# A Novel Hardware-Based UHD Video Up-Scaler Based on Local Structure Estimation

Hang Sun[1], Shengfu Dong[1,2,*], Xiaodong Xie[1], Meng Li[1], Xiaofeng Huang[1], and Wen Gao[1]

[1] National Engineering Laboratory for Video Technology, Peking University, Beijing, China

[2] Peking University Shenzhen Graduate School, Shenzhen, China

sunhang618@126.com, dongsf@pkusz.edu.cn, {xdxie,limeng,xfhuang,wgao}@jdl.ac.cn

**Abstract.** A novel HW-based video up-scaling method proposed in this paper performs the video image up-scaling. Based on the geometry similarity between the original image and the target image, the proposed algorithm produces new pixels by exploiting the directional weights from the errors estimated by a preset interpolation model in the local area, and using the weights to combine two pixel values predicted by the same model. The algorithm can well preserve the local structure, the edge sharpness in particular, and effectively reduce artifacts such as jaggies and ringings. Based on this method, an efficient HW architecture is developed which is capable of processing at real-time up-scaling to HD or UHD@30fps. Experimental results demonstrate that the proposed method even outperforms some methods using more complex algorithms in both objective and subjective image qualities. In addition, the low complexity of the algorithm simplifies the hardware implementation, which makes real-time enlargement of video image practical.

**Keywords:** local structure based image interpolation, video scaler, lowpass filter, VLSI design.

## 1 Introduction

Image up-scaling technique aims to reconstruct high resolution (HR) images from the original low resolution (LR) ones. This is a very prevalent technique in digital image applications, ranging from consumer electronics to professional display devices. Video scaler is a crucial component of these applications for resolution format conversion, for example, images enlarging from QCIF to CIF, or from standard definition (SD) video to high definition (HD) video or more recently to ultra high definition (UHD) video. The more advanced method in preserving the image quality usually increases the computational burden at the same time. But it is agreed in studies that for real applications, the criteria should be put not only on the perceptual quality of images, but also on the computational/implementation cost [1, 2].

[*] Corresponding author.

Classical linear interpolation methods, such as bilinear [3] or bicubic [4] interpolation, have their benefits on low computational cost and are hence widely used as scaling methods in early years. But these methods face difficulties in preserving the fast-changing geometry structures, such as sharp edges and fine textures. Unfortunately, this problem could not be solved by linear interpolation methods because of the unsuitability of their inherent space invariant model for a variety of natural geometry structures.

Non-linear interpolation method is the primary domain of research in recent years. The most outstanding advantage of non-linear interpolation is its possibility to preserve the sharpness of edges and textures, which is highly sensitive to human visual system. So a wide variety of methods [5-11] have been devoted toward this topic. The methods in [5, 6, 7] quantize the edge orientation into finite models and a VLSI architecture has been implemented in [7]. But the process of identifying edge orientations is quite complicated and may result in image distortion if the identification for orientation goes wrong. In order to adjust to an arbitrarily oriented edge, the methods in [8, 9, 10] apply auto-regression model to the estimation of the local interpolation coefficients. NEDI [8] proposed a method to estimate the covariance matrix of the high resolution image based on the corresponding low resolution one, and NLEDI [9] extends the choice of estimating window to larger areas. However, auto-regression model always introduces huge computation time by solving a set of linear equations: it is cited in [8] itself that the computation of each pixel will require about 1300 multiplications when the local window size is 8, which is obviously unacceptable for hardware design. In [11], two pre-estimated samples are combined by using the statistics obtained by linear minimum mean square-error estimation (LMMSE). However, the estimation is only taken in the pixel interpolation itself that ignores its connection with its surrounding pixels, which is not enough to get a satisfactory result.

In this paper, we propose a new image up-scaling method that exploits the local geometry structure by applying a preset interpolation model to calculate orientation estimation errors of four local neighboring pixels from LR image and interprets the structure into directional weights for HR image pixel interpolation. Due to the simplicity of this method, we propose a possible hardware architecture that accomplishes real-time HD to UHD image enlargement with limited resource consumption.

The rest of the paper is organized as follows. In Section 2, the proposed method is introduced. Section 3 gives the experimental results in objective and subjective quality. Section 4 proposes an architecture for implementation of our algorithm and shows the main resource consumption. The conclusion is provided in Section 5.

## 2 The Proposed Image Interpolation Method

Without losing generality, we first consider the case of scaling factor equal to 2, i.e. the problem of the LR image of size H W scaling up to the HR image of size 2H 2W. After this, we then consider the more general case where the scaling factor is an arbitrary number.

