DOI 10.1109/TIP.2016.2549362

Antipodally Invariant Metrics For Fast Regression-Based Super-Resolution

Eduardo Pérez-Pellitero, Jordi Salvador, Javier Ruiz-Hidalgo, and Bodo Rosenhahn

***Abstract*—Dictionary-based Super-Resolution algorithms usually select dictionary atoms based on distance or similarity metrics. Although the optimal selection of nearest neighbors is of central importance for such methods, the impact of using proper metrics for Super-Resolution (SR) has been overlooked in literature, mainly due to the vast usage of Euclidean distance. In this paper we present a very fast regression-based algorithm which builds on densely populated anchored neighborhoods and sublinear search structures. We perform a study of the nature of the features commonly used for SR, observing that those features usually lie in the unitary hypersphere, where every point has a diametrically opposite one, i.e. its antipode, with same module and angle, but opposite direction. Even though we validate the benefits of using antipodally invariant metrics, most of the binary splits use Euclidean distance, which does not handle antipodes optimally. In order to benefit from both worlds, we propose a simple yet effective Antipodally Invariant Transform (AIT) that can be easily included in the Euclidean distance calculation. We modify the original Spherical Hashing algorithm with this metric in our Antipodally Invariant Spherical Hashing scheme, obtaining the same performance as a pure antipodally invariant metric. We round up our contributions with a novel feature transform that obtains a better coarse approximation of the input image thanks to Iterative Back Projection. The performance of our method, which we named Antipodally Invariant Super-Resolution (AIS), improves quality (PSNR) and it is faster than any other state-of-the-art method.**

***Index Terms*—Super-Resolution, Regression, Antipodes, Spherical Hashing.**

I. Introduction

SUPER-RESOLUTION (SR) techniques aim to extend the resolution of a signal surpassing the limits of the original capture device. Increasing the resolution of an image without further information is a deeply ill-posed problem, and therefore, it needs to be addressed with a certain prior knowledge. Although SR is a relatively young research field, many of those priors have been proposed.

Some of the simplest and earliest image SR methods were based on piecewise linear and smooth priors (i.e. bilinear and bicubic interpolation, respectively), resulting in fast interpolation-based algorithms. Tsai and Huang [1] showed that it was possible to reconstruct higher-resolution images by registering and fusing multiple images, thus pioneering a vast amount of approaches on multi-image SR, often called reconstruction-based SR. This idea was further refined, among others, with the introduction of iterative back-projection for improved registration by Irani and Peleg [2], although further analysis by Baker and Kanade [3] and Lin and Shum [4] showed fundamental limits on this type of SR, mainly conditioned by registration accuracy. Learning-based SR, also known as example-based, overcame some of the aforementioned limitations by avoiding the necessity of a registration process and by building the priors from image statistics. The original work by Freeman et al. [5] aims to *learn* from patch- or feature-based examples to produce effective magnification well beyond the practical limits of multi-image SR.

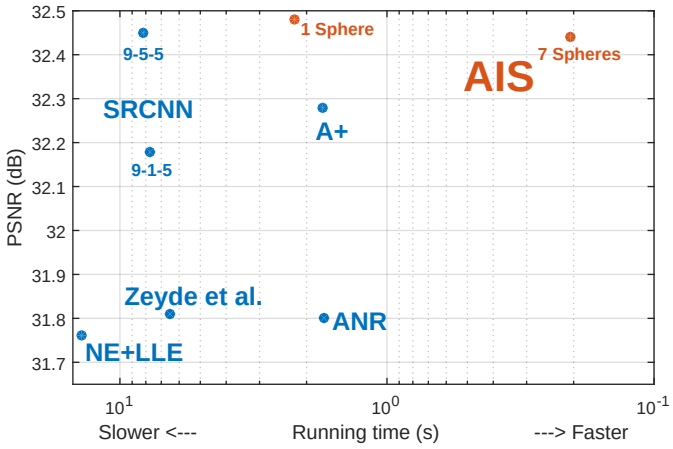


Fig. 1: The proposed AIS (Antipodally Invariant Super-Resolution) achieves both the best quality (PSNR) and the fastest speed (s) in comparison with most recent state-of-the-art methods. All algorithms run Set14 with a $\times 2$ upscaling factor, see Table III for more information.

Example-based SR approaches using dictionaries are usually divided into two categories: internal and external dictionary-based SR. The first exploits the strong self-similarity prior. This prior is learnt directly from the relationship of image patches across different scales of the input image. The opening work on this subcategory was introduced by Glasner et al. [6], presenting a powerful framework for fusing reconstruction-based and example-based SR. Further research on this category by Freedman and Fattal [7] introduced a mechanism for high-frequency transfer based on examples from a small area around each patch, thus better localizing the cross-scale self-similarity prior to the spatial neighborhood. The recent work of Yang et al. [8] further develops the idea of localizing the cross-scale self-similarity prior arriving to

E. Pérez-Pellitero is with Technicolor R&I Hannover and the TNT Lab, Leibniz Universität Hannover, Germany.
J. Salvador is with Technicolor R&I Hannover.
J. Ruiz-Hidalgo is with the Image Processing Group, Universitat Politècnica de Catalunya, Spain.
B. Rosenhahn is with the TNT Lab, Leibniz Universität Hannover.
Manuscript received March 20, 2015.

the in-place prior, i.e. the best match across scales is located exactly in the same position if the scale is similar enough.

External dictionary-based SR uses other images to build their dictionaries. A representative widely used approach is the one based on sparse decomposition. The main idea behind this approach is the decomposition of each patch in the input image into a combination of a sparse subset of entries in a compact dictionary. The work of Yang et al. [9] uses an external database composed of related low and high-resolution patches to jointly learn a compact dictionary pair. During testing, each image patch is decomposed into a sparse linear combination of the entries in the low-resolution (LR) dictionary and the same weights are used to generate the high-resolution (HR) patch as a linear combination of the HR entries. Both the dictionary training and testing are costly due to the L_1 regularization term enforcing sparsity. The work of Zeyde et al. [10] extends sparse SR by proposing several algorithmic speed-ups which also improve performance. However, the bottleneck of sparsity methods still remains in the sparse decomposition. In the neighbor embedding algorithm of Gao et al. [11] they propose a method that builds neighborhoods for linear embedding in a unified feature subspace spanned by LR–HR image patches rather than in the original LR feature space alone.

More recently, regression-based SR has received a great deal of attention by the research community. In this case, the goal is to learn a certain mapping from the manifold of the LR patches to that of HR patches, following the manifold assumption already used in the earlier work of Chang et al. [12].

The mapping of the manifold is assumed to be locally linear and therefore several linear regressors are used and anchored to the manifold as a piecewise linearization [13]. The key observation of performing the neighbor embedding just for a predefined set of anchor points allowed to preprocess them during training time, thus lessening substantially the testing time complexity. Our recent work [14] puts light in how to properly create the neighborhoods in such methods, obtaining sizable benefits in terms of reconstruction quality. In addition to that, we also explored sublinear search structures for the regressor nearest neighbor search, as this takes a significant quota of the running time from within the processing of the whole SR pipeline.

In this paper we follow the same research direction and we introduce the following contributions:

1) We study and provide insight about the behavior of distance metrics used during the regression process. We detect the importance of antipodal invariance.
2) We provide a suitable and efficient spherical hashing framework to exploit a fast regressor search. In order to benefit from antipodal invariance, we introduce a new simple yet effective transform which makes Euclidean distance (and also spherical hashing) antipodally invariant.
3) We present a novel feature transform based on gradients and Iterative Back Projection which improves the quality of the upscaled image thanks to a better first coarse approximation

The confluence of all the mentioned contributions yields state-of-the-art speed and quality results. With our proposed approximate regressor search we are the fastest (about $\times 4-11$ times faster) and we greatly improve in quality (about 0.2dB higher) compared to our predecessor A^+. In addition, when performing exhaustive search we obtain even better PSNR results (up to 0.32dB higher).

The rest of this paper is organized as follows. Section II defines the problem and reviews the state-of-the-art and related work. In Section III we describe our proposed algorithm. In Section IV we discuss and assess the performance of the presented contributions separately. Experimental results and comparisons with state-of-the-art methods are provided in Section V. We conclude the paper in Section VI.

II. Related work

In the upcoming section we introduce the SR problem and how example-based approaches tackle it, followed by a review of relevant methods among the state of the art.

A. Problem statement

Super-Resolution aims to upscale images which have an unsatisfactory pixel resolution while preserving the same visual sharpness, more formally

$$X = \uparrow (Y)\ s.t.\ \mathcal{X} \approx \mathcal{Y}, \tag{1}$$

where Y is the input image, X is the output upscaled image, $\uparrow(\cdot)$ is an upsampling operator and calligraphic font denotes the spectrum of an image.

