other sensory nerves.

Artificial vision systems are roughly classified into two types [3]: Non-retinal approaches and retinal approaches (e.g., Refs. [4], [5], [6], [7], [8]). At non-retinal approaches, cortical prosthesis and optic nerve prosthesis [9] have been researched. In cortical prosthesis, researches by Dobelle (e.g., Refs. [10], [11]), who was one of the pioneers, played a part in the dawn of portable artificial vision systems. Although there are difficulties in the mapping of stimulation to the visual cortex and occurring pseudo vision [12], the advantage of cortical prosthesis is that the visual cortex has a larger area to stimulate than the other visual nerves, which allows the use of larger size electrodes.

Artificial vision systems consist of four components: A camera device, an image-processing circuit, a stimulation control circuit, and stimuli electrodes, and they are often divided into two parts: The camera device and image-processing circuit are outside of the body, and the others are inside of the body. For the safety of artificial vision systems, in particular, it is important to decrease heat dissipation caused by the energy consumption of the inner components because excessive heat dissipation may damage organs touching the components. In general, systems include a wireless communication component for the prevention of infection, and, according to Ref. [13], a wireless communication subsystem occupies over 50% of the whole energy consumption. To reduce the energy consumption in wireless communication, one solution is to lower the data transmission rate of the wireless communication. However, this solution may increase delay in pseudo vision, which cannot satisfy the constraint of real-time

[1] Department of Information System Engineering, Graduate School of Information Science and Technology, Osaka University, Suita, 565–0871 Osaka, Japan
[2] The Global Center for Medical Engineering and Informatics, Osaka University, Suita, 565–0871 Osaka, Japan
[a)] s-tomoki@ist.osaka-u.ac.jp
[b)] imai@ist.osaka-u.ac.jp
[c)] yu.jaehoon@ist.osaka-u.ac.jp
[d)] takeuchi@ist.osaka-u.ac.jp

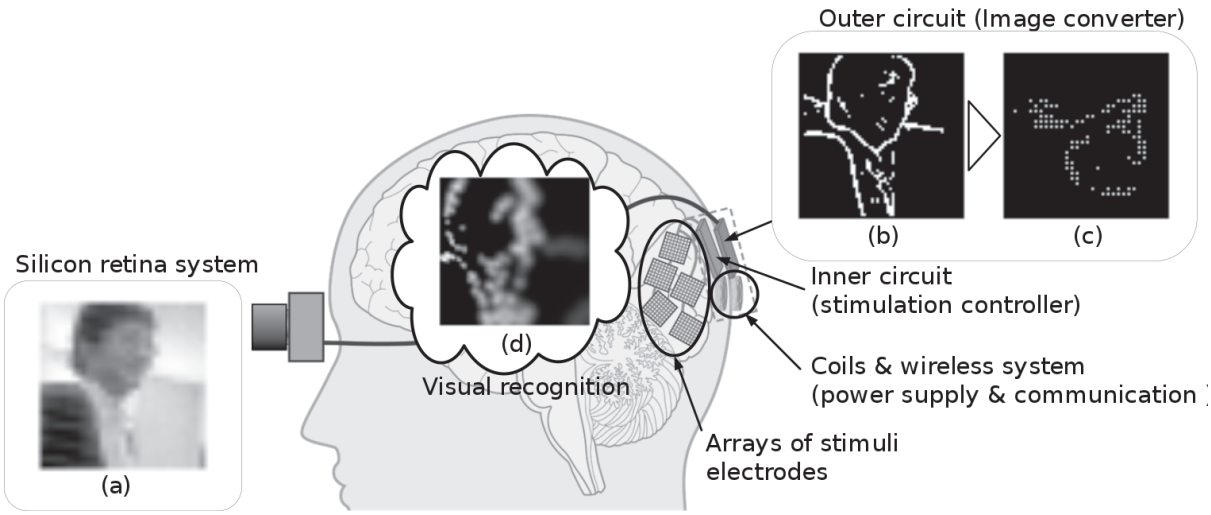


Fig. 1 Organization of the artificial vision system.

operation. Due to improvements of miniaturization of electrodes (e.g., Refs. [14], [15], [16], [17], [18]), the number of electrodes has increased: 100 electrodes in 1991 [14] to 320 electrodes in 2015 [16]. Whereas this improvement can raise the resolution of stimulation and pseudo vision, they also increase the amount of stimulation data to transfer.

Another solution is to reduce the amount of data transmission by applying a data compression method. In the research field of bio-information such as electrocardiogram and neuro-recording, data compression has been researched to cope with the increase of recoded data resolution and the number of electrodes [19], [20], [21]. For the compression of stimulation data for artificial vision systems, some different points are focused: lossless and real-time operation. To examine the relationship between the configuration of stimulation and generated pseudo vision, the data compression method should be lossless. To minimize the gap of the vision, the data compression method has to be not only effective but also of low computational complexity.

This paper proposes a lossless data compression method for stimuli position data used in artificial vision systems and its low energy implementation as application-domain specific instruction-set processor (ASIP). For the configurability of the systems, the implementation includes a processor as an inner stimulation controller. Paper [22] has proposed an ASIP implementation for stimuli position data compression previously. The novelty of this paper is that the ASIP this paper proposes achieves less energy consumption both in the compression and in the decompression process compared with Ref. [22]. This paper also evaluates the amount of reduction of energy consumption in wireless communication by the proposed implementation.

The remainder of this paper is organized as follows. Section 2 explains our targeted artificial vision systems. Section 3 explains the analysis results of targeted data for the compression, stimuli position data, and proposes a low-computational and high efficient data compression method. Section 4 introduces the implementation of the proposed compression method, and Section 5 shows its evaluation results. Finally, Section 6 concludes this paper.