Let $I_h$ be the HR image that is intended to be interpolated by the LR image, $I_l$. As in Fig.1, the black dots represent the $I_l$ pixels and the gray and white pixels represent the missing ones in $I_h$ that need to be interpolated. In our method, white pixels are able to be calculated only after all of the gray ones are obtained, hence the whole process is divided into two stages: first, gray pixels are interpolated by black pixels. Second, white pixels are interpolated by black and gray pixels together.

At the first stage, let's focus on P(2x+1, 2y+1) interpolation process as an example.

According to the assumption that HR image and its associated LR image have geometric similarity in the local area, we consider that the HR area centered at P(2x+1, 2y+1) and its four nearest surrounding LR areas centered at P(2x, 2y), P(2x+2, 2y), P(2x, 2y+2), P(2x+2, 2y+2) have the same geometric characteristics. In this paper, we mark the areas with two mutually orthogonal lines illustrated in Fig.1. The solid lines cover an HR local area, and the dotted lines cover one of the surrounding LR areas.

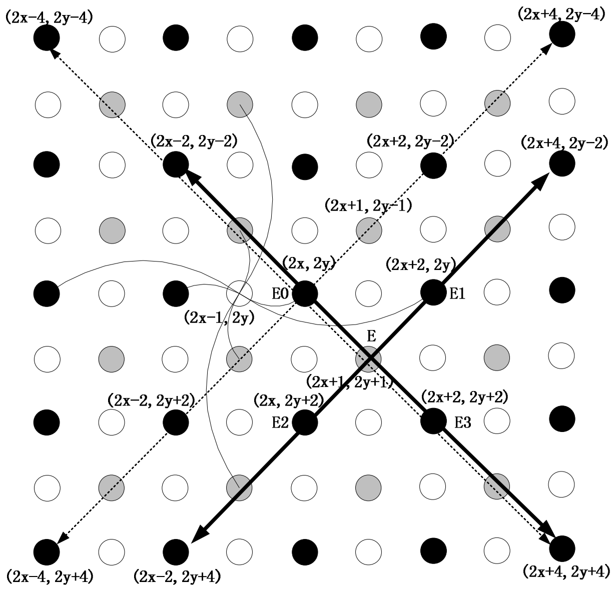

**Fig. 1.** HR image filled with LR image pixels and the missing pixels. The black dots represent the LR image pixels and the gray and white dots represent the missing HR pixels.

We find that P(2x+1, 2y+1), the center pixel of the HR local area, can be interpolated using the information extracted from the already existed LR pixels - P(2x, 2y), P(2x+2, 2y), P(2x, 2y+2) and P(2x+2, 2y+2). The question emerges naturally now: since the local areas share the same structure, is it possible to exploit the structure characteristics by the existing LR pixels, and to use this local structure to interpolate the missing ones? NEDI [8] proposed a possible method: it gets interpolation coefficients according to a principle of minimum mean square error in the local area, and then uses these coefficients to interpolate the missing samples. The disadvantage of this method is the high complexity of solving a set of linear equations for generating interpolation coefficients. Meanwhile, inspiration for an alternative method comes up in the opposite way: to save the computation time, we can prepare a preset interpolation model (two mutually orthogonal lines in Fig.1) and use the same model to interpolate P(2x+1, 2y+1) and its surrounding pixels- P(2x, 2y), P(2x+2, 2y), P(2x, 2y+2), P(2x+2, 2y+2). Since the surrounding pixels are already existing, prediction errors can be easily calculated between the true pixel values and their predicted ones, then the errors are used for generating a set of weighting factors to derive the more accurate P(2x+1, 2y+1). At the meantime, we introduce an edge Sensitive Factor (SF) to better represent the sharpness of edges and textures, which will be discussed in detail later. The procedure of our algorithm is described as follows:

At first, for missing pixel P(2x+1, 2y+1), we interpolate its two predicted values $\mathrm{E}_{45}$ and $\mathrm{E}_{135}$ along 45° and 135° respectively by using the eight nearest pixels on the two diagonal lines (4 for each). Then we fuse the two predicted values into the true value of P(2x+1, 2y+1) by a set of weighting factors $\mathrm{W}_{45}$ and $\mathrm{W}_{135}$, which remain to be determined.

