Improvement of pixel enhancement algorithm for high-speed camera imaging using 3D sparsity

Naoki Nogami*, Akira Hirabayashi*, Jeremy White* and Laurent Condat†
* College of Information Science and Engineering, Ritsumeikan University, Kusatsu, Japan
E-mail: akirahrb@media.ritsumei.ac.jp Tel: +81-77-507-0410
† University of Grenoble Alpes GIPSA-lab, F38000 Grenoble, France
laurent.condat@gipsa-lab.grenoble-inp.fr

***Abstract*—We improve the performance of a pixel enhancement algorithm for high-speed camera (HSC) imaging. HSCs have a principle problem that the number of pixels decreases when the number of frames per second (FPS) increases. To suppress this problem, our optical setup is organized with a digital mirror device (DMD) array to randomly select pixels in each frame. A small number of selected pixels are recorded by an image sensor. Then, our algorithm reconstructs the entire image only from those randomly selected pixels by exploiting not only the sparsity within the each frame, but also that of difference image between adjacent frame. In this paper, we improve the performance of the algorithm in the sense of two aspects. First, we improve the accuracy of the proposed algorithm by exchanging the role of two functions in the convex optimization algorithm. Further, we accelerate the algorithm by setting a better initial value. Simulation results show that the reconstructed image quality is slightly improved and the algorithm is accelerated by several percent.**

I. Introduction

A recent popular imaging tool is high-speed camera, which are capable of capturing images more than one hundred frames per second (fps). That enables us to observe things that are too fast to see for human eyes. Major fields that exploit high-speed cameras include engineering measurements, sports training, and entertainment. Casual use is also getting popular because "iPhone 6" series and "Go Pro Hero 4" are able to capture images at 240 and 120 fps, respectively.

One issue of high speed cameras is the decrease of pixels when fps increases. This is a principle problem, irrespective on products or companies. The reason of this phenomenon is that time for swipe out is proportional to the number of image pixels while the increase of fps number suppresses the time for swipe out. Our goal is to keep the number of pixels as high as possible even when fps increases. To this end, we suppose an optical setup shown in Fig. 1. The digital mirror devise (DMD) array randomly selects a small number of pixels, say 25%, and these pixels are recorded by the image sensor equipped above. This is a multiple pixel version of the so-called single pixel camera [1]. Then, an image processing technique recovers the entire original image by filling in the missing pixels. Many methods relevant to this recovery problem have been proposed so far [2]–[7]. Wakin *et al.* regarded this problem as a three dimensional sensing problem [2]. It is, however, difficult to implement such a sensing mechanism in high speed cameras. Kang and Lu exploited similarities between adjacent frames

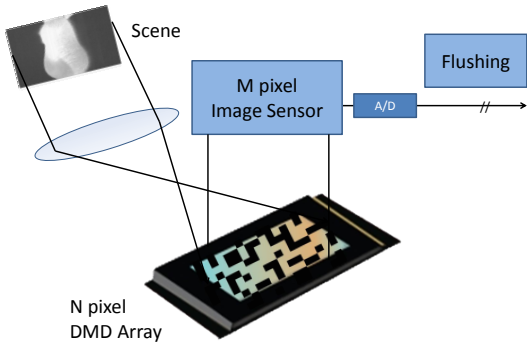


Fig. 1. Proposed optical setup for pixel enlargement in high speed camera image acquisition.

[3]. They changed the compression rate depending on key or non-key frames. However, such change of compression rate is difficult in high speed cameras because of a limited swipe out time. Vaswani proposed methods supposing that sparsity pattern (support of the sparsifying transform vector) changes slowly over time [4], [5]. It is, however, difficult to choose appropriate thresholds for adding and deleting a small part of the support. Another approach is based on dictionary learning [6]. This approach requires a very high computational cost. To reduce the cost, images are divided into small patches, which results in block noise and occasionally the need for post processing [8], [9]. Chang *et al.* treated a sequence as a long vector [7]. This approach also results in a huge computational cost.