2. Targeted Artificial Vision Systems

This section explains an overview of our targeted artificial vision system and its engineering requirements for stimulation controller. **Figure 1** illustrates our developing artificial vision system. It is divided into two components: outer unit and inner unit. The outer unit is attached to the top of the skull, and the inner unit is attached on the visual cortex. The outer unit is composed of the silicon retina system with image-processing circuit (see Refs. [23], [24], [25], [26]) and a wireless communication unit to exchange data with the inner unit. The silicon retina system in Ref. [26] can perform high-speed image processing, which mimics the human retina with analog CMOS integrated circuits, and it generates stimulation data, which consists of stimuli strength, stimuli timing, and stimuli position. Figure 1 (a) shows an example of an input image from a silicon retina system, and Fig. 1 (b) depicts intermediate data, which is the halfway of image processing: the edge enhancement and thresholding, and Fig. 1 (c) depicts a part of the stimulation data from the silicon retina system. The inner unit is composed of arrays of stimuli electrodes, a stimulation controller, and a wireless communication unit. The stimulation controller is divided into two parts: an analog circuit block and a digital circuit block. The analog circuit block deals with stimulations by controlling electrodes. The digital circuit block consists of a processor core, an interface to the analog circuit block, an external memory, and peripherals. When the digital circuit block receives the stimulation data via the wireless communication unit, it transfers data to buffer memory in the analog circuit block. Then, the analog block stimulates the visual cortex with the stimuli electrodes based on the data in the buffer memory. Figure 1 (d) illustrates the user's visual recognition by stimulation based on Fig. 1 (c).

Considering the medical demands for the artificial vision systems mentioned in Section 1, engineering requirements for the artificial vision systems are small size, low energy consumption, and real-time operation. To reduce energy consumption in the stimulation controller, it is effective to reduce data transaction in wireless communication. In general, power dissipation and exe-

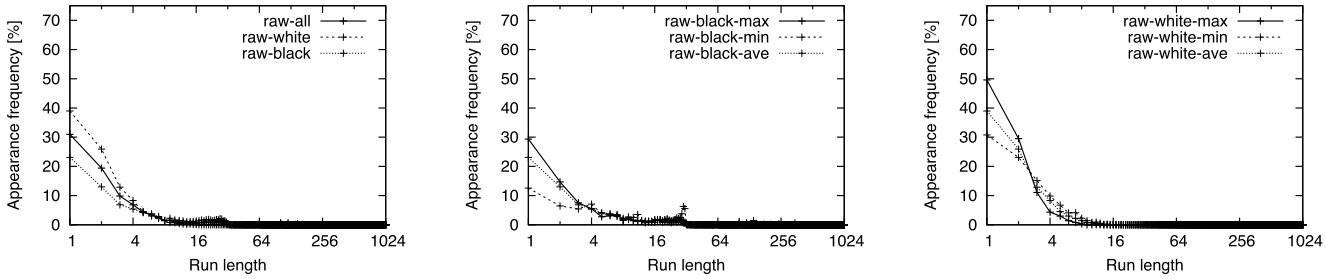


Fig. 2 Analysis results of appearance frequency of the run length of white and black dots in raw stimuli position data and difference data (a) Appearance frequency of the run length in raw stimuli position data, (b) Variance of appearance frequency of the run length of black dots in raw stimuli position data, (c) Variance of appearance frequency of the run length of white dots in raw stimuli position data.

cution time of the processing with a processor is larger than with an application specific integrated circuit (ASIC), so that the extra processing has to keep to a minimum from the aspect of real-time processing. On the other hand, the inner unit should avoid including ASIC for compression in terms of miniaturization. To satisfy these requirements, the compression process performed in the stimulation controller should be effective and low computational.

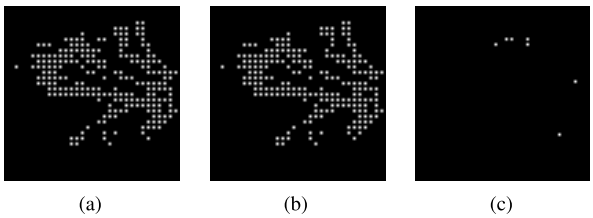


Fig. 3 Examples of stimuli position data and difference data (a) Stimuli position data, (b) Previous frame of Fig. 3 (a), and (c) Difference data between Fig. 3 (a) and Fig. 3 (b).

3. Data Compression Method

This section explains the statistical analysis of the stimuli position data and the proposed lossless data compression method. This paper focuses on the stimuli position data because stimuli position data occupies a large part of the traffic in wireless communication. Figure 1 (c) shows an example of an image representing stimulus position data as a binary bitmap. The stimulus data of the preliminary system is composed of 32×32, 1,024 bit in all. Each pixel is related to the coordinate of electrodes, and the value of each pixel represents whether the electrode at the same coordinate provides a stimulus or not. If the data is '1', represented as a white dot in Fig. 1 (c), the electrode stimulates. If the data is '0', represented as a black dot in Fig. 1 (c), the electrode does not.

3.1 Analysis of Stimuli Position Data

In Fig. 1 (c), black dots occupy much space and most of all these dots are successive in long length seen in the horizontal direction. On the other hand, some white dots are successive in short length and isolated. Therefore, we use the run length encoding to express stimulus data and we will analyze the distribution of run-length data to achieve the encoding method that marks higher compression efficiency. The data set used in this analysis consists of objects in the input image: walking people, running cars, bicycles, and motor bikes on the road. Forty-five samples of scenes are captured in various situations such as on narrow roads, in the cross of arterial roads, and at the station.

Figure 2 shows the appearance frequency of the run length of white dots and black dots in the stimuli position data. The x-axes of Fig. 2 (a), (b), and (c) represent the value of the run length, and the y-axes represent the appearance frequency of the run length. In Fig. 2 (a), 'raw-all' indicates the average of the whole run length from all data sets and 'raw-white' and 'raw-black' indicates the run length of white dots and black dots, respectively. Figure 2 (b) and (c) shows the variance of appearance frequency of the run length of each dot. In these figures, '-max' and '-min' indicate the data set that scores highest and lowest frequencies of run length equaled to one, respectively, and '-ave' indicates the average of appearance frequency from all data sets. Figure 2 shows that the probability distribution of 0's run-length data is different from that of 1's run-length data. In other words, 1's run length is shorter than 0's run length. A run length whose length is one occupies about 30% in 1's run length, while it occupies about 22% in 0's run length. Furthermore, a run length whose length is less than or equal to eight occupy about 90% in 1's run length, but on the contrary, they occupy about 65% in 0's run length.

3.2 Difference Data of Successive Stimuli Position Data

Figure 3 (a) and (b) are continuous frames of stimuli position data. Figure 3 (c) shows the difference between Fig. 3 (a) and (b) by calculating exclusive or. In Fig. 3 (c), white dots represent the change of the stimulus status from the previous frame as '1' in data, and black dots represent the same status in the previous frame as '0' in data. There are more successive black dots and less isolated white dots in Fig. 3 (c) than in Fig. 3 (a). In the rest of this paper, the stimuli position data is called raw data in distinction from difference data.

