Anchored Neighborhood Regression for Fast Example-Based Super-Resolution

Radu Timofte [1,2], Vincent De Smet [1], and Luc Van Gool [1,2]
[1] KU Leuven, ESAT-PSI / iMinds, VISICS
[2] ETH Zurich, D-ITET, Computer Vision Lab

Abstract

Recently there have been significant advances in image upscaling or image super-resolution based on a dictionary of low and high resolution exemplars. The running time of the methods is often ignored despite the fact that it is a critical factor for real applications. This paper proposes fast super-resolution methods while making no compromise on quality. First, we support the use of sparse learned dictionaries in combination with neighbor embedding methods. In this case, the nearest neighbors are computed using the correlation with the dictionary atoms rather than the Euclidean distance. Moreover, we show that most of the current approaches reach top performance for the right parameters. Second, we show that using global collaborative coding has considerable speed advantages, reducing the super-resolution mapping to a precomputed projective matrix. Third, we propose the anchored neighborhood regression. That is to anchor the neighborhood embedding of a low resolution patch to the nearest atom in the dictionary and to precompute the corresponding embedding matrix. These proposals are contrasted with current state-of-the-art methods on standard images. We obtain similar or improved quality and one or two orders of magnitude speed improvements.

1. Introduction

Super-resolution (SR) is a popular branch of image reconstruction that focuses on the enhancement of image resolution. In general, it takes one or more low resolution (LR) images as input and maps them to a high resolution (HR) output image. Super-resolution algorithms can be roughly subdivided into three subclasses: interpolation methods like Lanczos upsampling [5] and New Edge Directed Interpolation (NEDI) [10], multi-frame methods [6, 7, 11] which make use of the presence of aliasing in multiple frames of the same scene to produce one high resolution image, and finally learning-based methods. The latter use machine learning techniques and comprise methods like Gradient Profile Prior [13], which try to learn edge statistics from natural images, but also the recent and very popular dictionary- or example-based learning methods. Most of these dictionary-based methods build on the work of Freeman *et al.* [8] and Baker and Kanade [2].

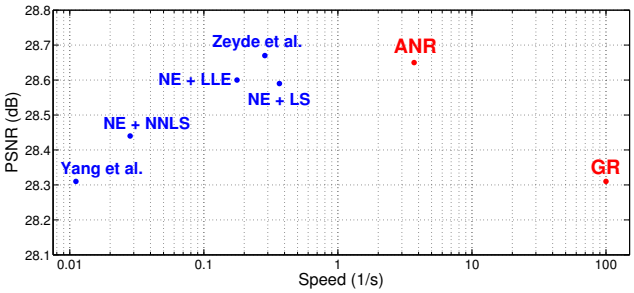


Figure 1. Speed vs. PSNR for the tested methods. Our ANR and GR methods (shown in red) provide both high speed and quality. More details in Table 1.

Dictionary-based methods use a patch- or feature-based approach to learn the relationship between local image details in low resolution and high resolution versions of the same scene. An input image is typically subdivided into overlapping patches, which together form a Markov Random Field (MRF) framework. By searching for nearest neighbors in a low resolution dictionary, a number of corresponding high resolution candidates can be retrieved. This results in an MRF with a number of HR candidate patches for each node. After associating a data cost to each candidate and a continuity cost for neighboring candidates the MRF can be solved by using techniques such as belief propagation or graph cuts.

One downside of these methods is their high computational complexity. Several methods have been proposed to overcome this problem, most notably neighbor embedding and sparse encoding approaches. Neighbor embedding super-resolution methods [4, 3] do not always explicitly focus on lowering computational complexity, but because of their inherent interpolation of the patch subspace they can be used to lower the number of image patch exemplars needed, thus reducing the algorithm's execution time. Sparse coding methods [17, 18] try to find a sparse coding for the input patches based on a compact dictionary (created by applying K-means or a similar algorithm to the training

patches). We give an overview of these methods in section 2. Bevilacqua *et al*. [3] compare the running times for several example-based super-resolution algorithms, mentioning minutes to hours for most state-of-the-art methods, depending on the size of the input image.

We propose a new method for example-based super-resolution that focuses on low computation time while keeping the qualitative performance of recent state-of-the-art approaches. We reach an average improvement in speed of between one and two orders of magnitude over other state-of-the-art methods (see Figure 1).

The remainder of the paper is organized as follows: we take a closer look at recent approaches for neighbor embedding and sparse coding for super-resolution in Section 2, we explain our proposed method in Section 3 and show experimental results in Section 4. The conclusions are drawn in Section 5.

2. Dictionary-based Super-Resolution

In this section we shortly review dictionary-based methods for SR. As we have briefly presented Freeman's original method [8] in the introduction, here we focus on neighbor embedding and sparse coding approaches.

2.1. Neighbor embedding approaches

Neighbor embedding (NE) approaches assume that small image patches from a low resolution image and its high resolution counterpart form low-dimensional nonlinear manifolds with similar local geometry. Chang *et al*. [4] proposed a super-resolution method based on this principle, using the manifold learning method Locally Linear Embedding (LLE) [12]. The LLE algorithm assumes that when enough samples are available, each sample and its neighbors lie on or near a locally linear patch of the manifold. Since the manifolds in LR and HR feature space are assumed to have a similar local geometry, this means that as long as enough samples are available, patches in the HR feature domain can be reconstructed as a weighted average of local neighbors using the same weights as in the LR feature domain. Chang *et al*. search for a set of K nearest neighbors for each input patch in LR feature space, compute K appropriate weights for reconstructing the LR patch by finding a constrained least squares solution, and eventually create an HR patch by applying these weights in HR feature space. The result image is then created by using the computed HR patches and averaging their contributions where they overlap. The recent Nonnegative Neighbor Embedding approach [3] is another example of NE used for super-resolution. It is based on the assumption that the local nonnegative least squares decomposition weights over the local neighborhood in LR space also hold for the corresponding neighborhood in HR space.