In the literature this transformation has usually been modeled backwards as the restoration of an original image that has suffered a certain degradation [9]

$$Y = \downarrow (B(X)), \tag{2}$$

where $B(\cdot)$ is a blurring filter and $\downarrow(\cdot)$ is a downsampling operator. The problem is usually addressed at a patch level, denoted with lower case (e.g. y, x) and extracted from the respective uppercase image (e.g. Y, X).

The example-based SR family tackles the super-resolution problem by finding meaningful examples from which a HR counterpart is already known, namely the couple of dictionaries D_l and D_h.

The low resolution patch y is decomposed as a linear combination of the atoms in the LR dictionary D_l.

$$\min_{\beta} \|y - D_l\beta\|_2^2 + \lambda \|\beta\|_p, \tag{3}$$

where β selects and weights the elements in the dictionary and λ weights a possible L_p-norm regularization term. Once this linear combination of atoms β is obtained, the high resolution patch x can be obtained by applying those weights to the HR dictionary D_h as $x = D_h\beta$. The L_p-norm selection and the dictionary-building process depend on the chosen priors and they further define the SR algorithm.

B. State of the art

Regression-based SR methods have been one of the most prolific SR families of the recent years. This section we reviews some of the most relevant publications taking advantage of regression strategies, with a special emphasis on, but not limited to, piecewise linear regression schemes.

In regression-based SR the objective of training a given regressor R is to obtain a certain mapping function from LR to HR patches. LR patches form an input manifold M of dimension m and HR patches form a target manifold N of dimension n. Formally, for training pairs $(y_F,\ x)$ with $y_F \in M$ and $x \in N$, we would like to infer a mapping $\Psi : M \subseteq \mathbb{R}^m \rightarrow N \subseteq \mathbb{R}^n$.

The work of Kim et al. [15] on example-based learning builds on the framework of Freeman et al. [5] and aims to recover the HF content of X through a non-linear regression stage, which is then added to the bicubic interpolation. The regression consist in a Kernel Ridge Regression (KRR) whose cost functional reads:

$$\mathcal{O}(\left\{f^1,\ldots,f^N\right\}) = \sum_{i=1}^{N}(\frac{1}{2}\sum_{j=1}^{l}(f^i(y_{Fj})-x_j^i)^2+\frac{1}{2}\lambda\left\|f^i\right\|^2), \quad (4)$$

where the f^i is expressed as a Gaussian kernel:

$$f^i(y_{Fj}) = \sum_{j=1}^{l} a_j^i \exp(-\left\|y_{Fj}-x_j^i\right\|^2/\sigma_k), \text{ for } i = 1,\ldots,N. \quad (5)$$

The solution to the minimization problem of (4) is found only for a certain basis set of l_b elements much smaller than the full training set size ($l_b \ll l$), due to prohibitive time complexity. These basis points are selected from the training datapoints through an incremental approach based on Kernel Matching Pursuit and gradient descent, having the necessity to apply additional simplifications to further reduce the complexity. The training time reported by the authors is still about a day in a desktop PC. After the KRR stage, the number of candidates for each final pixel is combined following a certain weighting learnt during training, contrasting with the usual overlapping weighting done in most SR algorithms. As the output of these two stages presents ringing artifacts, a modified natural image prior based on the one of Tappen et al. [16] is applied through Belief Propagation (BP).

The recent work of Timofte et al. [13] is especially remarkable for its low-complexity nature which achieves orders of magnitude speed-ups while having competitive quality results compared to the state-of-the-art. They propose a relaxation of the L_1-norm regularization commonly used in most of the Neighbor Embedding (NE) and Sparse Coding (SC) approaches, reformulating the problem as a least squares (LS) L_2-norm regularized regression, also known as Ridge Regression. While solving L_1-norm constrained minimization problems is computationally demanding, when relaxing it to a L_2-norm, a closed-form solution can be used. Their proposed minimization problem reads

$$\min_{\beta} \left\|y_F - N_l\beta\right\|_2^2 + \lambda\left\|\beta\right\|_2, \quad (6)$$

where N_l is the LR neighborhood chosen to solve the problem and y_F is a feature extracted from a LR patch. The algebraic solution is

$$\beta = (N_l^T N_l + \lambda I)^{-1} N_l^T y_F. \quad (7)$$

The coefficients of β are applied to the corresponding HR neighborhood N_h to reconstruct the HR patch, i.e. $x = N_h\beta$. This can also be written as the matrix multiplication $x = Ry_F$, where the projection matrix (i.e. regressor) is calculated as:

$$R = N_h(N_l^T N_l + \lambda I)^{-1} N_l^T \quad (8)$$

and can be computed offline, therefore moving the minimization problem from testing to training time.

They propose to use sparse dictionaries of d_s atoms size, trained with the K-SVD algorithm [17]. A regressor R_j is anchored to each atom d_j in D_l, and the neighborhood N_l in equation (8) is selected from a k-NN subset of D_l:

$$N_{l_j} = \mathrm{kNN}(d_j,\ D_l). \quad (9)$$

The SR problem can be addressed by finding the NN atom d_j of every input patch feature y_{F_i} and applying the associated R_j to it. In the specific case of a neighborhood size $k = d_s$, only one general regressor is obtained whose neighborhood comprises all the atoms of the dictionary and consequently does not require a NN search. This case is referred in the original paper as Global Regression (GR).

Other advances within this family of algorithms include the Naive Bayes SR Forest [18], that uses an ensemble of bimodal trees in order to benefit from antipodal invariance. It is also highly competitive in speed thanks to the naive Bayes selection procedure which applies a single regressor per patch, differently from other forest approaches, e.g. SR Forest of Schulter et al. [19], that apply as many regressors per patch as the number of trees in the ensemble. Another recent follow-up of ANR is the Multiple Linear Mappings (MLM) SR of Zhang et al. [20], which trains each regressor with a set of orthonormal basis representative of the correspondent subspace (i.e. a subdictionary) and later performs a post-processing stage based on non-local means regularization.

III. Proposed algorithm

In this section we present our algorithm for fast and efficient regression-based SR, divided in four subsections that describe the main pillars of our contributions. We show an overview of the complete algorithm in Figure 2.

A. Linear Regression Framework

As seen in Section II, selecting non-linear regression schemes (e.g. KRR of [15]) results in prohibitive training complexity, which is usually mitigated by reducing the training sets under certain assumptions. In testing time, although non-linear regression SR methods are reasonably fast, they still compare modestly with fast state of the art methods such as [13], [14], [21].

Linear Regression is the most simple regression scheme, i.e. for each output variable it performs a linear weighted sum

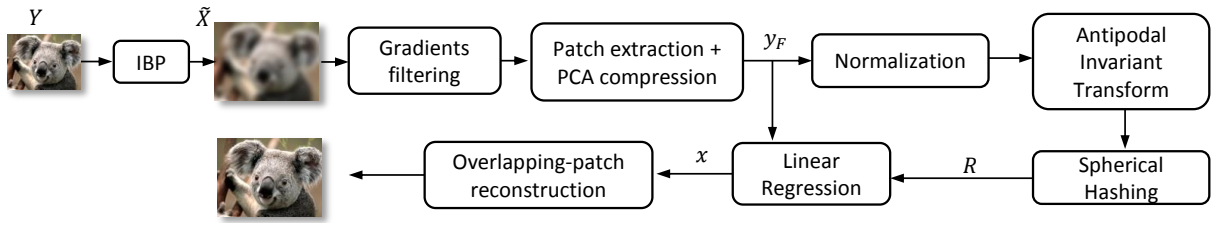


Fig. 2: Pipeline of our SR algorithm. Firstly, we extract feature vectors from the gradients of an image upscaled through IBP. Secondly, we find the best-fitted regressor R through spherical hashing using normalized features and our proposed AIT. We then apply the respective regressor to each feature vector and reconstruct the final image with overlapping patches.

of the input variables. This is an oversimplification of the upscaling problem if we only consider one linear regressor. Instead, several linear regressors are anchored to different points of the manifold, obtaining a finer piecewise linear regression model. In such strategies, data have to be split in training time and during testing time the proper regressor has to be selected.

For the SR problem, the regression is applied to the input features and aims to recover certain components of the patch, e.g. missing high frequency. We model the linear regression framework in a general way as:

$$x = \tilde{x} + R_i\, y_F,\ s.t.\ R_i = \underset{R_i \in \{R_k\}}{\operatorname{argmin}}\ \delta(R_i,\ y_F), \tag{10}$$

where $\tilde{x}$ is a coarse first approximation of the HR patch x, $\delta(\cdot)$ is a metric evaluating the dissimilarity from the input features to the ith regressor cluster or anchor point and $\{R_k\}$ is an ensemble of trained regressors.