$$E_{45} = \mathrm{C0} \times \mathrm{P}(2\mathrm{x}-2,2\mathrm{y}+4) + \mathrm{C1} \times \mathrm{P}(2\mathrm{x},2\mathrm{y}+2) + \mathrm{C2} \times \mathrm{P}(2\mathrm{x}+2,2\mathrm{y}) + \mathrm{C3} \times \mathrm{P}(2\mathrm{x}+4,2\mathrm{y}-2) \tag{1}$$

$$E_{135} = \mathrm{C0} \times \mathrm{P}(2\mathrm{x}-2,2\mathrm{y}-2) + \mathrm{C1} \times \mathrm{P}(2\mathrm{x},2\mathrm{y}) + \mathrm{C2} \times \mathrm{P}(2\mathrm{x}+2,2\mathrm{y}+2) + \mathrm{C3} \times \mathrm{P}(2\mathrm{x}+4,2\mathrm{y}+4) \tag{2}$$

$$\mathrm{P}(2\mathrm{x}+1,2\mathrm{y}+1) = W_{45} \times E_{45} + W_{135} \times E_{135} \tag{3}$$

The values C0, C1, C2 and C3 (1, 2) are coefficients derived using a typical low-pass filter model ($-\frac{1}{8}, \frac{5}{8}, \frac{5}{8}, -\frac{1}{8}$ for example). And the second step is to use the same model to interpolate predicted values for surrounding pixels, which are $E0_{45}$, $E0_{135}$ for P(2x, 2y), $E1_{45}$, $E1_{135}$ for P(2x+2, 2y), $E2_{45}$, $E2_{135}$ for P(2x, 2y+2) and $E3_{45}$, $E3_{135}$ for P(2x+2, 2y+2). Now taking the predicted values $E0_{45}$ and $E0_{135}$ as an example:

$$E0_{45} = \mathrm{C0} \times \mathrm{P}(2\mathrm{x}-4,2\mathrm{y}+4) + \mathrm{C1} \times \mathrm{P}(2\mathrm{x}-2,2\mathrm{y}+2) + \mathrm{C2} \times \mathrm{P}(2\mathrm{x}+2,2\mathrm{y}-2) + \mathrm{C3} \times \mathrm{P}(2\mathrm{x}+4,2\mathrm{y}-4) \tag{4}$$

$$E0_{135} = \mathrm{C0} \times \mathrm{P}(2\mathrm{x}-4,2\mathrm{y}-4) + \mathrm{C1} \times \mathrm{P}(2\mathrm{x}-2,2\mathrm{y}-2) + \mathrm{C2} \times \mathrm{P}(2\mathrm{x}+2,2\mathrm{y}+2) + \mathrm{C3} \times \mathrm{P}(2\mathrm{x}+4,2\mathrm{y}+4) \tag{5}$$

Then, we regard the sum of four prediction errors along a particular direction as the local structure estimation error along this direction:

$$Err_{45} = |\mathrm{P}(2\mathrm{x},2\mathrm{y}) - E0_{45}| + |\mathrm{P}(2\mathrm{x}+2,2\mathrm{y}) - E1_{45}| + |\mathrm{P}(2\mathrm{x},2\mathrm{y}+2) - E2_{45}| + |\mathrm{P}(2\mathrm{x}+2,2\mathrm{y}+2) - E3_{45}| \tag{6}$$

$$Err_{135} = |\mathrm{P}(2\mathrm{x},2\mathrm{y}) - E0_{135}| + |\mathrm{P}(2\mathrm{x}+2,2\mathrm{y}) - E1_{135}| + |\mathrm{P}(2\mathrm{x},2\mathrm{y}+2) - E2_{135}| + |\mathrm{P}(2\mathrm{x}+2,2\mathrm{y}+2) - E3_{135}| \tag{7}$$

$Err_{45}$ implies the possibility that we get the true value of P(2x, 2y) if we only make interpolation along 45° direction. The greater $Err_{45}$ is, the smaller this possibility will be, and vice-versa.

However, the experimental results (Fig.4 (b)) show that these prediction errors are not sufficient to reflect the sharpness of local structure. So before generating directional weighting factors, we introduce an edge Sensitive Factor (SF) to revise two prediction errors, which will make our algorithm more sensitive to edge structure:

$$ESF_{45} = (Err_{45})^{SF} \tag{8}$$

$$ESF_{135} = (Err_{135})^{SF} \tag{9}$$

Then, the weighting factors for both directions can be calculated as follows:

$$W_{45} = \frac{ESF_{135}}{ESF_{45}+ESF_{135}} \tag{10}$$

$$W_{135} = 1 - W_{45} \tag{11}$$

Finally, we can obtain P(2x+1, 2y+1) by substituting $W_{45}$ and $W_{135}$ to (3).

Once all the gray pixels in the entire image are interpolated in the first stage, they will be treated as available pixels in the second stage. Then the white pixels (for example, P(2x-1, 2y) covered by curved lines in Fig.1) can be obtained in the same fashion after rotating the image plane by π/4 clockwise.

