2011 Sixth International Conference on Image and Graphics

Image Interpolation Using Autoregressive Model and Gauss-Seidel Optimization

Ketan Tang, Oscar C. Au, Lu Fang, Zhiding Yu, Yuanfang Guo
Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology*

Abstract

In this paper we propose a simple yet effective image interpolation algorithm based on autoregressive model. Unlike existing algorithms which rely on low resolution pixels to estimate interpolation coefficients, we optimize the interpolation coefficients and high resolution pixel values jointly from one optimization problem. Although the two sets of variables are coupled in the cost function, the problem can be effectively solved using Gauss-Seidel method. We prove the iterations are guaranteed to converge. Experiments show that on average we have over 3dB gain compared to bicubic interpolation and over 0.1dB gain compared to SAI.

1 Introduction

Image interpolation is to get a High Resolution (HR) image from a corresponding Low Resolution (LR) image through interpolation technique. Popular interpolation methods are bilinear and bicubic interpolation. Although their complexity is relatively low, they have common drawback that they cannot adapt to varying pixel structures in an image because of the use of constant interpolators. As a result, they suffer from some inherent defects such as staircase effect, blurred details, and ringing artifacts.

To solve the problem, many sophisticated adaptive image interpolation methods have been proposed in recent years. To preserve edge structures, Li and Orchard propose to estimate the covariance of HR image from the covariance of the LR image, and then interpolate the missing pixels based on the estimated covariance [4]. Zhang and Wu propose to interpolate a missing pixel in multiple directions, and then fuse the directional interpolation results by minimum mean square-error (MMSE) estimation [3]. In [6] Zhang and Wu propose a Soft-decision Adaptive Interpolation (SAI) method, in which they model an image using a piecewise 2-D autoregressive (PAR) model, and recover the HR image block by block using MMSE technique. In [5] they apply PAR model on image compression and get promising results.

*This work has been supported in part by the Research Grants Council (GRF 610109) and the Innovation and Technology Commission (GHP/048/08) of Hong Kong, China

In this paper we model an image using piecewise 2-D autoregressive model similar to SAI [6]. The problem with SAI is that, the authors assume the correlation between pixels is unchanged in different scales, and then estimate the coefficients using LR pixels. This assumption is too strong that in many cases it does not hold. We propose a simplified optimization problem in which the interpolation coefficients and the HR pixel values are jointly estimated. We find that it is possible to discard the correlation assumption and still be able to solve the problem. The idea is to use Gauss-Seidel method to iteratively solve the interpolation coefficients and HR pixel values, instead of estimating coefficients from only LR pixels. By doing so we can estimate the interpolation coefficients and HR pixel values more accurately and robustly. Since our method is based on AutoRegressive model and Gauss-Seidel optimization, we refer to our method as **ARGS**.

2 Image interpolation using autoregressive model

Similar as in [6], we model an image with a piecewise autoregressive (PAR) process

$$X(i,j) = \sum_{(m,n)\in T} \alpha(m,n) X(i+m, j+n) + v_{i,j} \quad (1)$$

where T is a local window, $v_{i,j}$ is a random perturbation independent of spatial location and the image signal. Let I_h be the HR image to be estimated by interpolating the observed LR image I_l, which is a down sampled version of the HR image by a factor of two. Let $x_i \in I_l$ and $y_i \in I_h$ be the pixels of images I_l and I_h, $y_{i\diamond t}$ $(t = 1, 2, \dots)$ be the neighbors of pixel location

978-0-7695-4541-7/11 $26.00 © 2011 IEEE
DOI 10.1109/ICIG.2011.155

i in the HR image. Note that $x_i \in I_l$ implies $x_i \in I_h$, i.e. all LR pixels are naturally HR pixels.

The recovery process is done in local windows. Fig. 1 shows an example window, in which the dark dots are LR pixels, denoted by x, the white and dark dots are HR pixels, denoted by y. The problem is formulated as

$$\begin{aligned} \min_{\{y,a,b\}} \quad & F(y,a,b) = \\ & \sum_{i \in W} \left(\|y_i - \sum_t a_t y^{(8)}_{i \diamond t}\|^2 + \lambda \|y_i - \sum_t b_t y^{(4)}_{i \diamond t}\|^2 \right) \\ \text{s.t.} \quad & Sy = x \end{aligned} \tag{2}$$

where W is the window of HR pixels under consideration, which is the dotted line rectangle in Fig. 1; λ controls the importance of the horizontal and vertical correlation over diagonal correlation. Matrix S selects the HR pixels in W that are also in the LR image lattice. Note that this formulation is different from Zhang's formulation [6] in that we treat all pixels as HR pixels in the cost function, and LR pixels x serves as an equality constraint. By doing so the problem looks simpler and easier to analyze.