The choice of how to obtain $\tilde{x}$ affects the content to be learnt, since the regression output aims to recover $x - \tilde{x}$. We further discuss the importance of this in Section III-B together with the feature space selection for the transformation of y_F.

B. Feature Space and coarse approximation

SR algorithms are usually performed in a feature space other than that of the raw luminance pixel values. In the literature, a common rule for this feature transformation is to enforce mid and high frequencies of LR patches, under the observation that similarity between LR and HR patch structures is somehow improved and therefore the prediction is easier. As for the HR feature space (i.e. the output feature space), the same principle of enforcing high frequencies also applies, in this case under the assumption that the high frequency bands are conditionally independent of the lower frequency bands, and thus suppressing low-frequency bands from the HR feature space collapses the training data for all possible low-frequency values into one value [5]. Differently from the input LR feature space, in the HR feature space we need to be able to reverse the features into pixel-based values for the final image reconstruction.

Several features have been proposed: The early work of Freeman et al. [5] already used a simple high-pass filter which consisted in the subtraction of a low-pass filter. In the same direction, [12] and [9] used concatenated first- and second-order gradients, as an inexpensive solution to the same high-pass filter approximation. This type of feature was further refined by Zeyde et al. [10] by introducing PCA compression in order to reduce the feature dimensionality and memory usage.

It is important to remark that most feature transformations are computed from a first coarse approximation, i.e. the upscaled image $\tilde{X}$. We observed that the effect of this first approximation has been unnoticed in the literature, in which using bicubic interpolation or the patch-mean value is a common practice, both in LR and HR feature space. In this paper we propose a new feature transform which takes advantage of a better coarse approximation to obtain the LR input features, which we denote with y_F.

The main idea is to obtain an image approximation $\tilde{X}$ better than that obtained with bicubic interpolation but which is still within certain low-complexity boundaries. We present a feature transform based on a simplified Iterative Back Projection (IBP) algorithm of [2], together with unidimensional 1-st and 2-nd order gradients. We refer to this novel feature transform as Gradient IBP (GIBP).

Starting with an initial guess $\tilde{X}^{(0)}$ of the HR image, IBP simulates the imaging process to obtain a LR image $\tilde{Y}^{(0)}$ corresponding to the observed input image Y. The difference image $E^{(0)} =\uparrow (Y - \tilde{Y}^{(0)})$ is computed and used to improve the initial guess by back-projecting each error value onto the corresponding field in $\tilde{X}^{(0)}$, namely $\tilde{X}^{(1)} = \tilde{X}^{(0)} + E^{(0)}$. This process is repeated iteratively:

$$\tilde{X}^{(n)} = \tilde{X}^{(0)} + \sum_{j=1}^{n} E^{(j-1)} \tag{11}$$

In the original work of Irani and Peleg [2], the imaging process was modeled through a convolution with a given point spread function and a back-projection kernel. In our low-complexity approach, we model both functions with the simple and effective bicubic downscaling and upscaling kernel. With as few as $n = 2$ iterations the coarse approximation improves greatly when compared to bicubic. We filter this upscaled image $\tilde{X}$ with 1-st and 2-nd order unidimensional gradient filters (two vertical and two horizontal). At this point overlapping patches are extracted from each of the gradient images, and all the 4 gradient patches corresponding to the same patch position are concatenated together in a feature vector y_F. Note that dimensionality of this feature is four

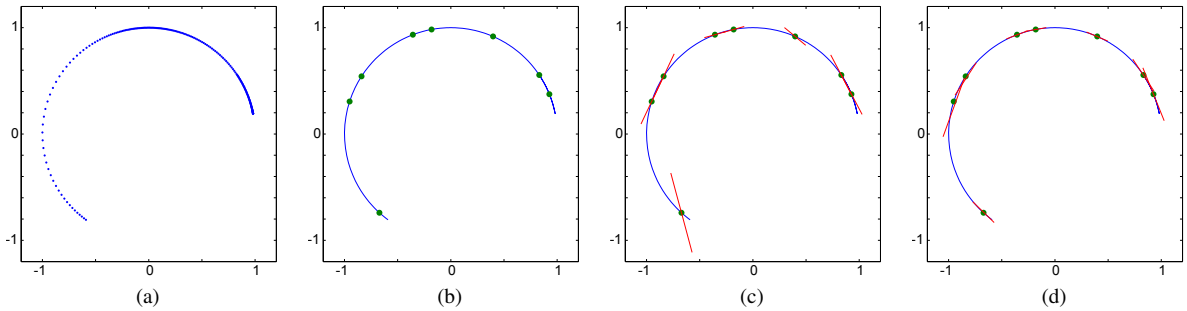


Fig. 3: A normalized degree 3 polynomial manifold illustrating the proposed approach compared to the one in [13]. (a) Bidimensional manifold samples. (b) The manifold (blue) and the sparse representation obtained with K-SVD algorithm (green) of 8 atoms. (c) Linear regressors (red) trained with the neighborhoods ($k = 1$) obtained within the sparse dictionary, as in [13]. (d) Linear regressors (red) obtained using our proposed approach: The neighborhoods are obtained within the samples from the manifold ($k = 10$).

times the patch size. If this eventually becomes a memory problem, a PCA compression can be applied with barely no information loss [10].

Section IV-A assess the performance of our proposed feature transform. As for the HR feature space, we consistently use IBP, without the non-reversible gradient step and PCA compression. During training, we form our HR features simply by the subtraction of the the first coarse approximation to the ground-truth patch $x - \tilde{x}^{(2)}$, so that our regression stage is specialized in correcting the errors that IBP is introducing. During testing, this HR feature transform requires substituting $\tilde{x}$ by $\tilde{x}^{(2)}$ in Equation 10.

C. Clustering and training

As we have previously seen, recent regression-based SR use linear regressors because they can be easily computed in closed form and applied as a matrix multiplication. However, the mapping Ψ is highly complex and non-linear [22]. To model the non-linearity nature of the mapping, an ensemble of regressors $\{R_k\}$ is trained, representing a locally linear parametrization of Ψ, under the assumption that both manifolds M and N have a similar local geometry. We analyze the effect on the distribution of those regressors in the manifold (i.e. the anchor points) and the importance of properly choosing the N_l in equation (8), concluding on a new training approach.

In the work of Timofte et al. [13], an overcomplete sparse representation is obtained from the initial LR training patches using K-SVD [17]. The atoms in this new reduced dictionary D_l are used both as anchor points to the manifold and datapoints for the regression training. In their GR, a unique regressor R_G is trained with all elements of the dictionary, therefore accepting higher regression errors due to the single linearization of the manifold. In order to achieve a more accurate regression they also propose the Anchored Neighborhood Regression (ANR). In ANR they use several regressors anchored to each sparse dictionary atom. They build for each of those atoms a neighborhood of k-NN composed by other atoms within the same sparse dictionary D_l, and finally train a ridge regressor with the corresponding neighborhood.

Performing a sparse decomposition of a high number of patches efficiently compresses data in a much smaller dictionary, yielding atoms which are representative of the whole training dataset, i.e. the whole manifold. For this reason they are suitable to be used as anchor points, but also sub-optimal for the neighborhood embedding. They are sub-optimal since the necessary local condition for the linearity assumption is likely to be violated. The atoms of a sparse dictionary are a set of centroids that represent the whole training set and, as a consequence, they are not likely to be similar among them. Important variations within sparse atoms are to be expected, therefore being unappropriated for locally linear embedding.

Following this observation, in [14] we proposed a different approach when training linear regressors for SR: Using sparse representations as anchor points to the manifold, but forming the neighborhoods with raw manifold samples (e.g. features, patches). By doing so, we find closer nearest neighbors and, therefore, fulfill better the local condition. Additionally, a higher number of local independent measurements is available (e.g. mean distance for 1000 neighbors in the raw-patch approach is comparable to a 40 atom neighborhood in the sparse approach) and we can control the number of k-NN selected, i.e. it is not upper-bounded by the dictionary size. We show a low-dimensional example of our proposed training scheme in Figure 3. Building dense and compact clusters through this methodology is a key contribution as it boosts performance at no complexity cost. Parallel in time, Timofte et al. presented a similar idea in their A+ algorithm [23].

In the present work we further optimize the training stage by reviewing the nearest neighbor search strategy, which follows on Section III-D, where we propose the usage of different metrics that allow even better locality conditions.

