FAST IMAGE INTERPOLATION WITH DECISION TREE

Jun-Jie Huang and Wan-Chi Siu

Centre for Signal Processing, Department of Electronic and Information Engineering
The Hong Kong Polytechnic University

ABSTRACT

This paper proposes a fast image interpolation method using decision tree. This new fast image interpolation with decision tree (FIDT) method can achieve state-of-the-art image interpolation performance and requires only 10% computational time of the soft adaptive interpolation (SAI) method. During training, the proposed method recursively divides the training data at a non-leaf node into two child nodes according to the binary test which can maximize the information gain of a division. At the end, for each of the leaf node, a linear regression model is learned according to the training data at that leaf node. In the image interpolation phase, input image patches are passed into the learned decision tree. According to the stored binary test at each non-leaf node, each input image patch will be classified into its left or right child node until a leaf node is reached. The high-resolution image patch of the input image patch can then be predicted efficiently using the learned linear regression model at the leaf node.

Index Terms— Image interpolation, decision tree, classification, regression and training

1. INTRODUCTION

The objective of image interpolation is to generate a high-resolution (HR) image from a low-resolution (LR) image, where the LR image is obtained by direct downsampling (without anti-aliasing pre-filtering) the original HR image. Image interpolation algorithms help to break the inherent limitation of the low-resolution imaging and better utilize the increasing resolution of the displays. They have wide applications, for example, HDTV, image resizing, image coding, surveillance systems, medical imaging and face recognition, etc.

The existing image interpolation methods can be categorized mainly into two classes: conventional polynomial based interpolation methods [1-4] and edge-directed interpolation methods [5-14].

The conventional polynomial based interpolation methods [1-4] interpolate the unknown pixels using the known surrounding pixels by non-adaptive [1] [2] or adaptive [3] [4] polynomial linear filters. For real-time applications, the conventional polynomial based interpolation methods (e.g. bicubic interpolation) are often adopted because their low computational complexity. However, they usually produce annoying blurry edges and jagging artifacts.

The edge-directed interpolation methods [5-15] explicitly or implicitly utilize the edge directional information to produce sharper edges and fewer artifacts. The explicit edge-directed methods [5-8] estimate the edge orientation and position and then interpolate the missing pixels using the edge information. The explicit methods are limited by the edge detection accuracy. To overcome this problem, implicit edge-directed methods [8-15] are proposed. The new edge-directed interpolation (NEDI) method [9] proposed to make use of the geometric duality property. The missing HR pixels are predicted using the classical Wiener filter by the HR pixel covariance which is estimated by LR pixel covariance. The directional filtering and data fusion (DFDF) method [13] fuses two noisy directional interpolation results by linear minimum mean square error estimation. The soft adaptive interpolation (SAI) methods [14] [15], which use the block-based soft-decision estimation, increase the orders of coefficients compared with NEDI and add feedback terms in the objective function, achieve great improvement in PSNR. The sophisticated edge-directed image interpolation algorithms can provide satisfactory results, however, often require huge computations.

In the literature, the decision tree [16-19] has been widely utilized for real-time applications, such as, object detection [20], fast keypoint detection [21], fast edge detection [22], image classification [23], image segmentation [24] etc. In this paper, we propose to apply decision tree for fast image interpolation. Different from the implicit edge-directed image interpolation methods, the proposed method does not learn a local statistical model from the input image for interpolation, but classifies the image patch into one of the classes in which a pre-learned image interpolation linear regression model is stored. Using the pre-learned linear regression model, the HR image patch can be easily predicted by multiplying the linear regression model with the LR image patch vector. Besides, the image patch detection and classification process is fast owing to the simple binary test used in the decision tree.

This paper is organized as follows. In Section 2, we briefly introduce the decision tree. In Section 3, we describe

how to apply decision tree for image interpolation. Section 4 presents the experimental results and Section 5 concludes the paper.

