IEICE TRANS. INF. & SYST., VOL.E101–D, NO.11 NOVEMBER 2018

PAPER

High-Performance Super-Resolution via Patch-Based Deep Neural Network for Real-Time Implementation

Reo AOKI[†,††a)], **Kousuke IMAMURA**[†††], **Akihiro HIRANO**[†††], ***Members***, ***and*** **Yoshio MATSUDA**[†††], ***Nonmember***

SUMMARY Recently, Super-resolution convolutional neural network (SRCNN) is widely known as a state of the art method for achieving single-image super resolution. However, performance problems such as jaggy and ringing artifacts exist in SRCNN. Moreover, in order to realize a real-time upconverting system for high-resolution video streams such as 4K/8K 60 fps, problems such as processing delay and implementation cost remain. In the present paper, we propose high-performance super-resolution via patch-based deep neural network (SR-PDNN) rather than a convolutional neural network (CNN). Despite the very simple end-to-end learning system, the SR-PDNN achieves higher performance than the conventional CNN-based approach. In addition, this system is suitable for ultra-low-delay video processing by hardware implementation using an application-specific integrated circuit (ASIC) or a field-programmable gate array (FPGA).

key words: *super-resolution, deep neural network, deep leaning, real-time processing*

1. Introduction

In recent years, the resolution of video streams of cameras and displays has increased to 4K/8K 60 fps. As such, the demand for upconverting from conventional image quality, such as Full-HD, in real time has arisen. In particular, in the fields of endoscopic surgery and telemedicine, it is difficult to increase the resolution of video sources because of physical restrictions in the imaging or transmission system. Therefore, a viewer-side real-time super-resolution upconverting system is needed.

Generally, there are two types of approaches to the implementation of real-time image processing. Software-based approaches, such as those involving a CPU or GPU, and hardware-based approaches, for example using an application-specific integrated circuit (ASIC) or field-programmable gate array (FPGA). However, in order to keep up with recent high-resolution trends such as 4K/8K 60 fps video and accomplish non-frame delay processing, which is crucial in the field of endoscopic surgery, hardware-based approaches are preferred for the display system.

Manuscript received March 5, 2018.
Manuscript revised June 22, 2018.
Manuscript publicized August 20, 2018.
[†]The author is with Natural Science & Technology, Kanazawa University, Kanazawa-shi, 920–1192 Japan.
[††]The author is with EIZO Corporation R&D, Visual Technologies (ASIC), Hakusan-shi, 924–8566 Japan.
[†††]The authors are with the Faculty of Engineering, Kanazawa University, Kanazawa-shi, 920–1192 Japan.
a) E-mail: reo.aoki@eizo.com
DOI: 10.1587/transinf.2018EDP7081

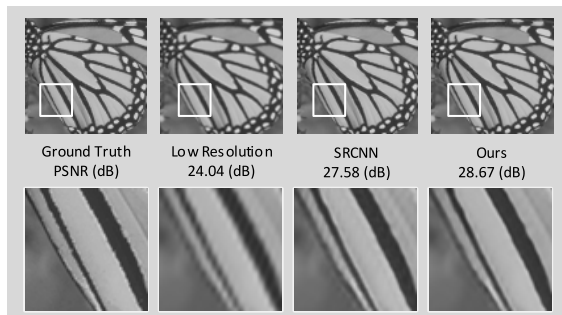

Fig. 1 The proposed super-resolution deep neural network outperforms the SRCNN. More details are provided in Sect. 4.2 (the *Set 5* dataset with an upscaling factor of 3).

Recently, learning based approaches for super resolution have been remarkably studied [1]. Especially, super resolution convolutional neural network (SRCNN) proposed by Dong et al. [2], [3] consists of feed-forward-processing architecture that does not require iterative processing. However, hardware cost for the SRCNN is high. Furthermore, jaggy and ringing still remain [4] as shown in Fig. 1.

The contribution of the present paper is to provide a novel neural-network-based architecture for super resolution. It is patch-based and can reduce jaggy and ringing artifacts. In addition, we also provide cost performance evaluation for the hardware implementation such as ASIC and FPGA.

The remainder of the present paper is organized as follows. Section 2 presents the detailed related works. Section 3 presents the proposed method. Section 4 presents the experimental results. Section 5 describes the ease of cost-performance tuning for the proposed method. Finally, Sect. 6 concludes the present paper.

2. Related Work

Single-image super-resolution is a classical problem in the image processing field, and numerous algorithms have been proposed thus far [1]. In recent years, machine learning approaches involving example-based super resolution have enabled remarkable achievements.

Example-based super resolution, first proposed by Freeman et al. [5], is a primitive dictionary-based method that restores the high-resolution image from a dictionary that consists of paired patch images of low resolution and corresponding high resolution.

Because of the patch-based approach, example-based super resolution can be applied to images of any size. However, practical performance requires a considerable number of paired images as well as a significant search cost.

Later, Yang et al. proposed sparse-coding super resolution (ScSR) [6], [7]. In ScSR, the patched image is encoded as coefficients of a few of (sparse) base functions that are pre-learned by machine learning, therefore, the dictionary size is remarkably reduced. From the viewpoint of real-time processing, ScSR is a good approach to compressing the prior information to a dictionary in offline processing. However, in online processing, iterative processing, such as orthogonal matting pursuit (OMP) [8], is required in order to obtain a sparse encoding of inputs.

In recent years, Dong et al. proposed SRCNN [2], [3]. The SRCNN uses a convolutional neural network (CNN) based on a machine learning system. The network internal parameters can be directly optimized via back propagation of end-to-end mapping between low- and high-resolution images. Here, the benefit of the end-to-end approach is that it can achieve a higher level optimization as compared to human-based design. Moreover, from the viewpoint of real-time processing, a feed-forward-based architecture is easier to implement than an iterative processing method, such as ScSR.

Because of the simple end-to-end learning approach and highly accurate restoration, the SRCNN is widely known as a current state-of-the-art method in the field of single-image super resolution, and the method is now applied to several applications, such as video super resolution [9] and character recognition [10].

However, SRCNN has a problem in terms of performance because it often suffers from jaggy and ringing artifacts (Fig. 1). Moreover, realization of real-time processing with a non-frame delay system involves some issues concerning processing delay and implementation cost.

As post-SRCNN studies, Timofte et al. [11] proposed seven techniques, such as data augmentation, back projection, cascading, etc., to improve performance of example-based single image super resolution. However the most effective cascading has an issue of significantly increasing costs of implementation. As a similar approach, Kim et al. [12] proposed a very deep convolutional network (VDSR) that further stratified SRCNN. Although it is a very excellent in performance, since it is composed of 20 layers of CNN, it is also a problem that a very large amount of calculation is necessary.