The present authors have proposed a pixel enhancement algorithm that quickly reconstructs the entire image frame by frame without changing the compression rate. This algorithm exploits sparsity of two types. One is that within each frame. The second is sparsity of difference image between adjacent frames. As is well-known, sparsity is promoted by the ℓ_1 norm. Hence, promotion of two types of sparsity is formulated as a cost function of the sum of two ℓ_1 norms of coefficients of sprarsifying transform and the difference of adjacent frames. By minimizing this cost function under the observation constraint, image sequence is reconstructed. The minimization problem is efficiently solved by a convex optimization technique of the Douglas-Rachford splitting (DRS)

algorithm [10].

In this paper, we further improve the performance of the algorithm in the sense of two aspects. First, we improve the accuracy of the proposed algorithm by exchanging the role of two functions in the convex optimization algorithm. In theory, the treatment of these two functions does not affect the results of the algorithm. In practice, however, the previous assignment of the two functions in the DRS algorithm enforced the observation constraint only approximately. If we replace the assignment, the constraint is enforced rigorously at each step. This improves the quality of the image at each step, and thus the final result too. Second, in the previous algorithm, the zero vector was set for the initial value. By changing this by a more appropriate vector, we accelerate the algorithm. We show the effectiveness of the proposed two improvements by computer simulations.

The rest of the present paper is organized as follows. In Section 2, we formulate the image reconstruction problem and review the previously proposed algorithm. Section 3 proposes revisions of the proposed algorithm and shows its effectiveness by simulations. Section 4 evaluates robustness of the proposed algorithm by changing the compression rate and the number of frames per second. Section 5 concludes the paper.

II. Problem Formulation

Suppose that a high speed camera captures a scene at a high frame rate and a sequence of images $\boldsymbol{x}_r \in \mathbb{R}^N$ $(r = 1, \ldots, R)$ is obtained[1]. Pixels in the rth image $\boldsymbol{x}_r$ are randomly selected in the following manner: each image is divided into local blocks of 2×2 or 4×4 pixels and several pixels out of each block are selected randomly. The number of remaining pixels for the entire image is denoted by M. Thus, the compression rate is M/N. Let A_r and $\boldsymbol{y}_r \in \mathbb{R}^M$ be a random selection matrix that reflects the aforementioned manner and a vector consisting of the selected pixels. Then, it holds that

$$\boldsymbol{y}_r = A_r \boldsymbol{x}_r, \qquad (r = 1, 2, \ldots, R). \tag{1}$$

Note that the random selection pattern in A_r is generated at every frame, not fixed. By selecting pixels randomly, instance nullspace property is satisfied with high probability. Hence, a good accuracy is achieved. Our goal is to estimate the image sequence $\{\boldsymbol{x}_r\}_{r=1,\ldots,R}$ from $\{A_r\}_{r=1,\ldots,R}$ and $\{\boldsymbol{y}_r\}_{r=1,\ldots,R}$. Because of this goal, we do not take blur nor noise into account. We solved this problem by using two priors. First, we suppose that each captured image is sparse in an appropriate sparsifying transform domain, such as discrete wavelet or cosine transform domains. In simulations, we adopted discrete cosine transform (DCT) for the sparsifying transform since the target image sequence is about a natural scene. Second, because of the high frame rate, the difference between adjacent frames is small. Further, if only a small part in the scene is moving and other objects do not move so much, then the difference is not only small, but also sparse. Based on these two assumptions, we reconstruct the image $\boldsymbol{x}_r$ by using the following cost function for the DCT coefficient vector $\boldsymbol{u} = (u_n) \in \mathbb{R}^N$:

$$\hat{\boldsymbol{u}}_r = \underset{A_r C^T \boldsymbol{u} = \boldsymbol{y}_r}{\arg\min} \{ \|\boldsymbol{u}\|_1 + \lambda \|C^T \boldsymbol{u} - \hat{\boldsymbol{x}}_{r-1}\|_1 \} \ (r = 2, \ldots, R), \tag{2}$$

