ADAPTABLE K-NEAREST NEIGHBOR FOR IMAGE INTERPOLATION

Karl S. Ni and Truong Q. Nguyen

ECE Dept, UCSD, La Jolla, CA 92093-0407
http://videoprocessing.ucsd.edu/

ABSTRACT

A variant of the k-nearest neighbor algorithm is proposed for image interpolation. Instead of using a static volume or static k, the proposed algorithm determines a dynamic k that is small for inputs whose neighbors are very similar and large for inputs whose neighbors are dissimilar. Then, based on the neighbors that the adaptable k provides and their corresponding similarity measures, a weighted MMSE solution defines filters specific to intrinsic content of a low-resolution input image patch without yielding to the limitations of a non-uniformly distributed training set. Finally, global optimization through a single pass Markovian-like network further imposes on filter weights. The approach is justified by a sufficient quantity of relevant training pairs per test input and compared to current state of the art nearest neighbor interpolation techniques.

Index Terms— interpolation, nearest neighbor

1. INTRODUCTION

Image interpolation relates to the problem of creating new detail from a discrete set of known points. The quality of the interpolation is usually considered subjectively, where content such as edges and textures are evaluated based on sharpness, continuity, and clarity. These properties can be generated by introducing new information to the problem in the form of a training set. Using a patch-based sliding window, the proposed algorithm interpolates images locally with a linear filter designed by the k nearest neighbors from a training set.

Standard interpolation techniques (not particularly specific to image applications) include static linear filtering such as bilinear interpolation and B-splines. It is widely acknowledged that the approaches incorrectly assume that relationships between local low and high-resolution content can be described with a single convolutional kernel. Rather, their complexity usually justifies their usage, and though numerical results are most likely incorrect, the visual quality does not overtly reflect this. In other words, the result relies on the assumption that the human visual system (HVS) is forgiving of estimation errors from reasonably designed linear filters. The assumption is fundamental to our effort because the damage of estimation errors due to insufficient training will appear perceptually mitigated.

A logical improvement on static interpolation methods would be to use unique filters adapted to their respective situations, achieved using the k nearest neighbors (k-NN) of a training set. The proposed method specially tailors filters to fit the test input based on the nearest low and high-resolution training pairs. The goal is to achieve specificity with regard to image content without any loss of generalization of application. That is, how detailed can we make an image look, and how broad a variety of images can we maintain?

This is actually a tradeoff intimately related to the quantity of training samples used per reconstruction filter. For images, more training points per filter, i.e. k large, equals more generality, meaning that errors and variations due to the training set are diminished. Alternatively, fewer training points per filter, i.e. k small, equals more specificity, meaning that the image reconstruction is clearer and more detailed. Because training points are distributed unevenly, the relevancy of the closest training points to the test input may be variable depending on the content (edges, surfaces, texture, etc.) within a single image. Should the neighbors be "close" to the test input, then fewer points are necessary because the subsequent filter design represents the test input well. Should the neighbors be "far", we obtain a large number of training points in the hopes that linear filtering masks the weaknesses of the training set.

This work proposes an adaptable k for the k-NN algorithm with special application to image interpolation, and explores the potential of the algorithm while comparing to related approaches. Sec. 2 describes the local interpolation techniques employed, i.e. the determination of k and minimum mean squared error (MMSE) filter coefficients. Then, Sec. 3 discusses a random field approximation to implement global considerations. Finally, Sec. 4 and Sec. 5 compare the algorithm to state of the art interpolation methods and state the conclusions to the introduced topics.

2. LOCAL IMAGE INTERPOLATION WITH K NEAREST NEIGHBORS

Using k-NN for superresolution is not in itself novel. Freeman's example-based approach draws from candidate nearest neighbors and chooses the "best" neighbor by heavily approximating a Markov network. Chang et. al's local linear embedding (LLE) [1] learns a k-NN defined regression after assuming certain relationships between low and high-resolution.