Figure 4 shows the appearance frequency of the run length of the changed point and unchanged point of the difference data. As in Fig. 2, the x-axes of Fig. 4 (a), (b), and (c) represent the value of run length and the y-axes of these figures represent the appearance frequency of the run length in the sample. In Fig. 4 (a), 'diff-all' indicates the average of the whole run length from all data sets and 'diff-white' and 'diff-black' indicate the run length of

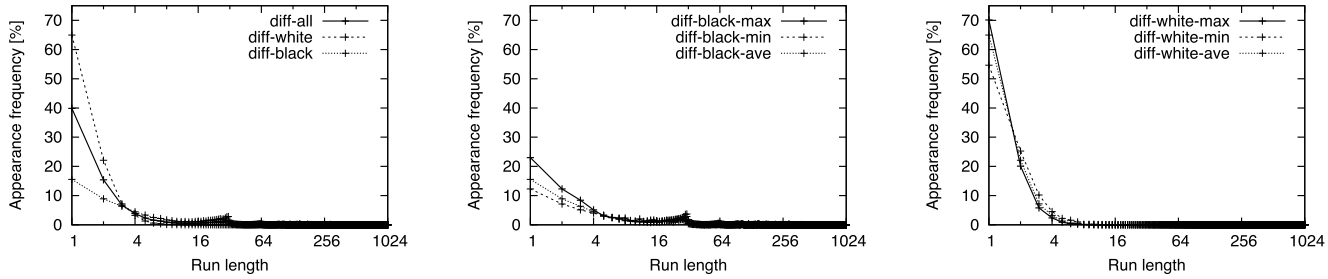


Fig. 4 Analysis results of appearance frequency of the run length of white and black dots in raw stimuli position data and difference data (a) appearance frequency of the run length in difference position data, (b) Variance of appearance frequency of the run length of black dots in difference data, and (c) Variance of appearance frequency of the run length of white dots in difference data.

white dots and black dots, respectively. Figure 4 (b) and (c) show the variance of the appearance frequency of the run length of each dot. In these figures, '-max' and '-min' indicate the data set that scores the highest and lowest frequencies of run length equaled to one, respectively, and '-ave' indicates the average of appearance frequency from difference data generated from all data sets. Figure 4 (a) shows that the frequency of the occurrence of 1's runs is extremely biased to be short in the difference data. For the length of 1's run, the appearance frequency of run whose length is one occupies about 64%, which is nearly twice as much as that in the raw data. For the length of 0's run, the frequency of occurrence of run whose length is less or equal to eight occupies about 40%, which is very different from that in the raw data, about 65%. In the difference data, the frequency of occurrence of 0's run whose length is more than 32 occupies about 20%, while they occupy about 3% in the raw data.

3.3 Proposed Data Compression Method

From the above analysis in Section 3.2, there is a different distribution between the length of 0's run and the length of 1's run in the raw data, and the difference data makes more biased that distribution. Then, a data compression method to take advantage of these characteristics is proposed, which is called α-exponential Golomb coding (α-EGC) in this paper.

The α-EGC compresses the stimuli position data in the following steps:

(1) It generates the difference data from the two successive frames of stimuli position data,
(2) counts the run of successive bits in the difference data with raster scanning, and
(3) encodes the length of runs into a bit stream
  (a) with the offset using the exponential Golomb coding (EGC) [27] if the run consists of unchanged bits, and
  (b) with the offset using the Elias alpha coding [28] if the run consists of changed bits.

The α-EGC encodes the length of runs with different encoding methods. For the length of 1's runs, the proposed method uses the Elias α coding, also called unary coding. **Algorithm 1** shows how the Elias α coding encodes an input integer value. Figure 4 (a) show that 94% of whole 1's run in the difference data is less or equal to four. Shown in Algorithm 1, it is suitable to encode short values because the Elias α coding can encode

```
Input: inVal
Output: codeword
1 codeword := '0'*(inVal - 1) & '1'
```

Algorithm 1: Elias alpha coding

```
Input: inVal, k
Output: codeword
1 tmp := inVal + 2^k;
2 dLen := the number of binary digits without leading zeros of (tmp - 1);
3 codeword := '1'*(dLen - k) & '0' & inVal[dLen .. 0];
```

Algorithm 2: Exponential Golomb coding

the numeric value to the code word, whose length is equal to the value encoded. Therefore, the Elias α coding encodes the length of 1's run in difference data. For the length of 0's runs, the proposed method uses the EGC. **Algorithm 2** shows how the EGC encodes an integer value. 'tmp' and 'dLen' are temporary variables. The distribution of 0's run in the difference data depends on the objects of the input data. Shown in Algorithm 2, the EGC has a parameter k, and the distribution of the length of the code word depends on the value of k; with smaller k, the code words make shorter in small value, and with larger k, the code words make shorter in large value. Therefore, the EGC can encode the 0's run in the difference data into less data size with suitable k. In addition, the α-EGC uses an offset value because the smallest length of both 0's runs and 1's runs are not zero but one, while the Elias α coding and the EGC begin to encode from zero.

3.4 Evaluation

To evaluate the efficiency of the α-EGC, it is compared with conventional compression methods, which are static Huffman coding (SHC), adaptive Huffman coding (AHC), Elias gamma coding (Gamma) [28], Golomb coding [29], and the original EGC. In this evaluation, compression ratio (CR) is used as an indicator of the effectiveness of compression methods, and the value of the CR is defined as follows:

$$\mathrm{CR} = \frac{D_r\,[\mathrm{bit}]}{D_c\,[\mathrm{bit}]}, \tag{1}$$

where D_c is compressed data size and D_r is raw data size, 1,024. The larger CR indicates that the target compression method can reduce a larger amount of data size.

Table 1 Compression ratio.

sample	SHC	AHC	Gamma	Golomb ($m = 7$)	EGC ($k = 2$)	Proposed (α-EGC) ($k = 3$)
bicy+car	1.55	1.29	2.17	2.00	2.36	2.48
bicycle	2.79	2.98	5.73	4.03	5.78	6.62
bike	1.60	1.35	2.55	2.13	2.54	2.74
bike+car	2.28	2.06	3.25	3.98	3.98	4.55
car	1.69	1.48	2.77	2.32	2.77	3.01
ped+car	2.07	1.70	3.00	2.82	2.98	3.31
pedestrian	2.46	2.54	4.55	3.44	4.65	5.20
face	2.40	2.34	4.45	3.36	4.46	5.08
average	1.12	2.07	3.68	3.00	3.68	4.16

Table 2 Relevance between values of k and CR in α-EGC.