where $\|\cdot\|_1$ is the ℓ_1 norm of the corresponding vector and C^T is the transpose of the two-dimensional DCT matrix C, thus the inverse transform. The rth frame $\boldsymbol{x}_r$ is then estimated by $\hat{\boldsymbol{x}}_r = (\hat{x}_{r,n}) = C^T \hat{\boldsymbol{u}}_r$. For $r = 1$, we obtain $\hat{\boldsymbol{u}}_1$ by setting 0 for λ, thus (2) amounts to the standard ℓ_1 norm minimization as in the compressed sensing [11].

The problem (2) was solved by using the Douglas-Rachford splitting (DRS) algorithm [10]. Let S be a set of $\boldsymbol{u}$ satisfying $A_r C^T \boldsymbol{u} = \boldsymbol{y}_r$. This is a convex set. Then, (2) is equivalent to

$$\hat{\boldsymbol{u}}_r = \underset{\boldsymbol{u} \in \mathbb{R}^N}{\arg\min} \{ \|\boldsymbol{u}\|_1 + \lambda \|C^T \boldsymbol{u} - \hat{\boldsymbol{x}}_{r-1}\|_1 + \imath_S(\boldsymbol{u}) \}, \tag{3}$$

where $\imath_S(\boldsymbol{u})$ is the indicator function that takes value 0 if $\boldsymbol{u} \in S$, $+\infty$ else. Now, our problem becomes the minimization of the sum of three convex terms, which are not differentiable but proximable. As is well known, the proximity operator of the first term $\|\boldsymbol{u}\|_1$ is $\mathrm{prox}_{\theta\|\cdot\|_1}(\boldsymbol{u}) = (\mathrm{softthreshold}(u_n, \theta)) \in \mathbb{R}^N$ with $\theta = 1$, where

$$\mathrm{softthreshold}(u, \theta) = \begin{cases} u - \theta & \text{if} \quad u \geq \theta, \\ u + \theta & \text{if} \quad u \leq -\theta, \\ 0 & \text{if} \quad -\theta < x < \theta. \end{cases} \tag{4}$$

Let us denote the sum of the second and the third terms by $g(\boldsymbol{u})$. The proximity operator of $g(\boldsymbol{u})$ can be computed by the following operations. First, apply the inverse two-dimensional DCT to $\boldsymbol{u}$ as $C^T \boldsymbol{u} \equiv \boldsymbol{x} = (v_n) \in \mathbb{R}^N$. Then, for the pixels in the mask A_r, replace the values v_n by the the corresponding element of $\boldsymbol{y}_r$. For the other pixels, apply the operation

$$v_n \leftarrow \mathrm{softthreshold}(v_n - \hat{x}_{r-1,n}, \lambda) + \hat{x}_{r-1,n}. \tag{5}$$

Finally, apply the two-dimensional DCT to the updated vector $\boldsymbol{x}$ for returning to the DCT domain. These operations result in $\mathrm{prox}_g(\boldsymbol{u})$.

Based on these observations, the problem (2) was solved as follows. First, (3) is viewed as

$$\hat{\boldsymbol{u}}_r = \underset{\boldsymbol{u} \in \mathbb{R}^N}{\arg\min} \{ \|\boldsymbol{u}\|_1 + g(\boldsymbol{u}) \},$$

where both terms are non-differentiable but proximal. Then, the problem (2) was solved by the following DRS algorithm:

