FAST ALGORITHM OF ARBITRARY FACTOR SUBPIXEL DOWNSAMPLING BASED ON FREQUENCY ANALYSIS

Ketan Tang†, Oscar C. Au†, Lu Fang‡, Jiahao Pang†, Yuanfang Guo†, Jiali Li†

†The Hong Kong University of Science and Technology
{tkt, eeau, eeandyguo, jiali, jpang}@ust.hk
‡CAS Key Laboratory of Technology in Geo-spatial Information Processingand Application System, University of Science and Technology of China, fanglu@ustc.edu.cn

ABSTRACT

Subpixel-based downsampling has shown its advantages over pixel-based downsampling in terms of preserving more spatial details along edges and generating sharper images, at the cost of certain amount of color-fringing artifacts in the downsampled image. To balance the sharpness and color-fringing artifacts, some algorithms are proposed to design optimal anti-aliasing (AA) filters, which are either image independent, or computationally too expensive. And all of the existing AA filters are designed for fixed downsampling factor, which makes them impractical for real applications. In this paper we propose two fast algorithms to design AA filter for arbitrary factor subpixel downsampling based on frequency analysis of the input image. The proposed algorithms generate image dependent AA filter which is as good as the state-of-the-art algorithm, but much faster.

Index Terms— subpixel-based downsampling, arbitrary factor, fast algorithm, frequency analysis

1. INTRODUCTION

Subpixel based downsampling (subpixel downsampling for short) is a new downsampling technique that can improve the apparent resolution of downsampled images. It is based on the fact that each pixel on a color LCD is actually composed of individually addressable red, green, and blue subpixel stripes [1–11]. Existing subpixel-downsampling algorithms include Direct Subpixel-Downsampling (DSD) [3] in which subpixel-downsampling is applied only in horizontal direction, and Direct Diagonal Subpixel-Downsampling (DDSD), in which subpixel-downsampling is applied diagonally in a 3×3 block, as shown in Figure 1.

The problem of directly applying DSD or DDSD is that color-fringing artifacts tend to be severe in the down-sampled images. Thus many algorithms are proposed to design an anti-aliasing (AA) filter to suppress the color-fringing artifacts while achieving high apparent resolution [2,4,5,8,9,11]. Some such algorithms include MMSE-SD [5] and DDSDFA [9]. In MMSE-SD [5] the authors treat the subpixel downsampling problem as a minimum mean square error problem and derives an image-independent AA filter in spatial domain, which can be precomputed offline. However MMSE-SD is only optimal in statistic sense and cannot adapt to the input image, thus optimality for a specific image is not guaranteed. In DDSDFA [9] the authors analyze the spectra of subpixel-downsampled images and derives an anti-aliasing filter in frequency domain which is optimal for the particular input image. However DDSDFA is computationally too expensive to be used in real applications, since it involves FFT, Laplace function fitting, spatial filtering, and so on. Besides, these two algorithms have a common problem that they are designed for downsampling factor of 1/3. To apply them for arbitrary factor downsampling needs to generate an intermediate image which is three times the size of the target image.

This paper aims to propose subpixel downsampling algorithms that are suitable for real applications, without losing much performance. Therefore the two aforementioned concerns need to be dealt with, i.e. capability of arbitrary factor downsampling and computational complexity. To maintain performance, we build our algorithm upon DDSDFA. Two fast algorithms of arbitrary factor subpixel downsampling are proposed, where alternative techniques are proposed to replace the original time-consuming steps in DDSDFA. Note that DDSDFA is even more time-consuming when applied for arbitrary downsampling factor, as an interpolation step is needed before applying DDSDFA. The first one removes the interpolation step before applying DDSDFA. The second one further removes the FFT step in designing the AA filter. Both of the proposed algorithms show improvement in speed without noticeable performance drop.

The rest of the paper is organized as follows. In Sec. 2 we briefly review the current method of performing arbitrary factor subpixel downsampling. In Sec. 3 we propose our fast algorithms. Experiment results are shown in Sec. 4. And in Sec. 5 we conclude our work.