D. Search strategy: Antipodally Invariant Spherical Hashing

In this section we study the most appropriate metrics for evaluating the similarity between features and anchor points, presenting some insight about their behavior with antipodal points and the importance of properly dealing with them. We also present a novel search strategy that speeds up the regressor selection and introduce a modification of the Spherical

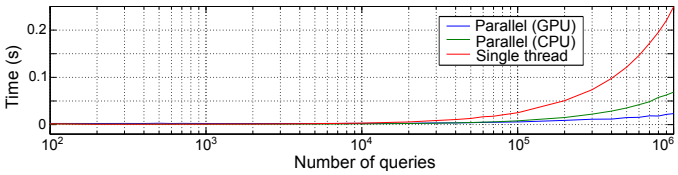


Fig. 4: Running times measured when computing 6-bit hash codes (6 hyperspheres) for increasing number of queries (in logarithmic axis and without re-ranking) for single-threaded (CPU) and parallel (CPU and GPU) implementations.

Hashing algorithm [24] in order to benefit from antipodal invariance.

SR algorithms require performing comparisons between vectors in order to guide the algorithm in its decisions, e.g. features, anchor points, dictionary atoms. The comparison is performed following a given metric function $\delta(a,\, b) a, b \in \mathbb{R}^m$ which typically has been designed to quantify the similarity or dissimilarity of the two compared vectors. It is recurrent in literature the usage of Euclidean distance for this purpose. We show that for the characteristics of the features used in our proposed algorithm (which are also shared in other SR algorithms [10], [13], [23], [14]), Euclidean distance is suboptimal as it fails to manage antipodal points (i.e. two points on the surface of a sphere are antipodal if they are diametrically opposite).

We find further insight of the ambiguous variations that our defined metric should ignore when looking at scalar matrix multiplication in linear regression, i.e. for a given scalar λ our linear regression reads $\lambda x = R(\lambda y)$. The regressor R and the associated linear operations are not modified by this scaling operation, as it is present at both sides of the equality. Normalizing feature vectors when comparing them is sometimes necessary as many dictionary optimization algorithms (as it is the case of K-SVD) enforce normalization to avoid a 0 solution. Even when it is not necessary, normalization is a good practice as it collapses training examples with similar structures but different scalar norms. During testing, the normalized features are used to find the best-suited regressor, while the non-normalized features are necessary for the final matrix multiplication.

However, vector normalization does not handle the cases where λ is negative, as the norms are strictly positive and can not compensate for $\lambda \in \mathbb{R}^-$. This is the case of antipodal points (i.e. there is a sign change).

Training and assigning different regressors for two antipodal points does not increase the performance by a better specialization, as the sign change is in both sides of the equality and the regressor and the associated linear operations are identical for two antipodal points (i.e. $x = R(y)$ and $-x = R(-y)$). Each regressor is associated with an anchor point, which describes a certain *mode* in the structure of patches, regardless of this structure being a positive or negative change (e.g. positive or negative change in the gradient), which is described by the sign of the normalized vector. The metric utilized for selecting the best regressor should therefore be able to associate two antipodal points to the same anchor point, thus having *antipodal invariance*. In the same way, when building the neighborhoods during training, this observation also applies.

We define an antipodal invariant metric as:

$$\delta(a,\, b) = \delta(-a,\, b) = \delta(a,\, -b) = \delta(-a,\, -b). \tag{12}$$

We propose the usage of the Absolute Value of the Cosine Similarity (AVCS) as a better alternative to compare feature vectors to anchor points. This metric $\delta_{|\cdot|}(a,\, b) = |a \cdot b|$ is antipodally invariant (respects Equation (12)), and furthermore, requires less operations than a common Euclidean distance calculation. During training, we can use it as there are no time constrains. Refer to Section IV to assess the improvement of using antipodally invariant metrics both for training and for testing.

In testing time, the best-fitted regressor has to be selected from within the ensemble of trained regressor. Although state-of-the art regression-based SR has already pushed forward the speed with regard to other dictionary-based SR [10], [9], finding the right regressor for each patch takes a considerable quota of the processing time. In the work of [13], most of the *encoding time* (i.e. time left after subtracting shared processing time, including bicubic interpolations, patch extractions, etc.) is spent in this task (i.e. $\sim 96\%$ of the time).

In order to reduce the search complexity, binary splits (e.g. trees or hashing functions) are trained forming a hierarchical structure composed by several split nodes. These split nodes divide the space in such a way that the children leaves improve their information gain. We face several challenges when adapting binary search structures to our algorithm: The outcome of our training stage is a set of anchor points associated with their respective regressors. This dictionary is obtained independently and ahead from the search structure, does not have any hierarchical structure and the neighborhoods, which can be interpreted as clusters, share elements among them. Furthermore, antipodal invariance is an important aspect on the search strategy as discussed previously. This search structure has to be built, therefore, adapted to this scenario.

We choose hashing techniques over tree-based methods for two main reasons: Hashing schemes provide low memory usage (the number of splitting functions in hashing-based structures is $O(\log_2(n))$ while in tree-based structures is $O(n)$, where n represents the number of clusters) and are highly parallelizable.

Binary hashing techniques aim to embed high-dimensional points in binary codes, providing a compact representation of high-dimensional data. Among their vast range of applications, they can be used for efficient similarity search, including approximate nearest neighbor retrieval, since hashing codes preserve relative distances. There has recently been active research in data-dependent hashing functions opposed to hashing methods such as [25] which are data-independent. Data-dependent methods intend to better fit the hashing function to the data distribution [26], [27] through an off-line training stage.

Among the data-dependent state-of-the-arts methods, we select the Spherical Hashing algorithm of Heo et al. [24],

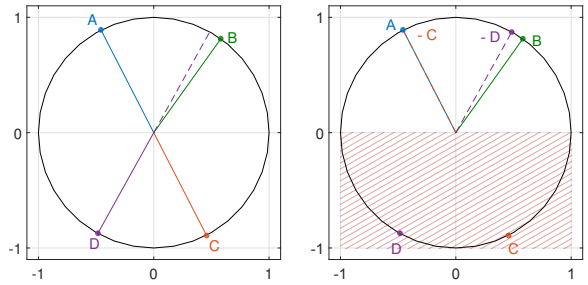


Fig. 5: Antipodally Invariant Transform. (a) A and C are antipodal points ($C = -A$) and B and D are close to be antipodes ($D \approx -B$). Although they are very similar features, Euclidean distance fails to find the proper nearest neighbor, as it can not handle the characteristics of antipodal points. (b) In our proposed Antipodally Invariant Transformation (AIT), we forbid a given half-space of the manifold (in this case the negative y axis, i.e. $y_F \cdot \mathbf{e}_y < 0$) and we project the features laying on it to its correspondent antipodal point. After transformation, Euclidean distance can properly deal with antipodal points.

which is able to define closed regions in $\mathbb{R}^m$ with as few as one splitting function. This hashing framework is useful to model the inverse search scheme and enables to benefit from substantial speed-ups by reducing the NN search into applying a precomputed function, which conveniently scales with parallel implementations, as shown in Figure 4.

Spherical hashing differs from previous approaches by setting hyperspheres to define hashing functions on behalf of the previously used hyperplanes. A given hashing function $H(y_F) = (h_1(y_F), \ldots, h_c(y_F))$ maps points from $\mathbb{R}^m$ to a base 2 $\mathbb{N}^c$, i.e. $\{0,1\}^c$. Every hashing function $h_k(y_F)$ indicates whether the point y_F is inside kth hypersphere, modeled for this purpose as a *pivot* $p_k \in \mathbb{R}^m$ and a distance threshold (i.e. radius of the hypersphere) $t_k \in \mathbb{R}^+$ as:

$$h_k(y_F) = \begin{cases} 0 & when\ \delta(p_k, y_F) > t_k \\ 1 & when\ \delta(p_k, y_F) \leq t_k \end{cases}, \tag{13}$$

where $\delta(p_k, y_F)$ denotes a distance metric between two points in $\mathbb{R}^m$ (Euclidean distance in the work of Heo et al.[24]). The advantages of using hyperspheres instead of hyperplanes is the ability to define closed tighter sub-spaces in $\mathbb{R}^m$ as intersection of hyperspheres. An iterative optimization training process is proposed in [24] to obtain the set $\{p_k, t_k\}$, aiming a balanced partitioning of the training data and independence between hashing functions. We perform this mentioned iterative hashing-function optimization in a set of input patch features from training images, so that $H(y_F)$ adapts to the natural image distribution in the feature space.

Our proposed spherical hashing search scheme becomes symmetrical as we can see in Figure 6, i.e. both image and anchor points have to be labeled with binary codes. This can be intuitively understood as creating NN subspace groups (we refer them as *bins*), which we label with a regressor by applying the same hashing functions to the anchor points. Relating a hash code with a regressor can be done during training time.

