Super Resolution for Automated Target Recognition

Maksim Levental

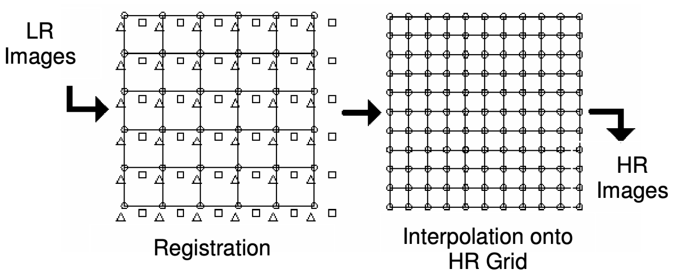


Fig. 1: LR image registration on an HR grid[2]

***Abstract*—Super resolution is the process of producing high-resolution images from low-resolution images while preserving ground truth about the subject matter of the images and potentially inferring more such truth. Algorithms that successfully carry out such a process are broadly useful in all circumstances where high-resolution imagery is either difficult or impossible to obtain. In particular we look towards super resolving images collected using longwave infrared cameras since high resolution sensors for such cameras do not currently exist. We present an exposition of motivations and concepts of super resolution in general and current techniques, with a qualitative comparison of such techniques. Finally we suggest directions for future research.**

1 Introduction

2 Background

3 Classical Algorithms

3.1 Registration

3.2 Interpolation

Suppose that H_k is linear spatial[1] and time invariant. Suppose further that A_k is affine. Then $H := H_k$ commutes with A_k[1] and eqn. 1 becomes

$$\begin{aligned} X_k &= (D \circ A_k \circ H)(Y) + \varepsilon \\ &= (D \circ A_k)(H(Y)) + \varepsilon \\ &= (D \circ A_k)(V) + \varepsilon \end{aligned} \tag{1}$$

where $V := H(Y)$. This naturally suggests interpolation in order to recover V (since X_k, in this framing, is simply shifted samples of V). Note in this context we use interpolation very broadly, i.e. to connote filling in missing values using neighboring (in some sense — not necessarily geometrically) values. This class of techniques proceed by first registering images on a high resolution grid (see figure 1) then interpolating at the "missing" pixels in the HR grid to recover V, and finally denoising and deconvolution (of H) to recover Y. Since in general consecutive X_k have non-uniform shifts (relative to X_1) the interpolation is non-uniform and improvisations on this theme use various weighting schemes for adjacent LR pixels[2].

[1]In analogy with Linear Time Invariant (i.e. linear and constant in space).

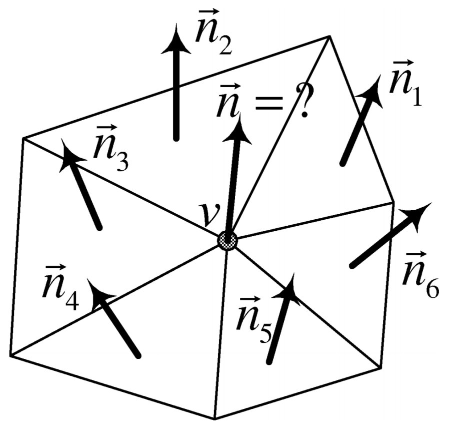


Fig. 2: Delaunay triangulation for fitting splines at LR pixels[4]. v is an LR pixel. Note that v is at z equal to the pixel value

For example Alam et al.[3] uses weighted nearest neighbors: for every pixel to be interpolated the three nearest pixels are weighted inversely by their distance (according to HR grid distance) and then their weighted sum is assigned to that pixel. This non-uniform interpolation is then followed by application of a Wiener filter whose design is informed by the OTF of the particular imaging system they study (which they do not estimate i.e. they assume they can model accurately). Lertrattanapanich et al.[4] base their algorithm on interpolants which require knowledge of gradients (e.g. splines) and mediate the non-uniform sampling by using a weighted average (by area) of those gradients in adjacent Delaunay cells; to be precise they produce a Delaunay triangulation of all LR pixels and compute the gradients (see figure 2) according to

$$\vec{n} = \sum_{i=1}^{m} \frac{B_j \vec{n_j}}{B} \text{ where } B = \sum_{i=1}^{m} B_i$$
$$\frac{\partial z}{\partial x} = -\frac{n_x}{n_z} \text{ and } \frac{\partial z}{\partial y} = -\frac{n_y}{n_z}$$

where B_i is the area of the ith Delaunay cell. Unfortunately this intricate solution is not robust to noise in real images.

A more sophisticated method for non-uniform interpolation uses parametric models for the auto-correlation between LR pixels and the cross-correlation between LR pixels and interpolated pixels to estimate wiener filter weights[5]. These weights are then used to average nearby pixel values. The algorithm operates on a sliding *estimation window* whose

[2]An LR pixel is one sampled from an LR image and embedded in an HR grid. An HR pixel is a pixel in an HR grid.

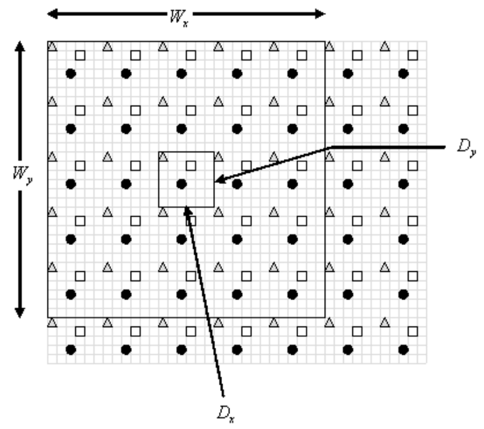


Fig. 3: Wiener filter super resolution estimation window of dimension $D_x \times D_y$ and observation window of dimension $W_x \times W_y$[5]

dimensions D_x, D_y are chosen such that the effective sampling rate exceeds the Nyquist rate for a given ρ_c. The pixel values for the estimation window are a function of the wiener filter weights of nearby LR pixels within an *observation window* whose dimensions W_x, W_y are an integer multiple of D_x, D_y (see figure 3). The weights $\boldsymbol{w}$ are defined as the solution to the minimum mean squared error (MMSE) filter problem, i.e. the finite impulse response (FIR) wiener filter:

$$\boldsymbol{w} = R^{-1}\boldsymbol{p} \tag{2}$$

where R is the auto-correlation of the LR pixels in the observation window and $\boldsymbol{p}$ is the cross-correlation between the pixels to be estimated and the LR pixels. Then R and $\boldsymbol{p}$ are both constructed by sampling a parametric model that weights pixels in the observation window according to distance: R is constructed by sampling from

$$C_1(r) \coloneqq \sigma_d^2 \rho^r * G(r) \tag{3}$$

and $\boldsymbol{p}$ is constructed by sampling from

$$C_2(r) \coloneqq \sigma_d^2 \rho^r * G(r) * G(-r) \tag{4}$$

In the case of R, r is distance on the HR grid, σ_d is related to the empirical variance of all LR pixels in a given observation window and $G(r)$ is a smoothing kernel (e.g. Gaussian). Thus by evaluating C_1 for all $r = r(n_1, n_2)$ distances between LR pixels n_1, n_2 we can construct R. Similarly for $\boldsymbol{p}$, $r = r(m, n)$ is the distance between pixel-to-be-estimated m and LR pixel n. Note that R is an $N \times N$ matrix where $N = KW_xW_y/D_xD_y$, i.e. how many LR pixels there are in the observation window, and $\boldsymbol{p}$ is an $N \times 1$ column vector uniquely computed for each pixel in the estimation window. The scheme is effective but suffers from issues with the spatial isotropy of the auto-correlation and cross-correlation models.