2.2. Sparse coding approaches

The NE approaches from the previous section use a dictionary of sampled patches from low and high resolution image pairs. These dictionaries can quickly become very large, especially when more or bigger training images are added to improve performance. Sparse coding (SC) approaches try to overcome this by using a learned compact dictionary based on sparse signal representation. Yang *et al*. [17] proposed an approach for super-resolution based on this idea. Low resolution patches are sparsely reconstructed from a learned dictionary using the following formulation:

$$\min_{\boldsymbol{\alpha}} \|F\mathbf{D}_l\boldsymbol{\alpha} - F\mathbf{y}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_0, \tag{1}$$

where F is a feature extraction operator, $\mathbf{D}_l$ is the learned low resolution dictionary, $\boldsymbol{\alpha}$ is the sparse representation, $\mathbf{y}$ is the low resolution input patch and λ is a weighting factor. The l_0-norm constraint leads to a NP-hard problem and, in practice, is relaxed to an l_1-norm constraint. Equation (1) is eventually also extended with a term which encourages similarity in the overlapping regions of nearby patches.

Sparse dictionaries are jointly learned for low and high resolution image patches, with the goal of having the same sparse representation for low resolution patches as their corresponding high resolution patches. This goal is reached for a set of training image patch pairs X_h, Y_l (high and low resolution patches resp.) by minimizing

$$\begin{aligned}\min_{\mathbf{D}_h,\mathbf{D}_l,Z} \frac{1}{N}\|X_h - \mathbf{D}_h Z\|_2^2 + \frac{1}{M}\|Y_l - \mathbf{D}_l Z\|_2^2 \\ + \lambda(\frac{1}{N} + \frac{1}{M})\|Z\|_1,\end{aligned} \tag{2}$$

where N and M are the dimensionality of the low and high resolution patches and Z is the coefficient vector representing the sparsity constraint. The resulting dictionary has a fixed size and thus the algorithm has the capability of learning from many training patches while avoiding long processing times due to an ever growing dictionary. Unfortunately, solving this sparse model still takes a large amount of time.

Zeyde *et al*. [18] build upon this framework and improve the execution speed by adding several modifications. The most important changes include using different training approaches for the dictionary pair (K-SVD [1] for the low resolution dictionary and direct approach using the pseudo-inverse for the high resolution dictionary), performing dimensionality reduction on the patches through PCA and using Orthogonal Matching Pursuit [16] for the sparse coding. They also show an improvement in quality with less artifacts and a higher average Peak Signal-to-Noise Ratio (PSNR) when compared to the results of Yang *et al*. [17].

3. Proposed Methods

We propose an anchored neighborhood regression method that conveys two situations, one being the general behavior where a neighborhood size is set and the other being the so called global case, where the neighborhood coincides with the whole dictionary in use. We refer to these in the following as the Anchored Neighborhood Regression (ANR) and its extreme case, the Global Regression (GR). We start with the global case for simplicity of the formulation, and then we consider the neighborhoods.

3.1. Global Regression

For most NE and SC approaches, the least squares (LS) problems are constrained or regularized using the l_1-norm of the coefficients, similar to equation (1). This is computationally demanding. We can reformulate the problem as a least squares regression regularized by the l_2-norm of the coefficients. Thus, we use Ridge Regression [14] (also known as Collaborative Representation [15]) and have a closed-form solution. The problem becomes

$$\min_{\boldsymbol{\beta}} \|\mathbf{y}_F - \mathbf{N}_l\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_2, \tag{3}$$

where $\mathbf{N}_l$ corresponds to the neighborhood in LR space that we choose to solve this problem, which in the case of neighborhood embedding would refer to the K nearest neighbors of the input feature $\mathbf{y}_F$ and in the case of sparse coding would refer to the LR dictionary. The parameter λ allows us to alleviate the singularity (ill-posed) problems and stabilizes the solution, which is the coefficient vector $\boldsymbol{\beta}$. The algebraic solution is given by

$$\boldsymbol{\beta} = (\mathbf{N}_l^T\mathbf{N}_l + \lambda\mathbf{I})^{-1}\mathbf{N}_l^T\mathbf{y}_F. \tag{4}$$

The HR patches can then be computed using the same coefficients on the high resolution neighborhood $\mathbf{N}_h$

$$\mathbf{x} = \mathbf{N}_h\boldsymbol{\beta}, \tag{5}$$

where $\mathbf{x}$ is the HR output patch and $\mathbf{N}_h$ the HR neighborhood corresponding to $\mathbf{N}_l$.

If we use the whole LR dictionary for this, meaning $(\mathbf{N}_h, \mathbf{N}_l) = (\mathbf{D}_h, \mathbf{D}_l)$, we get a global solution for the problem. An important observation here is that from equation (4) and equation (5), we obtain:

$$\mathbf{x} = \mathbf{D}_h(\mathbf{D}_l^T\mathbf{D}_l + \lambda\mathbf{I})^{-1}\mathbf{D}_l^T\mathbf{y}_F \tag{6}$$

where the projection matrix

$$\mathbf{P}_G = \mathbf{D}_h(\mathbf{D}_l^T\mathbf{D}_l + \lambda\mathbf{I})^{-1}\mathbf{D}_l^T \tag{7}$$

can be computed offline. This means that during the execution of the SR algorithm we only need to multiply the precomputed projection matrix $\mathbf{P}_G$ with the LR input feature vector, $\mathbf{y}_F$, to calculate the HR output patches $\mathbf{x}$. This formulation is the Global Regression (GR) approach, the extreme case of our ANR method.