Algorithm 1: Image recovery for rth frame

Input: $\boldsymbol{y}_r$, A_r, $\hat{\boldsymbol{x}}_{r-1}$
Output: $\hat{\boldsymbol{x}}_r$
1. Set $\gamma > 0$, $\delta \in]0, 2[$
2. Set zero vector for $\boldsymbol{v}$ as an initial value.
3. Repeat the following two operations:
   $\boldsymbol{u}_r \leftarrow \mathrm{prox}_{\gamma\|\cdot\|_1}(\boldsymbol{v})$,
   $\boldsymbol{v} \leftarrow \boldsymbol{v} + \delta\{\mathrm{prox}_{\gamma g}(2\boldsymbol{u}_r - \boldsymbol{v}) - \boldsymbol{u}_r\}$,
   until a stopping condition is met.
4. Compute $\hat{\boldsymbol{x}}_r = C^T \boldsymbol{u}_r$

[1]Images are raster scanned and regarded as column vectors.

The proximity operator $\mathrm{prox}_{\gamma g}$ is computed by replacing λ in (5) by $\gamma\lambda$. For simple presentation, we explained the algorithm using the raster scan, which results in huge sensing and DCT matrices. We implemented however, programs by Matlab exploiting a two-dimensional expression to reduce both computational time and memory use. Thus, we can compute relatively high dimensional images, say 256×256, which was not computed by the method in [5] provided from [12].

III. Improvement of Algorithm

To further improve the performance of the algorithm, we revise Algorithm 1 in the sense of two aspects. First, we exchange the assignment of two proximity operators in the algorithm. In theory, the treatment of these two functions does not affect the results of the algorithm. In practice, however, the assignment in Algorithm 1 did not completely guarantee the observation constraint at each step. By exchanging the assignment, the constraint is enforced rigorously at each step and directly onto the output variable $\boldsymbol{u}_r$ of the algorithm. This improves the quality of the image at each step, and thus the final result too. Second, in Algorithm 1, the zero vector was set as the initial value for the parameter $\boldsymbol{v}$. Instead, we use a DCT coefficient vector $\boldsymbol{v}_0$ for the image in which the missing pixels are filled by the observed pixel in the same block. This image should be closer to the target image than the black (all zero) image. Hence, we can expect a faster convergence than using the black image. Our algorithm is finally updated in the following form:

Algorithm 2: Image recovery for rth frame

Input: $\boldsymbol{y}_r$, A_r, $\hat{\boldsymbol{x}}_{r-1}$
Output: $\hat{\boldsymbol{x}}_r$
1. Set $\gamma > 0$, $\delta \in]0, 2[$
2. Compute the vector $\boldsymbol{v}_0$
3. Set $\boldsymbol{v}_0$ for $\boldsymbol{v}$ as an initial value.
4. Repeat the following two operations:
   $\boldsymbol{u}_r \leftarrow \mathrm{prox}_{\gamma g}(\boldsymbol{v})$,
   $\boldsymbol{v} \leftarrow \boldsymbol{v} + \delta\{\mathrm{prox}_{\gamma\|\cdot\|_1}(2\boldsymbol{u}_r - \boldsymbol{v}) - \boldsymbol{u}_r\}$,
   until a stopping condition is met.
5. Compute $\hat{\boldsymbol{x}}_r = C^T\boldsymbol{u}_r$

To evaluate the performance of the algorithm, we conducted simulations. Two image sequences were captured by a high speed camera, Optronis CR450x3, nac Image Technology. One is a water balloon bursting scene and the other is the moment of impact between a tennis racket and a ball. For both sequences, 210 frames of uncompressed 256×256 images at 6,000 fps were obtained.

Pixels in these images were selected in the aforementioned manner. Here, one pixel was selected out of 2×2 pixel blocks. Thus, the compression rate for this simulation is 25%. Figs. 4 and 5 show examples of reconstructed images by Algorithm 2 with $\lambda = 0.5$ and 0, respectively. Figs. 2 and 3 show the peak signal to noise ratio (PSNR) in dB of the reconstructed image $\hat{\boldsymbol{x}}_r$ with respect to the frame number r, defined as

$$\mathrm{PSNR}_r = 20\log_{10}\frac{255\sqrt{N}}{\|\hat{\boldsymbol{x}}_r - \boldsymbol{x}_r\|_2}[\mathrm{dB}],$$