As a different approach from deepening, Shi et al. [13] and Ohtani et al. [4] proposed a modified SRCNN structure in order to improve the performance. They focused on the fact that jaggy and ringing artifacts occurred from the bicubic interpolation upon making the input signal. Therefore, they tried to train SRCNNs independently for each interpolated pixel position. The performance was improved but this approach requires multiple SRCNNs according to the upscaling factor. Therefore, problems remain in terms of structural complexity and implementation cost.

As a method suitable for real-time calculation, Romano et al. [14] proposed a method of super-resolution with a simple optimized filter for each detected direction. However, their method is inferior to SRCNN in performance, in addition, Gaussian pyramid based Kalman filtering [15] is necessary for direction detection, so it is not suitable for low-delay hardware processing. Another, Zeng et al. [16] proposed the coupled deep auto-encoder (CDA), which is also a neural-network-based approach and consists of two types of auto-encoder for low resolution and high resolution image patches. The concept of the CDA is similar to that of ScSR. The paired auto-encoder works as a dictionary and the fully connected network between the auto-encoders is optimized to estimate high resolution features. Actually, the CDA is a good approach for real-time processing because the process is based on a patch-based approach. However, some performance issues remain because the authors did not consider interpolated pixel positions, and, essentially, the constraint of making the auto-encoder is not needed for restoration from an end-to-end perspective.

On the other hand, in terms of real-time implementation, Kim et al. [17] proposed GPU implementation techniques for the SRCNN. Normally, because the SRCNN is based on multistage and multidimensional filter processing, the memory cost of implementation is enormous, on the order of the image size. However, by decomposing the input image into small tile images, they reduced the implementation cost. Nevertheless, from the viewpoint of feasibility for 4K/8K 60 fps, it is not yet realistic.

For the above reasons, a method that is more suitable for low-delay real-time processing and has better performance than the SRCNN has not yet been proposed. In the present paper, we propose a novel high-performance super-resolution method for low-delay real-time processing with a patch based deep neural network (PDNN) based on the following concepts:

(a) *A PDNN is more suitable for super resolution than a CNN.*
In the case of a CNN, the referenced area spreads for each layer. This feature is important for understanding the global meaning, especially in the field of general image recognition. However, with regard to super resolution, excessively broad information leads to an increase in the difficulty of prediction. Therefore, a PDNN is better than a CNN, because the PDNN can estimate deeply with limited input information.

(b) *Target simplification is necessary in order to maximize the effectiveness of an end-to-end approach.*
In the case of CDA, the regularization term is included in the loss function for making a pair of auto encoders. However, the regularization term does not degrade the performance even if the value increases. Actually, SRCNN [2], [3] and VDSR [12] do not include the regularization term in their loss function, and their performance is better than CDA. (CDA performance is infe-

rior to SRCNN 9-5-5). Therefore, in our opinions, the regularization term is unnecessary from the viewpoint of end-to-end perspective. Furthermore, CDA does not consider interpolated pixel positions, which means the learning target includes unnecessary variations. Therefore, the result would be an averaged result.

In order to achieve the high throughput required for 4K/8K 60 fps, the present paper focuses on hardware implementation, such as an ASIC or a FPGA.

3. Proposed Method: SR-PDNN

In this section, we describe in detail the proposed super-resolution method via patch-based deep neural network (SR-PDNN).

3.1 PDNN Architecture

First, considering the definition of ideal input and output is very important in designing the target neural network system. For example, if we set the low resolution and high resolution images as direct input/output learning data, the target networks start by learning to approach the input level, followed by learning to obtain the output data. However, by using the concept of ResNet [18], the ideal output of the neural network becomes only for a residual component. Figure 2 shows the flow from *low resolution quality* (LR) image which is already upscaled by bicubic interpolation to estimated *high resolution quality* (HR) image. Here, even if the output of PDNN is zero, the final output is the same as the LR image. That's mean, the PDNN can focus on the restoration of lost high-frequency components that we would like to restore.

Here, the components of the image to be restored with super resolution are considered to be approximately the same as high-frequency components of HR, and the residual structure does not need to consider the lightness absolute values to fit the output lightness. This means that improvements in convergence speed and performance can be expected. For the reasons stated above, we define the PDNN role as:

$$at\ learning\ :\ PDNN(y) \leftarrow x - y \tag{1}$$

$$at\ estimation\ :\ \hat{x} = PDNN(y) + y \tag{2}$$

where $y \in \mathbb{R}^n$ and $x \in \mathbb{R}^n$ are column vectors of LR and HR patch images, and $\hat{x} \in \mathbb{R}^n$ is an estimated result of x. At a learning step, the PDNN is optimized to output the difference between y and x. In an estimation step, the PDNN outputs the estimated difference. Figure 3 (a) shows the structure and data flow of PDNN. There are three layers in PDNN. The first input layer is Feature Extraction, the second middle layer is Estimation, and the last output layer is Restoration. In the following, we described the details of these layers.

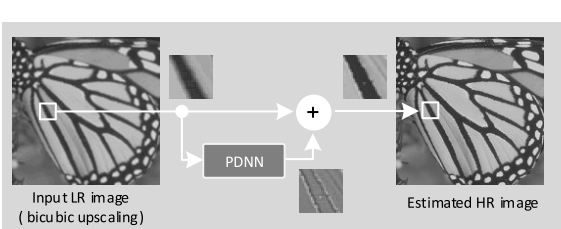

Fig. 2 Outline of the super resolution via patch-based deep neural network (SR-PDNN).

3.1.1 Restoration (*Output Layer*)

In the case of super resolution, the component to be restored can be regarded as mixed high-frequency components. Therefore, the output of $PDNN(y)$ can be expressed as a sum of base vectors without a bias component:

$$PDNN(y) = \Sigma_i[atom_i \times p(atom_i|y)] \tag{3}$$

$$= W_{atom} \times p(W_{atom}|y) \tag{4}$$

where $atom \in \mathbb{R}^n$ is a base vector to express a high-frequency component, and W_{atom}, which is an $f_h \times n$ matrix, represents a group of $atom_i$. Figure 3 (b) shows the relation between $atom$ and W_{atom}. As the figure, the $atom$ is an element of W_{atom} and the size is same as the output patch image. Moreover, $p(atom_i|y)$ is defined as a contribution of an $atom_i$ when input y is *given*, and $p(W_{atom}|y) \in \mathbb{R}^{f_h}$ represents a group of $p(atom_i|y)$ as well. For more detail, the concrete patterns of *atoms* are shown at Fig. 7 in Sect. 5.

As a non-bias expression, the optimization process for bias, which is conventionally required for SRCNN, becomes unnecessary. Moreover, the cost of implementation is expected to be reduced.