The search returns k-NN for each anchor point, thus not ensuring that all the input image patches have a related regressor (i.e. whenever the patch is not within the k-NN of any of the anchor points). Two solutions are proposed: (a) use a general regressor for the patches which are not in the k-NN of any anchor point, as proposed by [13] or (b) use the regressor of the closest labeled hash code calculated with the spherical Hamming distance, defined by [24] as $d_{SH}(a,b) = \frac{\sum(a \oplus b)}{\sum(a \wedge b)}$, where $\oplus$ is the XOR bit operation and $\wedge$ is the AND bit operation. Note that although not being guaranteed, it rarely happens that a patch is not within any of the k-NN regressors (e.g. for the selected parameter of 6 hyperspheres it never occurs). Since we have not observed significant differences in performance, we select (a) as the lowest complexity solution.

In a similar way, an inverse search might also assign two or more regressors to a single patch. It is common in the literature to do a re-ranking strategy to deal with this issue [28].

Antipodally Invariant SpH. During the opening of this section we discussed the goodness of properly dealing with the antipodes. However, as many other binary splits, the functioning of spherical hashing is unable to recognize antipodal vectors and put them together in a bin. This happens by cause of its thresholding mechanism, which is based on a directionless Euclidean distance from the center of a hypersphere to the feature point. As seen previously, antipodal patches should use the same regressor as the same linear operations are applied for both of them, with the unique difference of the input features' sign.

In order to benefit from the speed-ups of binary splits designed to use Euclidean distance, we propose a novel transformation which enables the benefits of antipodal invariance in the Euclidean space.

As described by the Borsuk-Ulam theorem [29], any continuous function from an m dimensional unit sphere S^m to an Euclidean m-space $\mathbb{R}^m$ maps two antipodal points into a single point. The goal of our transformation is similar: to design a function which maps two antipodal points into a single point. However, in the situation addressed in this paper there is a mismatch between the sphere space and the Euclidean space, as we deal with a function mapping from S^{m-1} to $\mathbb{R}^m$.

Our idea builds on designing a continuous hyperplane that crosses S^{m-1} passing through the origin of coordinates, thus mapping two points into a single one. In order to do that, we enforce a forbidden space region, corresponding to the negative half-space of the qth dimension, i.e. the features must be $y_F \cdot \mathbf{e_q} \in \mathbb{R}^+$, where $\mathbf{e_q}$ is the qth standard basis in the Euclidean m space.

In our proposed Antipodally Invariant Transformation (AIT), all the features lying outside that space are projected to their antipodes:

$$y_{F_{AIT}} = -y_F, \qquad \text{if } y_F \cdot \mathbf{e_q} < 0. \tag{14}$$

A low-dimensional illustrative example is shown in Figure 5. The dimension q should have balanced positive and

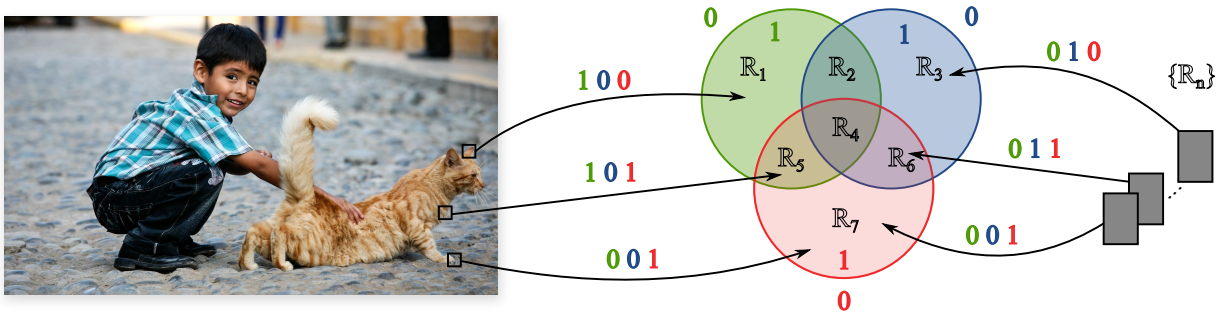


Fig. 6: Spherical hashing adapted to the regressor search for SR. Certain hashing functions are optimized on feature patch statistics creating a set of hyperspheres intersections that are directly labeled with a hash code. In training time, regressors fill this intersections (i.e. bins) and in testing time the hashing function is applied to each patch, which will directly map it to a regressor.

negative values so that the transformation is effective. For our training feature vectors (around 500K examples), all the dimensions were highly and similarly balanced, so we decided to select the dimension with less variation (PCA analysis).

If rather than transforming the points we just want to use an antipodally invariant metric which can operate in the Euclidean space, Equation (14) is applied in both vectors during the Euclidean distance calculation:

$$\delta_{AIT}(p_k, y_F) = \sqrt{\sum_m (p_{k_{AIT}}, y_{F_{AIT}})^2}, \qquad (15)$$

and then used in the distance metric $\delta(p_k, y_F)$ of Equation (13). Thanks to this we obtain an Antipodally Invariant Spherical Hashing (AISpH) which is optimal for the SR regression problem and can be used for any other problem which shares the same feature characteristics.

In Figure 7 we show the advantages of using our proposed AIT. In Figure 7(a), we confirm that neighborhoods created with AIT features and Euclidean distance metric have lower average distances than without transformation, hence obtaining a better local condition and having a higher number of samples available for a given maximum distance. In Figure 7(b), we asses the resilience to antipodal variance of AIT: The average angular distances obtained with AIT neighborhoods (Equation 15) are approaching those created with a pure antipodally invariant metric (i.e. AVCS). This is further validated by the results shown in Table II, where AIT and AVCS obtain similar PSNR performance.

IV. Configuration

In this section we assess and prove the optimal performance of the contributions of the paper separately. We use as baseline configuration our proposed algorithm, with 1 hypersphere and AIT, trained with the 91 training images from [8], using a single scale (about $500k$ training samples) and a sparse dictionary of 1024 atoms whenever is not specified otherwise. The size of the neighborhoods is set to 3000 samples and we weight the regularization term with $\lambda = 0.12$. We select a patch size of 3×3 and extract full overlapping patches in the LR reference image (i.e. the number of patches extracted in the upper scales is the same as in the LR reference scale, thus the patch size and overlap ratio are proportionally adapted).

	Set5		Set14	
	PSNR	time	PSNR	time
bic. gradients	32.55	0.216	23.27	0.407
GIBP	32.65	0.222	32.33	0.419

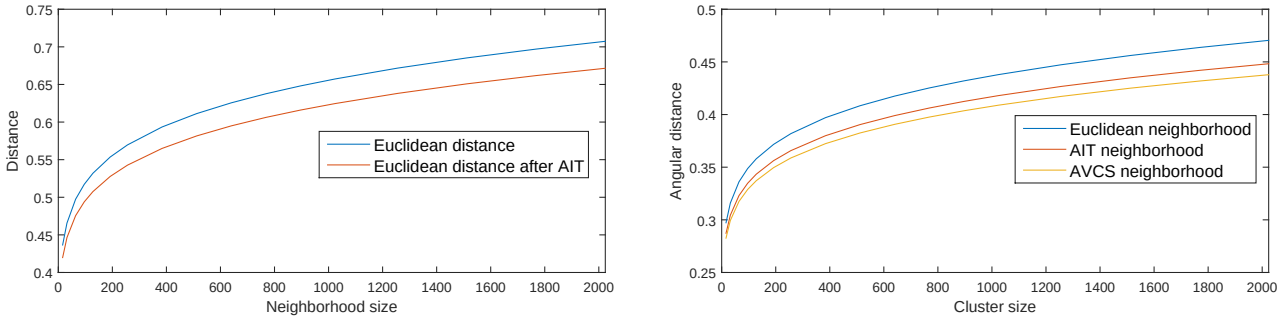
TABLE I: Average performance in terms of PSNR (dB) and time (s) for bicubic gradients features and our GIBP features, run on Set14 and Set5 on a $\times 2$ magnification factor.

A. Feature Space

We compare the performance of our proposed GIBP features with the features proposed by Zeyde et al. which are based on the gradients of the bicubic interpolation. We compress both features with PCA in order to reduce the dimensionality, so that not only the features are more compact, but specially the regressors. In Table I we show how by using our proposed features we consistently improve in quality (i.e. from 0.06dB to 0.10dB) with respect the previously used features. We also asses that, as expected, the computation time of the whole SR algorithm increases as GIBP requires the computation of more bicubic interpolations (three interpolation against a single one). Nevertheless the increase in running time has low incidence with respect the whole SR pipeline (i.e. about 3% of the total SR time).