3.2. Anchored Neighborhood Regression

The Global Regression approach reduces the super-resolution process to a projection of each input feature into the HR space by multiplication with a precomputed matrix. It is however a global solution and thus not tuned towards specific input features, but rather the entire dictionary, which is a representation of the features occurring in the training images. If instead of considering the whole dictionary as starting point for computing the projective matrix we consider local neighborhoods of a given size we allow more flexibility of the approach at the expense of increased computation – we will have more than one projective matrix and neighborhoods.

We start by grouping the dictionary atoms into neighborhoods. More specifically, for each atom in the dictionary we compute its K nearest neighbors, which will represent its neighborhood. If we start from a learned sparse dictionary, as in the sparsity approaches of Yang *et al*. [17] and Zeyde *et al*. [18], we find the nearest neighbors based on the correlation between the dictionary atoms rather than the Euclidean distance. The reason for this is that the atoms are a learned basis consisting of l_2-normalized vectors. If, conversely, we have a dictionary of features taken straight from the training patches, like in the NE approaches of Chang *et al*. [4] and Bevilacqua *et al*. [3], then the Euclidean distance is an appropriate distance measure. Once the neighborhoods are defined, we can calculate a separate projection matrix $\mathbf{P}_j$ for each dictionary atom $\mathbf{d}_j$, based on its own neighborhood. This can be done in the same way as in the previous section by using only the dictionary atoms that occur in the neighborhood rather than the entire dictionary, and can again be computed offline.

The super-resolution problem can then be solved by calculating for each input patch feature $\mathbf{y}_{iF}$ its nearest neighbor atom, $\mathbf{d}_j$, in the dictionary, followed by the mapping to HR space using the stored projection matrix $\mathbf{P}_j$:

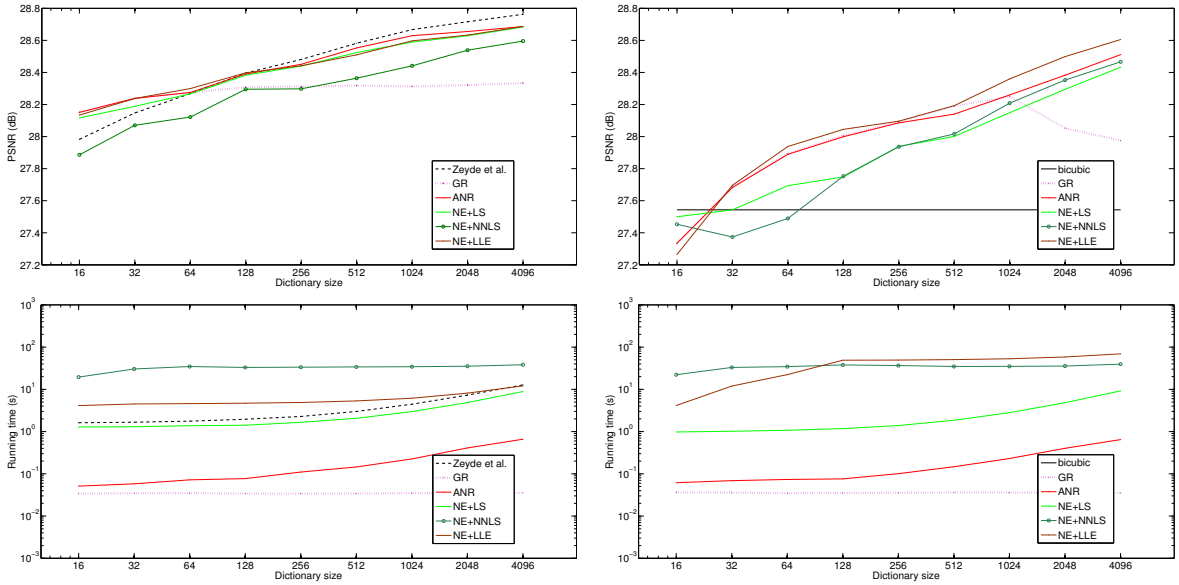
$$\mathbf{x}_i = \mathbf{P}_j\mathbf{y}_{iF}. \tag{8}$$

This is a close approximation of the NE approach, with a very low complexity and thus a vast improvement in execution time. We call our approach the Anchored Neighborhood Regression (ANR), since the neighborhoods are anchored to the dictionary atoms and not directly to the low resolution patches as in the other NE approaches.

4. Experiments

In this section we show experimental results[1] of our method and compare it quantitatively and qualitatively to other state-of-the-art methods. We first discuss some of the

[1] Source codes, images, and results are available at: http://www.vision.ee.ethz.ch/~timofter/



a) trained dictionary *b)* random dictionary

Figure 2. Dictionary size vs. average PSNR and average running time performance on the 14 images from Set14 with magnification $\times 3$. *Bicubic* is our reference. All the methods were used with their best neighborhood size. For the trained dictionaries, ANR uses a size of 40, NE+LS uses 12, NE+NNLS - 24, and NE+LLE - 24. For the random dictionaries, ANR uses a size of 128, NE+LS - 5, NE+NNLS - 24, and NE+LLE - 128, resp. For the running times we subtracted the shared processing time (collecting patches, combining the output) for all the methods.

details surrounding the algorithm such as used features, dictionary choices, similarity measures, size for neighborhood calculation and different patch embeddings.

4.1. Conditions

4.1.1 Features

One aspect which can influence the performance is the type of features used to represent the image patches. These features are almost always calculated from the luminance component of the image, while the color components are interpolated using a regular interpolation algorithm such as bicubic interpolation [9, 18, 17, 4, 3]. This is because the human visual system is much less sensitive to high frequency color changes than high frequency intensity changes, so for the magnification factors used in most papers the perceived difference between bicubic interpolation and SR of the color channels is negligible.