One of the most sophisticated of these non-uniform interpolation schemes employs the kernel regression framework and *steering kernels* (see 5). In this context we start with all X_k registered to a common HR grid and consider pixel values $Y(\boldsymbol{x}_i)$ at pixel coordinates $\boldsymbol{x}_i \coloneqq (x_{i1}, x_{i2})$ as the measured

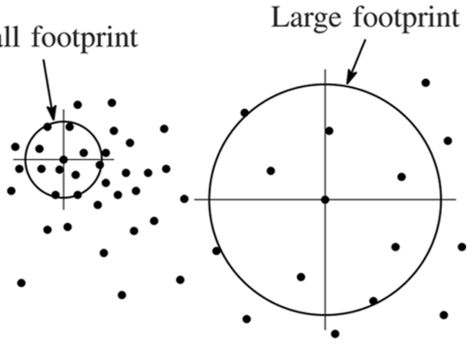


Fig. 4: Kernel footprint as a function of sample density[7]

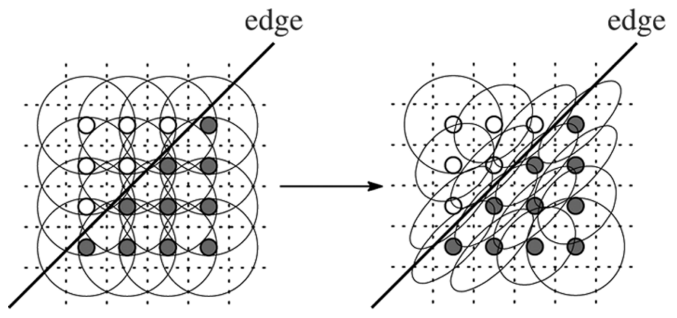


Fig. 5: Adapting kernel shape as a function of local directed structure[7]

data pairs $(\boldsymbol{x}_i, Y(\boldsymbol{x}_i))$. Recall that kernel regression frames the estimation problem as

$$Y(\boldsymbol{x}_i) = Z(\boldsymbol{x}_i) + \varepsilon \tag{5}$$

where Z is the to-be-estimated *regression function* that "predicts" Y as a function of $\boldsymbol{x}$. Then the Nadaraya–Watson estimator (NWE)[6] $\hat{Z}$ for Z is

$$\hat{Z}(\boldsymbol{x}) = \frac{\sum_{i=1}^{P} K(\boldsymbol{x} - \boldsymbol{x}_i) Y(\boldsymbol{x}_i)}{\sum_{i=1}^{P} K(\boldsymbol{x} - \boldsymbol{x}_i)} \tag{6}$$

where P indexes over all pixels in the HR grid and K is a *kernel function* whose purpose is to decay the contribution of $\boldsymbol{x}_i$ if it's in some sense far from $\boldsymbol{x}$. Note that $\hat{Z}(\boldsymbol{x})$ can also be seen as a weighted filtering of Y. In conventional kernel regression K might be any non-negative, symmetric, unimodal[8] function with augmented with an additional h parameter that controls the "bandwidth" or "footprint" of the kernel, i.e.

$$K_h(\boldsymbol{x} - \boldsymbol{x}_i) \coloneqq \frac{1}{h} K\left(h^{-1}(\boldsymbol{x} - \boldsymbol{x}_i)\right) \tag{7}$$

This bandwidth parameter h can be generalized to a *smoothing kernel* H in order to make $K = K_H$ adaptive to the local structure of the pixels, e.g. to have larger footprints in sparsely sampled regions and have smaller footprints in densely sampled regions (see figure 4). Ultimately though it is desirable to have kernels that can adapt to directed structure in the image, i.e. "steerable" kernels that filter strongly along an edge and weakly across an edge. This is accomplished by, for example, using a Gaussian as the kernel:

$$K_{H_i}(\boldsymbol{x}-\boldsymbol{x}_i) \propto \frac{\exp\left\{-(\boldsymbol{x}-\boldsymbol{x}_i)^T H_i^{-1}(\boldsymbol{x}-\boldsymbol{x}_i)\right\}}{\sqrt{\det H_i}} \tag{8}$$

and identifying H_i with $\nabla^2 Z(\boldsymbol{x}_i)$ (since gradients capture edge structure). An estimate $\hat{H}_i$ of $\nabla^2 Z(\boldsymbol{x}_i)$ can be obtained by looking at covariances of empirical gradients (i.e. the HR grid registered image convolved with a difference filter). Unfortunately this is a naive estimate that is often rank deficient or unstable (both leading to instances where $\hat{H}_i$ isn't invertible). One solution is to parameterize H_i:

$$H_i = \gamma_i U_{\theta_i} \Lambda_{\sigma_i} U_{\theta_i}^T$$

where U_{θ_i} is a rotation matrix, $\Lambda_{\sigma_i} = \operatorname{diag}\left(\sigma_i, \sigma_i^{-1}\right)$ is an "elongation" matrix, and γ_i is a scaling parameter, with each of $\gamma_i, \theta_i, \sigma_i$ estimated from the data in a more robust way. An alternative kernel is the bilateral kernel[9] that defines closeness according to geometric and radiometric distance:

$$\begin{aligned} K_S(\boldsymbol{x}-\boldsymbol{x}_i) &:= \exp\left\{-\frac{\|\boldsymbol{x}-\boldsymbol{x}_i\|^2}{2\sigma_S^2}\right\} \\ K_R(\boldsymbol{x},\boldsymbol{x}_i) &:= \exp\left\{-\frac{\|Y(\boldsymbol{x})-Y(\boldsymbol{x}_i)\|^2}{2\sigma_R^2}\right\} \\ K_B(\boldsymbol{x},\boldsymbol{x}_i) &:= K_S(\boldsymbol{x}-\boldsymbol{x}_i)K_R(\boldsymbol{x},\boldsymbol{x}_i) \end{aligned} \tag{9}$$

where σ_S parameterizes spatial distance weight and σ_R parameterizes "radiometric" distance weight.