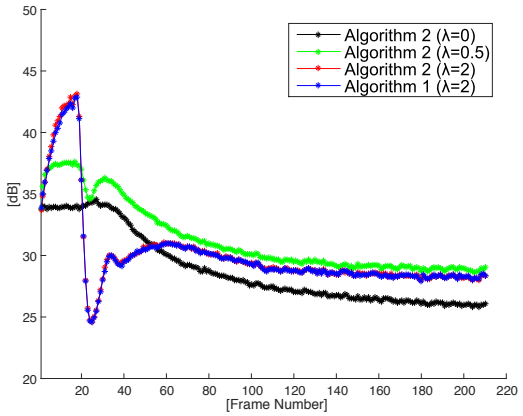


Fig. 2. PSNR [dB] of the reconstructed images about a water balloon bursting scene.

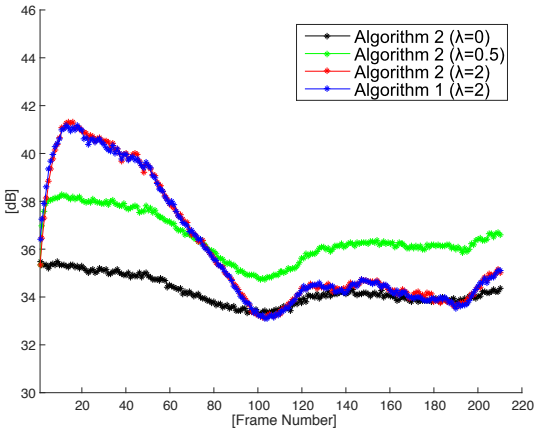
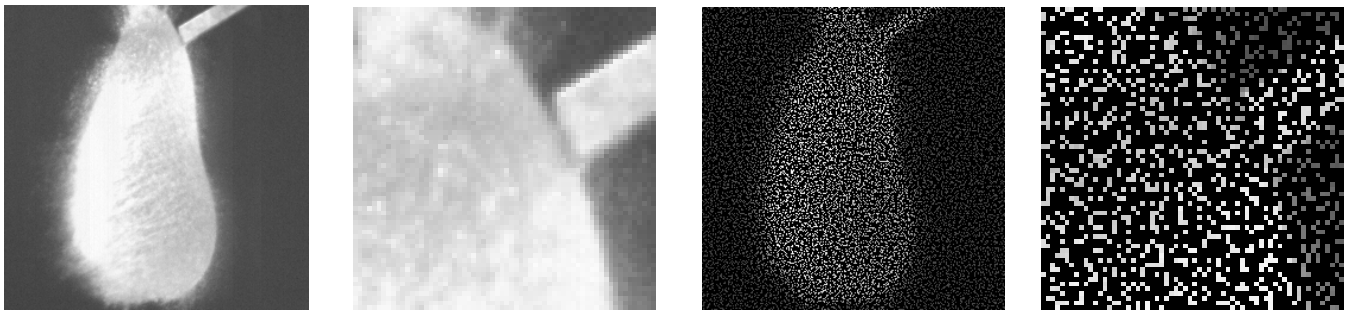
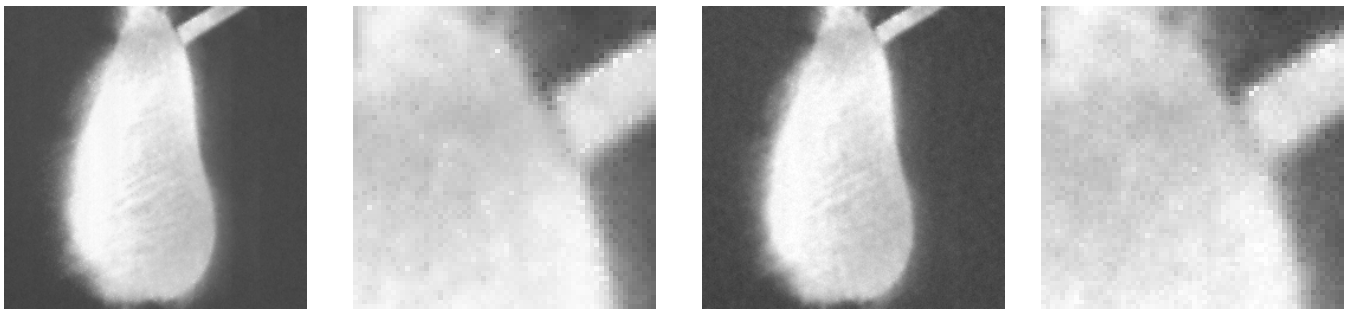