Neither of the approaches utilize advantages of the popular image processing technique of linear filtering, a unique asset to the proposed algorithm. Other shortcomings include issues in training set scaling; Freeman [2], through our experimentation, scales marginally according to training set size, and at times image quality even slightly deteriorates. Additionally, recent developments in [3] show that the underlying assumption of isometry in [1] (at least for Euclidean distances) between low-resolution neighbors and high-resolution neighbors is inherently false. In other words, the distance metric used by [1] at low-resolution does not correspond to its assigned high-resolution counterpart, and thus, the weights used at high-resolution are inappropriate.

While both approaches are based in lazy learning, other than its application to image interpolation, they do not improve upon the learning processes themselves by adjusting to the specific problem at hand. In hopes of observing the best traits of [1] and [2] while simulatenously rectifying their weaknesses, the proposed algorithm adapts both linear filtering and k-NN learning to the superresolution problem at hand in Sec. 2.1 and Sec. 2.2, respectively.

This work is support in part by a grant from Qualcomm, Inc.

2.1. Weighted Linear MMSE Interpolation

The k-nearest neighbor rule is among the simplest statistical learning tools in density estimation, classification, and regression. Trivial to train and easy to code, the nonparametric algorithm is surprisingly competitive and fairly robust to errors given good cross-validation procedures.

Let Ω be a training set of N low and high-resolution pairs. Then,

$$\Omega = \{(\mathbf{x}_1, \mathbf{y}), (\mathbf{x}_2, \mathbf{y}_2), \cdots, (\mathbf{x}_N, \mathbf{y}_N)\} \tag{1}$$

where $\mathbf{x}_i$ are d dimensional input vectors and $\mathbf{y}_i$ are u dimensional output vectors.

The traditional view of nearest neighbor algorithms [4] for density estimation is mathematically expressed as

$$\hat{f}(\mathbf{x}_o) = \frac{1}{N\sigma} \sum_{i=1}^{N} K\left(\frac{\mathbf{x}_o - \mathbf{x}_i}{\sigma}\right) \tag{2}$$

where K represents a kernel that integrates to one. The kernel function K of the proposed algorithm used in (2) is the RBF, stated as follows

$$K_{\mathbb{F}}(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{2\pi\|\Sigma\|} \exp\left\{d_{\mathbb{F}}(\mathbf{x}_i, \mathbf{x}_j)\right\} \leq 1 \tag{3}$$

where $(\mathbf{x}_i, \mathbf{y}_i)$, and $(\mathbf{x}_j, \mathbf{y}_i) \in \Omega$, and $d_{\mathbb{F}}(\mathbf{x}_i, \mathbf{x}_j)$ is the Mahalanobis distance or weighted Euclidean distance in feature space specified by $\frac{1}{2}(\mathbf{x}_i - \mathbf{x}_j)^T \Sigma^{-1}(\mathbf{x}_i - \mathbf{x}_j)$. Conceptually, (2) estimates $f(\mathbf{x})$ by placing kernels around every training point in order to describe a complete picture of the probability density function.

One family of generally applicable models for k-NN regression, known as *locally weighted regression* [5], defines a group of parameterized functions $g(\mathbf{x}, \boldsymbol{\beta})$, in which the output $\mathbf{y}_o$ is determined locally by functions based on how similar point $\mathbf{x}_i$ is to $\mathbf{x}_o$. The task then falls to estimating select parameters for reconstruction in (4).

$$\boldsymbol{\beta}^* = \text{argmin}_{\beta} \sum_{\mathbf{x}_i \in \text{neighborhood}} d_{\mathbb{R}}\left(g(\mathbf{x}_i, \boldsymbol{\beta}), \mathbf{y}_i\right) K\left(d_{\mathbb{F}}(\mathbf{x}_i, \mathbf{x}_t)\right) \tag{4}$$

where $d_{\mathbb{R}}$ and $d_{\mathbb{F}}$ are distance metrics in the range and feature space, respectively.