3 Joint optimization using Gauss-Seidel method

In general (2) is not a convex problem on $\{y, a, b\}$. However, if we fix y, then it is a convex problem on $\{a, b\}$, and vice versa. This observation enlighten us to solve the problem using Gauss-Seidel method. This section gives the iteration steps for jointly solving y and $\{a, b\}$ using Gauss-Seidel method, and the convergence proof of the iterations.

3.1 Gauss-Seidel iterations

Gauss-Seidel method is to alternatively fix one set of variables and optimize on the other set. The iterative equations of this problem are as follows.

$$\{a^{(n+1)}, b^{(n+1)}\} = \arg\min_{a,b} F(y^{(n)}, a, b) \tag{3}$$

$$y^{(n+1)} = \arg\min_{y} F(y, a^{(n+1)}, b^{(n+1)}) \tag{4}$$

Initial value of y can be obtained by bicubic interpolation.

For (3), since a and b are naturally decoupled, it can be divided into two sub-problems, and a and b have closed form solutions:

$$\begin{aligned} a^{(n+1)} &= (A^T A)^{-1} A^T u^{(n)} \\ b^{(n+1)} &= (B^T B)^{-1} B^T u^{(n)} \end{aligned}$$

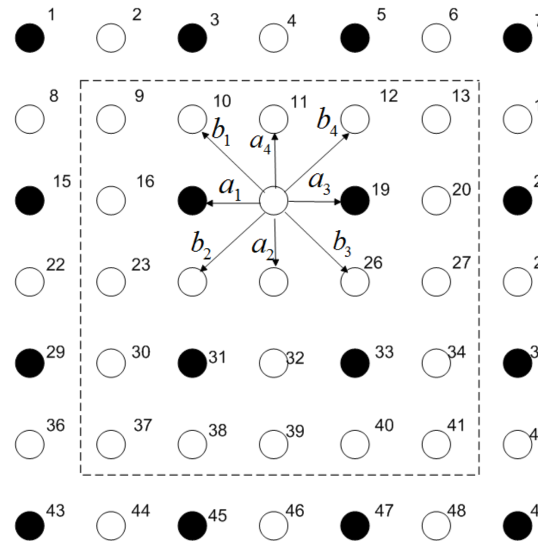


Figure 1: Spatial configuration of ARGS. The dark dots are LR pixels, the white dots are HR pixels. Note that LR pixels are also HR pixels.

where u is the vector of HR pixels that are the inside W. The ith row of matrix A consists of the 8-connected neighbors $y^{(8)}_{i \diamond t}$ of y_i, $t = 1, 2, 3, 4$. The ith row of matrix B consists of the 4-connected neighbors $y^{(4)}_{i \diamond t}$ of y_i, $t = 1, 2, 3, 4$. For the ease of representation we denote $a^{(n+1)}$ and $b^{(n+1)}$ by $\hat{a}$ and $\hat{b}$ in the following.

Similar to [5], the weighting factor λ is determined by the ratio of the fitting errors:

$$\lambda = \frac{e_1}{e_2}$$

where $e_1 = \sum_{i \in W} \|y_i - \sum_{1 \le t \le 4} \hat{a}_t y^{(8)}_{i \diamond t}\|^2$ and $e_2 = \sum_{i \in W} \|y_i - \sum_{1 \le t \le 4} \hat{b}_t y^{(4)}_{i \diamond t}\|^2$ are the fitting errors.