## 3 Experimental Result

We firstly down-sample HR images to LR images by eliminating even position pixels both in horizontal and vertical directions, then conduct different methods to reconstruct the LR images to their original size, and compare them with their original HR images in term of objective and subjective qualities. Extensive experiments are

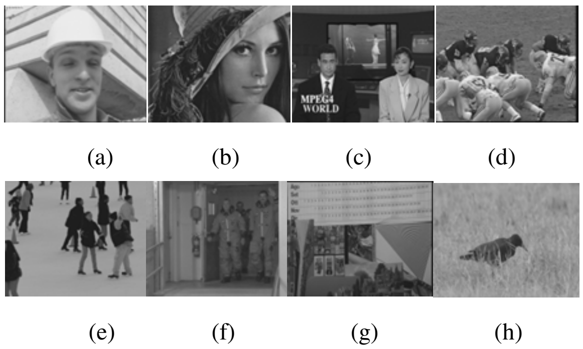

**Fig. 2.** Test images. CIF: (a)Foreman (b)Lena (c)News (d)Football. 4CIF: (e)Ice (f)Crew. HD: (g)Calendar (h)Raven.

conducted to evaluate our interpolation technique. Without the loss of generality, we choose test images with a variety of structures including parallel lines in Foreman and Crew, large curved edges in Lena and Football. High contrast contents including texts, individuals and big areas of still background are also presented in News and Ice. Calendar and Raven contain fine textures with plenty of details. Moreover, several universal resolutions are taken into consideration, including CIF (352×288), 4CIF (704×576) and HD (1280×720). Fig.2 depicts all these images.

To verify the vital role of Sensitive Factor introduced in Section 2, we choose integers 1~8 as the values of SF and substitute them into (8) and (9) respectively, then we compare the reconstructed images with their original ones and plot PSNR values in Fig.3. The PSNR results prove that the image quality is changing with the value of Sensitive Factor. To be more specific, in the interval between SF=1 and 8, an almost linear increase is detected for SF=1~3 and a decline trend is found for higher SF values from 4 to 8. The changing trend implies that the PSNR curves always have a peak point appearing around values of 3 and 4.

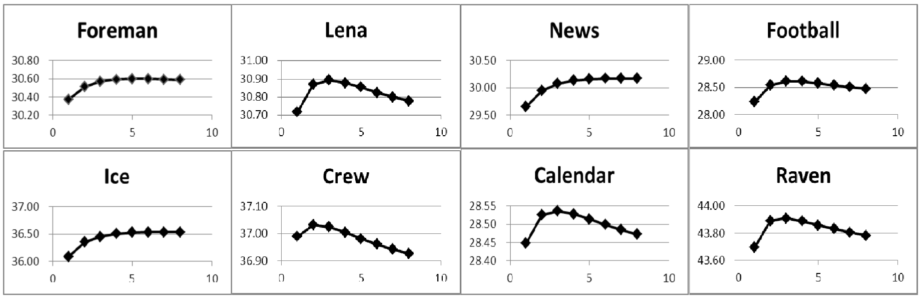

**Fig. 3.** The PSNR change trend from SF=1 to 8 for test images

The subjective quality measurements also verified this interesting phenomenon, and an example for enlarged Forman is illustrated in Fig.4. Not surprisingly, bicubic method introduces apparent jaggies along the sharp lines (Fig.4 (a)), whereas these artifacts can be amended by our method when SF=1 (b), but not satisfactory enough. Huge quality improvement is observed in Fig.4 (c) for SF=3, because the increase of Sensitive Factor gives more priority to the interpolation process that matches local fine structures. However, with the increase of SF value, the differences between Fig.4 (c) and (d) become almost invisible. In fact, the phenomena mentioned above are also observed in other test images.

We compared our method with other four methods including two classical methods of bilinear and bicubic, and two competitive methods in [8] and [11]. These different methods are applied to the eight test images of Fig.2 and PSNR results are tabulated in Table 1. Although a variety of the structures and resolutions are involved, the proposed method consistently ranks the first in terms of PSNR performance.