3.1.2 Feature Extraction (*Input Layer*)

The expected role of the input layer is to extract highly correlated features, which is useful for estimating $p(W_{atom}|y)$. Here, we define the features estimated by a bank of filters, the size of which is the same as that of input y. Then, the input layer is described as follows:

$$Features = softsign(\Sigma_j[filter_j \times y]) \tag{5}$$

$$= softsign(W_{filter} \times y) \tag{6}$$

where $filter_j \in \mathbb{R}^n$ is a filter, and W_{filter} is an $n \times f_l$ matrix that is used as a filter bank. Figure 3 (b) also shows the relation between $filter$ and W_{filter}. As the figure, the $filter$ is an element of W_{filter} and the size is same as the input patch image. And the concrete patterns of $filters$ are shown at Fig. 8 in Sect. 5. As a transfer function, we select $softsign(\alpha) = \frac{\alpha}{1+|\alpha|}$, because the feature that is expressed as a high-frequency component should be transmitted. Then, the feature is not dependent on the input bias that is approximately the same as the lightness expressed as a low-frequency component.

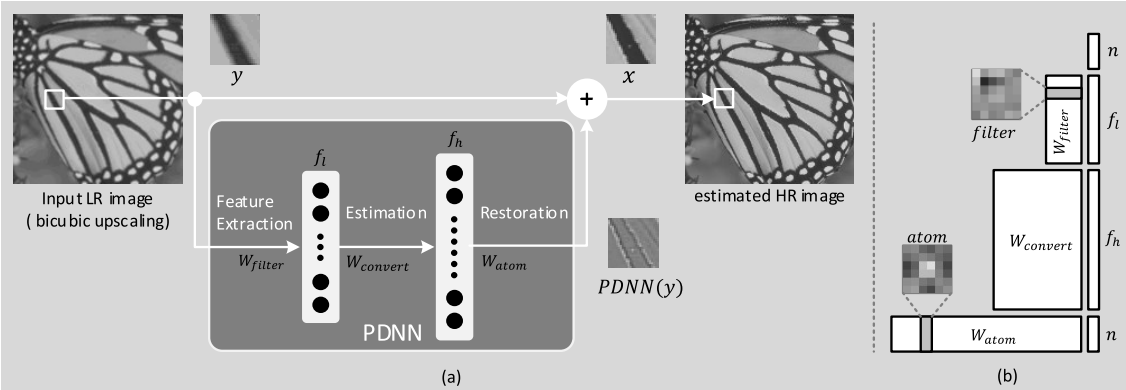

Fig. 3 The proposed PDNN structure for super resolution. (a) Data-flow and layer structure. (b) The size relations between weight matrices for each layer.

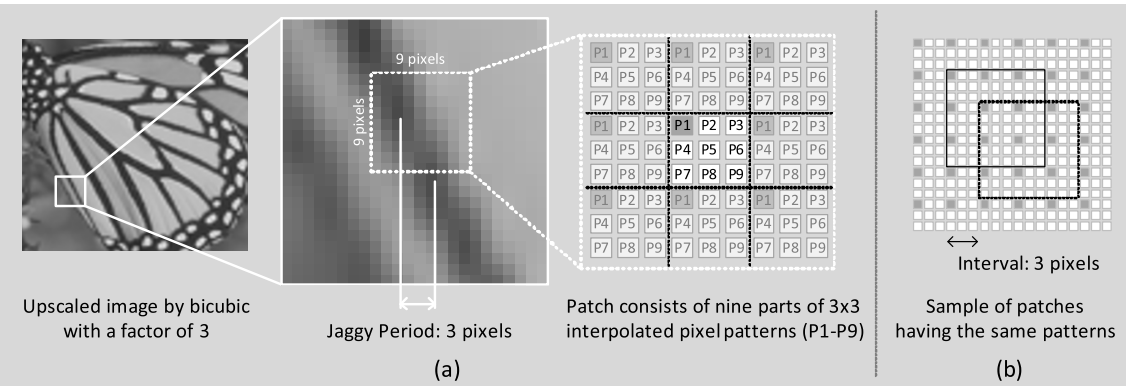

Fig. 4 Relationship of jaggy artifacts and patch extracting position against the interpolation pixel position in the case that the upscaling factor is 3. (a) The jaggy artifacts with 3 pixel period. (b) Patch images that have same interpolated pixel patterns.

3.1.3 Estimation (*Middle Layer*)

Based on the extracted features, $p(W_{atom}|y)$ is defined as the following nonlinear mapping:

$$p(W_{atom}|y) = softsign(W_{convert} \times Features) \tag{7}$$

where $W_{convert}$ is an $f_l \times f_h$ matrix as a fully connected estimation layer. In the present paper, we focus on the cost-performance balance. Therefore, we select a single-layer-based non-linear mapping for the estimation.

3.2 Training

Learning condition simplification is very important in machine learning in order to obtain good results. Figure 4 shows the relationship of jaggy artifacts and patch extracting position against the interpolation pixel position in the case that the upscaling factor is 3. In Fig. 4 (a), the left image is a sample of an input image degraded by bicubic interpolation. The center image is a part of the left image to show the jaggy clearly. Here, the size of jaggy period is same as upscaling factor. The white dash square area indicates a sample of an extracted patch region. In this case, the patch size is set to 9×9 pixels. The right figure shows periodic interpolated pixel patterns (P1-P9) of the extracted patch image. Here, the 3×3 area which is separated by the black dash lines corresponds to 1×1 pixel of the low resolution image which is not upscaled, and the number of interpolated pixel patterns is derived from the upscaling factor. In short, jaggy and ringing artifacts are occurred by bicubic interpolation, and the period of artifacts correspond to the positions of the interpolated pixel patterns defined by the upscaling factor. In other words, to reduce the artifacts, it is important to consider the interpolation pixel positions. As reported by Shi et al. [13] and Ohtani et al. [4], the SRCNN-based approach can improve the performance by using parallel independent CNNs to match the interpolation pixel position to each neuron. However, their approach requires multiple SRCNNs according to the upscaling factor. Therefore, problems remain in terms of structural complexity and implementation cost.

On the other hand, when we set the sliding step to the upscaling factor, each neurons are assigned to specific interpolation pixel positions. Figure 4 (b) shows an example of this situation with upscaling factor of 3. In the figure, the gray-boxes indicate the *P1* positions that are located at same positions in the 3×3 areas as same as Fig. 4 (a), and

the both patch of the solid line and the dash line have same interpolated pixel patterns. This position matching makes it possible to perform learning appropriate for each interpolation pixel positions, and it is expected to reduce the artifacts such as jaggy and ringing.