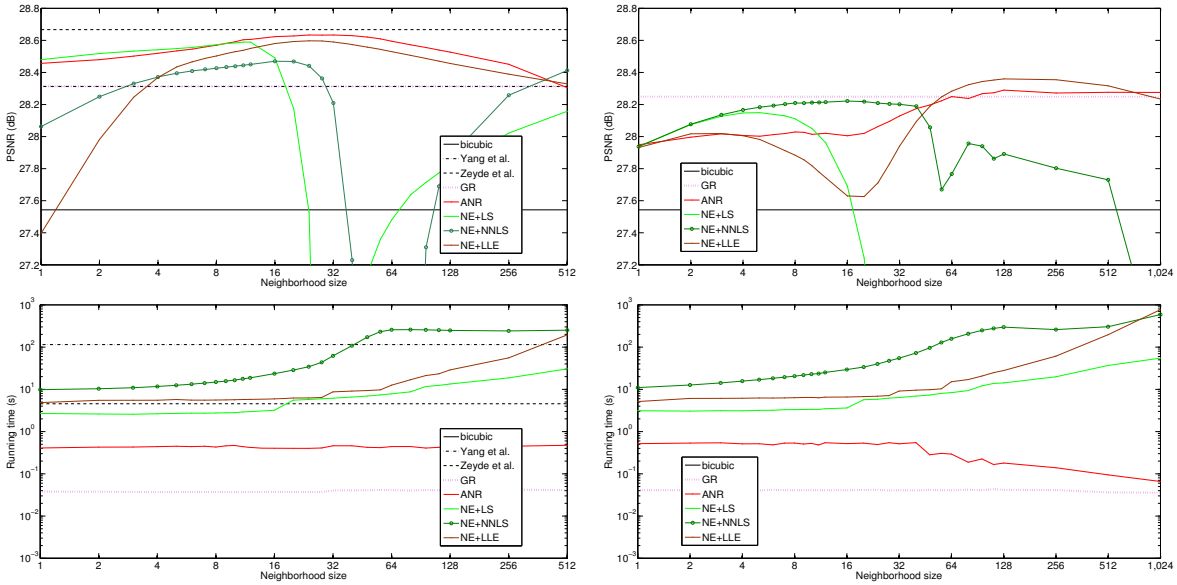
The most basic feature to use is the patch itself. This however does not give the feature good generalization properties, so a popular choice is to subtract the mean [3] and to normalize the contrast [8] by e.g. dividing by the standard deviation. An often used similar feature is the first- and second order derivative of the patch [4, 17, 18]. Both of these feature types seem to lead to similar performance, while Bevilacqua *et al*. [3] show that using only first order derivatives gives slightly worse performance than using only mean subtraction. We use the same features as Zeyde *et al*. [18], who start from the first- and second order gradients and apply PCA dimensionality reduction, projecting the features onto a low-dimensional subspace while preserving 99.9% of the average energy. This usually leads to features of about 30 dimensions for upscaling factor 3 and 3×3 low resolution patch sizes.

We subtract the bicubically interpolated LR image from the HR image to create normalized HR patches. The patches resulting from the SR process (*i.e*. equation (6) for GR) are added to the bicubically interpolated LR input image (with overlapping parts averaged) to create the output.

We use Zeyde *et al*.'s algorithm and their provided sources as a starting point for our implementations.

4.1.2 Embeddings

Apart from our comparisons with the sparse methods of Yang *et al*. [17] and Zeyde *et al*. [18], we also compare our results to neighbor embedding approaches adapted to our dictionary choices. The original LLE-based SR method of Chang *et al*. [4] does not use a learned dictionary, instead the dictionary consists simply of the training patches themselves. This makes direct comparison to our method and the sparse methods difficult, because the question then arises "which dictionary should we use to have a fair comparison?". The same can be said when we wish to compare to the nonnegative neighbor embedding of Bevilacqua *et al*. [3]. The solution is to use the same learned dictionary as Zeyde *et al*. and our method, with the respective SR methods of Chang and Bevilacqua implemented to solve the SR regression. We will refer to these as NE + LLE (Neighbor Embedding with Locally Linear Embedding) and NE



a) trained dictionary of size 1024 *b)* random dictionary of size 1024

Figure 3. Neighborhood size vs. average PSNR and average running time performance on the 14 images from Set14 with magnification $\times 3$. *Bicubic*, *Yang* et al., *Zeyde* et al., and *GR* are reported for reference. *Yang* et al. uses the original dictionary of 1022, while the other methods, except *Bicubic* interpolation, share the same dictionary. The decrease in the running time of the *ANR* method from random dictionary experiments is caused by applying a subsampling step of $\max\{1, \frac{neighborhoodsize}{30}\}$ for the anchor atoms, while for the trained one the step is 1. For the running times we subtracted the shared processing time (collecting patches, combining the output) for all the dictionary based methods, leaving only the encoding time for each method.

+ NNLS (Neighbor Embedding with NonNegative Least Squares), and we add results for a similar implementation that uses unconstrained least squares to solve the regression, to which we refer as NE + LS.

4.1.3 Dictionaries

The choice of the dictionary is critical for the performance of any SR method. Usually, the larger the dictionary the better the performance, however this comes with a higher computational cost. The dictionary can be built using the LR input image itself, in this case we have an "internal" dictionary. Glasner *et al*. [9] and their intensive exploitation of "patch redundancy" are the main advocates for this. However, many approaches prefer to build "external" dictionaries, external to the input query, using diverse images.

In our settings we work with the same set of external images as used by Zeyde *et al*. [18] and Yang *et al*. [17]. Also, we consider both randomly sampled dictionaries and learned dictionaries. For learning we use the K-SVD/OMP learning approach of Zeyde *et al*. [18].