	α-EGC						
sample	$k = 0$	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 6$
bicy+car	2.45	2.54	**2.57**	2.48	2.35	2.21	1.99
bicycle	5.85	6.20	6.43	6.62	6.78	**6.92**	6.54
bike	2.63	2.76	**2.80**	2.74	2.59	2.47	2.23
bike+car	4.06	4.31	4.48	**4.55**	4.52	4.48	4.15
car	2.86	3.00	**3.06**	3.01	2.88	2.76	2.49
ped+car	3.09	3.25	**3.34**	3.31	3.20	3.10	2.82
pedestrian	4.65	4.92	5.11	**5.20**	5.19	5.18	4.80
face	4.57	4.83	4.98	**5.08**	**5.08**	5.07	4.66
Average	3.77	3.98	4.11	**4.16**	4.13	4.09	3.81

In the comparison, the parameter k in the α-EGC is three and the offset is one, which makes the CR highest. This comparison experiment involved 45 sets of sample data, which are classified according to objects and their combination: pedestrians, bicycles, bikes, cars, and a person's face captured at close range. The experimental results are summarized in **Table 1**. The average CRs by object are in each row and all sets of samples are in the bottom row. The α-EGC exhibited the best CR compared with the other methods.

Table 2 shows the relevance between the values of k and CR in α-EGC evaluated with the same samples used in Table 1. This table indicates that the optimum value of the k in α-EGC depends on the objects in the input data.

4. Implementation for Compression and Decompression

This section explains how the proposed compression method α-EGC is implemented with a processor as an ASIP, which has dedicated functional units and an instruction-set of compression and decompression. To enhance the configurability of the stimulation, the inner controller circuits of artificial vision systems need to include a processor, and the ASIP can satisfy the demand for miniaturization of the inner components more than the implementation with processor and dedicated circuits connected via a system bus. A 16-bit reduced instruction-set computer (RISC) processor is selected as the base processor of the proposed ASIP and its instruction-set is shown in **Table 3**. The base RISC processor is low-energy consumption and it has enough performance to control the pulse-generation circuit of the artificial vision systems.

The proposed α-EGC compression procedure is performed following **Algorithm 3**, and with the base RISC processor, the encoding and decoding stages in the α-EGC occupies a large part of all the execution cycles in compression and decompression,

Table 3 Instruction Set of Base RISC Processer.

Operation class	Operations
Arithmetic Ops.	ADD, SUB, SLA, SRA
Logical Ops.	AND, OR, XOR, NOT, SLL, SRL
Comparison Ops.	SLT, SLTU, SEQ, SNEQ
Immediate Ops.	ADDI, SLLI, SRLI, LHI, LLI
Bit Ops.	SETB, CLRB, TSTB
Load/Store Ops.	LHB, LLB, SHB, SLB, LD, ST
Branch Ops.	BRZ, BRNZ
Jump Ops.	JP, JPL, JPR, JPRL
Interrupt Ops.	TRAP, RETI
Special Ops.	NOP, SLEEP

Input: stimuli position data
Output: encoded bit stream
1 Initialize buffers;
2 **repeat**
3 Load stimuli position data to buffer register;
4 Generate difference data from the previous frame;
5 **repeat**
6 **if** *curBit is different from preBit* **then**
7 **if** *curBit == '0'* **then** Exponential Golomb coding ;
8 **else** α-coding ;
9 Store encoded data to memory;
10 curBit := preBit;
11 **end**
12 **else**
13 Run length coding;
14 **end**
15 **until** *the tail of memory*;
16 **until** *the tail of buffer register*;

Algorithm 3: Compression procedure

Table 4 Breakdown of execution cycles of compression by base RISC processor.

Process	Execution cycles	Percentage [%]
1. Initialization & End process	24	0.06
2. Load & Generate difference data	9,338	25.23
3. Run length encoding	15,679	42.36
4. α-EGC encoding	10,448	28.23
5. Store code word	272	0.73
All	37,016	100.00

Table 5 Breakdown of execution cycles of decompression by base RISC processor.

Process	Execution cycles	Percentage [%]
1. Initialization & End process	26	0.03
2. Load encoded data	32	0.39
3. α-EGC decoding	3,694	45.33
4. Run length decoding	3,031	37.19
5. Restore & Store decoded data	1,364	16.74
All	8,149	100.00

respectively. The breakdowns of execution cycles of compression and decompression processes by the base RISC processor are shown in **Table 4** and **Table 5**, respectively. In the compression process, run length encoding is largely dominant and occupies about 42% in all the execution cycles, and the α-EGC encoding also occupies about 28%. In the decompression process, the α-EGC decoding accounts for approximately half of the whole execution cycles, which occupies about 45%, and the run length decoding occupies about 37%. From these analyses, it seems very effective to implement codecs of the α-EGC and run length cod-

ing to the base RISC processor.

4.1 Dedicated Instruction-set and Circuits

From the results of Table 4 and Table 5, this section describes the dedicated instruction-set of the proposed ASIP and its circuit. The proposed ASIP has the dedicated instruction-set for the run length coding and the α-EGC coding in order to reduce execution cycles effectively, and the proposed ASIP has codec for these coding methods to reduce execution cycles of bit operation, while the base RISC processor needs several instructions. The dedicated instruction-set is listed in **Table 6**. The proposed ASIP also has special purpose registers to save the number of working general-purpose registers (GPRs) and reduce memory access to the data memory, which needs several extra cycles by waiting for loading or storing the data. The special purpose registers are allocated as shown in **Table 7**.

Table 6 Dedicated instructions.