From the above viewpoint, the proposed method generates the learning dataset by the following steps.

Step 1. Generate a paired dataset of LR/HR

In order to obtain a paired dataset of HR and LR images, learning sample data are regarded as HR images, and the paired LR images are generated by degradation of the HR images. The degradation is achieved by downscaling and upscaling with the target scaling factor. And we select bicubic interpolation for the scaling processes as well as Yang et al. [7] and Dong et al. [2], [3]. Since the proposed method is a learning-based single image super-resolution, not a reconstruction-based, the low resolution image with intentional aliasing is unnecessary.

Step 2. Generate paired patch images

Paired patch size images are generated by cropping the dataset while scanning at a fixed interval that is the same value as the target scaling factor. Here, to match the sliding step for learning database creation and target scaling factor is important to get a good result. In Sect. 4.3, the effectiveness of this matching is described.

By preparing the training dataset by the above procedure, each pixel position in the patch image corresponds to the interpolation pixel position. Therefore, unnecessary variation of learning is suppressed.

In order to train SR-PDNN parameters, it is necessary to minimize the loss function, which is defined as follows:

$$Loss(\Theta) = \frac{1}{N_s}\sum_{n=1}^{N_s} \|PDNN(\mathbf{y_n};\Theta) - (\mathbf{x_n} - \mathbf{y_n})\|^2 \quad (8)$$

$$\Theta = \{W_{filter}, W_{convert}, W_{atom}\} \quad (9)$$

where N_s is defined as the number of samples in the dataset and Θ is the parameter set of all the matrices to be learned. Also, $PDNN(\mathbf{y_n};\Theta)$ is defined as the output of $PDNN(\mathbf{y_n})$ when the parameter Θ is set.

3.3 Super-Resolution

Since PDNN corresponds to a specific patch size, in order to achieve the super resolution for an arbitrary image size, it is necessary to decompose the image into patches and superimpose the respective processing results.

Here, we define $Y \in \mathbb{R}^{N_p}$ as a column vector of a input LR image of N_p pixels, R_p (p = 1, 2, · · · , $\mathrm{N_p}$) as an operator to extract the patch around pixel p, and y_p as a define as a patch image for pixel p. Then, the estimated HR patch image $\hat{x}_p$ is defined as follows:

$$y_p = R_p Y \quad (10)$$

$$\hat{x}_p = PDNN(y_p) + y_p \quad (11)$$

Then, the whole HR image $\hat{X}$ is calculated by averaging the overlapping regions between adjacent patches as follows:

$$\hat{X} = (\Sigma_p R_p^T R_p)^{-1} \Sigma_p R_p^T \hat{x}_p \quad (12)$$

As for Σ_p, each of the patch positions is determined by sliding step, and the sliding step for both horizontal and vertical directions is set to the upscaling factor. For example, in the case that the upscaling factor is 3, the all of gray boxes in Fig. 4 (b) correspond to the corner of the extracted patches as same as the patch positions in the figure. By doing this, the position of each patch image matches the pattern of the interpolation pixel as in learning. This is important to get a good result, the effectiveness of this pixel position matching is described in Sect. 4.3. In addition, because the proposed method is based on the square patch process, this averaging of adjacent patches by overlap is important to reduce the influence of the patch shape. Note that although the above equation is an expression of the entire image, when we design the hardware implementation, it can be processed in a local related area.

4. Experiments and Results

4.1 Basic Configuration

In order to achieve a fair comparison, *91 images* were used as the training dataset, and *Set 5* and *Set 14* were used as the test dataset† [3], [7]. The target scaling factor was set to 2, 3, and 4. In all of the experiments, we focused only on the luminance channel in the YCbCr space. The peak signal-to-noise ratio (PSNR) was calculated for objective evaluation.

As the parameter to specify the neural network, the input and output patch size was set to $n = 36(6\times6)$, $81(9\times9)$, and $144(12\times12)$ corresponding to upscaling factors of 2, 3, and 4, respectively. The number of filters was set to $f_l = 200$, and the number of atoms was set to $f_h = 800$.

In the training dataset generation, we set the sliding step to 2, 3, and 4, identical to the upscaling factor. We used all patterns that could be obtained from *91 images* as the dataset for each scaling factor. In particular, we prepared 4,916,516, 2,087,888, and 1,119,188 paired patches for each dataset corresponding to upscaling factors of 2, 3, and 4, respectively. Note that these numbers are the result of augmentation by rotations for 90°, 180°, and 270°.

For the training, we used the *mini-batch* approach. We randomly extracted 1,024 samples from the target training dataset for each process. As the number of training iterations, we used 5,000 epochs in total. Here, one epoch is equivalent to the number of times that the training set was turned around. In order to optimize the loss function, we used Adam Optimizer [19] with a learning rate of 0.0001. Before training, all of the learning targets $\{W_{atom}, W_{filter}, W_{convert}\}$ were initialized by the normal distribution with a zero mean and a standard deviation of 0.001.

†http://mmlab.ie.cuhk.edu.hk/projects/SRCNN.html

Table 1 Results for PSNR on *Set 5*.

image	scale	PSNR (dB)				
		Bicubic	ScSR	A+	SRCNN	**Ours**
baby	2	37.07	38.45	38.50	38.30	**38.53**
bird	2	36.81	40.56	41.10	40.64	**41.38**
butterfly	2	27.43	31.31	32.00	32.20	**33.48**
head	2	34.86	35.73	**35.80**	35.64	35.77
woman	2	32.14	34.95	35.30	34.94	**35.64**
average	2	33.66	36.20	36.55	36.34	**36.96**
baby	3	33.91	35.02	35.20	35.01	**35.28**
bird	3	32.57	34.35	35.50	34.90	**35.84**
butterfly	3	24.04	26.22	27.20	27.58	**28.67**
head	3	32.88	33.56	33.80	33.55	**33.80**
woman	3	28.56	30.33	31.20	30.91	**31.84**
average	3	30.39	31.90	32.59	32.39	**33.09**
baby	4	31.78	32.81	33.30	32.98	**33.30**
bird	4	30.18	31.55	32.50	31.97	**32.89**
butterfly	4	22.10	23.63	24.40	25.06	**25.79**
head	4	31.59	32.16	32.50	32.19	**32.52**
woman	4	26.46	27.66	28.60	28.20	**29.24**
average	4	28.42	29.56	30.29	30.08	**30.75**

Table 2 Results for PSNR on *Set 14*.