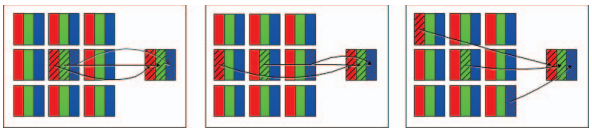


Fig. 1: Downsampling patterns. Left: DPD; Middle: DSD; Right: DDSD.

Fig. 2: Conventional arbitrary subpixel downsampling: bic+DDSDFA. First we do bicubic resizing (pixel-based), then we do subpixel downsampling (DDSDFA).

2. ARBITRARY FACTOR SUBPIXEL DOWNSAMPLING

Existing subpixel downsampling algorithms are all designed for 1/3 downsampling. For other downsampling factors, they have to resize the original large image to 3 times of target size, and then apply subpixel downsampling [9], as shown in Fig. 2. We suppose the original large image L is of size $M \times N$, the target small image S is of size $m \times n$, then the intermediate image I is of size $3m \times 3n$. The resizing process is usually implemented as interpolation, e.g. bicubic. In the following we denote this scheme as *bic+DDSDFA*. Another time- and memory-consuming step is fast Fourier transform (FFT) for deriving the AA filter of DDSDFA algorithm, which is briefly introduced in subsection 2.1.

2.1. DDSDFA algorithm

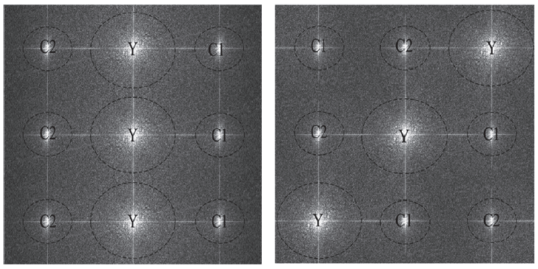
As shown in [9] that the luma spectrum of DDSD downsampled images have three luma components $\hat{Y}$ in the anti-diagonal direction, three chroma components $\hat{C}_1$ and three another chroma components $\hat{C}_2$ in other locations, as shown in Fig. 3. The definition for $\hat{Y}, \hat{C}_1, \hat{C}_2$ are as follows.

$$\hat{Y} = (\hat{R} + \hat{G} + \hat{B})/3, \quad (1)$$

$$\hat{C}_1 = (a_1\hat{R} + a_2\hat{G} + \hat{B})/3, \quad (2)$$

$$\hat{C}_2 = (a_2\hat{R} + a_1\hat{G} + \hat{B})/3. \quad (3)$$

where $a_1 = e^{-\frac{2\pi i}{3}} = -\frac{1}{2} - \frac{\sqrt{3}}{2}i$, $a_2 = e^{\frac{2\pi i}{3}} = -\frac{1}{2} + \frac{\sqrt{3}}{2}i$, $\hat{R}, \hat{G}, \hat{B}$ are the FFT of R, G, B. Optimal AA filter is chosen according to the actual size of the luma and chroma spectra of the input image, which are determined by approximating luma and chroma spectra with zero-mean symmetric Laplace functions and then finding the equal-magnitude frequency. The luma $\hat{Y}$ and chroma $\hat{C}_i$ spectra are fit as $\hat{Y}(r,\theta) = \frac{E_0}{2\lambda_0}\exp(-\frac{|r|}{\lambda_0})$, $\hat{C}_i(r,\theta) = \frac{E_i}{2\lambda_i}\exp(-\frac{|r|}{\lambda_i})$, where $E_0 = \iint_{r,\theta}\hat{Y}(r,\theta)drd\theta$, $E_i = \iint_{r,\theta}\hat{C}_i(r,\theta)drd\theta\,(i=1,2)$ are the integration of energy magnitudes over all frequencies. Note in [9] the two chroma spectra C_1 and C_2 are separately considered, which is unnecessary since they are conjugate to each other and their magnitudes are exactly the same. Hereafter we only consider C_1. The variance of the two symmetric Laplace functions are $2\lambda_i^2$, $i = 0, 1$. Then the cutoff frequency f_c are found as the equal-magnitude frequency of luma and chroma, i.e. $\hat{Y}(f_c) = \hat{C}_1(f_c - A)$, resulting in $f_c = \frac{\lambda_0\lambda_1}{\lambda_0+\lambda_1}(\ln\frac{\lambda_1 E_0}{\lambda_0 E_1} + \frac{A}{\lambda_1})$, where $A = \frac{1}{3}$ is the Nyquist frequency for 1/3 downsampling.