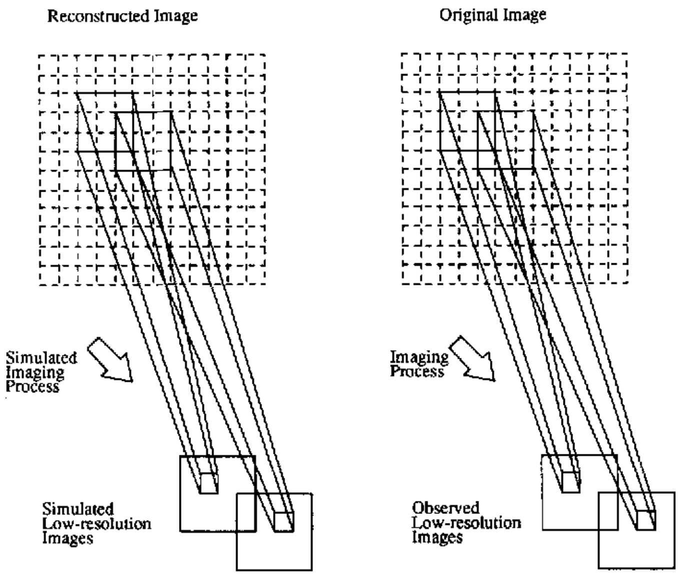
In general non-uniform interpolation techniques are intuitive and typically (relatively) computationally efficient but they assume an unrealistic observation model (namely that of affine flow).

3.3 Estimation

Statistical estimation methods cast SR as an inference problem. One of the earliest successful SR algorithms[10] proposed an iterative scheme inspired by the back-projection method commonly used to reconstruct 2-D objects from 1-D projections in computer-aided tomography. Recall eqn. **??**. Then the idea is to take the current estimate of the HR image $\hat{Y}^i$, see if after motion and down-sampling $(D \circ A_k)(\hat{Y}^i)$ it is near the LR samples X_k, and add a correction when it is not (see figure 6):

$$\hat{Y}^{i+1} = \hat{Y}^i + \sum_{k=1}^{K}(D\circ A_k)^{-1}\left((D\circ A_k)(\hat{Y}^i) - X_k\right) \tag{10}$$

where $\hat{Y}^i$ is the current estimate of the blurred HR image, $(D \circ A_k)(\hat{Y}^i)$ is the projection of the current estimate to low resolution, and $(D\circ A_k)^{-1}\left((D\circ A_k)(\hat{Y}^i) - X_k\right)$ is the *back-projection*. This process iterates until convergence i.e. $|(D \circ A_k)(\hat{Y}^i) - X_k| < \delta$ for some δ. Irani *et al.* [10] also convolve the back-projection with a smoothing kernel as a form of regularization since the estimation problem is in general ill-posed (there are many $\hat{Y}^i$ that will project down to a pixel-distance neighbor of X_k). It can be shown[11] that for ε_k distributed $(0, R_k)$-Normal, $\hat{Y}$ is none other than the



Reconstructed Image
Original Image
Simulated Imaging Process
Imaging Process
Simulated Low-resolution Images
Observed Low-resolution Images

Fig. 6: Iterative back projection[10]

maximum likelihood estimate (MLE) for Y. We can see this by recognizing that eqn. 10 is just the Richardson iterative[12] solution to

$$L(Y) = \frac{1}{2}\left\|\boldsymbol{X} - \begin{bmatrix} D_1\circ A_1 \\ D_2 \circ A_2 \\ \vdots \\ D_K \circ A_K \end{bmatrix} Y\right\|^2 \tag{11}$$

since

$$\nabla_Y L = 0 \iff \sum_{k=1}^{K}(D\circ A_k)^{-1}\left((D\circ A_k)(Y) - X_k\right) = 0$$

and therefore $\hat{Y}_i \to \hat{Y}$ is the MLE (since MLE is the solution to least squares[13]). Though this is one of the oldest SR algorithms it's recently been revisited and re-imagined as a deep neural network architecture (DNN)[14].

Another estimation technique employs a Kalman filter[15] to estimate Y. Let $\boldsymbol{x}_k = \operatorname{vec}(X_k)$ be the vectorization[3] of X_k and $\boldsymbol{y} = \operatorname{vec}(Y)$ likewise. If we assume linear models for each of D, A_k, H_k (i.e. all representable as matrices) and a well-behaved optical flow model (most pixels in image $\boldsymbol{x}_k$ appear in image $\boldsymbol{x}_{k-1}$) then we can image $\boldsymbol{y}$ as a sequence of images $\boldsymbol{y}_k$ related in time by

$$\boldsymbol{y}_k = A'_k \boldsymbol{y}_{k-1} + \eta_k \tag{12}$$

where A'_k is the *relative* motion operator, $A'_k\boldsymbol{y}_{k-1}$ is conventional matrix-vector multiplication, and η_k is the only source of

[3]
$$\operatorname{vec}\left(\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}\right) = [a_{11}, \ldots, a_{m1}, a_{12}, \ldots, a_{m2}, \ldots, a_{1n} \cdots a_{mn}]$$

new pixels (noise[4] distributed $(0, Q_k)$-Normal). Consequently eqn. **??** becomes

$$\boldsymbol{x}_k = DH_k \boldsymbol{y}_k + \varepsilon_k \tag{13}$$

and the pair of eqns. 12, 13 can be seen to constitute a linear dynamical system with $\boldsymbol{y}_k$ the state of the system, A'_k the state transition, η_k the state noise, $\boldsymbol{x}_k$ the measurement, DH_k the measurement model, and ε_k the measurement noise. Note that the HR image conditioned on all previous LR images (measurements)

$$\boldsymbol{y}_{k|s} \coloneqq \boldsymbol{y}_k | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_s \tag{14}$$

with $s \leq k$, is a Gaussian process (GP), with mean $\bar{\boldsymbol{y}}_{k|s} = E\left[\boldsymbol{y}_{k|s}\right]$ and covariance

$$C_{k|s} \coloneqq E\left[(\boldsymbol{y}_k - \bar{\boldsymbol{y}}_{k|s})(\boldsymbol{y}_k - \bar{\boldsymbol{y}}_{k|s})^T\right] \tag{15}$$

By definition the MMSE $\hat{\boldsymbol{y}}_{k|s}$ of $\boldsymbol{y}_{k|s}$ is $\bar{\boldsymbol{y}}_{k|s}$ and therefore $C_{k|s}$ is the covariance of the error of the estimate $\hat{\boldsymbol{y}}_{k|s}$. The Kalman filter proceeds in two steps: an *a priori* (before measurement) prediction step and an *a posteriori* (after measurement) update step. The prediction step iterates on the estimate (and its covariance) given all previous measurements:

$$\hat{\boldsymbol{y}}_{k|k-1} = A'_k \hat{\boldsymbol{y}}_{k-1|k-1} \tag{16}$$
$$C_{k|k-1} = A'_k C_{k-1|k-1} (A'_k)^T + Q_k \tag{17}$$

This can be seen as a one-step propagation of the estimate "in the direction" of the previous measurement. Then the update step incorporates new information from a measurement:

$$\hat{\boldsymbol{y}}_{k|k} = \hat{\boldsymbol{y}}_{k|k-1} + K_k(\boldsymbol{x}_k - DH_k \hat{\boldsymbol{y}}_{k|k-1}) \tag{18}$$
$$C_{k|k} = (I - K_k DH_k) C_{k|k-1} \tag{19}$$

where the Kalman gain

$$K_k \coloneqq \frac{C_{k|k-1}(DH_k)^T}{DH_k C_{k|k-1}(DH_k)^T + R_k} \tag{20}$$

weights the contribution of the prediction and the measurement[5]. Note that in the Kalman framework D, H_k, A'_k, R_k, Q_k are all assumed to be known. In Elad *et al.* [15] the assumption is D, H_k, R_k are known functions of camera parameters, A'_k can be estimated by an image registration algorithm and Q_k can be approximated:

$$Q_k \approx \alpha_k A'_k C_{k|k} (A'_k)^T \tag{21}$$

where α_k is chosen such that the approximation upper-bounds the true Q_k. They comment that this stronger auto-correlation for $\boldsymbol{y}_k$ adds "pseudo-noise" to the system and biases the Kalman filter to "rely" more on the measurements than the state transition model.