In Fig. 2 we depict the effect of the dictionary on the performance. As expected, usually the performance increases with the size of the dictionary. Moreover, we see again that using a learned dictionary is highly beneficial for all the methods – it allows for a reasonably high performance for small dictionary sizes. One needs a $16\times$ larger random sampled dictionary to reach the same performance as with the trained dictionary. Most of the methods exhibit a similar log-linear increasing trend with the dictionary size and the PSNR difference among ANR, Zeyde *et al*. and the NE approaches is quite small for their best settings (using optimal neighborhood size). The difference is made by the running time, where ANR and GR are the clear winners. GR is the fastest method, but as a global method it has its weaknesses, and for large dictionaries tends not to reach competitive PSNR levels.

4.1.4 Neighborhoods

As explained in Section 3.2, our ANR algorithm finds the nearest neighbor (atom) in the dictionary for each input feature and borrows the neighborhood and the precomputed projection matrix from this neighbor. The NE approaches also rely on the neighborhood to the input LR feature. The performance of the embedding methods, and hence the performance of the SR method, depends on the dimensionality of these neighborhoods.

The computation of the nearest neighbors is based on a similarity measure. The Euclidean distance is the choice of most NE approaches working directly with large dictionaries. We use the Euclidean distance for the setups with randomly sampled dictionaries. In the case of the learned sparse dictionaries, we obtain l_2-normalized atoms meant to form a basis spanning the space of the training samples while minimizing the reconstruction error. For this case our option is to obtain the nearest neighbors using the correlation expressed as the inner product.

Table 1. Magnification $\times 3$ performance in terms of PSNR (dB) and running time (s) per image on the Set14 dataset. All the original methods use the same training images from [17]. The methods share the same trained dictionary of 1024, except Bicubic interpolation and Yang *et al*. with a dictionary of 1022. The neighborhood sizes are as in Fig. 2. ANR is 5 times faster than Zeyde *et al*. If we consider only the encoding time, our ANR method takes 0.27s on average, being 13 times faster than Zeyde *et al*., and 10 times faster than NE+LS.

Set14	Bicubic		Yang *et al*. [17]		Zeyde *et al*. [18]		GR		**ANR**		NE+LS		NE+NNLS		NE+LLE	
images	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	**PSNR**	**Time**	PSNR	Time	PSNR	Time	PSNR	Time
baboon	23.2	0.0	23.5	142.1	23.5	4.4	23.5	0.7	**23.6**	**1.0**	23.5	3.6	23.5	37.8	23.6	6.6
barbara	26.2	0.0	26.4	150.4	26.8	7.6	26.8	1.1	**26.7**	**1.6**	26.7	6.1	26.7	66.5	26.7	11.5
bridge	24.4	0.0	24.8	172.7	25.0	4.7	24.9	0.7	**25.0**	**1.0**	24.9	3.8	24.9	41.8	25.0	7.1
coastguard	26.6	0.0	27.0	40.9	27.1	1.8	27.0	0.3	**27.1**	**0.4**	27.0	1.4	27.0	16.1	27.1	2.7
comic	23.1	0.0	23.9	60.3	24.0	1.6	23.8	0.2	**24.0**	**0.4**	23.9	1.3	23.8	13.9	24.0	2.4
face	32.8	0.0	33.1	23.7	33.5	1.4	33.5	0.2	**33.6**	**0.3**	33.5	1.1	33.5	11.7	33.6	2.0
flowers	27.2	0.0	28.2	88.8	28.4	3.3	28.1	0.5	**28.5**	**0.7**	28.3	2.6	28.2	28.4	28.4	4.9
foreman	31.2	0.0	32.0	30.6	33.2	1.8	32.3	0.3	**33.2**	**0.4**	33.2	1.5	32.9	15.9	33.2	2.7
lenna	31.7	0.0	32.6	78.7	33.0	4.8	32.6	0.8	**33.1**	**1.1**	33.0	4.0	32.8	41.0	33.0	7.2
man	27.0	0.0	27.8	123.7	27.9	4.7	27.6	0.7	**27.9**	**1.0**	27.9	3.8	27.7	41.2	27.9	7.2
monarch	29.4	0.0	30.7	130.5	31.1	7.2	30.4	1.1	**31.1**	**1.5**	30.9	5.8	30.8	63.3	30.9	10.8
pepper	32.4	0.0	33.3	75.6	34.1	4.8	33.2	0.7	**33.8**	**1.0**	33.9	3.8	33.6	41.4	33.8	7.2
ppt3	23.7	0.0	25.0	107.8	25.2	5.7	24.6	1.0	**25.0**	**1.3**	25.1	4.9	24.8	49.8	24.9	9.4
zebra	26.6	0.0	28.0	129.8	28.5	4.2	27.9	0.7	**28.4**	**0.9**	28.3	3.3	28.1	36.4	28.3	6.3
average performance	27.54	0.02	28.31	96.82	28.67	4.14	28.31	0.64	**28.65**	**0.90**	28.59	3.36	28.44	36.09	28.60	6.29
ANR speedup		**x0.02**		**x110**		**x4.6**		**x0.7**		**x1**		**x3.7**		**x40**		**x7**
average time for encoding		0.01		~90.00		3.51		0.01		**0.27**		2.73		35.46		5.66
ANR speedup for encoding		**x0.04**		**x330**		**x13**		**x0.04**		**x1**		**x10**		**x131**		**x21**

The neighborhood size is the major parameter for the NE techniques and for ANR as well. We show the effect of this size in Figure 3 for dictionaries of size 1024. The methods behave differently under the same settings. Moreover, the curves are not monotonic – as noticed also in [3]– and more investigation in this phenomenon is due. On the learned dictionary, NN + LS peaks at 12, NE + LLE and NE + NNLS at 24, while ANR peaks at 40. On the random dictionary, NN + LS peaks at 5, NE + LLE at 128, NE + NNLS at 24, while ANR peaks at 128. We will use these neighborhood sizes for the further experiments. The behavior of ANR and GR is also influenced by the choice of the regularization parameter λ, in all our experiments empirically set to 0.01.