(a) DSD (b) DDSD

Fig. 3: DSD and DDSD spectrum.

Based on the size of luma and chroma spectra, the optimal filter is chosen between square and circle shape filters, as shown in Fig. 4. Basically when the chroma spectrum is much smaller than luma spectrum, main aliasing will occur between the two luma spectra, and circle shape filter has larger area; when the chroma spectrum is comparable to luma spectrum, main aliasing will occur between luma and chroma spectra, then square shape filter has larger area. The final optimal filter F is chosen as

$$F = \begin{cases} \text{square}, r = f_c, & f_c \in (\frac{1}{2}A, \frac{\sqrt{2}A}{\sqrt{2}+1}A] \\ \text{square}, r = \frac{\sqrt{2}}{\sqrt{2}+1}A, & f_c \in (\frac{\sqrt{2}A}{\sqrt{2}+1}A, \frac{2}{\sqrt{\pi}}(\frac{\sqrt{2}}{\sqrt{2}+1}A)] \\ \text{circle}, r = f_c, & f_c \in (\frac{2}{\sqrt{\pi}}(\frac{\sqrt{2}}{\sqrt{2}+1}A), \frac{\sqrt{2}}{2}A] \\ \text{circle}, r = \frac{\sqrt{2}}{2}A, & f_c \in (\frac{\sqrt{2}}{2}A, A] \end{cases} \quad (4)$$

where r is the radius for circle filter, and half of side length for square filter. Note in [9] the authors use rectangle and ellipse filters, which are inaccurate since the filter is always symmetric in x, y directions.

3. FAST SUBPIXEL DOWNSAMPLING

In order to decide the size of luma and chroma spectra, Laplace function fitting is used. However this procedure is very time-consuming and unnecessary. We start from removing the Laplace function fitting. Let x be the spectrum that needs to be fit (x could be Y or C_1, C_2), then we have the standard deviation and energy of x being $\lambda = \frac{1}{\sqrt{2}}\sqrt{\text{var}(x)}$, $E = \sum_k |x(k)|$. In experiment we find out that

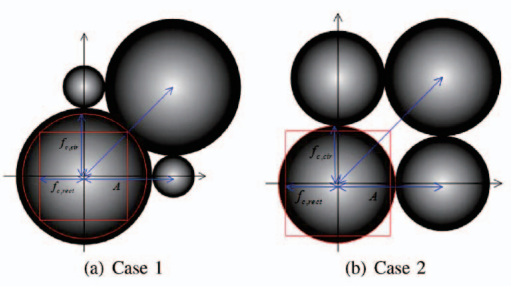


Fig. 4: Optimal filter is chosen between (a) circle shape filter and (b) square shape filter .

for most natural images $\frac{E_0}{\lambda_0} \approx \frac{E_1}{\lambda_1}$, which implies $\ln(\frac{\lambda_1 E_0}{\lambda_0 E_1}) \approx 0$. Therefore the cutoff frequency is

$$f_c = \frac{\lambda_0\lambda_1}{\lambda_0+\lambda_1}(\ln\frac{\lambda_1 E_0}{\lambda_0 E_1} + \frac{A}{\lambda_1}) \approx \frac{\lambda_0}{\lambda_1+\lambda_0}A \approx \frac{E_0}{E_1+E_0}A. \tag{5}$$

As a result we do not need to fit a Laplace function for any of Y, C_1, C_2. Instead, we just need to compute the energy E_0, E_1, E_2. And since C_1 and C_2 are conjugate we know $E_1 = E_2$, so only E_0 and E_1 need to be computed.