image	scale	PSNR (dB)				
		Bicubic	ScSR	A+	SRCNN	**Ours**
baboon	2	24.86	25.59	25.65	25.62	**25.74**
barbara	2	28.00	**28.70**	**28.70**	28.59	28.64
bridge	2	26.58	27.67	27.78	27.70	**27.91**
coastguard	2	29.12	30.58	30.57	30.49	**30.73**
comic	2	26.02	27.99	28.29	28.27	**28.80**
face	2	34.83	35.71	**35.74**	35.62	**35.74**
flowers	2	30.37	32.72	33.02	33.03	**33.55**
foreman	2	34.14	36.91	36.94	36.23	**37.05**
lenna	2	34.70	36.48	36.60	36.50	**36.72**
man	2	29.25	30.69	30.87	30.82	**31.07**
monarch	2	32.94	36.52	37.01	37.18	**38.29**
pepper	2	34.95	36.73	37.02	36.73	**37.08**
ppt3	2	26.87	29.52	30.09	30.40	**31.01**
zebra	2	30.63	33.37	33.59	33.29	**33.96**
average	2	30.23	32.08	32.28	32.18	**32.59**
baboon	3	23.21	23.54	23.60	23.60	**23.69**
barbara	3	26.25	**26.70**	26.50	26.66	26.58
bridge	3	24.40	25.02	25.20	25.07	**25.31**
coastguard	3	26.55	27.18	27.30	27.20	**27.34**
comic	3	23.12	24.04	24.40	24.39	**24.71**
face	3	32.82	33.52	**33.80**	33.58	33.79
flowers	3	27.23	28.51	29.00	28.97	**29.44**
foreman	3	31.16	33.19	34.40	33.40	**34.41**
lenna	3	31.68	33.04	33.50	33.39	**33.77**
man	3	27.01	27.91	28.30	28.18	**28.46**
monarch	3	29.43	31.30	32.10	32.39	**33.38**
pepper	3	32.38	33.81	34.70	34.34	**34.90**
ppt3	3	23.71	25.06	26.10	26.02	**26.75**
zebra	3	26.63	28.38	29.00	28.87	**29.64**
average	3	27.54	28.66	29.13	29.00	**29.44**
baboon	4	22.44	22.66	22.70	22.70	**22.77**
barbara	4	25.15	25.57	25.70	25.70	**25.87**
bridge	4	23.15	23.58	23.70	23.65	**23.91**
coastguard	4	25.48	25.65	25.90	25.94	**26.07**
comic	4	21.69	22.28	22.50	22.53	**22.80**
face	4	31.55	32.09	32.40	32.12	**32.51**
flowers	4	25.52	26.41	26.90	26.84	**27.30**
foreman	4	29.38	30.45	32.20	31.46	**32.49**
lenna	4	29.83	30.81	31.40	31.20	**31.63**
man	4	25.70	26.38	26.80	26.65	**26.96**
monarch	4	27.46	28.80	29.40	29.89	**30.48**
pepper	4	30.59	31.70	32.90	32.34	**33.12**
ppt3	4	21.98	22.71	23.60	23.84	**24.48**
zebra	4	24.08	25.38	25.90	25.97	**26.55**
average	4	26.00	26.75	27.32	27.20	**27.64**

4.2 Comparisons to State-of-the-Art Methods

The results of comparisons with conventional state-of-the-art SR methods are shown in Tables 1 and 2. In these tables, the left *image* column shows the name of the images and the *scale* column shows the upscale factor. And, the other columns show the PSNR values against HR images. In particular, *bicubic* is same as input of proposed method and *ours* is the result of proposed method. For the conventional methods, we selected ScSR [7], A+ [20], and SRCNN [3]. Regarding implementation, we used publicly available codes provided by the authors. The performance of the proposed method was found to be highest for most images. In particular, in comparison with the SRCNN, the average PSNR increased approximately 0.6 dB in *Set 5* and approximately 0.4 dB in *Set 14* for all scaling factors.

The visual effects of these performance improvements are shown in Fig. 5. The six images of Fig. 5 (c) are the results of "butterfly" (*Set5*) with an upscaling factor of 4, and the white rectangle of Fig. 5 (a) indicates the cropped region. Also, the six images of Fig. 5 (d) are the results of "comic" (*Set14*) with an upscaling factor of 3, and the white rectangle of Fig. 5 (b) indicates the cropped region. The circle marked areas of Fig. 5 (a) and Fig. 5 (b) indicate the observed locations for jaggy and ringing. Specifically, ringing artifacts can be seen at the circle 1 area and jaggy artifacts can be seen at the circle 2 and 3 areas. As seen in the figures, the ScSR results contain strong jaggy and ringing. The A+ results are better than ScSR, but the ringing still remains especially in circle 1 area. The SRCNN results still contain jaggy and ringing, obviously. In contrast, our results remove the most of jaggy and ringing artifacts, and the visual improvement can be observed at all of the circle areas. As is well known, subjective image quality is not in proportion to PSNR. In fact, in terms of the butterfly images, the PSNR of the SRCNN is higher than that of the A+, as shown in Table 1. However the A+ result seems better in visual than the SRCNN result. Similarly, our PSNR improvement is not high, but the visual improvement is obvious.

4.3 Effectiveness of Fitting to Interpolation Patterns

As described in Sect. 3, it is important to fit the patch image extraction position to interpolation pixel position in both restoration and learning phases.

Table 3 shows the change of performance against the sliding steps of learning and restore. Where s_r represents the sliding step for restoration and s_l represents the sliding step for learning. And all values are average PSNR on *Set5* with an upscaling factor of 3. The other parameters are the same as those in Sect. 4.1. Note that the patch size is 9×9 pixels in this case, and the sliding step for restoration defines the overlapping width. For example, when we set $s_r = 9$, there is no overlap.