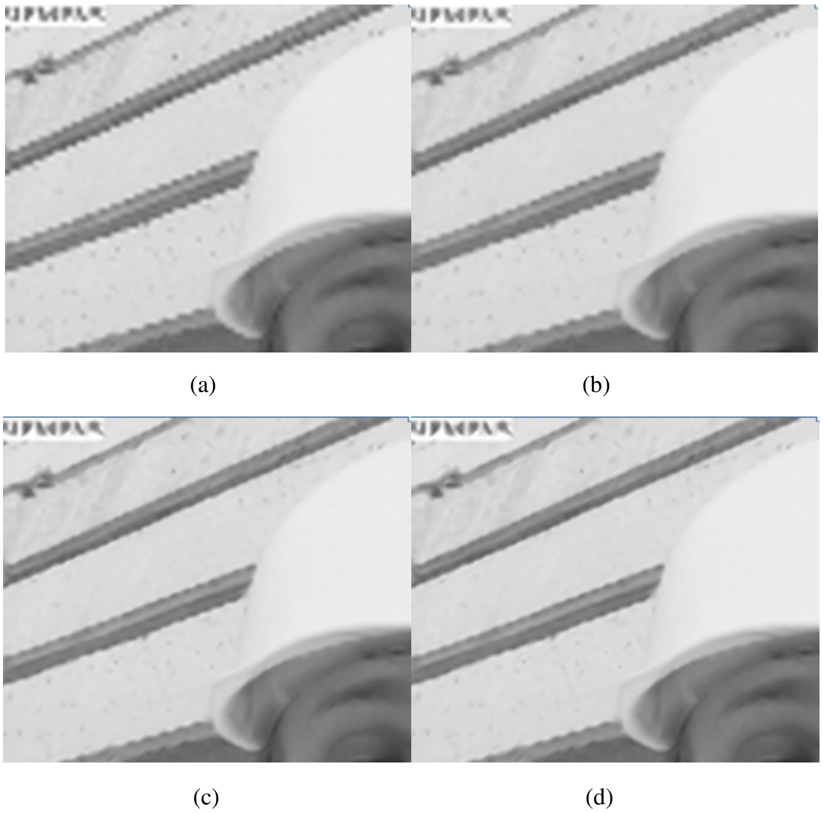

**Fig. 4.** Reconstructed images enlarged to compare different SF values. (a)bicubic (b)SF=1 (c)SF=3 (d)SF=8.

**Table 1.** PSNR results of the reconstructed HR image by different methods

| **Images** | bilinear | bicubic | Method in[8] | Method in [11] | **Proposed** |
|---|---|---|---|---|---|
| **Foreman 352x288** | 29.82 | 30.01 | 30.49 | 30.46 | **30.60** |
| **Lena352x288** | 30.13 | 30.42 | 30.53 | 30.27 | **30.89** |
| **News 352x288** | 28.84 | 29.38 | 28.73 | 29.16 | **30.17** |
| **Football 352x288** | 27.42 | 28.04 | 27.38 | 27.38 | **28.61** |
| **Ice 704x576** | 35.36 | 35.74 | 35.92 | 36.42 | **36.54** |
| **Crew704x576** | 36.31 | 36.98 | 35.81 | 36.56 | **37.03** |
| **Calendar 1280x720** | 28.10 | 28.25 | 27.64 | 28.32 | **28.54** |
| **Raven 1280x720** | 42.52 | 43.68 | 42.63 | 43.03 | **43.91** |
| **Average** | 32.31 | 32.81 | 32.39 | 32.70 | **33.29** |

The subjective qualities of test images for these methods are demonstrated by News and Football in Fig.5 and Fig.6. The performance of using bilinear interpolation is similar to that of using bicubic interpolation as shown in (b) and (d). Both of the methods tend to blur the details and generate jaggies throughout the whole image, so the visual qualities are inferior to others.