For (4), by defining $y = \begin{bmatrix} u \\ v \end{bmatrix}$, $C = \begin{bmatrix} C_1 \\ \lambda C_2 \end{bmatrix} = \begin{bmatrix} D_1 & D_2 \end{bmatrix}$, $S = \begin{bmatrix} S_1 & S_2 \end{bmatrix}$, we have

$$\begin{aligned} y^{(n+1)} = \quad & \arg\min \quad \|C_1 y\|^2 + \lambda \|C_2 y\|^2 \triangleq \|Cy\|^2 \\ & \triangleq \|D_1 u + D_2 v\|^2 \\ \text{s.t.} \quad & S_1 u = x \end{aligned} \tag{5}$$

where v is the vector of pixels outside W, and

$$C_1(i,j) = \begin{cases} 1, & \text{if } y_j \text{ is the } i\text{th pixel inside } W \\ -\hat{a}_t, & \text{if } y_j \text{ is the neighbor } y_{i\diamond t}^{(8)} \text{ of } y_i \\ 0, & \text{otherwise} \end{cases}$$

$$C_2(i,j) = \begin{cases} 1, & \text{if } y_j \text{ is the } i\text{th pixel inside } W \\ -\hat{b}_t, & \text{if } y_j \text{ is the neighbor } y_{i\diamond t}^{(4)} \text{ of } y_i \\ 0, & \text{otherwise} \end{cases}$$

$$S(i,j) = \begin{cases} 1, & \text{if } y_j \text{ is the } i\text{th LR pixel inside } W \\ 0, & \text{otherwise} \end{cases}$$

D_1 consists of the columns of C corresponding to u, D_2 consists of the remaining columns of C corresponding to v. S_1 and S_2 are constructed similarly. If we use Matlab command to construct D_1 and D_2, then $D_1 = C(:, index_u)$ and $D_2 = C(:, index_v)$, where $index_u$ and $index_v$ are the indexes for u and v respectively.

We divide y into inner pixel vector u and outer pixel vector v because in the cost function of (2) the subtractions are centered at pixels in u. As a result, only the estimation of u is reliable. Thus we estimate v using bicubic interpolation or previously recovered result, and solve u from (5). And only pixels in u are updated. As shown in §10.1.1 of [2], this equality constrained quadratic optimization can be effectively solved with following KKT system

$$\begin{bmatrix} D_1^T D_1 & S_1^T \\ S_1 & 0 \end{bmatrix} \begin{bmatrix} u \\ \nu \end{bmatrix} = \begin{bmatrix} -D_1^T D_2 u \\ x \end{bmatrix}$$

where ν is Lagrange multiplier. Equivalently we have closed form solution of u:

$$\begin{bmatrix} u \\ \nu \end{bmatrix} = \begin{bmatrix} D_1^T D_1 & S_1^T \\ S_1 & 0 \end{bmatrix}^{-1} \begin{bmatrix} -D_1^T D_2 u \\ x \end{bmatrix}$$

3.2 Convergence proof

The strength of Gauss-Seidel method is that we do not need to compute any subgradients, so it is easy to implement. Now it remains to show that the iterations ((3) and (4)) converge.

As shown in Proposition 2.7.1 of [1], for the following problem

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & x \in X_1 \times X_2 \times \cdots \times X_m, \end{aligned}$$

the Gauss-Seidel iteration

$$x_i^{k+1} = \arg\min_{\xi \in X_i} f(x_1^{k+1}, \ldots, x_{i-1}^{k+1}, \xi, x_{i+1}^k, \ldots, x_m^k),$$

will converge as long as the above minimum is uniquely obtained. For our problem, as long as the two minimum

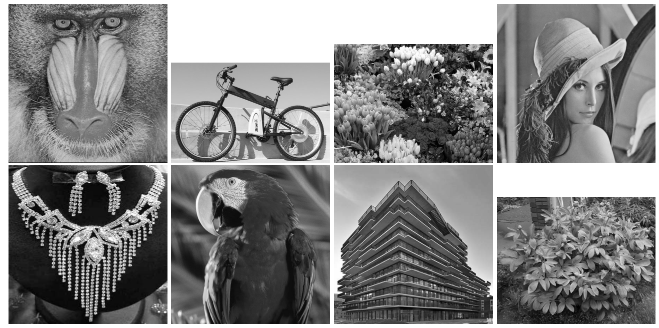
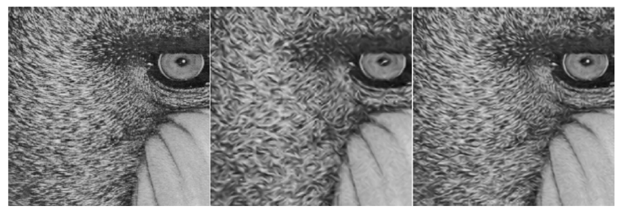


Figure 2: Example images.