4.2. Performance

In this section we will show quantitative and qualitative results as well as running times for our proposed method and compare them to the other discussed dictionary-based SR algorithms. More specifically, we compare our results to the sparse coding algorithms of Yang *et al*. [17] and Zeyde *et al*. [18], as well as to our implementations based on the LS regressions used by Chang *et al*. [4] (NE + LLE) and Bevilacqua *et al*. [3] (NE + NNLS) as described in Section 4.1.2. Tables 1 and 2 summarize the results, showing PSNR and running time values for a number of test images. The images are divided into two sets; **Set14** was used by Zeyde *et al*. to show their results and **Set5** was used by Bevilacqua *et al*. The effect of dictionary size is explored in Fig. 2, while Fig. 3 shows the relationship between neighborhood size, PSNR and time.

4.2.1 Quality

When using the optimal neighborhood size for each method the PSNR of Zeyde *et al*. [18], NE + LLE, NE + LS, and our ANR method reach comparable average values. The approach of Zeyde *et al*. reaches the highest PSNR in all experiments, slightly above our ANR method. On the Set14 dataset Zeyde *et al*. get an average of 28.67 dB, while ANR gets 28.65 dB, and NE + LS and NE + LLE get an average of 28.6 dB. Our GR method and Yang *et al*. get the same average PSNR of 28.31 dB, while NE + NNLS lies in between with 28.44 dB. A similar behavior can be seen on the Set5 database, where ANR can be better than Zeyde *et al*.

Visual examples are shown in Figures 4, 5, and 6. From these we can conclude that ANR gets very similar quality performance as the top methods it was compared to.

4.2.2 Running Time

Our implementation of NNLS has similar computation time as what is reported by Bevilacqua *et al*. [3], which is in the order of 10 seconds for a magnification factor of $3\times$. This can also be observed in Figure 2 and Figure 3. We compare with their algorithm because it is a very recent method aimed at low complexity and high processing speed while still keeping high quality results, and is therefore an ideal candidate for reference.

When we compare the processing times it is clear that our Global Regression algorithm is the fastest by far, followed by our Anchored Neighborhood Regression. The last row of Table 1 as well as Fig. 2 and 3 show the difference of the encoding time, which is the processing time that is left after subtracting the shared processing time of 0.63 seconds of the algorithms (pre/post processing, bicubic interpolation, patch extraction, etc.).

The Global Regression algorithm is useful when speed is the most important aspect, however the general ANR algorithm gives a better speed-performance trade-off. That being said, when we look at the results for $3\times$ magnification, GR reaches a speedup of $350\times$ when compared to Zeyde *et al*., $9000\times$ when compared to Yang *et al*., $560\times$ when compared to the NE + LLE method inspired by Chang *et al*. and $3500\times$ to NE + NNLS inspired by Bevilacqua *et al*. For the

Table 2. Magnification $\times 2$, $\times 3$, and $\times 4$ performance in terms of PSNR (dB) and running time (s) per image on the Set5 dataset. All the original methods use the same training images from [17]. All the methods share the same trained dictionary of 1024, except Bicubic interpolation and Yang *et al*. with a dictionary of 1022. We use the same neighborhood sizes as in Fig. 2. For upscaling factor 3, ANR is 5 times faster than Zeyde *et al*. being 94 times faster than Yang *et al*. and 4 times faster than NE+LS with 12 neighbors.

Set5		Bicubic		Yang *et al*. [17]		Zeyde *et al*. [18]		GR		**ANR**		NE+LS		NE+NNLS		NE+LLE	
images	*Scale*	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	**PSNR**	**Time**	PSNR	Time	PSNR	Time	PSNR	Time
baby	2	37.1	0.0	–	–	38.2	10.8	38.3	1.2	**38.4**	**1.8**	38.1	8.3	38.0	98.4	38.3	15.8
bird	2	36.8	0.0	–	–	39.9	3.3	39.0	0.4	**40.0**	**0.6**	39.9	2.5	39.4	30.6	40.0	4.9
butterfly	2	27.4	0.0	–	–	30.6	2.7	29.1	0.3	**30.5**	**0.4**	30.4	2.0	30.0	23.7	30.4	3.8
head	2	34.9	0.0	–	–	35.6	3.1	35.6	0.4	**35.7**	**0.5**	35.5	2.4	35.5	28.3	35.6	4.6
woman	2	32.1	0.0	–	–	34.5	3.2	33.7	0.4	**34.5**	**0.5**	34.3	2.4	34.2	28.8	34.5	4.6
average	2	33.66	0.00	–	–	35.78	4.63	35.13	0.54	**35.83**	**0.78**	35.66	3.53	35.43	41.94	35.77	6.74
baby	3	33.9	0.0	34.3	88.5	35.1	4.8	34.9	0.7	**35.1**	**1.0**	35.0	3.9	34.8	41.3	35.1	7.2
bird	3	32.6	0.0	34.1	35.7	34.6	1.5	33.9	0.2	**34.6**	**0.3**	34.4	1.2	34.3	12.9	34.6	2.2
butterfly	3	24.0	0.0	25.6	32.2	25.9	1.2	25.0	0.2	**25.9**	**0.3**	25.8	0.9	25.6	10.0	25.8	1.7
head	3	32.9	0.0	33.2	24.1	33.6	1.4	33.5	0.2	**33.6**	**0.3**	33.5	1.1	33.5	11.9	33.6	2.1
woman	3	28.6	0.0	29.9	30.2	30.4	1.4	29.7	0.2	**30.3**	**0.3**	30.2	1.1	29.9	11.9	30.2	2.1
average	3	30.39	0.00	31.42	42.14	31.90	2.08	31.41	0.33	**31.92**	**0.45**	31.78	1.65	31.60	17.61	31.84	3.05
baby	4	31.8	0.0	–	–	33.1	2.9	32.8	0.6	**33.0**	**0.8**	32.9	2.4	32.8	22.5	33.0	4.2
bird	4	30.2	0.0	–	–	31.7	0.9	31.3	0.2	**31.8**	**0.2**	31.6	0.7	31.5	6.9	31.7	1.3
butterfly	4	22.1	0.0	–	–	23.6	0.7	23.1	0.2	**23.5**	**0.2**	23.4	0.6	23.3	5.3	23.4	1.0
head	4	31.6	0.0	–	–	32.2	0.9	32.1	0.2	**32.3**	**0.3**	32.2	0.7	32.1	6.5	32.2	1.2
woman	4	26.5	0.0	–	–	27.9	0.9	27.4	0.2	**27.8**	**0.2**	27.6	0.7	27.6	6.4	27.7	1.2
average	4	28.42	0.00	–	–	29.69	1.25	29.34	0.26	**29.69**	**0.34**	29.55	1.00	29.47	9.52	29.61	1.77