3.1. Fast subpixel downsampling with no interpolation

As resizing is one of the key parts of arbitrary factor DDSDFA algorithm and interpolation is computationally forbidden for many mobile devices, we perform resizing in frequency domain instead of in spatial domain. Note that since the anti-aliasing filter is designed in frequency domain, FFT is needed anyway, therefore we do not spend extra computation for frequency domain resizing. As we get rid of spatial domain interpolation, we name this fast algorithm FSDNI (Fast Subpixel Downsampling with No Interpolation), which is based on two properties of FFT:

1. **resizing in frequency domain:** padding zeros in frequency domain is equivalent to zooming up in spatial domain; truncation in frequency domain is equivalent to zooming down in spatial domain.
2. **filtering in frequency domain:** multiplication in frequency domain is equivalent to convolution in spatial domain.

The first property is used for deriving the intermediate image as in Fig. 2. Suppose the spectra of RGB channels of image L are $\hat{R}^L, \hat{G}^L, \hat{B}^L$ (size $M \times N$), the spectra of image I are $\hat{R}^I, \hat{G}^I, \hat{B}^I$ (size $3m \times 3n$), then

- for upsampling:

$$\hat{C}^I(u,v) = \begin{cases} \hat{C}^L(u,v), & 0 \le |u| \le \frac{M}{2}, 0 \le |v| \le \frac{N}{2} \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

- for downsampling:

$$\hat{C}^L(u,v) = \hat{C}^I(u,v),\ 0 \le |u| \le \frac{3m}{2}, 0 \le |v| \le \frac{3n}{2} \tag{7}$$

The second property is used for AA filtering in frequency domain. To prevent severe ringing artifact, the AA filter is designed as a sinc padded lowpass filter (SPLF), which is composed of ideal lowpass filter (ILF) q with cutoff frequency f_c in low frequency part, and a sinc filter p padded in high frequency part. For square shape filter, the padded sinc filter is

$$p = \text{sinc}(\frac{|x|-f_c}{w_x})\,\text{sinc}(\frac{|y|-f_c}{w}), \tag{8}$$

where w is the bandwidth of the sinc filter (we set $w = \frac{1}{3(2n+1)}$ where n is the tap number of the corresponding spatial domain filter and is typically set to be 4), due to the symmetry of x, y directions. Then SPLF $\tilde{q}$ for square shape filter is

$$\tilde{q}(x,y) = \begin{cases} 1, & \text{if } |x| \le f_c, |y| \le f_c, \\ \text{sinc}(\frac{|x|-f_c}{w})\,\text{sinc}(\frac{|y|-f_c}{w}), & \text{otherwise.} \end{cases} \tag{9}$$

For circle shape filter, the padded sinc filter is defined in polar coordinate:

$$p = \text{sinc}(\frac{(\sqrt{x^2+y^2}-f_c}{w}), \tag{10}$$

Then SPLF $\tilde{q}$ for circle shape filter is

$$\tilde{q}(x,y) = \begin{cases} 1, & \text{if } x^2+y^2 \le f_c^2, \\ \text{sinc}\left(\frac{\sqrt{x^2+y^2}-f_c}{w}\right), & \text{otherwise.} \end{cases} \tag{11}$$

Fig. 5 compares ILF and SPLF for circle shape filter in frequency domain. One can see that SPLF has smoother frequency response than ILF, thus has less ringing effect in spatial domain. One may also try other existing filtering techniques such as Chebyshev or Butterworth filter, however we have found SPLF mains more high frequency information than existing filters without causing ringing effects for our subpixel downsampling problem. Our fast algorithm is detailed in Algorithm 1.

3.2. Fast subpixel downsampling with no FFT

The drawback of FSDNI is that it still needs FFT which needs significant amount of computation. Thus the second algorithm, fast subpixel downsampling with no FFT (FSDNF), is proposed to further reduce computation complexity. The basis of this algorithm is Parserval's theorem, which says the energy in frequency domain is equivalent to the energy in spatial domain. Therefore without FFT we still can compute the energy of luminance and chrominance, and then the cutoff frequency of AA filter.