Instruction	Used in Compression	Used in Decompression
EGINIT (EGc INITialize)	Y	Y
RLEN (Run Length ENcoding)	Y	-
ALEN (Alpha ENcoding)	Y	-
EGENC (EGc ENCode)	Y	-
PACK (PACKing)	Y	-
STDATA (STore encoded DATA)	Y	-
EGFLASH (EGc FLASH)	Y	-
LDDATA (LoaD encoded DATA)	-	Y
ALDEC (ALpha DECoding)	-	Y
EGDEC (EGc DECoding)	-	Y
UNPACK (UNPACKing)	-	Y
INITRLD (INITialize Run Length Decoding)	-	Y
RLDEC (Run Length DECoding)	-	Y
LDRL (LoaD Run Length)	-	Y

Table 7 Special purpose registers for codecs.

name	width [bit]	description
bufpnt	6	index where data remain in codebuf
codebuf	48	buffer register for encode data
codelen	8	code length of code word from codec
codewd	33	code word from codec
membuf	48 (16x3)	code word snippets in one word (use in encoding only)
param	16	parameter for coding (including k for EGC and offset)
pn	2	control signal for the times of memory access
rlpntr	4	pointer of tail of decoded run length data
rlbuf	16	buffer register for decoded run length data

As shown in Table 4, the compression procedure is divided roughly into five stages. In the stage of initialization, instruction EGCINT initializes the special purpose registers; clears the bufpnt register, the codebuf register, the pn register, and the codewd register, and sets the k of EGC and offset of coding to the param register. In the stage of the run length encoding, instruction RLEN is used to count the run of successive bit. Two operands of this instruction indicate the data to encode and the index of start to count up, and it writes the index of the switching position of bits in the data, where the bit is changed from zero to one and vice versa, to the GPR. In the stage of the α-EGC encoding, instruction ALEN and EGENC encode the value of the run to each code word individually, and the code word and its code length are stored to the codewd register and the codelen register, respectively. Instruction PACK is used at every step after execution of ALEN or EGENC. This instruction loads the code word data from the codewd register and stores the data to the tail of data in the codebuf register referring the bufpnt register, and then the instruction also updates the bupnt register, the pn register, and the membuf register. In the stage of storing the code word, instruction STDATA stores the encoded data in the codebuf register to the data memory. This instruction refers the pn register that stores the number of times for storing the data in the membuf register. When the STDATA instruction is carried out, the data in the membuf register is stored in 16 bits, and the membuf register is shifted to MSB and the value of the pn register is subtracted by one. If the value of the pn register is zero, the STDATA instruction is not carried out, which is equal to the NOP instruction. In the stage of the end process, the instruction EGFLASH stores the remains of the code words in the codebuf register to the data memory.

As shown in Table 5, the decompression procedure is divided roughly into five stages. The dedicated instruction-set is used in the stage of initialization, loading, and decoding the code word. In the initialization, instruction EGCINT is also used as in the compression, and the instruction INITRLD is used to clear registers in the run length decoder. In the stage of loading the code word, instruction LDDATA loads encoded data to the codebuf register from the data memory, which requires the address of the data memory as an operand. In the stage of α-EGC decoding, instruction ALDEC and EGDEC decode the code word in the codebuf register to the numerical value of the run of bit stream, and then store the data to the GPR indicated as an operand. After

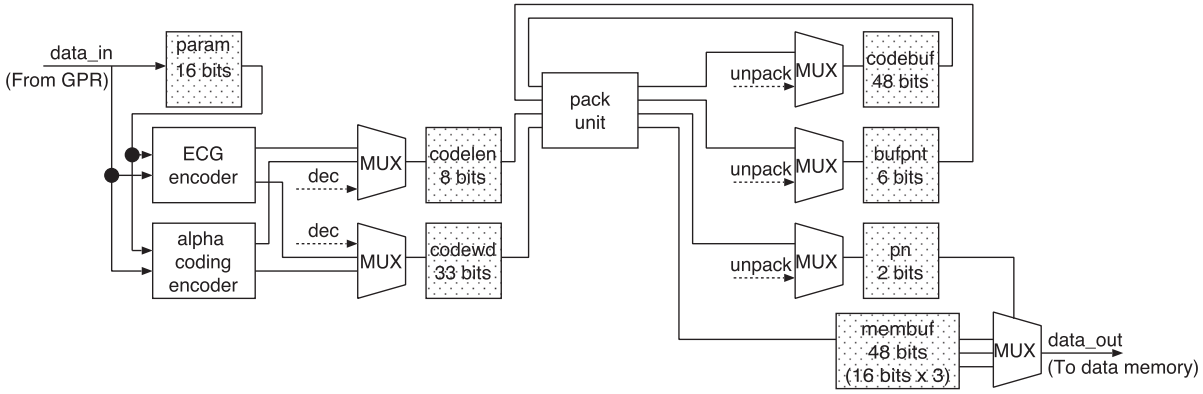


Fig. 5 Dedicated circuits and registers for compression.

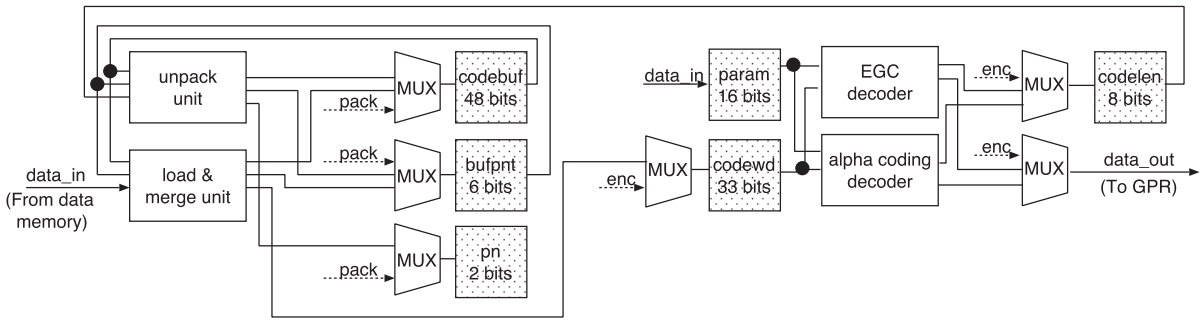


Fig. 7 Dedicated circuits and registers for decompression.

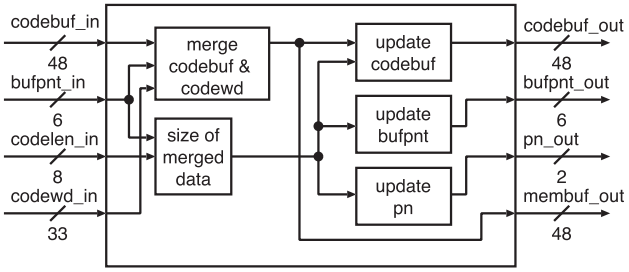


Fig. 6 Circuit for packing codebuf register and updating bufpnt register, pn register, and membuf register in encoding.