same magnification, ANR reaches speed improvements of $13\times$, $330\times$, $21\times$, and $131\times$, resp.

5. Conclusions

We proposed a new example-based method for super-resolution called Anchored Neighbor Regression which focuses on fast execution while retaining the qualitative performance of recent state-of-the-art methods. We also proposed an extreme variant of this called Global Regression which focuses purely on high execution speed in exchange for some visual quality loss. The main contributions of this paper are twofold: i) we present the ANR approach, which uses ridge regression to learn exemplar neighborhoods offline and uses these neighborhoods to precompute projections to map LR patches onto the HR domain, and ii) we show through our analysis of existing neighborhood embedding SR methods that most of these can reach a similar top performance based on using the appropriate neighborhood size and dictionary; the sparse learned dictionaries in combination with neighbor embeddings methods were shown to be a faster alternative to full sparse coding methods.

We plan to extend our method to make full use of the extra dimension of time for the case of video sequences, with real-time streaming super-resolved video as a goal.

Acknowledgment. The authors are grateful for support by the Flemish iMinds framework and FWO Levenslijn.

References

[1] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Trans. on Sig. Proc.*, 54(11), 2006. 4322

[2] S. Baker and T. Kanade. Hallucinating faces. In *Automatic Face and Gesture Recognition*, 2000. 4321

[3] M. Bevilacqua, A. Roumy, C. Guillemot, and M.-L. Alberi Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. In *BMVC*, 2012. 4321, 4322, 4323, 4324, 4326

[4] H. Chang, D.-Y. Yeung, and Y. Xiong. Super-resolution through neighbor embedding. *CVPR*, 01:275–282, 2004. 4321, 4322, 4323, 4324, 4326

[5] C. E. Duchon. Lanczos Filtering in One and Two Dimensions. *J. Appl. Meteorology*, 18:1016–1022, 1979. 4321

[6] S. Farsiu, M. D. Robinson, M. Elad, and P. Milanfar. Fast and robust multiframe super resolution. *Trans. Img. Proc.*, 13(10):1327–1344, Oct. 2004. 4321

[7] R. Fransens, C. Strecha, and L. Van Gool. Optical flow based super-resolution: A probabilistic approach. *Computer Vision and Image Understanding*, 106(1):106–115, 2007. 4321

[8] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael. Learning low-level vision. *International Journal of Computer Vision*, 40(1):25–47, 2000. 4321, 4322, 4324

[9] D. Glasner, S. Bagon, and M. Irani. Super-resolution from a single image. In *ICCV*, 2009. 4324, 4325

[10] X. Li and M. T. Orchard. New edge-directed interpolation. *Trans. Img. Proc.*, 10(10):1521–1527, 2001. 4321

[11] M. Protter, M. Elad, H. Takeda, and P. Milanfar. Generalizing the non-local-means to super-resolution reconstruction. In *Trans. Img. Proc.*, 2009. 4321

[12] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. In *ICCV*, 2001. 4322

[13] J. Sun, Z. Xu, and H.-Y. Shum. Image super-resolution using gradient profile prior. In *CVPR*, 2008. 4321

[14] A. N. Tikhonov and V. Y. Arsenin. *Solution of Ill-posed Problems*. Winston & Sons, 1977. 4323

[15] R. Timofte and L. Van Gool. Adaptive and weighted collaborative representations for image classification. *Pattern Recognition Letters*, 2013. 4323

[16] J. A. Tropp and A. C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 53(12), 2007. 4322

[17] J. Yang, J. Wright, T. Huang, and Y. Ma. Image super-resolution via sparse representation. *Image Processing, IEEE Transactions on*, 19(11):2861–2873, 2010. 4321, 4322, 4323, 4324, 4325, 4326, 4327

[18] R. Zeyde, M. Elad, and M. Protter. On single image scale-up using sparse-representations. In *Curves and Surfaces*, pages 711–730, 2010. 4321, 4322, 4323, 4324, 4325, 4326, 4327