Algorithm 1 FSDNI

1. Compute FFT of RGB channels of original large image L and get $\hat{Y^L}, \hat{C_1^L}, \hat{C_2^L}$.
2. Calculate energies and then the cutoff frequency f_c by Eqn. (5).
3. Design SPLF $\tilde{q}$ as Eqn. (11).
4. Get the RGB spectra of intermediate image I, $\hat{R^I}, \hat{G^I}, \hat{B^I}$, by Eqn. (6,7).
5. Multiply $\hat{R^I}, \hat{G^I}, \hat{B^I}$ with filter $\tilde{q}$, then apply inverse FFT to get spatial domain image I'.
6. Apply DDSD downsampling on I' to get target small image S.

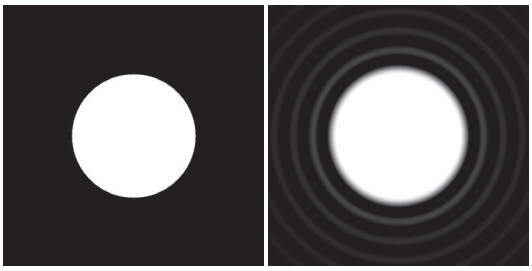


Fig. 5: Comparison between ILF and SPLF in frequency domain. Left: ILF q; right: SPLF $\tilde{q}$.

In the original DDSDFA algorithm to compute cutoff frequency of AA filter we need to compute the energy ratio of Y and C_1 and C_2: E_0/E_1 and E_0/E_2, where

$$E_0 = \|\hat{Y}\|^2 = \sum_{i,j} \frac{1}{9} |\hat{R}(i,j) + \hat{G}(i,j) + \hat{B}(i,j)|^2 \quad (12)$$

$$E_1 = \|\hat{C_1}\|^2 = \sum_{i,j} \frac{1}{9} |\hat{R}(i,j) + a_1\hat{G}(i,j) + a_2\hat{B}(i,j)|^2 \quad (13)$$

We skip E_2 since $E_2 = E_1$ due to the fact that C_2 is conjugate to C_1. From Parserval's theorem we know the energy in frequency domain equals to energy in spatial domain:

$$E_0 = \sum_{i,j} \frac{1}{9} |R(i,j) + G(i,j) + B(i,j)|^2 \quad (14)$$

$$E_1 = \sum_{i,j} \frac{1}{9} |R(i,j) + a_1 G(i,j) + a_2 B(i,j)|^2 \quad (15)$$

Although a_1 and a_2 are complex numbers, directly computing Y, C_1, C_2 in spatial domain is possible.

Note now we do not have FFT, thus designing AA filter in frequency domain is no longer possible. We need to directly perform arbitrary factor subpixel resizing in spatial domain. After cutoff frequency f_c is obtained using the energies in Eqn. (14-**??**), a lanczos filter with cutoff frequency f_c is designed. As stated in [12], lanczos filter is a sinc function weighted sinc function. Compared to simple sinc function (which is ideal low pass filter in spatial domain), lanczos filter has the advantage of reducing ringing artifacts. A 1-D lanczos filter with cutoff frequency f_c has the following form

$$q(x) = \frac{2nf_c \sin(2\pi f_c x) \sin(\pi x/n)}{\pi^2 x^2} \quad (16)$$

where n is the order of the lanczos filter. Usually n is set to be 2 or 3. In our algorithm we set $n = 3$.

The advantage of having a continuous domain downsampling kernel lanczos is that, the AA filtering process can be fused into the downsampling process, therefore we don't need to generate an intermediate image and the computation complexity is reduced. Basically for a pixel in channel c of the downsampled image S located at (x^S, y^S), we find its corresponding location in the original large image L via following equations

$$x^L(c) - 0.5 = \alpha_x(x^S + p_x(c) - 0.5) \quad (17)$$

$$y^L(c) - 0.5 = \alpha_y(y^S + p_y(c) - 0.5) \quad (18)$$

where $\alpha_x = N/n, \alpha_y = M/m$ are the scaling factors, $p_x(c), p_y(c)$ are the horizontal and vertical subpixel shifts for channel c. -0.5 in both sides of equations (17,18) is to follow the downsampling convention that 0.5 in input space maps to 0.5 in output space (this conversion is widely used in academy and industry, including Matlab©, OpenCV, etc.). For pixel-based downsampling $p_x(c) = 0, p_y(c) = 0$ for all three channels, but for subpixel-based downsampling they are different. Take DDSD for example,