SR can also be posed as a Bayesian maximum a posterior (MAP) estimation problem. Let $\boldsymbol{H} \coloneqq (H_1, \ldots, H_k)$ be the vector of blur operators applied to Y in order to produce $\boldsymbol{X}$ and similarly $\boldsymbol{A}$. Then

$$\begin{aligned} \hat{Y} &= \underset{Y}{\operatorname{argmax}}\ P\left(Y|\boldsymbol{X}\right) \\ &= \underset{Y}{\operatorname{argmax}} \int_{D,\boldsymbol{H},\boldsymbol{A}} P\left(Y, D, \boldsymbol{H}, \boldsymbol{A}|\boldsymbol{X}\right) \\ &= \underset{Y}{\operatorname{argmax}} \int_{D,\boldsymbol{H},\boldsymbol{A}} \frac{P\left(\boldsymbol{X}|Y, D, \boldsymbol{H}, \boldsymbol{A}\right) P(Y) P(D, \boldsymbol{H}, \boldsymbol{A})}{P(\boldsymbol{X})} \end{aligned} \tag{22}$$

$$= \underset{Y}{\operatorname{argmax}} \int_{D,\boldsymbol{H},\boldsymbol{A}} P\left(\boldsymbol{X}|Y, D, \boldsymbol{H}, \boldsymbol{A}\right) P(Y) P(D, \boldsymbol{H}, \boldsymbol{A}) \tag{23}$$

where in eqn. 22 we've used the independence of Y and $D, \boldsymbol{H}, \boldsymbol{A}$[16] and in eqn. 23 we've used that $\boldsymbol{X}$ is a constant with respect to the maximization. While there exist reasonable priors for Y, marginalizing over $D, \boldsymbol{H}, \boldsymbol{A}$ is still difficult due to the high-dimensionality of each. Therefore assuming $D, \boldsymbol{H}, \boldsymbol{A}$ can be estimated independently as $\hat{D}, \hat{\boldsymbol{H}}, \hat{\boldsymbol{A}}$, eqn. 23 becomes

$$\hat{Y} = \underset{Y}{\operatorname{argmax}}\ P\left(\boldsymbol{X} \middle| Y; \hat{D}, \hat{\boldsymbol{H}}, \hat{\boldsymbol{A}}\right) P(Y) \tag{24}$$

where the semicolon indicates $\hat{D}, \hat{\boldsymbol{H}}, \hat{\boldsymbol{A}}$ are known parameters of the conditional distribution. This casts $\hat{Y}$ the standard MAP estimate of Y. Note that an equivalent formulation of MAP maximizes the log-likelihood instead of maximizing the likelihood:

$$\begin{aligned} \hat{Y} &= \underset{Y}{\operatorname{argmax}} \left[\log\left(P\left(\boldsymbol{X} \middle| Y; \hat{D}, \hat{\boldsymbol{H}}, \hat{\boldsymbol{A}}\right) P(Y)\right)\right] \\ &= \underset{Y}{\operatorname{argmax}} \left[\log P\left(\boldsymbol{X} \middle| Y; \hat{D}, \hat{\boldsymbol{H}}, \hat{\boldsymbol{A}}\right) + \log P(Y)\right] \end{aligned} \tag{25}$$

Various choices for $P\left(\boldsymbol{X} \middle| Y; \hat{D}, \hat{\boldsymbol{H}}, \hat{\boldsymbol{A}}\right)$ and the prior $P(Y)$ (and consequent choice of optimization strategies) characterize this class of SR techniques. Again assume D, H_k, A_k linear and given ε_k in eqn. **??** distributed $(0, rI)$-Normal

$$P\left(\boldsymbol{X} \middle| Y; \hat{D}, \hat{\boldsymbol{H}}, \hat{\boldsymbol{A}}\right) \propto \exp\left\{-\frac{\left\|\boldsymbol{X} - \hat{D}\hat{\boldsymbol{H}}\hat{\boldsymbol{A}}\boldsymbol{y}\right\|^2}{2r^2}\right\} \tag{26}$$

where here $\boldsymbol{X} = (\text{vec}(X_1), \ldots, \text{vec}(X_k))$ and

$$\hat{D}\hat{\boldsymbol{H}}\hat{\boldsymbol{A}} \coloneqq \begin{bmatrix} \hat{D}\hat{\boldsymbol{H}}_1\hat{\boldsymbol{A}}_1 \\ \hat{D}\hat{\boldsymbol{H}}_2\hat{\boldsymbol{A}}_2 \\ \vdots \\ \hat{D}\hat{\boldsymbol{H}}_K\hat{\boldsymbol{A}}_K \end{bmatrix} \tag{27}$$

After a suitable choice for the prior it can be seen that eqn. 25 is just regularized regression. For example when choosing a Gibbs[16] distribution as the prior, i.e.

$$P(Y) \propto e^{-\alpha B(Y)} \tag{28}$$

where $B(Y)$ is called the *potential*, eqn. 25 becomes

$$\hat{\boldsymbol{y}} = \underset{\boldsymbol{y}}{\operatorname{argmin}} \left[\left\|\boldsymbol{X} - \hat{D}\hat{\boldsymbol{H}}\hat{\boldsymbol{A}}\boldsymbol{y}\right\|^2 + \lambda B(Y)\right] \tag{29}$$

[4]Noise here doesn't necessarily mean unwanted high frequency variation but simply a source of randomness. This is in close affinity with how generative machines such as variational auto-encoders and generative adversarial networks are understood.

[5]This is better understood in the more general linear dynamic systems case where $\boldsymbol{y}_k = A_k\boldsymbol{y}_k + B_k\boldsymbol{u}_k + \varepsilon_k$ and $B_k\boldsymbol{u}_k$ is known "controlled" input. Then the prediction includes a $B_k\boldsymbol{u}_{k-1}$ term and the Kalman gain effectively mediates between controlled and uncontrolled inputs.

where the aggregate regularization parameter λ absorbs α from $P(Y)$ and r from ε_k. There are many other choices for the prior in eqn. 25, whose effect is to bias the estimator $\hat{\boldsymbol{y}}$ towards "natural" images. Alternatively we can (and will) take eqn. 29 as our starting point and explicitly choose the regularizer $B(Y)$.