2. DECISION TREE

Decision tree was firstly proposed by Breiman et al. [16] in 1984, and now is a commonly used data mining algorithm. The general idea of decision tree is to predict an unknown input sample according to several known training samples. A decision tree recursively partitions a region in the training data space into two child regions according to the information gain of the partition and assign a classification or regression model to the region which cannot be further divided.

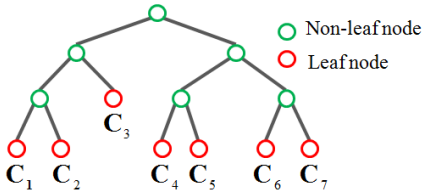

Fig.1. A decision tree.

A decision tree ***T*** has non-leaf nodes and leaf nodes as shown in Fig.1. Each non-leaf node classifies the input sample **x** into its left or right child node according to the result of the split function $h(\mathbf{x},\theta)$ which uses the stored binary test θ. The predictions are performed on leaf nodes using the trained classification or regression model (e.g. $\mathbf{C}_1$, $\mathbf{C}_2$, …) associated in each leaf node.

The binary test adopted in this paper is specified by three parameters $\theta = \{p_1, p_2, \tau\}$. The first two parameters p_1 and p_2 represent two positions on the feature vector of the input sample, and τ is a threshold value. The split function is defined as below:

$$h(\mathbf{x},\theta) = \begin{cases} 0, \text{ if } \mathbf{x}(p_1) < \mathbf{x}(p_2) + \tau \\ 1, \quad \text{otherwise.} \end{cases} \tag{1}$$

2.1. Decision Tree Training

The decision tree is trained in a recursive manner. For each non-leaf node j, the objective is to find the best binary test θ_j, which can maximize the information gain among K randomly selected binary tests. For each binary test, all the three parameters are randomly generated within a range.

$$\begin{cases} 0 \le p_1, p_2 \le D-1 \\ 0 \le \tau \le 255 \end{cases}, \tag{2}$$

where D is the dimension of the feature vector.

Let us define the training data at node j as S_j. The binary test can split the training data at node j into its left child node S_j^L or right child node S_j^R.

$$\begin{aligned} S_j^L &= \{x \in S_j \mid h(\mathbf{x},\theta) = 0\} \\ S_j^R &= S_j \setminus S_j^L. \end{aligned} \tag{3}$$

The information gain is defined to evaluate the effectiveness of a binary test:

$$I(S,\theta) = H(S) - \sum_{n\in\{L,R\}} \frac{|S^n|}{|S|} H(S^n), \tag{4}$$

where $H(S)$ is the entropy for the training data S to be defined in (12).

The randomly generated binary tests tend to split the training data into two unbalanced child nodes (i.e. one child node occupies the vast majority of the training data). According to (4), the little drop of the entropy at the large child node would mask the great rise of the entropy at the small child node, while large entropy means a higher level of dissimilarity among the training data. This would affect the training effectiveness. To solve this problem, we propose to insert a constraint to select the binary test. Only the binary test which fulfills (5) will be picked into the K randomly selected binary tests. The constraint parameter λ is selected as 0.75 by cross-validation.

$$\max(|S^L|, |S^R|) \times \lambda \le \min(|S^L|, |S^R|). \tag{5}$$

If the highest information gain $I(S, \theta_j)$ is larger than a threshold I^T, the training data at node j will be mapped into its two child nodes according to θ_j. Besides, the binary test θ_j will be stored associated with node j.

There are three situations to declare a node as the leaf node: (i) $I(S, \theta_j)$ is smaller than I^T; (ii) the number of training data of a node is less than the minimum number of training data N^T for further split; (iii) the depth of a node reaches the maximum tree depth D_{max}.

For each leaf node, a regression model will be constructed for prediction.