$$p_x(r) = -1/3, p_y(r) = -1/3; \quad (19)$$

$$p_x(g) = 0, p_y(g) = 0; \quad (20)$$

$$p_x(b) = 1/3, p_y(c) = 1/3. \quad (21)$$

When x^L, y^L are found, the downsampled pixel value is computed as

$$S(x^S, y^S, c) = \sum_{(x_i,y_i)\in N_{x^L,y^L}} q(\frac{x_i - x^L(c)}{\alpha_x}) q(\frac{y_i - y^L(c)}{\alpha_y}) \cdot L(x_i, y_i, c),\ c \in \{r, g, b\}, \quad (22)$$

where $S(x^S, y^S, c)$ is the pixel value of downsampled image S at position (x^S, y^S) in channel c. N_{x^L,y^L} is a square neighborhood of (x^L, y^L) which depends on scaling factor α_x and α_y. Usually the size of N_{x^L,y^L} is $4\alpha_x \times 4\alpha_y$. (x_i, y_i) are the integer pixel positions in original image L. In practice we can separate the 2-D filtering of (22) into two 1-D filtering:

$$S(x^S, y^S, c) = \sum_{y_i \in N_{y^L}} q(\frac{y_i - y^L(c)}{\alpha_y}) \sum_{x_i \in N_{x^L}} q(\frac{x_i - x^L(c)}{\alpha_x}) \cdot L(x_i, y_i, c),\ c \in \{r, g, b\}. \quad (23)$$

Table 1: Average objective quality measurements for 1/2 downsampling.

	sharpness	aliasing	contrast	PSNR_U	PSNR_V	PSNR_z	time (s)
bic+DDSDFA	1.06	1.08	1.03	29.03	29.70	24.72	23.13
FSDNI	0.93	0.50	0.95	30.52	30.97	26.49	1.29
FSDNF	0.90	0.31	0.93	31.64	32.28	26.70	0.55

Table 2: Average objective quality measurements for 1/2.5 downsampling.

	sharpness	aliasing	contrast	PSNR_U	PSNR_V	PSNR_z	time (s)
bic+DDSDFA	1.04	1.15	1.03	28.54	28.98	24.93	16.42
FSDNI	0.92	0.50	0.94	30.15	30.54	25.42	0.98
FSDNF	0.90	0.33	0.93	31.08	31.63	24.23	0.16

Algorithm 2 FSDNF

1. Compute E_0, E_1 by Eqn. (15), and then cutoff frequency $f_c = AE_0/(E_0 + E_1)$.
2. Design AA filter q in spatial domain by Eqn. (16).
3. Apply lanczos kernel based downsampling to directly get the downsampled image.

The whole algorithm is described in Algorithm 2. Compared to FSDNI, this algorithm is even faster, but it has disadvantage of losing sharpness and contrast of the downsampled images. This is caused by the intrinsic smoothing feature of lanczos filter. It is up to users to decide which method to use.

4. EXPERIMENT RESULTS

Extensive experiments are conducted to evaluate the performance of the proposed two fast algorithms FSDNI and FSDNF. The test image set includes the whole Kodak image database and 24 more images which are even sharper than Kodak images[1]. Fig. 6 shows the comparison of bic+DDSDFA and the proposed algorithms for downsampling factor 1/2. We can see that the visually there is almost no difference between the two results. Table 1 and Table 2 show the average objective measurements for 1/2 and 1/2.5 respectively. The running time is measured on Matlab R2013a, Windows 7 system with i3 540 CPU. We notice that although sharpness and contrast of FSDNI and FSDNF are slightly smaller than bic+DDSDFA, their aliasing is much smaller and PSNR_U, PSNR_V are larger, which shows that the two fast algorithms achieves much better color performance with slightly worse sharpness and contrast.