B. Search strategy

In order to confirm the superiority of antipodally invariant metrics, we test Euclidean distance against AVCS and our proposed AIT+Euclidean distance. We only want to evaluate the incidence of the different metrics, excluding from the experimental setup the effect of the approximate search introduced with the AISpH (as this comparison is explicitly shown in the Results section). We perform experiments with the three different metrics used for both training and testing separately. The results clearly show the advantage of using an absolute invariant metric, as using them improves greatly the PSNR (+0.18 dB). We also prove the invariance of our



(a) (b)

Fig. 7: Average distance of the neighborhoods to their anchor points for increasing neighborhood sizes. (a) shows the Euclidean distance of neighborhoods created before and after AIT of the features and (b) shows angular distance (i.e. the distance derived from cosine similarity) for the neighborhoods obtained with different metrics: Euclidean distance, AIT Euclidean distance and AVCS. In (a) we show how thanks to our AIT the clusters are tighter in the Euclidean space. In (b) we assess how we improve the invariance to antipodality with respect to Euclidean distance, being very close to the curve obtained with a pure antipodally invariant metric (i.e. AVCS).

		Testing		
		Cosine	Euclidean	AIT
Training	Cosine	32.33	32.21	32.33
	Euclidean	32.27	32.15	32.26

TABLE II: Performance of different metrics for training and testing run on Set14 and $\times 2$ magnification factor.

AIT as the quality obtained with it is equivalent to the one of AVCS. We also observe that both training and testing improve the performance separately when handling properly antipodes (which is $+0.12$dB for the testing, and at least $+0.06$dB for the training).

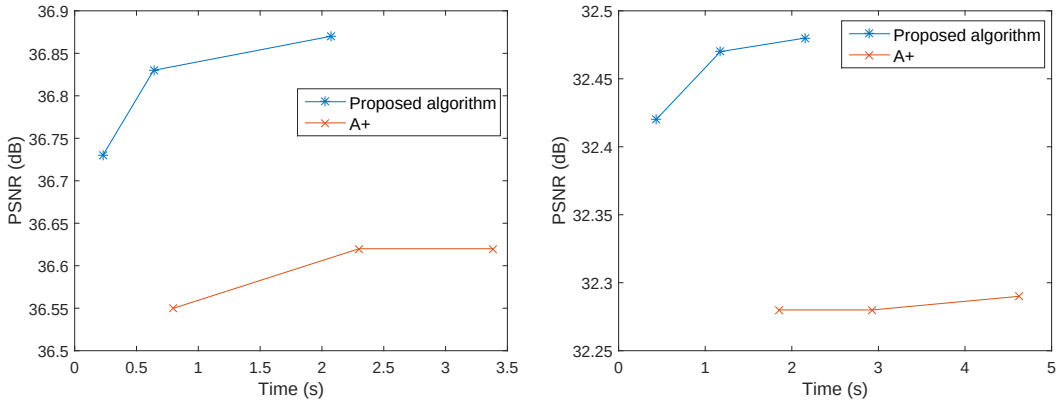
C. Dictionary and spheres ratio

In our presented algorithm the size of the sparse dictionary is not necessarily associated with the number of hyperspheres used during testing time. Our hashing scheme defines several hash codes or buckets, and the regressors are labeled with them during training time. In the case of having more than one regressor per bucket, a reranking strategy is followed and thus the best-suited regressor is obtained from the bucket's candidates. The ratio between the number of sparse atoms and the number of buckets (2^s where s is the number of hyperspheres) gives an average number of regressors per hash code. In this section we show that our algorithm scales well in terms of quality when increasing the size of the sparse dictionary, and therefore, is worth increasing its size and adapting the number of hyperspheres to obtain the desired quality and speed trade-off. In Figure 9 we show how our algorithm scales better by increasing the dictionary size than A+, which improvement is always smaller and tends to saturate earlier. We obtain maximum quality by setting our dictionary size to 8192 elements and, afterwards, fixing a number of hyperspheres which gives us the desired speed. We aim to obtain the same speed as the original work of [14] where they used 1024 atoms and 6 hyperspheres. We obtain a very similar time figure (while obtaining substantially improved PSNR quality) with 7 spheres and 8192 atoms. Note also that with this particular configuration the difference between exhaustive search and approximate search is almost non-existent, specially for upscaling factor $\times 4$. We would like to remark that a comparison point to point in Figure 9 assess the speed-up thanks to our parallel implementation, since both algorithms are using exhaustive search and the algorithm complexity is therefore the same. The figure also shows that our methods is consistently obtaining better PSNR values (about 0.2dB higher) for different dictionary sizes, and that even with 1024 we perform better than A+ with 8192 atoms.

V. Results

In this section we show experimental results of our proposed method and we compare its performance in terms of quality and execution time to other state-of-the-art recent methods. All the experiments were run on a Intel Xeon W3690 @ 3.47GHz equipped with 12GB of RAM memory. The methods included in our benchmark are: bicubic as a baseline, the K-SVD method of Zeyde et al. [10] (denoted as *sparse*), the SR method based on Deep Learning of Dong et al. [21], both their opening work SRCNN [21] (referred as *DL 9-1-5*) and their most recent publication [30] (referred as *DL 9-5-5*), the ANR method of Timofte et al. [13] and its improved version A+ [23]. The datasets and the whole experimental setup follows the one used in [8], [9], [21], [14], with the addition of the kodak dataset and the $\times 2$ magnification factor. All the codes were obtained from their respective author's website. In the case of DL, the implementation they provide is slower than the used in their publication, which makes time comparison problematic. However, their reported times for *DL 9-5-5* are slower than those of A^+ in their benchmark [30]. All methods are trained with the training dataset of [9], which contains image crops with high frequencies with the exception of *DL 9-5-5* that uses ImageNet (in the order of hundred of thousand images). For the neighborhood embedding, in both our AIS and A+ we use

Fig. 8: Close-ups of the results for visual qualitative assessment of a $\times 2$ (three first rows) and $\times 3$ (last four rows) magnification factors from the datasets in the benchmark. Best-viewed zoomed in.



(a) Set5 (b) Set14

Fig. 9: PSNR vs time values for dictionary sizes of 1024, 4096 and 8192 atoms (from left to right in the lines) for our proposed method with exhaustive search and A+ .

TABLE III: Performance of $\times 2$, $\times 3$ and $\times 4$ magnification in terms of averaged PSNR (dB) and averaged execution time (s) on datasets Set5, Set14 and Kodak. Best results in bold.

		Bicubic		Sparse [10]		ANR [13]		DL 9-1-5 [21]		DL 9-5-5 [30]		A+ [23]		AIS, $s = 1$		AIS, $s = 7$	
	MF	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time
Set5	2	33.66	0.002	35.78	3.181	35.83	0.712	36.34	3.953	36.66	4.434	36.55	0.761	**36.87**	1.167	36.80	**0.105**
	3	30.39	0.002	31.90	1.474	31.92	0.449	32.39	3.916	32.75	4.950	32.59	0.467	**32.79**	0.583	32.75	**0.080**
	4	28.42	0.002	29.69	0.916	29.69	0.348	30.09	4.031	**30.48**	10.284	30.29	0.346	30.46	0.382	30.45	**0.075**
Set14	2	30.23	0.002	31.81	6.506	31.80	1.717	32.18	7.695	32.45	8.204	32.28	1.739	**32.48**	2.223	32.44	**0.205**
	3	27.54	0.002	28.67	3.003	28.65	0.933	29.00	7.646	**29.29**	8.098	29.13	0.963	29.26	1.075	29.23	**0.153**
	4	26.00	0.002	26.88	1.862	26.85	0.696	27.20	7.944	**27.50**	8.305	27.32	0.714	27.45	0.716	27.42	**0.155**
Kodak	2	30.85	0.003	32.19	11.286	32.24	2.938	32.63	13.121	32.81	14.367	32.71	3.161	**32.89**	3.943	32.84	**0.339**
	3	28.43	0.003	29.22	5.250	29.21	1.615	29.43	12.805	29.65	15.026	29.57	1.678	**29.68**	1.771	29.65	**0.246**
	4	27.23	0.003	27.83	3.228	27.80	1.199	27.94	13.315	**28.17**	14.069	28.10	1.226	**28.17**	1.186	28.15	**0.245**

TABLE IV: Performance of $\times 2$, $\times 3$ and $\times 4$ magnification in terms of averaged IFC and averaged SSIM on datasets Set5, Set14 and Kodak. Best results in bold.