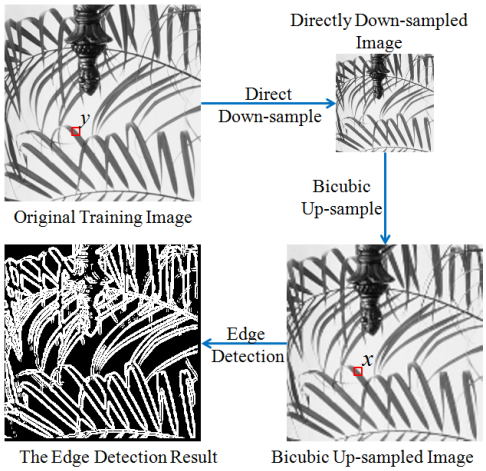

Fig.2. Training data sampling process.

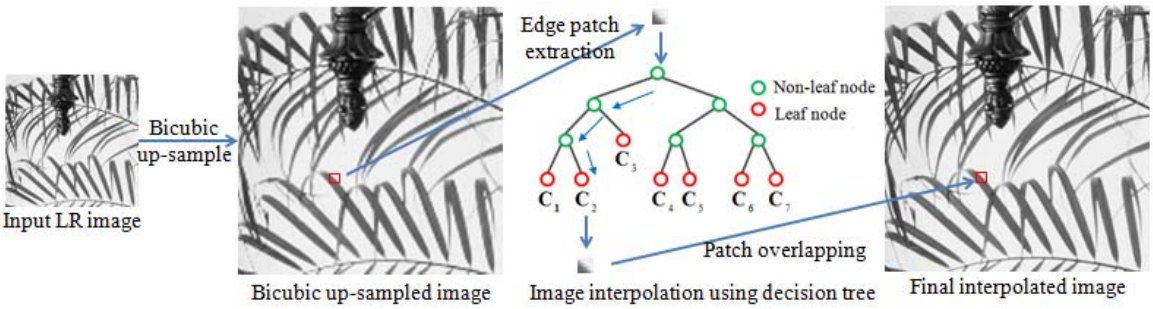

Fig.3. Image interpolation using decision tree.

3. IMAGE INTERPOLATION USING DECISION TREE

The image interpolation is regarded as a regression process, i.e. relating the LR image patch to its desired HR image patch.

According to extremely randomized trees [18], for regression problem, the number of randomly selected binary tests K should be:

$$K = D(D-1), \tag{6}$$

where D is the dimension of the feature vector.

The decision tree is good at dealing with multiple features, however, only the intensity feature is adopted, for the sake of simplicity. For further investigation, more features can be attached.

Each training data P is a LR-HR image patch pair.

$$P = (\mathbf{x}, \mathbf{y}), \tag{7}$$

where $\mathbf{x} \in R^d$ is the LR image patch vector sampled from the bicubic upsampled image, $\mathbf{y} \in R^d$ is the corresponding HR image patch vector of $\mathbf{x}$ sampled from the original HR image and $\sqrt{d}$ is the patch size.

Considering the training efficiency, only the patches from the edge areas are sampled, as shown in Fig.2. The edge areas are the positions with edge magnitude larger than 60 after performing the Canny edge detection.

Assuming there are l training data reached at node j:

$$S_j = \{P_i \mid i = 1,...,l\}. \tag{8}$$

All the LR image patch feature vectors and HR image patch feature vectors in S_j can be grouped into matrix form $\mathbf{X} \in R^d \times R^l$ and $\mathbf{Y} \in R^d \times R^l$. The relationship between $\mathbf{X}$ and $\mathbf{Y}$ in node j is model by a regression model $\mathbf{C}_j$ which can minimize the mean squared error between the reconstructed HR image patch, ($\mathbf{C}_j\mathbf{X}$), and the ground truth HR image patch, $\mathbf{Y}$, within the training data on node j.

$$\mathbf{C}_j = \arg\min \| \mathbf{Y} - \mathbf{C}_j \mathbf{X} \|^2 . \tag{9}$$

And (9) can be simply solved by the least squares method with a closed form solution:

$$\mathbf{C}_j = \mathbf{Y}\mathbf{X}^T (\mathbf{X}\mathbf{X}^T)^{-1}. \tag{10}$$