Fig. 3. PSNR [dB] of the reconstructed images about the moment of impact by a tennis racket and a ball.

where N is the number of pixels. The red line shows PSNR_r of the reconstructed images by the revised algorithm (Algorithm 2) with $\lambda = 2$ while the blue line shows PSNR_r of the reconstructed images by Algorithm 1 with $\lambda = 2$. The averages of PSNRs were 30.21dB and 36.33dB for Figs. 2 and 3, respectively, when images were reconstructed by Algorithm 2. On the other hand, the averages were 30.15dB and 36.29dB for Figs. 2 and 3, respectively, when images were reconstructed by Algorithm 1. This means that the image quality was improved by 0.06dB and 0.04dB, respectively. The green line shows PSNR_r of reconstructed images by Algorithm 2 with $\lambda = 0.5$. The averages are 31.31dB and 36.46dB for Figs. 2 and 3, respectively. The maximum value with $\lambda = 2$ is better than that with $\lambda = 0.5$, but the average is 1.1dB better with $\lambda = 0.5$ than the Algorithm 2 with $\lambda = 2$ on Fig. 2. The black line shows PSNRs obtained by Algorithm 2 with $\lambda = 0$, which means that images are reconstructed without referring the previous frame. The average of PSNRs were 28.92dB and 34.42dB for Figs.