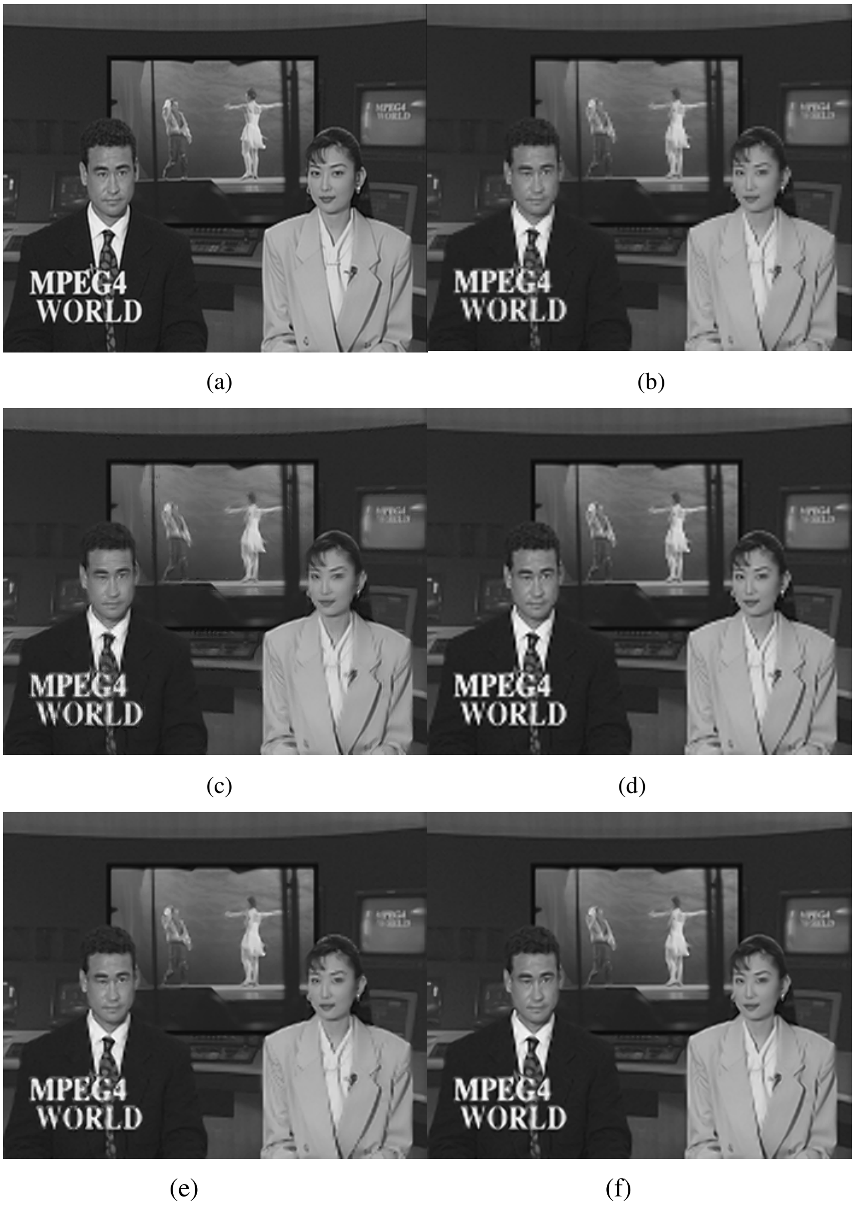

**Fig. 5.** Comparison of methods on News. (a)original image (b)bilinear (c)method in [8] (d)bicubic (e)method in [11] (f)proposed.

In Fig.5, differences among (c), (e) and (f) are easily discovered by evaluating the text quality of "MPEG4 WORLD" in the bottom left of the images. The text deformation in (c) is probably caused by the incorrect estimation of covariance [9, 10], which may generate speckle artifacts around high contrast curvatures. That is the reason that the PSNR measure for method [8] is the second lowest on average, next