With the obtained regression model between the LR image patch and the HR image patch, the predicted HR image patch $\mathbf{y}^R$ is reconstructed using the regression model $\mathbf{C}$ and the LR image patch $\mathbf{x}$ as follows.

$$\mathbf{y}^R = \mathbf{C}\mathbf{x}. \tag{11}$$

The entropy $H(S)$ defined in (4) is the mean squared error between the predicted HR image patch and its corresponding original HR image patch.

$$H(S) = \frac{1}{|S|} \sum_{\mathbf{y} \in S} \| \mathbf{y}^R - \mathbf{y} \|^2. \tag{12}$$

Each leaf node will store the regression model constructed by the training data reached at this leaf node as (10).

Fig.3. shows the procedures for image interpolation using decision tree. The input LR image will be firstly up-sampled by bicubic interpolation. Each image patch on the edge area will be collected and vectorized, then passed into the learned decision tree. According to the binary tests stored at the non-leaf nodes, the input image patch will be recursively classified into left or right child node until a leaf node is reached. Using the regression model at the reached leaf node, the predicted HR image patch of the input patch can be obtained by (11). As the input patches of the decision tree are overlapped with each other and the patches in smooth area are interpolated by bicubic interpolation, each pixel will have multiple prediction values (from decision tree prediction or bicubic interpolation). The final interpolated image is obtained by averaging all the predictions.

4. EXPERIMENTAL RESULTS

To train the decision tree, 22 images from training images of [25] are selected as shown in Fig.4 The patch size was selected as 5×5, the maximum tree depth D_{max} is 12, the minimum number of training data N^T for further split is 200 and the threshold for information gain I^T is 0.01. Around 830000 LR-HR patch pairs are used for training. After training, there are 2958 leaf nodes in the decision tree.

As shown in Fig.5, 8 commonly used images are selected for testing. The PSNR is adopted to evaluate the objective comparison between the proposed method and the competing methods, including NEDI [9], DFDF [13], SAI [14]. From Table 1, we can find that the PSNR of the

Images	Bicubic (C)		NEDI [9] (Matlab)		DFDF [13] (Matlab)		SAI [14] (C)		Proposed FIDT (C)	
	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time
kodim03	34.99	0.003	35.27	12.076	35.25	8.475	35.79	2.584	35.77	0.190
kodim05	27.29	0.003	27.29	13.033	27.40	7.698	28.34	2.631	28.42	0.461
kodim07	34.50	0.004	34.19	12.388	34.58	7.633	35.53	2.648	35.72	0.279
kodim08	23.68	0.003	23.40	13.222	23.74	7.655	23.81	2.642	23.95	0.463
kodim15	33.68	0.004	33.82	12.439	33.77	7.632	34.15	2.614	34.28	0.193
kodim19	28.24	0.004	27.59	12.841	28.38	7.593	27.97	2.609	28.67	0.269
kodim20	32.06	0.004	32.53	12.162	32.49	7.969	32.98	2.349	33.15	0.208
kodim23	36.14	0.004	36.41	12.032	35.98	7.643	37.28	2.585	37.14	0.149
Average	***31.32***	***0.004***	***31.31***	***12.524***	***31.45***	***7.787***	***31.98***	***2.583***	***32.14***	***0.277***

Table.1. PSNR (dB) and the computational time (s) of the proposed method and other competing methods.

Fig.4. Sample training images.

Fig.5. 8 testing images. From left to right and top to bottom: *kodim03, kodim05, kodim07, kodim08, kodim15, kodim19, kodim20* and *kodim23*.