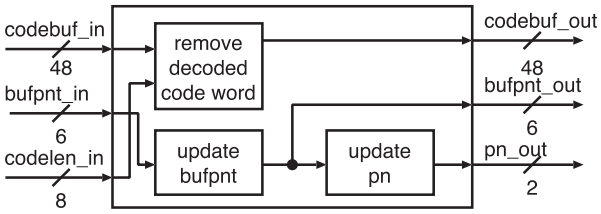


Fig. 8 Circuit for detaching decoded code word from codebuf register and updating bufpnt register and pn register in decoding.

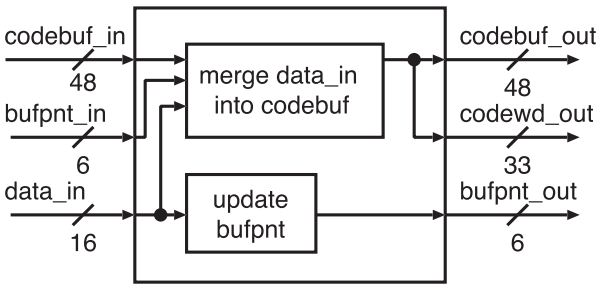


Fig. 9 Circuit for merging data from data memory to codebuf register and updating bufpnt register in decoding.

decoding by ALDEC or EGDEC, instruction UNPACK is used and loads the encoded data from the codebuf register and stores one code word to the codewd register in parallel with decoding the header of the code word in order to get the code length of the code word. In the stage of the run length decoding, instruction RLDEC and LDRL are used. Instruction RLDEC expands the numerical run in one of the operands to bit stream to the rlbuf register. After execution, instruction RLDEC subtracts the value that is equal to the length of the bit stream instruction which expands from the numerical run. If the rlbuf register is full, instruction RLDEC writes the status to the GPR indicated by an operand, and then instruction LDRL is used to load the bit stream in the rlbuf register to the GPR indicated by an operand.

Figure 5 illustrates components realizing the dedicated instruction-set for compression, and **Fig. 6** illustrates the block diagram of 'pack unit' in Fig. 5. In Fig. 5, 'data_in' indicates input from one of the GPRs indicated by an operand, and 'data_out' indicates output to the access unit of the data memory in the base RISC processor. **Figure 7** also illustrates components realizing the dedicated instruction-set for decompression, and **Fig. 8** and **Fig. 9** illustrate the block diagrams of 'unpack unit' and 'load & merge unit', respectively. In Fig. 7, 'data_in' indicates output from access unit of the data memory, and 'data_out' indicates input to the one of the GPRs. In both Fig. 5 and Fig. 7, shadowed boxes indicate registers that are initialized by instruction ECGINIT.

5. Evaluation

This section explains the comparison of the proposed ASIP and the base processor in terms of the following points: area, execution cycles, power consumption, and energy consumption. The area and power consumption based on the switching probabilities of each gate were estimated using Design Compiler from Synopsys, Inc., with which the TSMC 0.18 μm typical library was used. The power consumption was measured within the synthesis limitation of the minimum area. The execution cycles and the switching probabilities of each gate were based on the simulation results from ModelSim by Mentor Graphics Corp. The energy consumption E is defined as follows.

$$E = P\frac{N}{f}, \tag{2}$$

where P is power [W], N is the number of execution cycles, and f is operation frequency [Hz].

Table 8 Comparison of execution cycles in compression.

Process	Base	Proposed	Diff. [%]
1. Initialization & End process	24	47	+95.83
2. Load & Generate difference data	9,338	1,282	−86.27
3. Run length encoding	15,679	2,148	−86.30
4. α-EGC encoding	10,448	1,326	−87.30
5. Store code word	272	272	±0.00
All	37,016	5,075	−86.30

5.1 Results

The comparisons of execution cycles for compression and decompression with the proposed processor and the base RISC processor are summarized in **Table 8** and **Table 9**, respectively. For compression, the proposed processor reduced the number of execution cycles by about 86% and 87% for run length encoding and

Table 13 Comparison with proposed compression method and other methods in energy consumption.

Method	P_d [μW/MHz]	E_d [nJ]	n [bit]	E_r (BLE [30]) [nJ]	E_r (BAN [31]) [nJ]
w/o compression	0	0	1,024	15,667 (±0.00)	4,198 (±0.00)
Gamma	46.42	117	278.26	4,374 (−72.08)	1,258 (−70.04)
EGC	46.52	100	278.26	4,358 (−72.19)	1,241 (−70.44)
Proposed	47.62	128	246.15	3,894 (−75.15)	1,137 (−72.93)

Table 9 Comparison of execution cycles in decompression.

Process	Base	Proposed	Diff. [%]
1. Initialize & End process	26	215	+826.92
2. Load encoded data	32	234	+731.25
3. α-EGC decoding	3,694	64	−98.27
4. Run length decoding	3,031	1,372	−54.73
5. Restore & Store decoded data	1,364	792	−41.94
All	8,149	2,678	−67.14

Table 10 Comparison with base processor and proposed processor.

		Base	Proposed	Diff. [%]
Area [μm^2]		53,102	122,764	+131.19
Max Freq. [MHz]		303	222	−26.70
Power [μW/MHz]	Compression	37.6	33.3	−11.44
	Decompression	41.5	47.6	+14.42
Energy [nJ]	Compression	1,391	169	−87.86
	Decompression	338	128	−62.43

Table 11 Characteristics of low-power wireless communications.

	Standard	V_s [V]	P_r [mW]	T [kbps]	e_r [nJ/bit]
DA14580 [30]	BLE	2.35–3.3	15.3	1,000	15.3
ISSCC2014 [31]	BAN	0.74	3.66	971.4	4.1

Table 12 Comparison of estimation of energy consumption in receiver.

	n [bit]	E_d [nJ]	E_r (BLE [30]) [nJ] (Diff. [%])	E_r (BAN [31]) [nJ] (Diff. [%])
w/o compression	1 024	0	15,667 (±0.00)	4,198 (±0.00)
Base	246.15	339	4,104 (−73.81)	1,347 (−67.92)
Proposed	246.15	108	3,894 (−75.15)	1,137 (−72.92)

the α-EGC encoding, respectively, and reduced the total number of execution cycles by 86%. For decompression, the proposed processor reduced the number of execution cycles by 98% and 54% for the α-EGC decoding and run length decoding. As shown in **Table 10**, the power consumption of the proposed processor decreased by 11% and increased 14% for compression and decompression, respectively, and the proposed processor reduced energy consumption by 87% and 62% for compression and decompression, respectively.