One of the simplest priors is a $(0, Q)$-Normal, where Q is symmetric positive definite[6] (PD) and captures the covariance. This corresponds to the regularizer taking the form

$$B(Y) = \boldsymbol{y}^T Q \boldsymbol{y} \tag{30}$$

where $\boldsymbol{y} = \text{vec}(Y)$. Since Q is symmetric PD eqn. 29 becomes

$$\hat{\boldsymbol{y}} = \underset{\boldsymbol{y}}{\text{argmin}} \left[\left\| \boldsymbol{X} - \hat{D}\hat{\boldsymbol{H}}\hat{\boldsymbol{A}}\boldsymbol{y} \right\|^2 + \lambda \left\| \sqrt{Q}\boldsymbol{y} \right\|^2 \right] \tag{31}$$

where $\sqrt{Q}$ is U of the Cholesky decomposition $Q = U^T U$. Equation 31 is Tikhonov regularized regression, or Ridge regression if $\sqrt{Q} = I$. Letting $G = \hat{D}\hat{\boldsymbol{H}}\hat{\boldsymbol{A}}$ the closed-form solution to eqn. 31 is

$$\hat{\boldsymbol{y}} = \left(G^T G + \lambda Q \right)^{-1} G^T \boldsymbol{X} \tag{32}$$

Nguyen *et al.* [17] use cross-validation to determine the regularization parameter λ (by partitioning the pixels into a "fit" and "validate" set). In general, rather than explicitly computing inverses in eqn. 32, in practice $\hat{\boldsymbol{y}}$ is found by solving the regression problem via optimization (for high-dimensional matrices optimization is faster than inversion). Nguyen *et al.* use conjugate gradient descent[7] to optimize eqn. 31. They argue that $\hat{D}\hat{\boldsymbol{H}}\hat{\boldsymbol{A}}$ is ill-conditioned[8] and to that end, since the convergence rate of conjugate gradients is dependent on the condition number[18], propose pre-conditioners[9] to improve the converge rate.

An issue with the multivariate Normal prior is that it strongly enforces global smoothness, penalizing sharp edges. One solution is to use Huber loss[19]

$$L_\delta(a) := \begin{cases} a^2 & \text{for } |a| \leq \delta \\ 2\delta|a| - \delta^2 & \text{otherwise} \end{cases} \tag{33}$$

to explicitly parameterize the penalty for gradients (i.e. high-frequency features). Huber loss enforces local smoothness (since it's quadratic for $|a| \leq \delta$) but permits edges (since it's linear for $|a| > \delta$). Capel *et al.* [20] implement this by composing $L_\delta(a)$ with a first-order gradient operator[10] as the potential in eqn. 28. Another gradient penalty that encourages sparse gradients (i.e. local smoothness and steep edges) is Total Variation (TV) norm[21]:

$$\|u\|_{\text{TV}} := \int_\Omega \|\nabla u\| \, d\Omega \tag{34}$$

where u is a smooth image of bounded variation (i.e. such that the integral converges) over domain Ω. In the context of SR this amounts to setting

$$B(Y) = TV(Y) := \|\nabla Y\|_1 \tag{35}$$

Farsiu *et al.* [22] introduce a *bilateral* TV norm

$$BTV(Y) := \sum_{k=0}^{N} \sum_{l=0}^{N} \alpha^{k+l} \|Y - S_x^k S_y^l Y\|_1 \tag{36}$$

where S_x^k, S_y^l are shift operators (i.e. S_x^k shifts Y by k pixels in the horizontal) and N is arbitrary. The bilateral TV norm factors in gradients at several scales ($Y - S_x^2 Y$ is an approximation of the horizontal gradient at twice the scale of $Y - S_x^1 Y$) and decays their contribution according to their spatial distance (α^{k+l} decays as a function of shift distance). It can also be shown to be equivalent[23] to filtering $\boldsymbol{X}$ by the bilateral kernel (see eqn. 9).

Though there are many priors (or regularizers) that have been studied none really capture natural images in all of their variation. For that we need to actually learn from real data i.e. examples images.

3.4 Example based

Example based techniques *learn* a mapping from LR patches to HR patches based on a training set and then use that mapping to predict details in new (unseen) images. Freeman *et al.* [24] argues that the most important features to reconstruct are the high-frequency features. Therefore they store only the correspondence between high-pass filtered, contrast-normalized versions of example LR patches and HR patches. Note that they pre-process the LR images by cubic-spline interpolating to the higher pixel sampling resolution and that HR patches are sampled to overlap by one or more pixel widths at their edges. Naive mosaicing of these matching HR patches produces poor results due to noise and the ill-posed nature of super-resolution. Their solution to this issue is using a hidden Markov random field[11] (HMRF) to model spatial consistency between adjacent HR patches y_i and between LR-HR patch pairs x_i, y_i (see figure 7). They then compute the MAP estimate of the HMRF to obtain the smoothest assignment of HR patches: the HMRF model postulates that the conditional probability of any HR patch assignment $\boldsymbol{y}$ given observed LR patches $\boldsymbol{x}$ is

$$P(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{P(\boldsymbol{x})} \prod_{i \bullet\!\!-\!\!\bullet j} \psi(y_i, y_j) \prod_i \phi(x_i, y_i^{LR}) \tag{37}$$

[6]A symmetric real matrix Q is positive definite if $\boldsymbol{y}^T Q \boldsymbol{y} > 0$ for all non-zero $\boldsymbol{y}$.

[7]Two vectors $\boldsymbol{u}, \boldsymbol{v}$ are conjugate with respect to G if $\boldsymbol{u}^T G \boldsymbol{v} = 0$. Conjugate gradient descent is gradient descent but with conjugate gradients (it has better convergence properties).

[8]The condition number κ of a function is a measure of how sensitive it is to small perturbations; for a matrix G it is defined $\kappa(G) := \sigma_{\max}(G)/\sigma_{\min}(G)$ where $\sigma_{\max}(G), \sigma_{\min}(G)$ are the maximum and minimum singular values of G respectively.

[9]A pre-conditioner of a matrix G is an approximation of G that has a better condition number. Nguyen *et al.* propose a pre-conditioner with singular values clustered around 1 in order that $\kappa(G) \approx 1$.

[10]Let $u = (1, 2, 1)$ and $v = (1, 0, -1)$ then $h = uv^T$ is the first order Sobel filter and $\nabla Y = \left(h * Y, h^T * Y \right)$.

[11]A Markov random field (MRF) is a collection of random variables $x_i, y_j, \ldots$ with conditional dependence represented by pairings and satisfying the *pairwise Markov* property: any two random variables that aren't paired are conditionally independent of each other given (conditioned on) all other variables. A *hidden* Markov random field is simply a MRF where some of the random variables aren't observed.