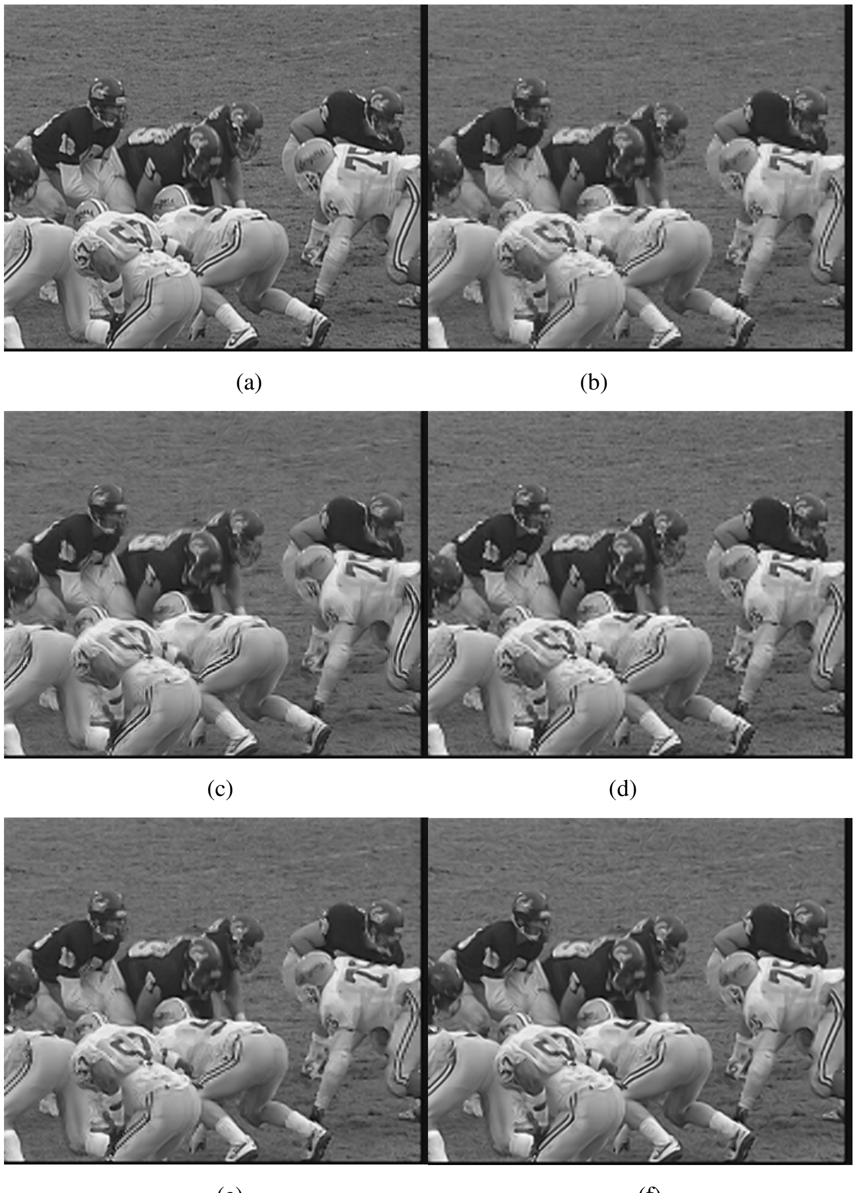

**Fig. 6.** Comparison of different methods on Football. (a) original image (b) bilinear (c) method in [8] (d) bicubic (e) method in [11] (f) proposed.

only to the bilinear interpolation. Fig.5 (e) presents “ringing” around the texts and the contents are discontinuous. A possible explanation for the undesired artifacts is that the method of [11] fails to catch the changing trend of nearby area structure as a whole since its LMMSE estimation is only effective in limited area. As a result, its PSNR ranks in between bicubic and the method of [8], while the same conclusion is also drawn in [10]. However, our proposed method produces favorable image (f) with less artifacts, the most noticeable places are in texts and female’s collar.

The same situation occurs in Fig.6 as well. The reconstruction for parallel stripes on the sport pants is a good case to evaluate different interpolation methods, because the parallel stripes consist of multiple characteristics such as thin lines, step edges and non-directional curves, any of which is a serious challenge for image interpolation. The proposed method succeeds in reconstructing a fine image (f) with continuous lines and smooth edges rather than other images (a~e) consisting of jaggies, ringings and distortion artifacts.

## 4 Hardware Architecture

Except for the high reconstruction quality achieved by our method, low complexity is also an important factor. Based on our method, a novel hardware architecture is proposed in Fig.7. As mentioned in Section 2, the whole process is divided into two stages: calculating all the gray pixels (Fig.1) in the first stage and completing the others in the second stage. However, unlike the stage separating method in our algorithm, extra memory for frame buffering between the two stages is not necessarily required. The first pipeline stage processes pixels one by one and directly stores them in Parallel Memory Unit in Second-stage; the Second-stage starts its process while all the required pixels arrive.

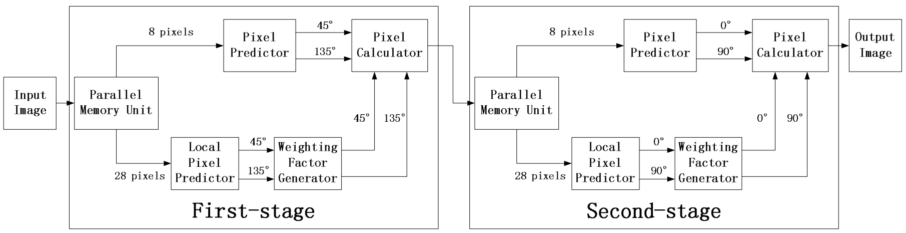

**Fig. 7.** Block diagram of proposed hardware architecture

The First-stage is composed of five blocks: Parallel Memory Unit, Missing Pixel Predictor, Local Pixel Predictor, Weighting Factor Generator, and Pixel Calculator. According to the data dependency deduced from Fig.1, Parallel Memory Unit composes of a six-line-buffer array for buffering input image pixels line by line. And the pixels are pre-fetched one at a time from each buffer by the Missing Pixel Predictor to warrant no conflict in memory accessing, then the predicted values for both directions are generated and stored in registers temporarily. At the same time, Local Pixel Predictor generates two groups of predicted values and sends them to the next unit for weighting factor generation. The Pixel Calculator produces final pixel value by the time that both the weights and the predictions for current missing pixel are available. For each new pixel processing, twenty eight original pixels are required in total except the overlapped ones. The Pixel Calculator requires two multipliers for weighted summation operation (3). Weighting Factor Generator (Fig.8) needs four multipliers for SF enhancement (8, 9) when SF=3 (recommended value referring to PSNR change trend) and one divider for the following weight assignment (10). In addition, the consumption of multipliers and divider can be totally avoided by configuring a lookup table directly indexed by prediction errors. As an optimized result, main resources consumed in the First-stage are only two multipliers in Pixel Calculator since the rest modules are only composed of shifters and adders. It is estimated that the proposed method consumes about 1% of hardware resources that would otherwise required in method [8].