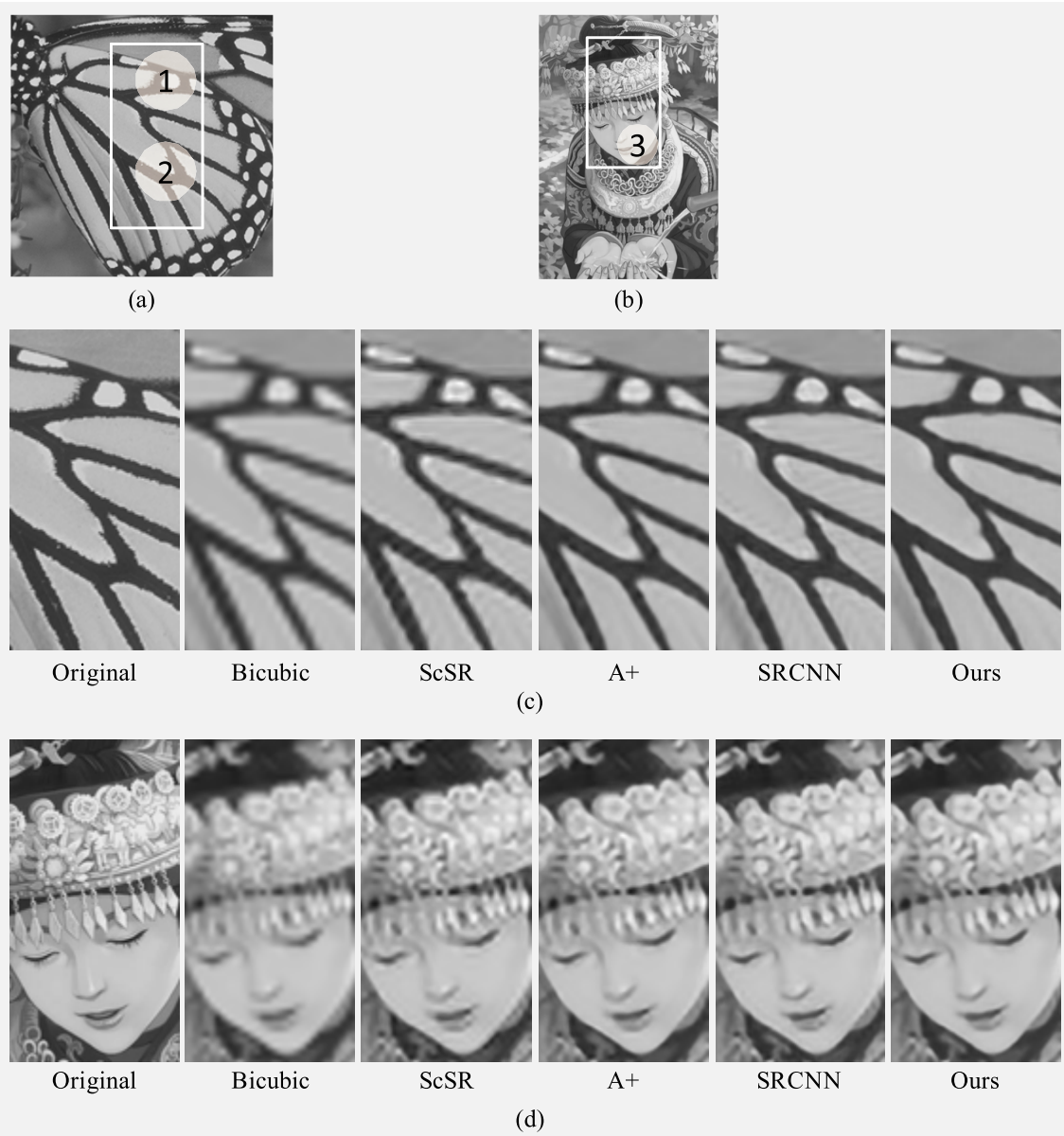

Fig. 5 The comparison of subjective image quality. (a) Thumbnail of "butterfly" (*Set5*). (b) Thumbnail of "comic" (*Set14*). (c) Results for "butterfly" (*Set5*) with an upscaling factor of 4. (d) Results for "comic" (*Set14*) with an upscaling factor of 3.

In Table 3, the best performance was observed when we set $(s_l, s_r) = (3, 3)$. It means the effectiveness of matching the positions of the learning and the restoration was confirmed.

In particular, Fig. 6 shows the case of $s_l = 3$ in Table 3. Also in the case of $s_r = 6$ or 9, the PSNR becomes high scores because the positions are also matched in these cases. However, due to the reduction of overlapping effect, the values are less than the case of $s_r = 3$. In addition, when the case of $s_r = 1$, 2, 4, 5, 7, and 8, the performance drops remarkably. This is because of the mismatch about the positions of learning and restoration.

On the other hand, in the case where the $s_l = 1$ or 2, since all the interpolation patterns are learned to each neu-

Table 3 The change of performance against the sliding steps of learning (s_l) and restore (s_r).

s_r	PSNR (dB)		
	$s_l = 1$	$s_l = 2$	$s_l = 3$
1	32.98	32.93	30.77
2	32.97	32.91	30.68
3	33.01	32.94	**33.09**
4	32.92	32.85	29.47
5	32.85	32.79	29.92
6	32.86	32.82	32.94
7	32.73	32.67	28.12
8	32.64	32.57	28.16
9 (without overlapping)	32.62	32.55	32.75

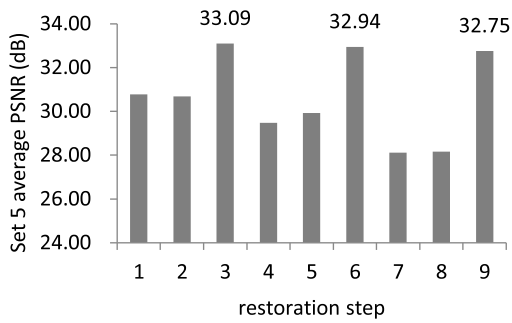

Fig. 6 The relation between averages PSNR (dB) on *Set5* with upscaling factor of 3 and sliding step for restoration (s_r) when the sliding step for learning (s_l) is set as 3.

ron, the influence of the s_r setting is reduced. However, since the variations of different interpolation pixel patterns are included, the performance is lower than the matched case.

From the above results, it was confirmed that if the patch extraction position of learning and restoration is set to match, the performance of image quality improves. This effect also contributes to the reduction of artifacts such as jaggy and ringing, and the effect can be visually confirmed from Fig. 5.

4.4 Cost Performance Comparison in Real-Time Processing

Since both the SR-PDNN and SRCNN are based on feed-forward processing, fixed time hardware processing can theoretically be realized. However, in order to actually design the hardware, it is necessary to consider the process delay with respect to the input and the implementation cost.

4.4.1 Hardware Implementation Cost of the SRCNN

In order to specify the processing delay, it is necessary to consider that the real-time video stream is input as a raster scan. When we regard the video stream image itself as an input image to the super resolution system, frame delay must occur because the conventional SRCNN uses the filtered result of the entire image as the intermediate result. Therefore, in order to realize the non-frame delay system, we need to divide the video stream image into small images as input for the SRCNN, in the manner of Kim et al. [17].

On the other hand, in order to estimate the hardware cost, parameters such as filter size, number of features, and input image size are needed. Dong et al. [3] proposed two models in their paper: SRCNN (9-1-5) for cost, and SRCNN (9-5-5) for performance. Using these models, the minimum hardware cost required in order to obtain one pixel output was calculated as shown in Table 4.

Here, Table 4 shows that SRCNN (9-1-5) requires a 13×13 sub image as input, 181,600 multiplication steps, and 2,569 *pixels* of memory for saving the intermediate results. In the table, a *pixel* denotes a virtual unit for saving one value.