5.2 Discussion

From the evaluation results in Section 5.1, we estimate a reduction of energy consumption in wireless communication by the proposed implementation. Some examples of low-power wireless communication implementation and their parameters are listed in **Table 11** (see Ref. [32] to get more details). In Table 11, V_s is supply voltage, P_r is power consumption of receiver in activation, T is data rate in wireless communication, and e_r is energy consumption per one bit in receiving. In communication standards, Bluetooth low energy (BLE) and body area network (BAN), which is also called IEEE 802.15.6 standard, are well-known low-power communication standards, and BLE has been already used for various purposes. The energy consumption of receiving one stimuli position data in wireless communication E_r is defined as follows:

$$E_r = ne_r + E_d, \tag{3}$$

where n [bit] is the amount of received traffic data per one frame of stimuli position data and E_d [nJ] is the energy consumption in compression processing. Estimation results are summarized in **Table 12** that compares the proposed implementation with the base RISC processor and **Table 13** that compares the proposed implementation of the α-EGC and other simpler methods: gamma coding and EGC. This table shows that the α-EGC can reduce the energy consumption of a wireless communication receiver more significantly than the other methods, and it also shows that the increment of the energy consumption by adding data compression processing is negligible in both standards.

The evaluation results also show that the proposed ASIP can perform decompression in terms of real-time processing. It can decompress data in 303 μs per one frame in 10 MHz. Compared with the base RISC processor, the proposed processor reduced execution times in decompression, which can achieve the equivalent performance to the base RISC processor with less operation frequency.

6. Conclusion

This paper proposed a lossless data compression method, α-EGC, for stimuli position data used in artificial vision systems. According to analyses of the stimuli position data, the proposed method marked higher CR, 4.16, compared with the conventional lossless coding methods. The implementation of the α-EGC was also proposed as an ASIP, which encodes and decodes by smaller number of instructions using a dedicated instruction-set. Evaluation results showed that the proposed implementation reduced the number of execution cycles by about 86% and 67% for compression and decompression, respectively, and reduced energy consumption by about 87% and 62%, respectively. This paper revealed that the ASIP implementation could reduce energy consumption to about one tenth in compression with thirteen tenths in the area of circuits. Based on the evaluation results, this paper showed that the α-EGC can reduce energy consumption in wireless communication. Our future work is to evaluate fabricated chips and verify the efficiency of the α-EGC to measure energy consumption in wireless communication in observation.

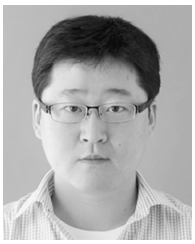
Acknowledgments This work was supported in part by the MEXT project, “Creating Hybrid Organs of the future” at Osaka University. Authors express their special thanks to Prof’s Tetsuya Yagi, Yuki Hayashida, Hirotsugu Okuno, Seiji Kameda, and Takatsugu Kamata from Osaka University for their valuable discussions about visual prosthesis and artificial vision systems.