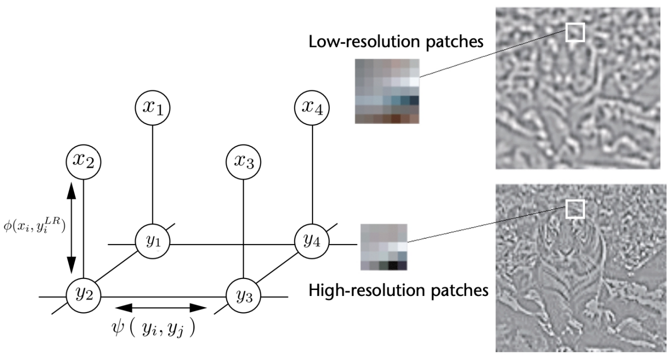


Fig. 7: Hidden Markov random field modeling spatial relationships between LR patches x_i and example LR patches y_i^{LR} and HR patches y_i[24]

where x_i are observed LR patches and y_i are unknown (to-be-inferred) HR patches (along with their learned-mapping example LR patches y_i^{LR}). Note $P(\boldsymbol{x})$ is a normalization constant, $i \bullet\!\!-\!\!\bullet j$ indicates the product is only over adjacent HR patches, $\psi(y_i, y_j)$ encodes the compatibility of adjacent HR patches according to

$$\psi(y_i, y_j) = \exp\left\{-\frac{\|y_i - y_j\|}{2\sigma^2}\right\} \tag{38}$$

where $\|y_i - y_j\|$ is only computed on the overlapping pixels. Similarly $\phi(x_i, y_i^{LR})$ encodes the compatibility between the observed LR patch x_i and the example LR patch y_i^{LR} corresponding to the estimated HR patch y_i. To make the MAP computation tractable they choose only 16 candidate example LR patches y_i^{LR}. They compute the MAP estimate using belief propagation[12].

The main problem with the direct LR-HR patch pair technique is that it requires an enormous database of patches in order for it to generalize. Chang *et al.* remedy this problem by using ideas from a manifold[13] learning[14] technique called locally linear embedding[27] (LLE) that they call nearest neighbor embedding (NNE). First they imagine both the space of LR patches and HR patches as manifolds. Then based on the intuition that a well-sampled smooth manifold is locally linear (i.e. an LR patch x_i on the LR manifold and its neighbors $x_j, x_k, \dots$ lie in a locally linear subspace of the manifold) they characterize the local geometry. One can characterize local geometry by finding convex reconstruction weights W_{ij} of any x_i from only its neighbors by minimizing reconstruction error, i.e.

$$\hat{W} = \underset{W_{ij}}{\operatorname{argmin}} \sum_i \left\| x_i - \sum_{x_j \in N(x_i)} W_{ij} x_j \right\|^2 \quad \text{s.t.} \quad \sum_{x_j \in N(x_i)} W_{ij} = 1 \text{ for all } i \tag{39}$$

[12]An efficient way to compute marginals of joint probabilities that have structure by reusing partial sums (i.e. passing messages)[25].

[13]A collection of points that locally resembles Euclidean space.

[14]Dimensionality reduction.

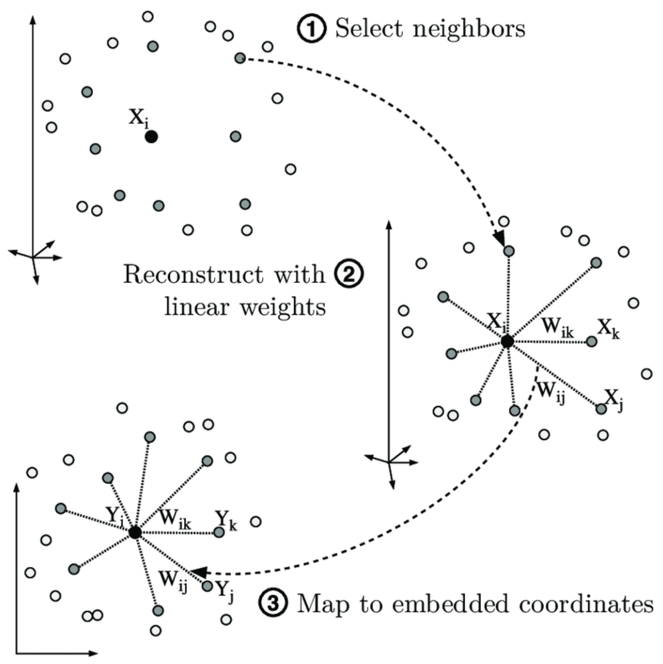


Fig. 8: NNE algorithm from LR patches x_i to HR patches y_i[26]

where $N(x_i)$ is the neighborhood of x_i. These weights W_{ij} are invariant with respect to rotation, rescaling, and translation of the LR patch and its neighborhood[27] and therefore should remain valid in an embedding space (i.e. the HR manifold) coordinate system (see figure 8). The HR patches y_i in the HR manifold are then reconstructed using the same weights, i.e.

$$y_i = \sum_{x_j \in N(x_i)} W_{ij} y_j \tag{40}$$

Note that neighboring patches in HR space are constrained to overlap and in fact the neighborhoods are computed using gradient features of the LR and HR patches rather than the raw patches (they argue that this is what allows for better generalization i.e. much smaller patch databases).

The NNE approach suffers from its own issues; using a fixed number of nearest neighbors results in blurring (due to over-fitting or under-fitting). Yang *et al.* [28] resolve this issue by using sparse coding (SC) as a framework for learning a sparse mapping from LR patches to HR patches (see figure 9). This enables them to essentially leave the number of nearest numbers unspecified (or, alternatively, learn the correct number of neighbors on a patch by patch basis[15]). To be precise, let $D \in \mathbb{R}^{n \times K}$ be an *over-complete dictionary*[16] of K atoms $\boldsymbol{d}_i \in \mathbb{R}^n$, and suppose a "signal" $\boldsymbol{y} \in \mathbb{R}^n$ can be represented as a sparse linear combination of these atoms i.e.

$$\boldsymbol{y} = D\boldsymbol{\alpha} \tag{41}$$

such that $\|\boldsymbol{\alpha}\|_0 \ll K$ (few non-zero entries). The sparse vector $\boldsymbol{\alpha}$ is variously called the code, reconstruction, or sparse

[15]No pun intended.