Table 4 SRCNN hardware implementation cost. Here, PSNR refers to the *Set 14* average PSNR with an upscaling factor of 2.

Model	Input	Memory	Multi.	PSNR
SRCNN (9-1-5)	13×13	2,569 *pixels*	181,600 steps	32.18 dB
SRCNN (9-5-5)	17×17	6,273 *pixels*	1,700,704 steps	32.45 dB

Table 5 SR-PDNN hardware implementation cost. Here, PSNR refers to the *Set 14* average PSNR with an upscaling factor of 2.

Model	Input	Memory	Multi.	PSNR
$f_h = 36$	6×6	272 *pixels*	3,924 steps	32.26 dB
$f_h = 200$	6×6	436 *pixels*	13,600 steps	32.52 dB
$f_h = 800$	6×6	1,036 *pixels*	49,000 steps	32.59 dB

In SRCNN (9-5-5), although the memory cost is drastically increased, the performance improvement is only a 0.27 dB increase. Note that SRCNN (9-1-5) was trained using the *91 images* in Sect. 4.1, whereas SRCNN (9-5-5) used 395,909 images obtained from ImageNet [21].

4.4.2 Hardware Implementation Cost of the SR-PDNN

In order to specify the implementation cost of the proposed method, the size of the input/output patch images and the matrices should be defined. Here, in order to allow a fair comparison, we determined the performance change for the proposed method when the number of atoms (f_h) was varied. Figure 9 shows dependence of the performance on the number of atoms. The PSNR increased with increasing f_h, and it overtook the level of SRCNN (9-1-5) at $f_h = 36$, and of SRCNN (9-5-5) at $f_h = 200$. Note that all of the experimental conditions were as described in Sect. 4.1 with an upscaling factor of 2, except for the number of atoms.

Based on the above results, we summarized the relationship between the cost and performance of the proposed method, as shown in Table 5. Comparing Tables 4 and 5, the proposed method has better cost performance than the SRCNN. Specifically, in the case of $f_h = 36$, the model corresponding to SRCNN (9-1-5), the required memory cost is reduced by 89% and the accumulation number is reduced by 98%.

5. Discussion

In the following, we describe another advantage of the proposed method, namely an ease method of cost performance adjustment. First, the results of learning atoms and filters with an upscaling factor of 2 and $f_h = 200$ are shown in Figs. 7 and 8. As shown in the figures, these atoms and filters can be expressed as images because their size is the same as that for the input/output patch image.

The atoms in Fig. 7 were arranged in order of the sum of absolute values for each atom. The larger value means the more significant atom that contributes to the output. Figure 10 shows the distribution for the case of $f_h = 36$, 200, and 800. As a trend, the number of *valid* atoms increased with f_h. However, in the case of $f_h = 800$, the number

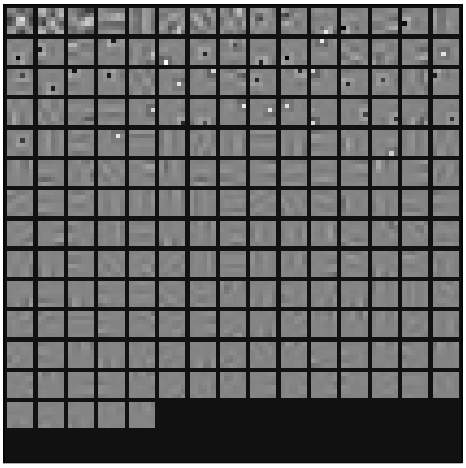

Fig. 7 Learned atoms in the restoration layer (f_h = 200).

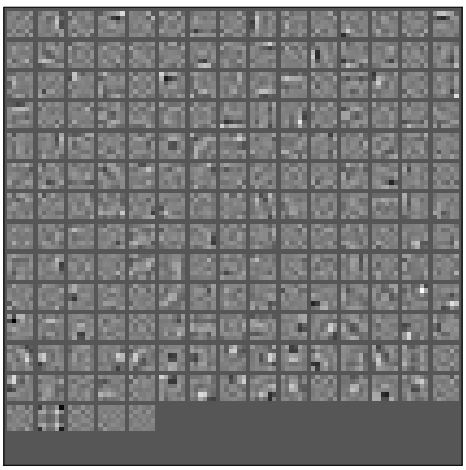

Fig. 8 Learned filters in the feature extraction layer (f_l = 200).

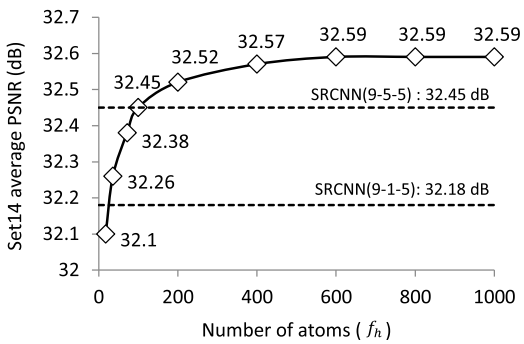

Fig. 9 Relationship between f_h and performance. The PSNR is the *Set 14* average PSNR with an upscaling factor of 2, and the points denote f_h = 18, 36, 72, 100, 200, 400, 600, 800, and 1000, respectively.

of *valid* atoms did not increase beyond approximately 600. This means that a performance improvement cannot be expected, even if f_h is further increased, which agrees with the results shown in Fig. 9.

In other words, the proposed method can determine whether the performance limit has been reached by checking

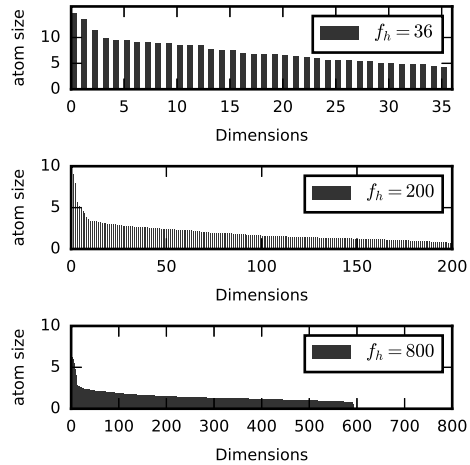

Fig. 10 Number of *valid* atoms for various f_h. The *atom size* means the sum of the absolute values for each atom.

the number of *valid* atoms. In addition, when the network has *invalid* atoms, as in the case of f_h = 800, cost reduction can be achieved without re-learning by simply deleting the components of W_{atom} and $W_{convert}$ corresponding to the unnecessary atoms.

In this way, an advantage of the PDNN is that the cost can be easily reduced after learning. However, in the case of the CNN, it is difficult to separate each layer after learning. Moreover, this approach also suggests that it is possible to adjust the cost performance balance based on the *valid* atom. This is a new approach to adjusting cost performance, and there is still potential for practical applications.