References

[1] WHO Media centre: Visual Impairment and Blindness (2014).
[2] Potts, A.M. and Inoue, J.: The Electrically Evoked Response of the Visual System (EER): III. Further Contribution to the Origin of the EER, *Investigative Ophthalmology & Visual Science*, Vol.9, No.10, pp.814–819 (1970).
[3] Fernandes, R.A.B., Diniz, B., Ribeiro, R. and Humayun, M.: Artificial Vision through Neuronal Stimulation, *Neuroscience Letters*, Vol.519, No.2, pp.122–128 (2012).
[4] Rothermel, A.: Recent results with subretinal stimulation, *Proc. IEEE Biomedical Circuits and Systems Conference*, pp.220–223 (2014).
[5] Liu, W. et al.: A Neuro-Stimulus Chip with Telemetry Unit for Retinal Prosthetic Device, *IEEE Journal of Solid-State Circuits*, Vol.35, No.10, pp.1487–1497 (2000).
[6] Shire, D. et al.: Development and Implantation of a Minimally Invasive Wireless Subretinal Neurostimulator, *IEEE Trans. Biomedical Engineering*, Vol.56, No.10, pp.2502–2511 (2009).
[7] Toshihiko, N. et al.: Performance Improvement and Functionalization of an Electrode Array for Retinal Prosthesis by Iridium Oxide Coating and Introduction of Smart-Wiring Technology using CMOS Microchips, *Sensors and Actuators A: Physical*, Vol.211, pp.27–37 (2014).
[8] Liu, W. et al.: Electronic Visual Prosthesis, *Artificial Organs*, Vol.27, No.11, pp.986–995 (2003).
[9] Pezaris, J.S. and Reid, R.C.: Simulations of Electrode Placement for a Thalamic Visual Prosthesis, *IEEE Trans. Biomedical Engineering*, Vol.56, No.1, pp.172–178 (2009).
[10] Dobelle, W.H., Mladejovsky, M.G. and Girvin, J.P.: Artificial Vision for the Blind: Electrical Stimulation of Visual Cortex Offers Hope for a Functional Prosthesis, *Science*, Vol.183, No.4123, pp.440–444 (1974).
[11] Dobelle, W.H.: Artificial Vision for the Blind by Connecting a Television Camera to the Visual Cortex, *ASAIO Journal*, Vol.46, No.1, pp.3–9 (2000).
[12] Polimeni, J.R., Balasubramanian, M. and Schwartz, E.L.: Multi-area Visuotopic Map Complexes in Macaque Striate and Extra-striate Cortex, *Vision Research*, Vol.46, No.20, pp.3336–3359 (2006).
[13] Landsiedel, O., Wehrle, K. and Götz, S.: Accurate Prediction of Power Consumption in Sensor Networks, *Proc. 2nd IEEE Workshop on Embedded Networked Sensors* (2005).
[14] Campbell, P. et al.: A Silicon-Based, Three-Dimensional Neural Interface: Manufacturing Processes for an Intracortical Electrode Array, *IEEE Trans. Biomedical Engineering*, Vol.38, No.8, pp.758–768 (1991).
[15] Wise, K.D. and Najafi, K.: Microfabrication Techniques for Integrated Sensors and Microsystems, *Science*, Vol.254, No.5036, pp.1335–1342 (1991).
[16] Shulyzki, R. et al.: 320-Channel Active Probe for High-Resolution Neuromonitoring and Responsive Neurostimulation, *IEEE Trans. Biomedical Circuits and Systems*, Vol.9, No.1, pp.34–49 (2015).
[17] Wise, K.D.: Silicon Microsystems for Neuroscience and Neural Prostheses, *IEEE Engineering in Medicine and Biology Magazine*, Vol.24, No.5, pp.22–29 (2005).
[18] Normann, R.A., Maynard, E.M., Rousche, P.J. and Warren, D.J.: A Neural Interface for a Cortical Vision Prosthesis, *Vision Research*, Vol.39, No.15, pp.2577–2587 (1999).
[19] Chua, E. and Fang, W.-C.: Mixed Bio-Signal Lossless Data Compressor for Portable Brain-Heart Monitoring Systems, *IEEE Trans. Consumer Electronics*, Vol.57, No.1, pp.267–273 (2011).
[20] Olsson, R. and Wise, K.: A Three-Dimensional Neural Recording Microsystem with Implantable Data Compression Circuitry, *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Vol.1, pp.558–559 (2005).
[21] Lapolli, A., Coppa, B. and Heliot, R.: Low-Power Hardware for Neural Spike Compression in BMIs, *Proc. 35th Annual Int. Conf. of the IEEE Engineering in Medicine and Biology Society*, pp.2156–2159 (2013).
[22] Sugiura, T., Yu, J., Takeuchi, Y. and Imai, M.: A Low-Energy ASIP with Flexible Exponential Golomb Codec for Lossless Data Compression toward Artificial Vision Systems, *Proc. IEEE Biomedical Circuits and Systems Conference*, pp.97–100 (2015).
[23] Shimonomura, K., Kameda, S., Iwata, A. and Yagi, T.: Wide-Dynamic-Range APS-Based Silicon Retina with Brightness Constancy, *IEEE Trans. Neural Networks*, Vol.22, No.9, pp.1482–1493 (2011).
[24] Kameda, S. and Yagi, T.: An Analog VLSI Chip Emulating Sustained and Transient Response Channels of the Vertebrate Retina, *IEEE Trans. Neural Networks*, Vol.14, No.5, pp.1405–1412 (2003).
[25] Kameda, S. and Yagi, T.: An Analog Silicon Retina with Multichip Configuration, *IEEE Trans. Neural Networks*, Vol.17, No.1, pp.197–210 (2006).
[26] Okuno, H. and Yagi, T.: Image Sensor System with Bio-Inspired Efficient Coding and Adaptation, *IEEE Trans. Biomedical Circuits and Systems*, Vol.6, No.4, pp.375–384 (2012).
[27] Lei, L. and Krishnendu, C.: On Using Exponential-Golomb Codes and Subexponential Codes for System-on-a-Chip Test Data Compression, *Journal of Electronic Testing*, Vol.20, No.6, pp.667–670 (2004).
[28] Elias, P.: Universal Codeword Sets and Representations of the Integers, *IEEE Trans. Inf. Theory*, Vol.21, No.2, pp.194–203 (1975).
[29] Golomb, S.: Run-length encodings, *IEEE Trans. Inf. Theory*, Vol.12, No.3, pp.399–401 (1966).
[30] Dialog Semiconductor: *DA14580 Low Power Bluetooth Smart SoC*, 3.1.0 edition (2015).
[31] Bachmann, C. et al.: 10.6 A 0.74 V 200 μW Multi-Standard Transceiver Digital Baseband in 40 nm LP-CMOS for 2.4 GHz Bluetooth Smart/ZigBee/IEEE 802.15.6 Personal Area Networks, *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, pp.186–187 (2014).
[32] Blanckenstein, J., Klaue, J. and Karl, H.: A Survey of Low-Power Transceivers and Their Applications, *IEEE Circuits and Systems Magazine*, Vol.15, No.3, pp.6–17 (2015).

Tomoki Sugiura received his B.E. degree and Master of Information Science and Technology from Osaka University, Osaka, Japan in 2013 and 2015, respectively. He is now a Ph.D. cadidate at Graduate school of Information Science and Technology, Osaka University. His current research interests include biomedical data compression, low-energy technology for implantable biomedical device, and cortical visual prosthesis.

Masaharu Imai received his B.S. degree in Electrical Engineering from Nagoya University, Nagoya, Japan in 1974, then M.S. and Ph.D. degrees in Information Science from Nagoya University in 1976 and 1979, respectively. From April 1979 through March 1996, he has been with the Department of Information and Computer Sciences, Toyohashi University of Technology, Toyohashi, Japan, where his final title was a professor. He has been a visiting professor in the University of South Carolina, Columbia, SC, U.S.A. from 1984 to 1985. From April 1996 to present, he is with Osaka University, Osaka, Japan, where he is a professor of the Department of Information Systems Engineering, Graduate School of Information Science and Technology. His research interest includes ASIP (Application domain Specific Instruction set Processor) design automation, hardware/software codesign, VLSI architecture, and system level design methodology of embedded systems, since 1991, he has been working for EDA standardization including VHDL under IEEE and JEITA (Japan Electronics and Information Technology Industries Association). He is a member of IEEE, ACM, and IEICE of Japan.



Jaehoon Yu received his B.E. degree in Electrical and Electronic Engineering and M.S. degree in Communications and Computer Engineering from Kyoto University, Kyoto, Japan, in 2005 and 2007, respectively, and received Ph.D. degree in Information Systems Engineering from Osaka University, Osaka, Japan, in 2013. He is currently an assistant professor in the Department of Information Systems Engineering, Osaka University. His research interests include computer vision, machine learning, pattern recognition, and system level design. He is a member of IEICE, and IEEE.

Yoshinori Takeuchi received his B.E., M.E. and Dr.Eng. degrees from Tokyo Institute of Technology in 1987, 1989 and 1992, respectively. From 1992 through 1996, he was a research associate of Department of Engineering, Tokyo University of Agriculture and Technology. From 1996, he has been with the Osaka University. He was a visiting scholar in University of California, Irvine from 2006 to 2007. He is currently an associate professor of Graduate School of Information Science and Technology at Osaka University. His research interests include system level design, VLSI design and VLSI CAD. He is a member of IEICE of Japan, ACM, and SP, CAS and SSC Society of IEEE.