[16]A collection of vectors (known as *atoms*) that spans a space but is also linearly dependent.

representation of $\boldsymbol{y}$. Yang *et al.* construct two dictionaries D_{LR}, D_{HR} consisting corresponding LR and HR patches $\boldsymbol{x}_i, \boldsymbol{y}_i$. Then given a new input LR patch $\boldsymbol{x}$ they compute the code for a "featurized" representation of $\boldsymbol{x}$ in D_{LR} by minimizing the code subject to reconstruction error tolerance:

$$\hat{\boldsymbol{\alpha}} = \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \|\boldsymbol{\alpha}\|_0 \text{ s.t. } \|F\boldsymbol{x} - FD_{LR}\boldsymbol{\alpha}\|^2 \leq \epsilon \tag{42}$$

where ϵ is the reconstruction error tolerance and F is the feature extraction operator (first and second order gradients), and construct the HR image using the same sparsely represented HR patch (i.e. $\hat{\boldsymbol{y}} = D_{HR}\hat{\boldsymbol{\alpha}}$). As stated the optimization in eqn. 42 is non-convex (due to $\|\boldsymbol{\alpha}\|_0$) and is in fact NP-hard[17][29]. Fortunately the L_1 (i.e. $\|\boldsymbol{\alpha}\|_1$) relaxation is often good enough[30]:

$$\hat{\boldsymbol{\alpha}} = \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \left[\lambda \|\boldsymbol{\alpha}\|_1 + \frac{1}{2} \|F\boldsymbol{x} - FD_{LR}\boldsymbol{\alpha}\|^2\right] \tag{43}$$

where we've also transformed to the unconstrained optimization using the Lagrange multiplier λ. Solving eqn. 43 input-patch by input-patch does not enforce consistency between adjacent patches. They use a raster-scan technique discussed in Freeman *et al.* [24] to enforce consistency. They further comment that since eqn. 42 doesn't demand equality between the LR patch $\boldsymbol{x}$ and its reconstruction $D_{LR}\boldsymbol{\alpha}$ the reconstructed image Y might not satisfy the imaging model (see eqn. **??**). They resolve this by using iterative back projection (see eqn. 10) as a final step in the algorithm. While the sparse representation approach mitigates issues having to do with local bias (fixed number K patch neighbors) it still requires a large database of raw image patches. The same group's follow-on paper[31] addresses this issue directly by learning the LR, HR dictionaries with a fixed number of atoms K. In general dictionary learning minimizes the reconstruction error of the codes across all J example inputs $\boldsymbol{Z} := (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_J)$:

$$\hat{D} = \underset{D,\boldsymbol{A}}{\operatorname{argmin}} \|\boldsymbol{Z} - D\boldsymbol{A}\|^2 + \lambda \|\boldsymbol{A}\|_1 \tag{44}$$

where $\boldsymbol{A} := (\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_J)$. Yang *et al.* "jointly" learn D_{LR}, D_{HR} by making the identifications

$$\boldsymbol{Z} = \begin{bmatrix} \frac{1}{\sqrt{N}}\boldsymbol{Y} \\ \frac{1}{\sqrt{M}}\boldsymbol{X} \end{bmatrix} \text{ and } D = \begin{bmatrix} \frac{1}{\sqrt{N}}D_{HR} \\ \frac{1}{\sqrt{M}}D_{LR} \end{bmatrix}$$

where $\boldsymbol{Y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_P)$ are the example HR patches, $\boldsymbol{X}$ similarly the example LR patches, and N, M are the number of HR and LR pixels respectively in each example patch pair $\boldsymbol{y}_i, \boldsymbol{x}_i$ (in order to normalize the error between $\boldsymbol{y}_i, \boldsymbol{x}_i$).

Though SC achieves good results for reasonably sized example sets and dictionary sizes it suffers from the inherent computational complexity of solving the optimization problem (see figure 10). Timofte *et al.* [32] argues that the chief inefficiency is the L_1 norm on the code. By relaxing the L_1 norm in eqn. 43 to the L_2 norm they recover the Ridge regression form of the problem (see eqn. 32) and get a closed form solution for the code:

$$\hat{\boldsymbol{\alpha}} = \left(D_{LR}^T D_{LR} + \lambda I\right)^{-1} D_{LR}^T \boldsymbol{x} \tag{45}$$

The code can then be used to compute the HR patch:

$$\begin{aligned} \boldsymbol{y} &= D_{HR}\hat{\boldsymbol{\alpha}} && (46)\\ &= D_{HR}\left(D_{LR}^T D_{LR} + \lambda I\right)^{-1} D_{LR}^T \boldsymbol{x} && (47) \end{aligned}$$

Factoring out

$$P = D_{HR}\left(D_{LR}^T D_{LR} + \lambda I\right)^{-1} D_{LR}^T \tag{48}$$

we see that P can be pre-computed, thereby turning inference into simply a matrix multiplication. They then go on to argue,

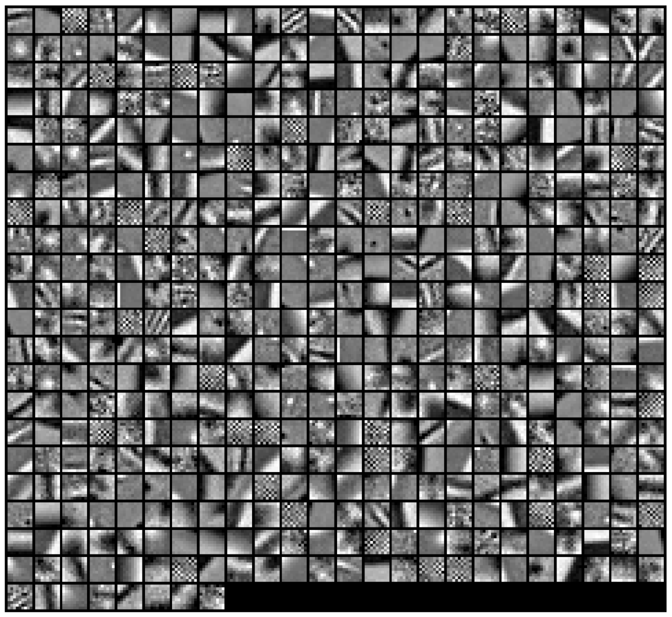


Fig. 9: 512 entry 9×9 atom HR image patch dictionary trained using 100,000 HR and LR image patch pairs[31]

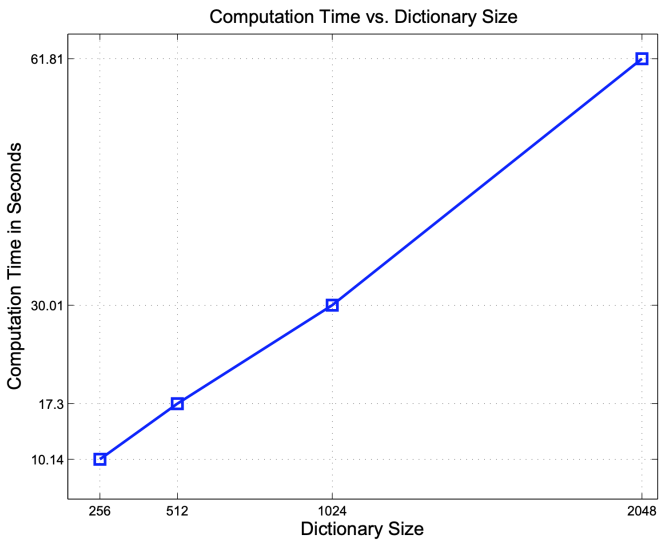


Fig. 10: Dictionary inference time as a function of dictionary size[31]

[17]Problems in NP are those which are unknown to have a deterministic polynomial time solutions. NP-hard admit a polynomial time reduction from any problem in NP (colloquially they are as hard as any problem in NP).