Comparing FSDNI and FSDNF, FSDNF has better aliasing and chrominance distortion performance than FSDNI, but slightly worse sharpness and contrast. It is up to the user to choose which one to use in a particular application. We also notice that the proposed algorithms cost much less time than bic+DDSDFA. We can conclude that FSDNI has almost same performance both subjectively and objectively, while costing much less time than bic+DDSDFA.

[1]FSD (Fast Subpixel Downsampling) dataset, public available at `http://ihome.ust.hk/~tkt/research_subpixel.html`

5. CONCLUSION

In this paper we propose two fast algorithms based on DDSDFA. Both algorithms show the ability of retaining the advantages of bic+DDSDFA for arbitrary factor subpixel downsampling, but cost much less time than bic+DDSDFA. FSDNI removes spatial interpolation based on the properties of FFT. FSDNF further removes FFT based on Parserval's theorem. Also since in FSDNF we don't generate an intermediate image, memory requirement is largely reduced.

Acknowledgment

This work was supported in part by the Hong Kong Government Research Grants Council (GRF Project no. 610112), and the Hong Kong Government Innovation and Technology Fund and the State Key Laboratory on Advanced Displays and Optoelectronics Technologies (Project No: ITC-PSKL12EG02), Natural Science Foundation of China (NSFC) under contract No. 61303151 and 61331015.

6. REFERENCES

[1] S. J. Daly, "Analysis of subtriad addressing algorithms by visual system models," *SID Digest*, pp. 1200–1203, 2001.

[2] J. Kim and C. Kim, "A filter design algorithm for subpixel rendering on matrix displays," in *European Signal Processing Conference (EUSIPCO)*, Poznan, Poland, 2007.

(a) bic+DDSDFA (b) FSDNI (c) FSDNF

Fig. 6: Comparison of bic+DDSDFA and the proposed FSDNI, FSDNF for 1/2 downsampling.

[3] D. S. Messing and S. Daly, "Improved display resolution of subsampled colour images using subpixel addressing," in *Image Processing (ICIP), International Conference on*, vol. 1, 2002, pp. 625–628.

[4] J. C. Platt, "Optimal filtering for patterned displays," *Signal Processing Letters, IEEE*, vol. 7, no. 7, pp. 179–181, 2000.

[5] L. Fang, O. C. Au, K. Tang, X. Wen, and H. Wang, "Novel 2-D MMSE subpixel-based image down-sampling," *Circuits and Systems for Video Technology, IEEE Transactions on*, no. 99, 2011.

[6] L. Fang, O. C. Au, and A. K. Katsaggelos, "Adaptive joint demosaicing and subpixel-based down-sampling for bayer image," in *Multimedia and Expo (ICME), IEEE International Conference on*, 2011, pp. 1–6.

[7] F. Lu, O. C. Au, Y. Yi, T. Weiran, and W. Xing, "A new adaptive subpixel-based downsampling scheme using edge detection," in *Circuits and Systems (ISCAS), IEEE International Symposium on*, 2009, pp. 3194–3197.

[8] L. Fang and O. C. Au, "Novel 2-D MMSE subpixel-based image down-sampling for matrix displays," in *Acoustics Speech and Signal Processing (ICASSP), IEEE International Conference on*, 2010, pp. 986–989.

[9] L. Fang, K. Tang, O. Au, and A. Katsaggelos, "Anti-aliasing filter design for subpixel down-sampling via frequency domain analysis," *Image Processing, IEEE Transactions on*, no. 99, 2011.

[10] L. Fang, O. C. Au, Y. Yang, W. Tang, and X. Wen, "Subpixel-based image downsampling-some analysis and observation," in *Multimedia and Expo (ICME), IEEE International Conference on*, 2009, pp. 1576–1577.

[11] S. J. Daly and R. R. K. Kovvuri, "Methods and systems for improving display resolution in images using sub-pixel sampling and visual error filtering," US Patent 6 608 632, Aug 19, 2003.

[12] C. E. Duchon, "Lanczos filtering in one and two dimensions," *Journal of Applied Meteorology*, vol. 18, pp. 1016–1022, 1979.