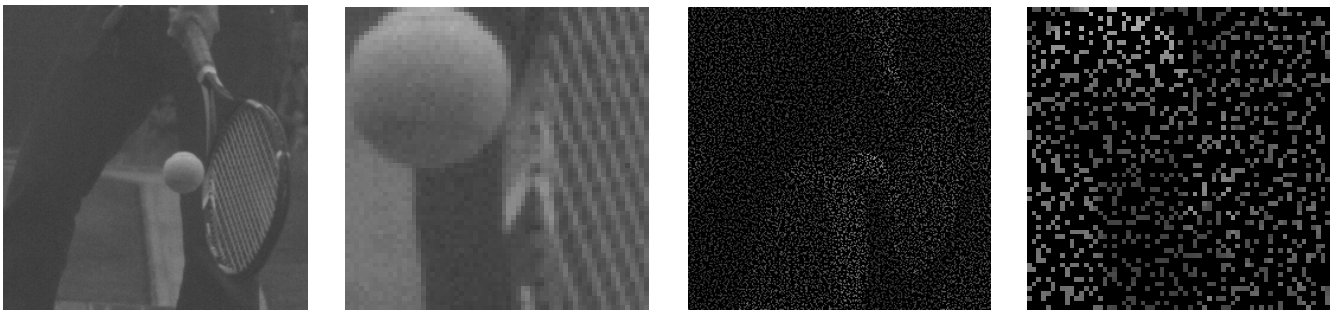


Original image Block random observation

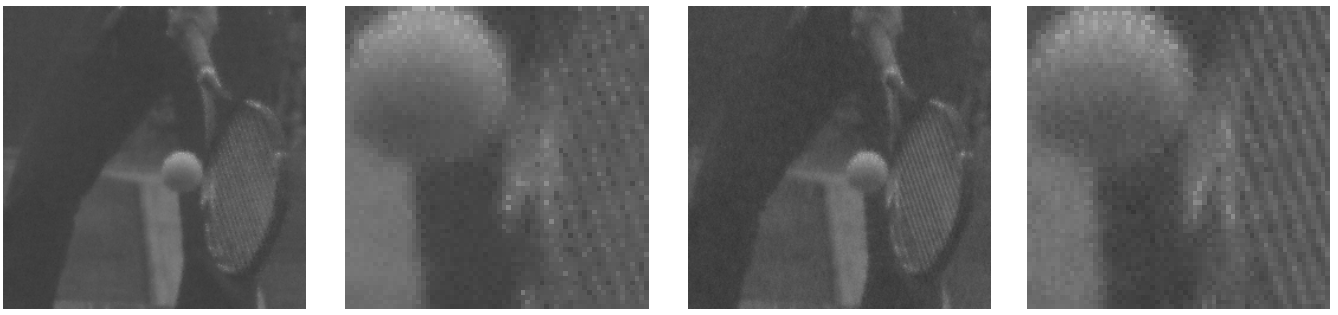


Reconstructed image of Algorithm 2 with λ = 0.5 (34.92dB) Reconstructed image of Algorithm 2 with λ = 0 (33.05dB)

Fig. 4. The image is a water balloon bursting scene.



Original image Block random observation



Reconstructed image of Algorithm 2 with λ = 0.5 (34.9dB) Reconstructed image of Algorithm 2 with λ = 0 (33.43dB)

Fig. 5. The reconstructed image which is impact between a racket and a boll.

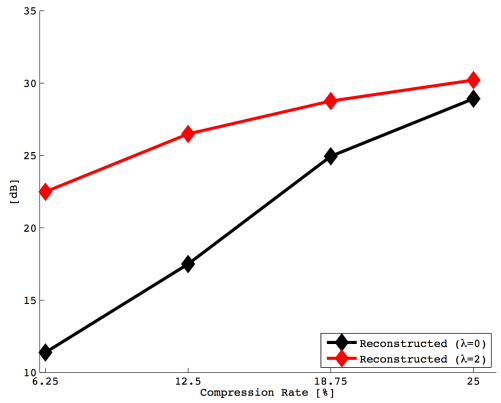


Fig. 6. PSNR [dB] of the reconstructed images about a water balloon bursting scene.

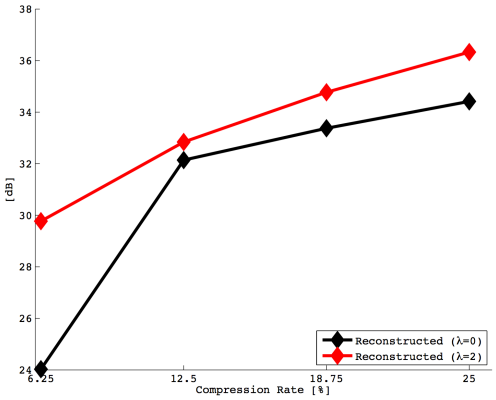


Fig. 7. PSNR [dB] of the reconstructed images about the moment of impact by a tennis racket and a ball.

2 and 3, respectively. These results show the effectiveness of the reference to the previous frame.

Computational time for the water-ballon sequence was 0.460 and 0.475 seconds by Algorithms 2 and 1, respectively. This means that the computational time was reduced by 3.16%. On the other hand, that for the tennis sequence was 0.456 and 0.472 seconds by Algorithms 2 and 1, respectively. This means that the computational time was reduced by 3.39%. These results show the effectiveness of the revised algorithm.

IV. Evaluation of Robustness

To evaluate the robustness of the proposed algorithm, we first reduced the compression rate as 25%, 18.75%, 12.5% and 6.25%. This is done by choosing 1, 2, 3, and 4 pixels from 4×4 blocks. The same image sequences as in Section 3 were used in this simulation. Figs. 6 and 7 show the average of PSNRs of reconstructed images with respect to the compression rate for the water balloon and the tennis image sequences, respectively. The red and black lines indicates the PSNRs when λ is 2 and 0, respectively. We can confirm that the proposed algorithm keeps a better PSNR average than that without referring the previous frame.