On the principle that isometry is not a realistic scenario for image superresolution [3], (4) becomes a viable alternative, where the assumption is that linear filtering yields an excellent approximation for local image construction as opposed to assuming some kind of duality between low-resolution and high-resolution manifolds in [1]. Hence, the $g(\mathbf{x}, \boldsymbol{\beta})$ in (4) can be reduced to an MMSE filter formulation, and we can find $\mathbf{y}_o$ by

$$\mathbf{y}_o = E\left[\mathbf{y}_o | \mathbf{x}_o\right] \approx g(\mathbf{x}_o, \boldsymbol{\beta}) = G\mathbf{x}_o \tag{5}$$

where G is a $u \times d$ matrix constructed by probability parameters β and neighboring low-resolution and high-resolution pairs.

The focus from (5) now becomes finding G, which is accomplished by slightly modifying traditional MMSE equations. To manage the data, let us assemble all neighboring low-resolution column vectors $\mathbf{x}_i$ and high-resolution column vectors $\mathbf{y}_i$ into X and Y matrices, respectively. From (3), we construct a matrix P for a given neighborhood of $\mathbf{x}_o$ such that if $\mathbf{p}$ is a vector of similarity measures whose i^{th} entry is the value $K_{\mathbb{F}}(\mathbf{x}_i, \mathbf{x}_o)$, then

$$P = \mathbf{1}^T \mathbf{p} \tag{6}$$

where $\mathbf{1}$ is a k dimensional column vector of all ones. Hence, P has dimensions $k \times p$.

The purpose of P is to establish a proper weighting of point $\mathbf{x}_i \in \mathcal{N}(\mathbf{x}_t)$. Since one of the arguments to the $K_{\mathbb{F}}(\mathbf{x}_i, \mathbf{x}_j)$ in (3) is always $\mathbf{x}_o$, weighting schemes usually observe the similarity between $\mathbf{x}_i$ given $\mathbf{x}_o$ as a Gaussian PDF with mean centered at $\mathbf{x}_t$ and the elements in P as how probable that neighborhood vector is relevant.

The sample autocorrelation matrix $R_{XX} = XX^T$ and cross-correlation matrix $R_{XY} = XY^T$ lead to the final expression for G in (7), which is roughly equivalent to the derivations from [6].

$$G = \left(R_{XX} \cdot P\right)^{-1} \left(R_{XY} \cdot P\right) \tag{7}$$

2.2. Choosing the Correct k

The required weighting scheme in Sec. 2.1 is a consequence that remedies the built-in error of uneven training data collection in unsupervised learning. Similarly, non-uniformly distributed training data also has implications that k^* may potentially be significantly different for any two given test points, where again, the goal is to find the right k for a desired tradeoff.

As one may guess, $k < k^*$, where k^* is the optimum value of k, overfits the training set by specializing by G too much, and the manifestation is a grainy and discontinuous image. Furthermore, if k were exceedingly small, $k \ll k^*$, G could become non-singular. This is intuitive because training points near $\mathbf{x}_o$ could be very close together causing (7) to be underdetermined. Analytically speaking, vectors in X that are too similar can mean that R_{XX} is rank deficient and thus non-invertible. This is a dilemma because while k-NN should find the most relevant data, it is designed such that the collected vectors based on $\mathbf{x}_o$ are similar to each other. Hence, though it is counterintuitive, it is important to choose a large enough neighborhood in $\mathbb{F}$ so that diversity in the $\mathcal{N}(\mathbf{x})$ exists.

To find the optimal k^*, we introduce η such that k^*, a function of $\mathbf{x}_o$ and the training set Ω, is determined by

$$\begin{aligned} k^*(\mathbf{x}, \Omega) = \text{argmin}_k \quad & \sum_{i=1}^{N} W_i(\mathbf{x}_o, \Omega, k) K(\mathbf{x}_i, \mathbf{x}) \geq \eta \\ \text{where} \quad & W_i(\mathbf{x}_o, \Omega, k) \in \{0, 1\} \end{aligned} \tag{8}$$

The expression in (8) obtains k^* by finding the minimum number of neighbors whose sum of similarity measures exceeds a threshold η. Moreover, η is a minimum bound of k since $K(\mathbf{x}_i, \mathbf{x}_o) \leq 1$ for all $\mathbf{x}_i$.