(a) Original (b) SAI[6] (c) ARGS

Figure 3: Comparison on Baboon image. The image in (b) is photocopied from the paper [6].

(3) and (4) are uniquely obtained at each iteration, then convergence is guaranteed. In this problem, since (3) and (4) are all quadratic minimization, we have closed form solution for each of them, which is unique. Thus the convergence of our Gauss-Seidel iteration is guaranteed.

4 Experiment results

Extensive experiments are conducted to evaluate the proposed interpolation algorithm. Fig. 2 shows some of the example images we use. Most of the images contains complex textures which are difficult for interpolation. Similar to SAI ([6]), the LR image is produced by directly downsampling the original HR image by a factor of two (ref. Fig. 1). The binary executable program of SAI is kindly provided by the author[1].

Table. 1 shows the PSNR values for the example images. From the table we can see that on average ARGS has over 3 dB gain compared to bicubic interpolation. Note that here although some images have same names as in [6], they are actually different images. Thus it is not strange that the bicubic PSNR values are different from that of [6]. Fig. 4 shows comparison with bicubic results for image “building”. We can see that ARGS performs much better than bicubic interpolation when

[1] http://www.ece.mcmaster.ca/~xwu/

Table 1: PSNR (dB) comparison of ARGS and bicubic interpolation and SAI. **gain1** is the gain compared to bicubic, **gain2** is the gain compared to SAI.

Image	Bicubic	SAI	ARGS	gain1	gain2
baboon	22.12	23.61	23.68	1.56	0.07
bike	22.79	25.99	26.28	3.49	0.29
flower	20.50	22.65	22.66	2.16	0.01
lena	30.13	34.76	34.73	4.60	-0.04
necklace	19.51	21.96	22.08	2.57	0.12
parrot	30.64	34.09	34.29	3.65	0.20
building	22.74	26.97	27.36	4.63	0.40
tree	25.52	29.13	28.94	3.42	-0.20
Average	-	-	-	3.26	0.11

(a) Bicubic

(b) ARGS

Figure 4: Comparison with bicubic results.

directional patterns exist in the image.

ARGS performs particularly well on texture region compared to SAI [6]. Fig. 3 shows the results on Baboon. We can see that SAI generates a lot of fake textures, however ARGS recovers the HR image quite well.

5 Conclusion

In this paper we propose a simple yet effective interpolation algorithm based on autoregressive model. Unlike existing algorithms, we do not need to assume that the pixel correlation does not change under different scales. Instead, we optimize the interpolation coefficients and HR pixel values jointly from one optimization problem. Although the problem looks more difficult than before, it turns out that Gauss-Seidel method can effectively solve it, and we prove the convergence is guaranteed. As long as we can find a close initial value of HR pixel values - which could be the bicubic interpolation result - local optimality is guaranteed. Experiments show that the algorithm converges in two or three iterations in complex regions, and in only one step in flat region, thus this algorithm does not increase much computational complexity. We have obtained a simpler model compared to SAI, yet experiments show that on average we have over 3dB gain compared to bicubic interpolation, and over 0.1dB gain compared to SAI.

References

[1] D. P. Bertsekas. *Nonlinear Programming.* Massachusetts Institute of Technology, 2 edition, 2003.

[2] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

[3] Z. Lei and W. Xiaolin. An edge-guided image interpolation algorithm via directional filtering and data fusion. *Image Processing, IEEE Transactions on*, 15(8):2226–2238, 2006.

[4] X. Li and M. T. Orchard. New edge-directed interpolation. *Image Processing, IEEE Transactions on*, 10(10):1521–1527, 2001.

[5] X. Wu, X. Zhang, and X. Wang. Low bit-rate image compression via adaptive down-sampling and constrained least squares upconversion. *Image Processing, IEEE Transactions on*, 18(3):552–561, 2009.

[6] X. Zhang and X. Wu. Image interpolation by adaptive 2-d autoregressive modeling and soft-decision estimation. *Image Processing, IEEE Transactions on*, 17(6):887–896, 2008.