		Bicubic		Sparse [10]		ANR [13]		SRCNN[21]		SRCNN [30]		A+ [23]		AIS, $s = 1$		AIS, $s = 7$	
	MF	IFC	SSIM	IFC	SSIM	IFC	SSIM	IFC	SSIM	IFC	SSIM	IFC	SSIM	IFC	SSIM	IFC	SSIM
Set5	2	6.083	0.9299	7.856	0.9493	8.090	0.9499	7.524	0.9521	8.036	0.9542	8.477	0.9544	**8.683**	**0.9560**	8.628	0.9557
	3	3.579	0.8682	4.483	0.8968	4.606	0.8968	4.313	0.9033	4.658	0.9090	4.929	0.9088	**5.060**	**0.9121**	5.022	0.9111
	4	2.329	0.8104	2.935	0.8428	3.005	0.8419	2.844	0.8530	2.991	0.8628	3.249	0.8603	**3.348**	**0.8655**	3.319	0.8643
Set14	2	6.105	0.8687	7.663	0.8988	7.846	0.9004	7.237	0.9039	7.785	0.9067	8.140	0.9056	**8.292**	**0.9081**	8.261	0.9076
	3	3.473	0.7736	4.218	0.8075	4.317	0.8093	4.026	0.8145	4.338	0.8215	4.535	0.8188	**4.643**	**0.8227**	4.609	0.8218
	4	2.237	0.7019	2.725	0.7342	2.792	0.7353	2.614	0.7413	2.751	0.7513	2.961	0.7491	**3.034**	**0.7537**	3.009	0.7526
Kodak	2	5.711	0.8694	7.025	0.8993	7.187	0.9013	6.746	0.9050	7.150	0.9073	7.381	0.9075	**7.493**	**0.9102**	7.471	0.9096
	3	3.214	0.7781	3.827	0.8064	3.906	0.8083	3.656	0.8116	3.895	0.8177	4.053	0.8174	**4.132**	**0.8208**	4.109	0.8198
	4	2.026	0.7186	2.431	0.7430	2.472	0.7438	2.330	0.7454	2.430	0.7540	2.593	0.7539	**2.650**	**0.7572**	2.632	0.7563

pyramid of multiple scales to obtain more training samples (12 scales). We upscale images by the magnification factors $\times 2$, $\times 3$ and $\times 4$ with the authors' recommended configurations and measure PSNR, time, Structural Similarity (SSIM) [31] and Information Fidelity Criterion (IFC) [32], which has the highest correlation with perceptual scores for SR evaluation [33].

Our proposed AIS has a parallel implementation, and runs in the same CPU platform used for all methods. We use a LR patch size of 3×3 with LR full overlapping. We use a K-SVD sparse dictionary of 8192 elements and the chosen neighborhood size of 4250 k-NN, regularized with $\lambda = 0.12$. A+ uses a dictionary of 1024 atoms and a neighborhood size of 2048 atoms, as setting it to 4250 degraded their quality results. See Section IV-C for further discussion about the sparse dictionary size and the number of hyperspheres selected both in AIS and how it relates to A+. We present two configurations of our algorithm: 1 hypersphere (i.e. exhaustive search) which sets an upper quality limit and 7 hyperspheres which is our optimal configuration in terms of quality vs speed trade-off. By showing both configurations we evaluate the effect of the approximate search both in quality drop and in time speed-up, showing at the same time the full potential of the antipodal search and GIBP features.

We show objective evaluation of all methods in Table III (PSNR) and Table IV (IFC and SSIM). First of all, the PSNR obtained with our AIS is consistently around 0.2dB higher than A+, which is the most related compared method. The

TABLE V: Performance of $\times 2$, $\times 3$ and $\times 4$ magnification in terms of PSNR (dB) and execution time (s) on the Set5 dataset. Best results in bold.

Set5		Bicubic		Sparse [10]		ANR [13]		SRCNN [21]		SRCNN [30]		A+ [23]		AIS, $s=1$		AIS,$s=7$	
images	MF	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time
baby	2	37.1	0.003	38.2	7.537	38.4	1.571	38.5	8.563	38.5	8.860	38.5	1.617	**38.6**	2.644	38.6	**0.234**
bird	2	36.8	0.002	39.9	2.251	40.0	0.457	40.6	2.943	40.9	3.462	41.1	0.549	**41.7**	0.887	41.6	**0.079**
butterfly	2	27.4	0.002	30.6	1.793	30.5	0.447	32.2	2.555	32.8	3.088	32.0	0.476	**32.7**	0.678	32.5	**0.065**
head	2	34.9	0.002	35.6	2.170	35.7	0.546	35.6	2.831	35.7	3.437	35.8	0.576	**35.8**	0.831	35.8	**0.073**
woman	2	32.1	0.002	34.5	2.151	34.5	0.540	34.9	2.870	35.4	3.321	35.3	0.588	**35.6**	0.795	35.5	**0.072**
average	2	33.66	0.002	35.78	3.181	35.83	0.712	36.34	3.953	36.66	4.434	36.55	0.761	**36.87**	1.167	36.80	**0.105**
baby	3	33.9	0.003	35.1	3.471	35.1	1.061	35.0	8.226	35.2	9.077	35.2	1.101	**35.3**	1.240	35.3	**0.174**
bird	3	32.6	0.002	34.6	1.084	34.6	0.329	34.9	2.931	35.5	4.067	35.5	0.338	**35.9**	0.404	35.8	**0.065**
butterfly	3	24.0	0.002	25.9	0.854	25.9	0.246	27.6	2.626	**28.0**	3.622	27.2	0.256	27.6	0.383	27.5	**0.050**
head	3	32.9	0.002	33.6	0.973	33.6	0.305	33.5	2.896	33.7	4.060	33.8	0.321	**33.9**	0.444	33.8	**0.057**
woman	3	28.6	0.002	30.4	0.987	30.3	0.305	30.9	2.902	**31.4**	3.926	31.2	0.319	31.4	0.445	31.3	**0.057**
average	3	30.39	0.002	31.90	1.474	31.92	0.449	32.39	3.916	32.75	4.950	32.59	0.467	**32.79**	0.583	32.75	**0.080**
baby	4	31.8	0.003	33.1	2.213	33.0	0.844	33.0	8.531	33.1	9.550	33.3	0.826	**33.4**	0.824	33.3	**0.162**
bird	4	30.2	0.002	31.7	0.650	31.8	0.251	32.0	3.065	32.5	10.200	32.5	0.245	**32.8**	0.282	32.8	**0.054**
butterfly	4	22.1	0.002	23.6	0.499	23.5	0.194	25.1	2.693	**25.5**	9.465	24.4	0.196	24.7	0.253	24.7	**0.051**
head	4	31.6	0.002	32.2	0.614	32.3	0.226	32.2	2.925	32.4	10.875	32.5	0.235	**32.6**	0.262	32.6	**0.055**
woman	4	26.5	0.002	27.9	0.604	27.8	0.227	28.2	2.939	**28.9**	11.133	28.6	0.229	28.8	0.290	28.8	**0.053**
average	4	28.42	0.002	29.69	0.916	29.69	0.348	30.09	4.031	**30.48**	10.284	30.29	0.346	30.46	0.382	30.45	**0.075**

speed-up with respect A+ ranges from $\times 4.6$ to 9.3, and it increases for other methods, e.g. DL 9-1-5 and 9-5-5. We are competitive in terms of PSNR when compared with *DL 9-5-5,* and we are substantially faster. We are consistently the best-performers both in SSIM and IFC for all datasets and magnification factors, confirming the good performance of our method.

Secondly, the algorithmic speed up of our AISpH (i.e. comparison between $s = 1$ and $s = 7$) ranges from $\times 4.8$ to 11 depending on the upscaling factors. The drop in quality is very reduced and ranges from 0.01 to 0.07dB. With $s = 7$ we clearly outperform all the state-of-the-art methods in running time (with the exception of bicubic) while being highly competitive in quality (PSNR, IFC, SSIM). In Figure 8 we show close-ups for visual inspection. This subjective evaluation is in consonance with the objective results, as images present less ringing artifacts and sharper edges.