To analyze (8) for a given $\mathbf{x}_o$, if there are only a few $\mathbf{x}_i$ with high probability of being related to it, that is $\sum_i K(\mathbf{x}_i, \mathbf{x}_o)$ is small, then the proposed algorithm will need to consider more points in hopes of generalizing well. Alternatively, if there are many $\mathbf{x}_i$ that are related to $\mathbf{x}_o$, i.e. $\sum_i K(\mathbf{x}_i, \mathbf{x}_o)$ is large, it is unnecessary to use other points where the similarity is low because the specialized filter generated by the points within $\sum_i K(\mathbf{x}_i, \mathbf{x}_o) \leq \eta$ is very likely to be accurate. Conceptually, we can visualize a ring that extends further and further depending on whether or not there are enough points inside the ring and whether those points have high enough similarity values.

2.3. Heuristics for Insufficient Training

k-NN algorithms assume there are enough points to blanket the entire domain, providing a good density estimate of the input space. Problems then arise from insufficient training because the further

the ring of values under consideration extends, the smaller the similarity values, and the less suited any additional training point is to complete the task of reaching η. In extreme cases, η may not even be reached before the entire training set is exhausted of points. Thus, we require the incorporation of a simple heuristic of limiting the maximum value of k that is allowed to be used. When the maximum value of k is reached, then k-NN is no longer adequate for $\mathbf{x}_o$ because there are too few points in Ω that are relevant.

Let ζ denote the maximum value of k. The question now becomes finding what kind of interpolation algorithm should replace k-NN when ζ is reached. Is there a particular type of image patch that the k-NN algorithm consistently disfavors? Moreover, based on this bias, are there certain properties of these patches that allow an informed decision to determine high-resolution content? The answer is yes on both accounts. After running several tests, we came across a peculiar reoccurring theme in generic training and testing images: texture patches never reached ζ and appeared at high quality, but edge patches often did and required an alternative interpolation technique, seen in Fig 1.

(a) Original Interpolation (b) Patches w/Insufficent Training

Fig. 1. With only 200 thousand data points, we cannot actively reconstruct many edges on the lighthouse because training patches don't occur frequently enough and there aren't close enough matches. Texture, however, *can* be, and much of the texture interpolation occurs because $k^* < \zeta$.

Fig. 1 reveals much about the patch-based domain, where it is reasonable to assert that the distribution of low-resolution patches in Euclidean space clusters around texture. The phenomenon makes sense because image content is usually dense in texture with sparse, albeit structure-defining, edges.

Though texture results in high peak signal to noise ratios (PSNR), to be presented in Sec. 4, the human visual system (HVS) focuses on edges. Fortunately, research into edge-oriented image filtering has been well-studied. In our framework, we agglomerate a bank of edge-oriented filters that do "well-enough" when the "best" filter through k-NN is unavailable, effectively reducing the implementation to a specialized version of [6] with an added MRF improvement (the next section, Sec. 3) through [7].

3. GLOBAL APPROXIMATION USING RANDOM FIELDS

Freeman [2] makes a good argument that globalization in terms of relating neighboring patches is necessary. Consequently, we have followed their lead by considering the usage of Markov network in modeling spatial relationships between patches. These modeling techniques usually require the use of some annealing process, which is usually computationally intractable for most realistic applications. Therefore, with the aid of [7], we have implemented a simpler-than-MRF, single-pass technique to enhance coherency from patch to patch.

Single pass algorithms include extra arguments into the decision making process that increase propensity towards one neighbor over another. Because our algorithm observes multiple neighbors per input patch, the structure of the one pass algorithm must be modified somewhat. Given the filter construction process in (7), we can take advantage of an expression that is already designed to penalize or reward training points through a matrix P. To review conceptually, elements within P denote the importance of a particular training point. After determining the k^* values for all image patches (or realistically, just the ones surrounding the test patch being evaluated), the logical course of action would be to reward those states that contain high values for $K(\mathbf{x}_o, \mathbf{x}_i)$ and $\Psi\left(\mathbf{z}_o, \mathbf{z}_n\right)$, where Ψ defines the similarity of states $\mathbf{z}_o$ and $\mathbf{z}_n$. Adding a cross-validated scaling factor α, a very simple conditioning scheme could be

$$P_{(i,\cdot)} = K(\mathbf{x}_o, \mathbf{x}_i) + \alpha \sum_{n \in \mathcal{N}} \sum_j \Psi\left(\mathbf{z}_j^{(n)}, \mathbf{z}_i\right) \tag{9}$$

Here, $\mathbf{z}_j^{(n)}$ refers to the j^{th} candidate state of the n^{th} low-resolution block in the neighborhood $\mathcal{N}$, the neighborhood of the input block.