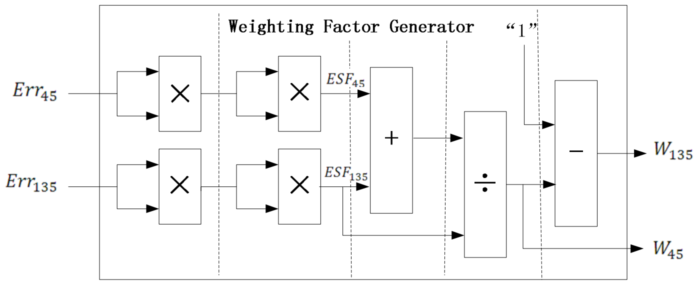

**Fig. 8.** Block diagram of Weighting Factor Generator

The architecture and operation process of Second-stage are quite similar to that of First-stage, which are only distinguished by the memory accessing pattern if we rotate the interpolation model by π/4 clockwise (curved lines in fig.1).

Because of the intrinsic pipeline nature of the architecture, it is found that our proposed design can easily support the up-scaling to the UHD@30fps resolution, which would require the operating clock frequency of 297MHz.

## 5 Conclusion

A HW efficient image up-scaling method is introduced to solve the contradiction between algorithm complexity and visual quality. Specifically, we exploit local structure by a preset interpolation model to eliminate the computation brought by either edge identification or auto-regression. The edge Sensitive Factor performs well in edge regions while suppressing the artifacts and preserving the sharpness. The experimental results have proven an overall superiority through objective and subjective measurements. Finally, the advantage in much reduced HW resource and efficient architecture makes the proposed method appropriate for real-time imaging applications, even for UHD applications.

**Acknowledgements.** This work is partially supported by grants from the Chinese National Natural Science Foundation under contract No.61171139 and No. 61035001, and National High Technology Research and Development Program of China (863 Program) under contract No.2012AA011703.

## References

1. Wünschmann, J., Zanker, S., Günter, C., Rothermel, A.: Reduction of Computational Cost for High Quality Video Scaling. IEEE Trans. on Consumer Electronics 56(4), 2584–2591 (2010)
2. Wang, Z., Zhai, J., Zhou, M.: A Fast Autoregression Based Image Interpolation Method. In: IEEE International Conference on Networking, Sensing and Control, pp. 1400–1404 (2008)
3. Gonzalez, R.C., Woods, R.E.: Digital Image Processing. Publishing House of Electronics Industry, Reading (2010)
4. Hou, H.S., Andrews, H.C.: Cubic splines for image interpolation and digital filtering. IEEE Trans. on Signal Processing 26, 508–517 (1978)
5. Jensen, K., Anastassiou, D.: Subpixel edge localization and the interpolation of still images. IEEE Trans. on Image Processing 4(3), 285–295 (1995)
6. Chen, M., Huang, C., Lee, W.: A fast edge-oriented algorithm for image interpolation. Image and Vision Computing 23, 791–798 (2005)
7. Raghupathy, A., Chandrachoodan, N.: Algorithm and VLSI Architecture for High Performance Adaptive Video Scaling. IEEE Trans. on Multimedia 5(4), 489–502 (2003)
8. Li, X., Orchard, M.T.: New edge-directed interpolation. IEEE Trans. on Image Processing 10(10), 1521–1527 (2001)
9. Zhang, X., Ma, S., Zhang, Y., Zhang, L., Gao, W.: Nonlocal Edge-directed Interpolation. In: Muneesawang, P., Wu, F., Kumazawa, I., Roeksabutr, A., Liao, M., Tang, X. (eds.) PCM 2009. LNCS, vol. 5879, pp. 1197–1207. Springer, Heidelberg (2009)
10. Zhang, X., Wu, X.: Image Interpolation by Adaptive 2-D Autoregressive Modeling and Soft-Decision Estimation. IEEE Trans. on Image Processing 17(6), 887–896 (2008)
11. Zhang, L., Wu, X.: Image interpolation via directional filtering and data fusion. IEEE Trans. on Image Processing 15(8), 2226–2238 (2006)