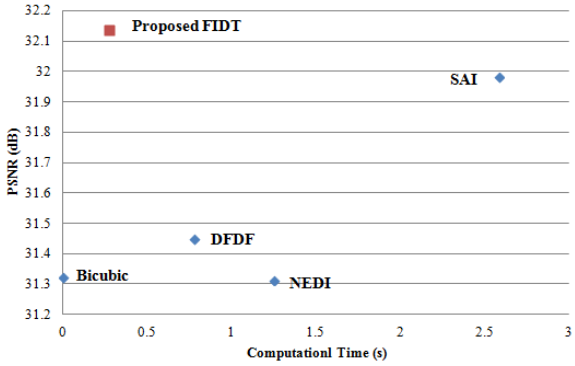

Fig.6. Approximated C computational time (s) vs. PSNR (dB) of the proposed FIDT method and other methods.

proposed fast image interpolation with decision tree (FIDT) method is 0.81 dB, 0.82 dB, 0.68 dB and 0.16 dB higher than Bicubic, NEDI, DFDF and SAI, respectively.

Since the source code from the author of NEDI and DFDF are implemented in Matlab and other methods are implemented in C, in order to compare the computational time of different methods, we assume that Matlab implementation is 10 times slower than C implementation; hence the computational time of NEDI and DFDF is scaled down by 10 times in Fig.6, even though in some cases Matlab program can be as faster as C++ program.

From Fig.6, we can find that the proposed FIDF method can achieve high quality image interpolation while use 10 percent computational time of the SAI. The reason is that the proposed FIDT method does not learn a statistical model from a local window in the input image, but classifies the input image patches into one of the learned image interpolation model. Besides, the image patch classification process is very fast, since only several pairs of pixels have to be compared. Most of the computational time is spent on the matrix multiplication between the input image patch and the learned image model. So, the overall complexity is very low for the proposed fast image interpolation with decision tree method. The high quality image interpolation results come from the good classification properties of the decision tree with the proposed constraint and benefit from the huge amount of training data.

5. CONCLUSIONS

In this paper, we have presented a fast image interpolation method with decision tree. The proposed fast image interpolation with decision tree (FIDT) method can offer state-of-the-art interpolation results as well as high computational efficiency. Different from the explicit-edge directed image interpolation methods, the proposed method uses off-line learned image interpolation models rather than on-line learning image interpolation models. The simple image patch classification process enables fast image interpolation. While the decision tree with the proposed binary test constraint guarantees the quality of training results and the image interpolation results.

For future work, random ferns can be used to replace the decision tree to explore fast image interpolation method, since the classification process is even simpler. The decision tree has a good property to be realized in parallel, so another direction is to realize image interpolation with decision tree in GPU or multi-process.

6. REFERENCES

[1] R. G. Keys, "Cubic convolution interpolation for digital image processing," *IEEE Trans. Acoust., Speech, Signal Process*., vol. 29, no. 6, pp. 1153–1160, Dec. 1981.

[2] H. S. Hou and H. C. Andrews, "Cubic splines for image interpolation and digital filtering," *IEEE Trans. Acoust., Speech, Signal Process*., vol. 26, no. 6, pp. 508–517, Dec. 1978.

[3] T. Blu, P. The´venaz and M. Unser, "Linear interpolation revitalized," *IEEE Trans. Image Process*., vol. 13, no.6,, pp. 710–719, 2004.

[4] T. Lehmann, C. Go¨nner, K. Spitzer, "Addendum: B-spline interpolation in medical image processing," *IEEE Trans. Med. Imaging*, vol. 20, no. 7, pp. 660–665, 2001.

[5] K. Jensen and D. Anastassiou, "Subpixel edge localization and the interpolation of still images," *IEEE Trans. Image Process*., vol. 4, no. 3, pp. 285–295, Mar. 1995.

[6] H. Shi and R. Ward, "Canny edge based image expansion," in Proc. IEEE Int. Symp. Circuits Syst., vol. 1, pp. 785–788, May 2002.

[7] Q. Wang and R. K. Ward, "A new orientation-adaptive interpolation method," *IEEE Trans. Image Process*., vol. 16, no. 4, pp. 889–900, Apr. 2007.