4. RESULTS

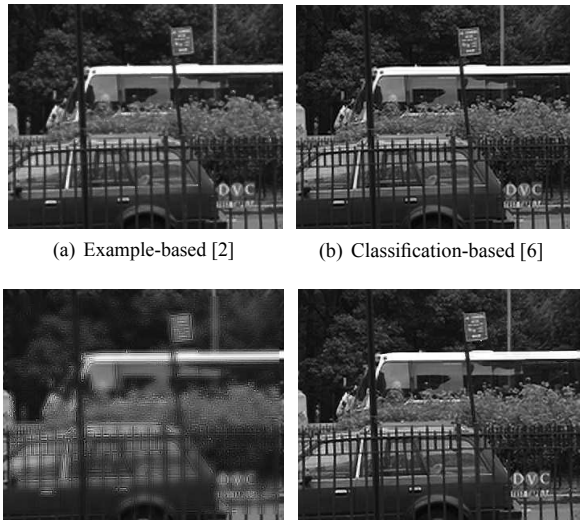
The proposed algorithm is designed with the assumption that a very large training set is available. (Training set size N on the order of 10^7.) Even training and testing on the same image will not yield any meaningful result should the testing image remain the only training available. In our experiments, a *minimum* of 10 images, where there are at least a few million image patches, is common. Requiring large N returns to previous explanations of determining filter coefficients from the proposed algorithm's choice of k. As it turns out, creating a nonsingular matrix is surprisingly difficult, and very often insufficient data plagues the filter generation effort.

With an RBF kernel, we can alter two degrees of freedom in our experiments, σ, the bandwidth parameter of the Gaussian used to extend around the observation, or η, the minimum number of training points necessary. Between the two, it is easier to alter η because the bandwidth σ is a squared exponent term and is difficult to control.

From descriptions in Sec. 2, values of k are directly correlated with the amount of training, N. Increasing N usually results in decreasing k, and this is verified through experimentation because k tends to stay around select values, k_{avg}, for particular N. In Fig. 2, k has a standard deviation of 59.79 on average staying around 151.76. Often $\zeta = 10^6$ is reached, but it is usually either hit or miss, where either k is be close to k_{avg} or ζ isreached, in which case, image values are created using classified edge filters.

Fig. 2 shows the qualitative results of other nearest neighbor and statistical classification algorithms alongside our own, and Table 1 gives the quantitative results for various images. The total number of training for this set of test runs is $N = 4,309,914$ points.

Fig. 2(a) and Fig. 2(b) show a decidedly edge-centric result, which can be explained by the neighborhood regularization done through Markov networks and limited class usage, respectively. The algorithms perform especially well in areas where the original image contains a disparity between textures such as the border between the bus and the background, but gives average performance where edges are less well-defined (see the pole above the bus in Fig. 2(a) and bus window pane in Fig. 2(b)). The same phenomenon can be seen for



(a) Example-based [2] (b) Classification-based [6]

(c) Neighbor-Embedding [1] (d) Adaptive k-NN

Fig. 2. Comparisons of the bus image for various statistical learning interpolation techniques. Neighbor embedding [1] in (c) seems to work for select images only.