We also reduced the number of frames per second as 1000, 500, 250, and 125. The average of the PSNRs are shown in Fig. 8. We can see that, even though the number of FPS decreases, the average of the PSNR does not decrease so much, showing the robustness against the FPS reduction.

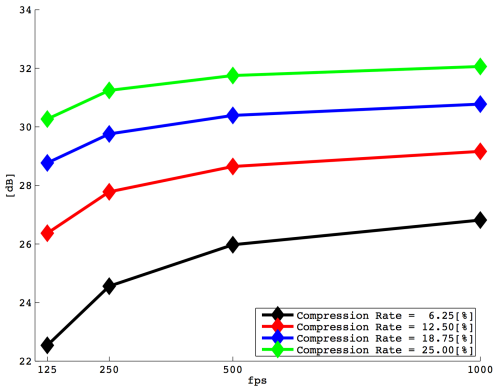


Fig. 8. Average of PSNRs in terms of the frame per second.

V. Conclusion

We proposed revision of the previously proposed pixel enhancement algorithm for high-speed camera image acquisition in the sense of two aspects. First, we improved the accuracy of the proposed algorithm by exchanging the role of two functions in the convex optimization algorithm. Second, we accelerated the algorithm by setting a better initial value. Simulation results showed the effectiveness of the proposed revision.

Acknowledgements

The authors wish to thank nac Image Technology for providing the high speed camera systems.

References

[1] R. Baraniuk, "Compressive sensing [lecture notes]," *IEEE Signal Processing Magazine*, vol. 24, no. 4, pp. 118–121, July 2007.

[2] M. Wakin, J. Laska, M. Duarte, D. Baron, S. Sarvotham, D. Takhar, K. Kelly, and R. Baraniuk, "An architecture for compressive imaging," in *2006 IEEE International Conference on Image Processing*, 2006, pp. 1273–1276.

[3] L. Kang and C. Lu, "Distributed compressive video sensing," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2009)*, 2009, pp. 1169–1172.

[4] N. Vaswani, "Kalman filtered compressed sensing," in *15th IEEE International Conference on Image Processing (ICIP 2008)*, 2008, pp. 893–896.

[5] ——, "LS-CS-Residual (LS-CS): Compressive sensing on least squares residual," *IEEE Transactions on Signal Processing*, vol. 58, no. 8, pp. 4108–4120, 2010.

[6] H. Chen, L. Kang, and C. Lu, "Dictionary learning-based distributed compressive video sensing," in *Picture Coding Symposium (PCS)*, 2010, pp. 210–213.

[7] S. Chan, R. Khoshabeh, K. Gibson, P. Gill, and T. Nguyen, "An augmented lagrangian method for total variation video restoration," *IEEE Transactions on Image Processing*, vol. 20, no. 11, pp. 3097–3111, 2011.

[8] J. Yang, J. Wright, T. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Transactions on Image Processing*, vol. 19, no. 11, pp. 2861–2873, 2010.

[9] Y. Song, Z. Zhu, Y. Lu, Q. Liu, and J. Zhao, "Reconstruction of magnetic resonance imaging by three-dimensional dual-dictionary learning," *Magnetic Resonance in Medicine*, vol. 71, no. 3, pp. 1285–1298, 2014.

[10] H. Bauschke, R. Burachik, P. Combettes, V. Elser, D. Luke, and H. Wolkowicz, Eds., *Fixed-point algorithms for inverse problems in science and engineering*. New York: Springer, 2011.
[11] E. Candes and M. Wakin, "An introduction to compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21–30, March 2008.
[12] SequentialCS, "http://www.ece.iastate.edu/ namrata/research/sequentialcs.html."