[8] D. D. Muresan, "Fast edge directed polynomial interpolation," Proceedings, pp. 990-993, *2005 IEEE International Conference on Image Processing (ICIP'05)*, vol. 2, 11-14 Sep. 2005, Genova, Italy.

[9] X. Li and M. T. Orchard, "New edge-directed interpolation," *IEEE Trans. Image Process*., vol. 10, no. 10, pp. 1521–1527, Oct. 2001.

[10] W.S. Tam, C.W. Kok and W.C. Siu, "A Modified Edge Directed Interpolation for Images," *Journal of Electronic Imaging*, vol.19 (1), 013011, pp.13011_1-20, Jan-March 2010.

[11] C.S. Wong and W.C. Siu, "Further Improved Edge-directed Interpolation and Fast EDI for SDTV to HDTV Conversion," Proceedings,*18th European Signal Processing Conference (EUSIPCO'2010)*, pp.309-313, 23-27 August, 2010, Aalborg Denmark.

[12] C.S. Wong and W.C. Siu, "Adaptive Directional Window Selection for Edge-Directed Interpolation," Proceedings, pp. 1-6, *2010 International Conference on Computer Communications and Networks (ICCCN'10)*, 2-5 August, 2010, Zurich, Switzerland.

[13] L. Zhang and X. Wu, "An edge guided image interpolation algorithm via directional filtering and data fusion," *IEEE Trans. Image Process*., vol. 15, no. 8, pp. 2226–2238, Aug. 2006.

[14] X. Zhang and X. Wu, "Image interpolation by adaptive 2D autoregressive modeling and soft-decision estimation," *IEEE Trans. Image Process*., vol. 17, no. 6, pp. 887–896, Jun. 2008.

[15] K.W. Hung and W.C. Siu, "Robust Soft-decision Interpolation using weighted Least Squares," vol.21, no.3, pp.1061-1069, March 2012, IEEE Transactions on Image Processing, USA.

[16] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, "Classification and Regression Trees," *CRC press*, 1984.

[17] L. Breiman, "Random forests," *Machine Learning*, 45.1 (2001): 5-32.

[18] P. Geurts, D. Ernst, and L.Wehenkel, "Extremely randomized trees," *Machine Learning*, 63.1 (2006):3–42.

[19] A. Criminisi, J. Shotton, and E. Konukoglu, "Decision Forests: A Unified Framework," *Foundations and Trends in Computer Graphics and Vision*, vol. 7, pp. 81-227, 2012.

[20] J. Gall, A. Yao, N. Razavi, L. Gool, and V. Lempitsky, "Hough forests for object detection, tracking, andaction recognition," *IEEE Trans. Pattern Anal. Mach. Intel*., vol. 33, no. 11, pp. 2188–2202, Nov. 2011.

[21] Lepetit, Vincent, and Pascal Fua, "Keypoint recognition using randomized trees," *IEEE Trans. Pattern Anal. Mach. Intel*., vol. 28, no. 9, pp. 1465-1479, Sept. 2006.

[22] Piotr Dollár and C. Lawrence Zitnick, "Structured Forests for Fast Edge Detection," Proceedings, pp. 1841-1848, *2013 IEEE International Conference on Computer Vision (ICCV'13)*, 1-8 Dec. 2013, Sydney, Australia.

[23] A. Bosch, A. Zisserman, and X. Munoz, "Image classification using random forests and ferns," Proceedigns, pp. 1-8, *2007 IEEE International Conference on Computer Vision (ICCV'07)*, 14-21 Oct. 2007, Rio de Janeiro, Brazil.

[24] J. Shotton, M. Johnson, R. Cipolla, "Semantic Texton Forests for Image Categorization and Segmentation," Proceedings, p. 1-8, *2008 IEEE International Conference on Computer Vision and Pattern Recognition (CVPR'08)*, 23-28 June, 2008, Anchorage, USA.

[25] J. Yang, J. Wright, T. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Trans. Image Process*., vol. 19, no. 11, pp. 2861–2873, Nov. 2010.