Table 1. Miscellaneous PSNR comparisons

METHOD	PSNR Values		
	Pirates	Lighthouse	Bus,4
Bicubic	29.28 dB	28.87 dB	24.55 dB
NEDI [8]	26.82 dB	27.44 dB	22.57 dB
SEL [9]	26.67 dB	27.38 dB	22.63 dB
128 Class RS [6]	28.91 dB	28.98 dB	26.48 dB
LLE [1]	21.97 dB	22.70 dB	18.04 dB
Example-Based [2]	27.28 dB	28.75 dB	25.52 dB
Proposed Algorithm	29.62 dB	29.12 dB	26.25 dB

involved textures (the bushes in the lower right hand corner). This result is interesting, and could be a byproduct of the particularly large emphasis Freeman places on texture regularity. More likely, Sec. 2.3 reasons that the trend could be due to an incorrect choice of a single neighbor from insufficient edge information. Fig. 2(d) follows this model, but here, the benefits of k rather than a single neighbor becomes apparent, significantly reducing the artifacts that afflict [2].

Again, our algorithm does not have the luxury of an initial interpolation stage, instead filtering from scratch, and when there is insufficient training, cross-validation for η and σ is difficult albeit possible. Owing to this fact, the proposed algorithm manages smaller N sizes at the lower end of $\alpha < 10$ in $\alpha \times 10^5$ worse than [2]. Yet, when N is large, it often outperforms [2].

Interestingly enough, while [1] is the most similar in theory to our algorithm, the results (verified by code from the original author) did not reflect this. No interpolation effort with any reasonably cross-validated parameters yielded acceptable results. Experiments in [1] trained on single images, but the feature space of [1] in those cases was an astonishing 100 dimensions, which mandates an immense N. The errors may be due in part to the fact that neighboring patch information is not considered. Also, because the neighborhood preservation rate of the output patch is on average less than 10% [3], we can expect little continuity in the image result, justifying the methods in Sec. 3 to consider adjacent image patches.

5. CONCLUSIONS AND FUTURE WORK

A k-NN algorithm with optimal filters and a variable k, determined by relevant training, has been proposed, tested, and compared to the state of the art. The analysis of the proposed algorithm leads to the following conclusions:

- For small training sets, edges cannot be accurately depicted with any nearest neighbor algorithm (using Euclidean distance).
- Fast neighboring-patch approximations of a Markov Network elucidate edges and provide good continuity but sometimes hinder texture synthesis in cases where the training is limited.
- Linear filtering is a good mask and covers up considerable estimation errors.
- The direct application of k-NN regression with slight modifications exhibits competitive image quality and offers detailed texture.

The investigation of k-NN for image interpolation opens several avenues of analysis. While empirically touched upon in Sec. 2.2, theoretical analysis of domain representations remains a pressing topic. Along with domain considerations comes the more immediate need for a good distance metric, which may be manifested in such techniques as dimensionality reduction and feature-space mapping.

6. REFERENCES

[1] Hong Chang, Dit-Yan Yeung, and Yimin Xiong, "Super-resolution through neighbor embedding," *IEEE Conference on Computer Vision and Patter Recognition*, vol. 01, pp. 275–282, 2004.

[2] William T. Freeman, Thouis R. Jones, and Egon C. Pasztor, "Example-based super-resolution," *IEEE Computer Graphics and Applications*, vol. 22, no. 2, pp. 56–65, March/April 2002.

[3] Kevin Su, Qi Tian, Qing Que, Nicu Sebe, and Jingsheng Ma, "Neighborhood issue in single-frame image superresolution," 2005, IEEE Computer Society.

[4] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, Wiley Interscience, 2nd edition, 2000.

[5] W. S. Cleveland and S. J. Delvin, "Locally weighted regression: An approach to regression analysis by local fitting," *Journal of American Statistical Association*, vol. 83, no. 403, pp. 596–610, 1988.

[6] C. Brian Atkins and C. Bouman, *Classification based methods in optimal image interpolation*, Ph.D. thesis, Purdue University, 1998.

[7] M. Li and T. Nguyen, "Discontinuity-adaptive de-interlacing scheme using markov random field model," October 2006.

[8] X. Li and M. Orchard, "New edge-directed interpolation," *IEEE Transactions on Image Processing*, vol. 10, pp. 1521–1527, 2001.

[9] K. Jensen and D. Anastassiou, "Subpixel edge localization and the interpolation of still images," *IEEE Transactions on Image Processing*, vol. 4, pp. 285–295, 1995.