6. Conclusion

In the present paper, we proposed a novel high-performance and cost-effective super-resolution via patch-based deep neural network (SR-PDNN).

Using a PDNN-based architecture suitable for super resolution and learning considering interpolated pixel positions, a performance increase of approximately 0.6 dB for *Set 5* and approximately 0.4 dB for *Set 14* was obtained, as compared to SRCNN.

The SR-PDNN is also suitable for real-time hardware processing, for example, using an ASIC or an FPGA. When the performance is comparable to that of the SRCNN, the memory cost may be reducible by 89% and the multiplication logic cost may be reducible by 98%.

In the future, it will be necessary to investigate the multilayering of the estimation. Moreover, the calculation accuracy, including the transfer function, should be investigated in order to realize hardware implementation.

References

[1] C.-Y. Yang, C. Ma, and M.-H. Yang, "Single-image super-resolution: A benchmark," European Conference on Computer Vision, vol.8692, pp.372–386, Springer, 2014.

[2] C. Dong, C.C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," European Conference

on Computer Vision, vol.8692, pp.184–199, Springer, 2014.
[3] C. Dong, C.C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," IEEE Trans. Pattern Anal. Mach. Intell., vol.38, no.2, pp.295–307, 2016.
[4] S. Ohtani, Y. Kato, N. Kuroki, T. Hirose, and M. Numa, "Multichannel convolutional neural networks for image super-resolution," IEICE Trans. Fundamentals, vol.E100-A, no.2, pp.572–580, 2017.
[5] W.T. Freeman, T.R. Jones, and E.C. Pasztor, "Example-based super-resolution," IEEE Computer graphics and Applications, vol.22, no.2, pp.56–65, 2002.
[6] J. Yang, J. Wright, T. Huang, and Y. Ma, "Image super-resolution as sparse representation of raw image patches," Computer Vision and Pattern Recognition, 2008, CVPR 2008, IEEE Conference on, pp.1–8, IEEE, 2008.
[7] J. Yang, J. Wright, T.S. Huang, and Y. Ma, "Image super-resolution via sparse representation," IEEE Trans. Image Process., vol.19, no.11, pp.2861–2873, 2010.
[8] Y.C. Pati, R. Rezaiifar, and P.S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," Signals, Systems and Computers, 1993, 1993 Conference Record of The Twenty-Seventh Asilomar Conference on, pp.40–44, IEEE, 1993.
[9] A. Kappeler, S. Yoo, Q. Dai, and A.K. Katsaggelos, "Video super-resolution with convolutional neural networks," IEEE Transactions on Computational Imaging, vol.2, no.2, pp.109–122, 2016.
[10] C. Dong, X. Zhu, Y. Deng, C.C. Loy, and Y. Qiao, "Boosting optical character recognition: A super-resolution approach," arXiv preprint arXiv:1506.02211, 2015.
[11] R. Timofte, R. Rothe, and L.V. Gool, "Seven ways to improve example-based single image super resolution," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp.1865–1873, 2016.
[12] J. Kim, J.K. Lee, and K.M. Lee, "Accurate image super-resolution using very deep convolutional networks," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp.1646–1654, 2016.
[13] W. Shi, J. Caballero, F. Huszár, J. Totz, A.P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp.1874–1883, 2016.
[14] Y. Romano, J. Isidoro, and P. Milanfar, "Raisr: rapid and accurate image super resolution," IEEE Transactions on Computational Imaging, vol.3, no.1, pp.110–125, 2017.
[15] X. Feng and P. Milanfar, "Multiscale principal components analysis for image local orientation estimation," Signals, Systems and Computers, 2002, Conference Record of the Thirty-Sixth Asilomar Conference on, pp.478–482, IEEE, 2002.
[16] K. Zeng, J. Yu, R. Wang, C. Li, and D. Tao, "Coupled deep autoencoder for single image super-resolution," IEEE Trans. Cybern., vol.47, no.1, pp.27–37, 2017.
[17] S.Y. Kim and P. Bindu, "Realizing real-time deep learning-based super-resolution applications on integrated gpus," Machine Learning and Applications (ICMLA), 2016 15th IEEE International Conference on, pp.693–696, IEEE, 2016.
[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp.770–778, 2016.
[19] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.
[20] R. Timofte, V. De Smet, and L. Van Gool, "A+: Adjusted anchored neighborhood regression for fast super-resolution," Asian Conference on Computer Vision, pp.111–126, Springer, 2014.
[21] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," Computer Vision and Pattern Recognition, 2009, CVPR 2009, IEEE Conference on, pp.248–255, IEEE, 2009.

Reo Aoki was boarn in Gifu, Japan, on Jun 28, 1981. He received the B.E. from Tokyo Metropolitan Institute of Technology in 2004, and M.S. from Japan Advanced Institute of Technology in 2006. Since 2006, He has been in EIZO corporation, Hakusan, Japan. Since 2016, he has also been in the doctoral course of Natural Science and Technology of Kanazawa University.

Kousuke Imamura received the B.E., M.E. and Dr. Eng. degrees in Electrical Engineering and Computer Science in 1995, 1997 and 2000, respectively, all from Nagasaki University. He is currently an Associate Professor in the Institute of Science and Engineering at Kanazawa University. His research interests include video/image processing.

Akihiro Hirano received the B. Eng., M. Eng. and Dr. Eng. degrees from Kanazawa University, Kanazawa, Japan in 1987, 1989 and 2000, respectively. He was affiliated with NEC Corporation, Kawasaski, Japan from 1989 to 1998, where he had been a Research Engineer in the Research and Development Group. He joined Faculty of Engineering, Kanazawa University in 1998. He is currently an Assistant Professor in Graduate School of Natural Science & Technology, Kanazawa University. He has been engaged in researches on adaptive signal processing and neural networks. He was awarded the 1995 Academic Encouragement Award by IEICE. Dr. Hirano is a member of the *Institute of Electrical and Electronics Engineers* (IEEE).

Yoshio Matsuda was born in Ehime, Japan, on October 26, 1954. He received the B.S. degree in physics and the M.S. and Ph.D. degree in applied physics from Osaka University in 1977, 1979, and 1983, respectively. He joined the LSI Laboratory, Mitsubishi Electric Corporation, Itami, Japan, in 1985. He was engaged in development of DRAM, advance CMOS logic, and high frequency devices and circuits of compound semiconductors. Since 2005, he has been a professor of Institute of Science and Engineering at Kanazawa University, Japan. His research is in the fields of integrated circuits design where his interests include multimedia system, low power SoCs, and image compression processors.