VI. Conclusions

In this paper we extend our previous work on SR based on densely trained regressors and spherical hashing search [14]. In our initial work, we achieve high quality performance and vast speed-ups compared to the original work of [13] and other contemporary methods. In this paper, we further analyze the features used and the metrics involved during the regression process. The contributions of the paper are threefold: (1) We detect and study the importance of antipodal invariance in our search space, proposing the use of the absolute value of the cosine similarity for exhaustive search whenever time is not a constrain (i.e. during training), (2) we propose a novel transform which boosts the antipodal invariance in the Euclidean space, which we embed in the Spherical Hashing algorithm of Heo et al. [24], thus obtaining an Antipodally Invariant Spherical Hashing, and (3) we present a feature transform that performs better thanks to a better coarse approximation of the upscaled gradients obtained by IBP. The regressors obtained with an antipodally invariant metric show a neat gain in PSNR over those obtained with Euclidean distance and, furthermore, AISH is optimally adapted to the search as the loss in quality compared to an exhaustive search is minimal. Finally, in our experimental results we compare our algorithm with recent state-of-the-art methods, proposing two main configurations of our algorithm which result in (a) exhaustive search or (b) approximate search. Our approximate search configuration widely ranks top in terms of execution time, while showing highly competitive quality results, e.g. improving up to 0.25dB in terms of PSNR when compared to $A+$ and up to $\times 9$ faster.

Acknowledgment This work has been partially supported by the project TEC2013-43935-R (financed by the Spanish Ministerio de Economía y Competitividad and the European Regional Development Fund) ; by the ERC-Starting Grant (Dynamic MinVIP) and the Cluster of Excellence rebirth. The authors gratefully acknowledge the support.

References

[1] R. Tsai and T. Huang, "Multiple frame image restoration and registration," in *Proc. Advances in Computer Vision and Image Processing*, vol. 1, 1984.

[2] M. Irani and S. Peleg, "Improving resolution by image registration," *CVGIP: Graphical Models and Image Processing*, vol. 53, no. 3, 1991.

[3] S. Baker and T. Kanade, "Limits on super-resolution and how to break them," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 9, pp. 1167–1183, 2002.

[4] Z. Lin and H.-Y. Shum, "Fundamental limits of reconstruction-based superresolution algorithms under local translation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 26, no. 1, 2004.

[5] W. Freeman, T. Jones, and E. Pasztor, "Example-based super-resolution," *IEEE Trans. Computer Graphics and Applications*, vol. 22, no. 2, 2002.

[6] D. Glasner, S. Bagon, and M. Irani, "Super-resolution from a single image," in *Proc. IEEE International Conf. on Computer Vision*, 2009.

[7] G. Freedman and R. Fattal, "Image and video upscaling from local self-examples," *ACM Trans. on Graphics*, vol. 30, no. 2, pp. 12:1–12:11, 2011.

[8] J. Yang, Z. Lin, and S. Cohen, "Fast image super-resolution based on in-place example regression," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2013.

[9] J. Yang, J. Wright, H. T.S., and Y. Ma, "Image super-resolution via sparse representation," *IEEE Trans. on Image Processing*, vol. 19, no. 11, pp. 2861–2873, 2010.

[10] R. Zeyde, M. Elad, and M. Protter, "On single image scale-up using sparse-representations," in *Proc. International Conf. on Curves and Surfaces*, 2012.

[11] X. Gao, K. Zhang, D. Tao, and X. Li, "Joint learning for single-image super-resolution via a coupled constraint," *IEEE Trans. on Image Processing*, vol. 21, no. 2, pp. 469–480, 2012.

[12] H. Chang, D.-Y. Yeung, and Y. Xiong, "Super-resolution through neighbor embedding," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2004.

[13] R. Timofte, V. D. Smet, and L. V. Gool, "Anchored neighborhood regression for fast example-based super-resolution," in *Proc. IEEE International Conf. on Computer Vision*, 2013.

[14] E. Pérez-Pellitero, J. Salvador, I. Torres, J. Ruiz-Hidalgo, and B. Rosenhahn, "Fast super-resolution via dense local training and inverse regressor search," *Proc. Asian Conf. on Computer Vision, Lecture Notes in Computer Science*, 2014.

[15] K. I. Kim and Y. Kwon, "Single-image super-resolution using sparse regression and natural image prior," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 32, no. 6, 2010.

[16] M. Tappen, B. Russel, and W. Freeman, "Exploiting the sparse derivative prior for super-resolution and image demosaicing," 2003.

[17] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. on Signal Processing*, vol. 54, no. 11, 2006.

[18] J. Salvador and E. Pérez-Pellitero, "Naive Bayes Super-Resolution Forest," in *Proc. IEEE Int. Conf. on Computer Vision*, 2015, pp. 325–333.

[19] S. Schulter, C. Leistner, and H. Bischof, "Fast and accurate image upscaling with super-resolution forests," in *IEEE Conf, on Computer Vision and Pattern Recognition*, June 2015.

[20] K. Zhang, D. Tao, X. Gao, X. Li, and Z. Xiong, "Learning multiple linear mappings for efficient single image super-resolution," *IEEE Trans. on Image Processing*, vol. 24, no. 3, pp. 846–861, 2015.

[21] C. Dong, C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *Proc. European Conf. on Computer Vision*, 2014.

[22] G. Peyré, "Manifold models for signals and images," *Computer Vision and Image Understanding*, vol. 113, no. 2, 2009.

[23] R. Timofte, V. D. Smet, and L. V. Gool, "A+: Adjusted anchored neighborhood regression for fast super-resolution," in *Proc. Asian Conf. on Computer Vision, Lecture Notes in Computer Science*, 2014.

[24] J.-P. Heo, Y. Lee, J. He, S.-F. Chang, and S.-E. Yoon, "Spherical hashing," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2012.

[25] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *Proc. of the Thirtieth Annual ACM Symposium on Theory of Computing*, ser. STOC '98, 1998.

[26] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," 2008.

[27] J. Wang, S. Kumar, and S.-F. Chang, "Semi-supervised hashing for scalable image retrieval," 2010.

[28] K. He and J. Sun, "Computing nearest-neighbor fields via propagation-assisted kd-trees," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2012.

[29] J. Matousek, *Using the Borsuk-Ulam Theorem: Lectures on Topological Methods in Combinatorics and Geometry*. Springer Publishing Company, Incorporated, 2007.

[30] C. Dong, C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. PP, no. 99, 2015.

[31] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *Image Processing, IEEE Trans. on*, vol. 13, no. 4, pp. 600–612, April 2004.

[32] H. Sheikh, A. Bovik, and G. de Veciana, "An information fidelity criterion for image quality assessment using natural scene statistics," *IEEE Trans. Image Processing*, vol. 14, no. 12, 2005.

[33] C.-Y. Yang, C. Ma, and M.-H. Yang, "Single-image super-resolution: A benchmark," in *Proc. European Conf. on Computer Vision*, 2014.

Eduardo Pérez-Pellitero received the B.Sc. in image and sound engineering and the M.Sc. in telecomunications engineering from the Universitat Politècnica de Catalunya (UPC) in 2010 and 2012, respectively. Since 2012, he has been working towards a Ph.D. within an agreement between Leibniz Universität Hannover and Technicolor, both in Hanover, Germany. His research has been concerned with super-resolution upscaling, machine learning, and fast regression structures.

Jordi Salvador is project leader at Technicolor R&I in Hannover and member of the Technicolor's Fellowship Network since 2014. His main research focus is on machine learning for image super resolution and restoration. Formerly, he obtained the Ph.D. degree in 2011 from the Universitat Politècnica de Catalunya (UPC), where he contributed to projects of the Spanish Science and Technology System (VISION, PROVEC) and also to a European FP6 project (CHIL) as research assistant on multiview 3D reconstruction. He has also served as reviewer in several conferences and journals. His research interests include 3D reconstruction, real-time and parallel algorithms, image and video restoration, inverse problems and machine learning.

Javier Ruiz-Hidalgo received a degree in Telecommunications Engineering at the Universitat Politècnica de Catalunya (UPC), Barcelona, Spain in 1997. From 1998 to 1999, he developed an MSc by Research on the field of Computer Vision by the University of East Anglia (UEA) in Norwich, UK. During 1999 he joined the Image Processing Group at UPC working on image and video indexing in the context of the MPEG-7 standard. In 2006, he received his PhD. in the field of image processing. Since 1999 he has been involved in various European Projects as a researcher from the Image Processing Group at UPC. Since 2001 he is an Associate Professor at the Universitat Politècnica de Catalunya. He is currently lecturing on the area of digital signal and systems and image processing. His current research interests include 3D video coding, 3D analysis and super-resolution.

Bodo Rosenhahn studied Computer Science (minor subject Medicine) at the University of Kiel. He received the Dipl.-Inf. and Dr.-Ing. from the University of Kiel in 1999 and 2003, respectively. From 10/2003 till 10/2005, he worked as PostDoc at the University of Auckland (New Zealand), funded with a scholarship from the German Research Foundation (DFG). In 11/2005-08/2008 he worked as senior researcher at the Max-Planck Institute for Computer Science in Saarbruecken. Since 09/2008 he is Full Professor at the Leibniz-University of Hannover, heading a group on automated image interpretation.