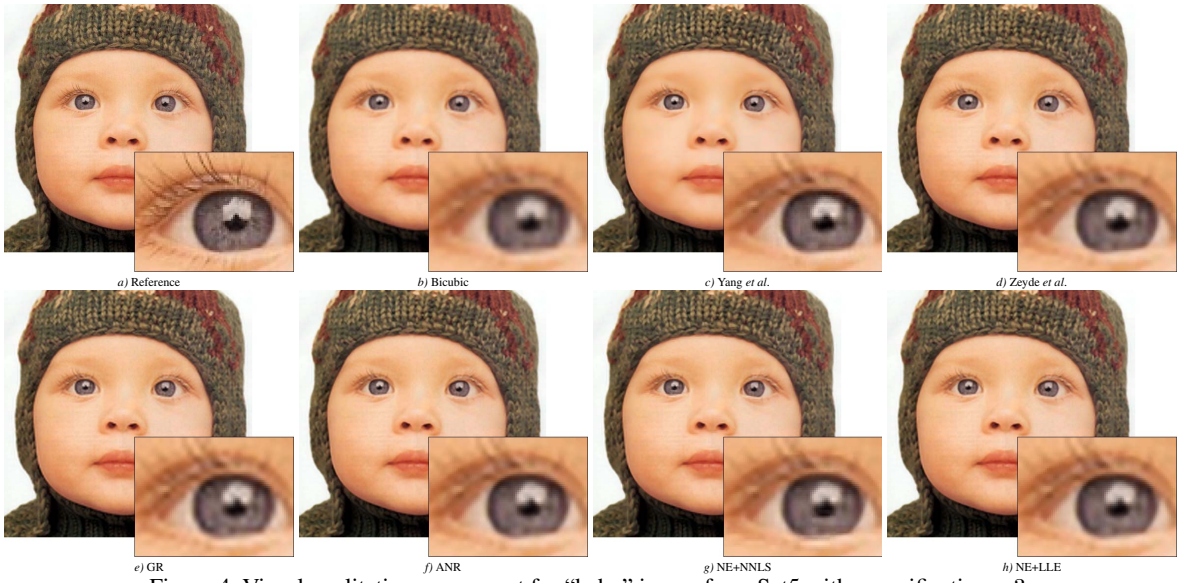


Figure 4. Visual qualitative assessment for "baby" image from Set5 with magnification $\times 3$.

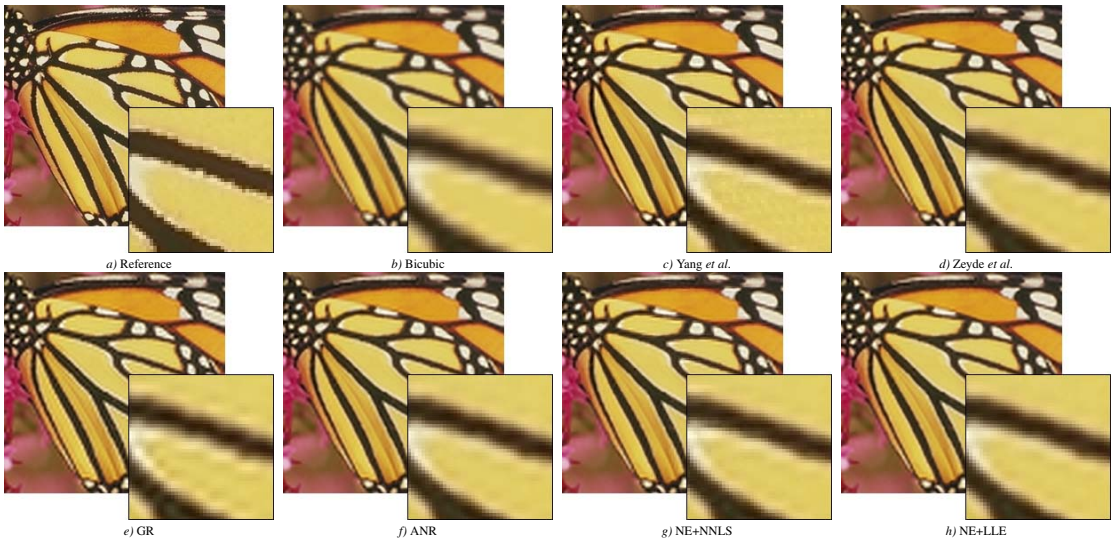


Figure 5. Visual qualitative assessment for "butterfly" image from Set5 with magnification $\times 3$.

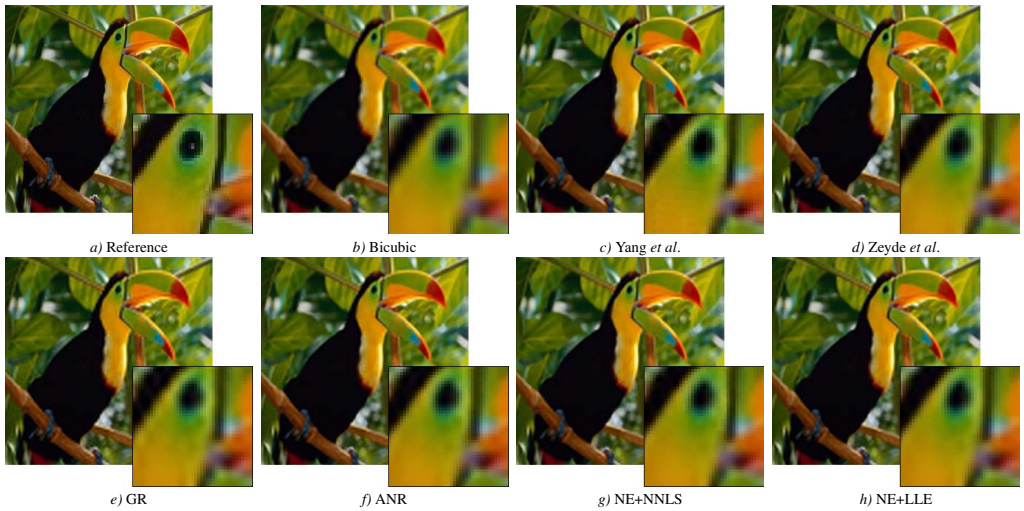


Figure 6. Visual qualitative assessment for "bird" image from Set5 with magnification $\times 3$.