reasoning by analogy with NNE, that the projection should be locally sensitive to input patches, i.e. instead of using the entire dictionary to construct the projection only the atoms nearest to the input LR patch should be used. Thus for every atom in the dictionary, they find its $K' < K$ nearest neighbors, and using only those K' atoms they pre-compute K distinct projection operators $P_k \in \mathbb{R}^{n\times K'}$:

$$P_k = D_{HR}^k \left((D_{LR}^k)^T D_{LR}^k + \lambda I\right)^{-1} (D_{LR}^k)^T \tag{49}$$

where D_{LR}^k, D_{HR}^k are the neighborhood of the LR atom and the neighborhood of its corresponding HR atom. They call these P_k *anchored regressions* because they effectively weight the contributions of LR and HR dictionary atoms. For a new input LR patch x they find the nearest atom in the D_{LR} and recover the HR patch $\boldsymbol{y} = P_k\boldsymbol{x}$. Note that ANR, just as in SC, does not operate on the raw image patches themselves but rather "featurized" image patches: they apply Principle Components Analysis (PCA) to the first and seconder gradients of the patches and, as in Zeyde *et al.* [33], and keep only keep only components explaining 99.9% of the variance. This reduces the dimensions of their atoms to about 30.

Inspired by the anchored regressions of ANR Schulter *et al.* [34] propose a technique called Super-Resolution Forests: their point of departure is a "data-dependent" function $W(\cdot)$ such that

$$\hat{\boldsymbol{y}} = W(\boldsymbol{x}) \cdot \boldsymbol{x} \tag{50}$$

This is similar to eqn. 49. They model $W(\cdot)$ as an n-tree regression tree forest. Putting aside for the moment how the trees are trained, each tree T_t independently splits LR patch space into disjoint subspaces by routing patches down the tree (see figure 11). For each tree t each subspace associated with a leaf l is then used to fit a linear basis-regression model defined by

$$_t^lW(\boldsymbol{x}) = \sum_{i=1}^{I} {}_t^l\hat{w}_i \phi_i(\boldsymbol{x}) \tag{51}$$

$$_t^l\hat{\boldsymbol{w}} = \underset{_t^l\boldsymbol{w}}{\text{argmin}} \sum_{j=1}^{J} \left\| \boldsymbol{y}_j - \sum_{i=1}^{I} {}_t^l w_i \phi_i(\boldsymbol{x}_j) \right\|^2 \tag{52}$$

where ${}_t^l\boldsymbol{w} := \left({}_t^l w_1, \dots, {}_t^l w_I\right)$ and $\phi_i(\boldsymbol{x}_j)$ are basis functions such as the radial basis function

$$\phi_i(\boldsymbol{x}_j) = \exp\left\{\frac{\|\boldsymbol{x}_j - \boldsymbol{\mu}_i\|^2}{\sigma_i}\right\} \tag{53}$$

Since the objective in eqn. 52 is linear in ${}_t^l w_i$ the leaf models are fit using the closed form solution to least squares:

$$_t^l\hat{\boldsymbol{w}} = \left(\Phi(\boldsymbol{X}_t^l)^T\Phi(\boldsymbol{X}_t^l) + \lambda I\right)^{-1} \Phi(\boldsymbol{X}_t^l)^T \boldsymbol{Y}_t^l \tag{54}$$

where $\boldsymbol{X}_t^l$ are only those LR patches that get routed to leaf l in tree t, similarly $\boldsymbol{Y}_t^l$, and $\Phi(\boldsymbol{X}_t^l)$ is matrix $(\phi_i(\boldsymbol{x}_j))$ with $\boldsymbol{x}_j$ being one of $\boldsymbol{X}_t^l$. Finally the inferred HR patch $\boldsymbol{y}$ is the average of patches ${}_t^lW(\boldsymbol{x})$ inferred by all n trees in the forest:

$$\hat{\boldsymbol{y}} = \frac{1}{n}\sum_{t=1}^{n} {}_t^lW(\boldsymbol{x}) = \frac{1}{n}\sum_{t=1}^{n}\sum_{i=1}^{I} {}_t^l\hat{w}_i\phi_i(\boldsymbol{x}) \tag{55}$$

where $l = l(t)$ is the leaf that sample $\boldsymbol{x}$ is routed to for each tree t. Schulter *et al.* train the trees on example LR-HR patch pairs $\boldsymbol{x}, \boldsymbol{y}$ by recursively splitting the data using parameterized splitting functions

$$\sigma(\boldsymbol{x}, i, j, \tau) := \begin{cases} \text{left} & \text{if } \text{vec}(\boldsymbol{x})\,[i] - \text{vec}(\boldsymbol{x})\,[j] - \tau < 0 \\ \text{right} & \text{otherwise} \end{cases} \tag{56}$$

where $\text{vec}(\boldsymbol{x})\,[i]$ is the ith pixel of $\text{vec}(\boldsymbol{x})$ and similarly $\text{vec}(\boldsymbol{x})\,[j]$. The tree is grown as a function of the splitting function, current depth, and current goodness of fit. At each node they select the splitting function parameters i, j, τ by evaluating all such possible pixel pairs and thresholds according to a "quality measure" whose purpose is to cluster similar LR patches:

$$Q(\boldsymbol{X}, \boldsymbol{Y}) = \sum_k \left(\|\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k\|^2 + \kappa\,\|\boldsymbol{x}_k - \bar{\boldsymbol{x}}\|^2\right) \tag{57}$$

where $\hat{\boldsymbol{y}}_k$ is the estimate produced by the linear regression fit using data that's arrived at that node. The first term in eqn. 57 measures fitting error and the second term is the clustering regularizer. They decide whether to actually split according to the *error reduction* in performing the split:

$$R(i, j, \tau, \boldsymbol{X}, \boldsymbol{Y}) = Q(\boldsymbol{X}, \boldsymbol{Y}) - \sum_{c\in\{L,R\}} \frac{|\boldsymbol{X}^c|}{|\boldsymbol{X}|} Q(\boldsymbol{X}^c, \boldsymbol{Y}^c)$$

where $\boldsymbol{X}^L$ are all of the $\boldsymbol{x}_k$ such that $\sigma(\boldsymbol{x}_k, i, j, \tau) = $ left and $\boldsymbol{Y}^L$ are all the corresponding $\boldsymbol{y}_k$ (and similarly for $\boldsymbol{X}^R, \boldsymbol{Y}^R$). The best set of parameters $\hat{i}, \hat{j}, \hat{\tau}$ are those that maximize $R(i, j, \tau, \boldsymbol{X}, \boldsymbol{Y})$. Then if $R\left(\hat{i}, \hat{j}, \hat{\tau}, \boldsymbol{X}, \boldsymbol{Y}\right)$ is greater than zero (there's an error reduction gained from performing the split) the data is split amongst the left child and right child. Each non-leaf node stores its best set of parameters and each leaf node stores its learned linear regression.

4 Deep Learning Algorithms

5 Future Research

6 Conclusion

7 Appendix

TODO: work out diffraction circular aperture TODO: workout poisson noise TODO: workout conjugate gradients TODO: workout belief propagation

Acknowledgments

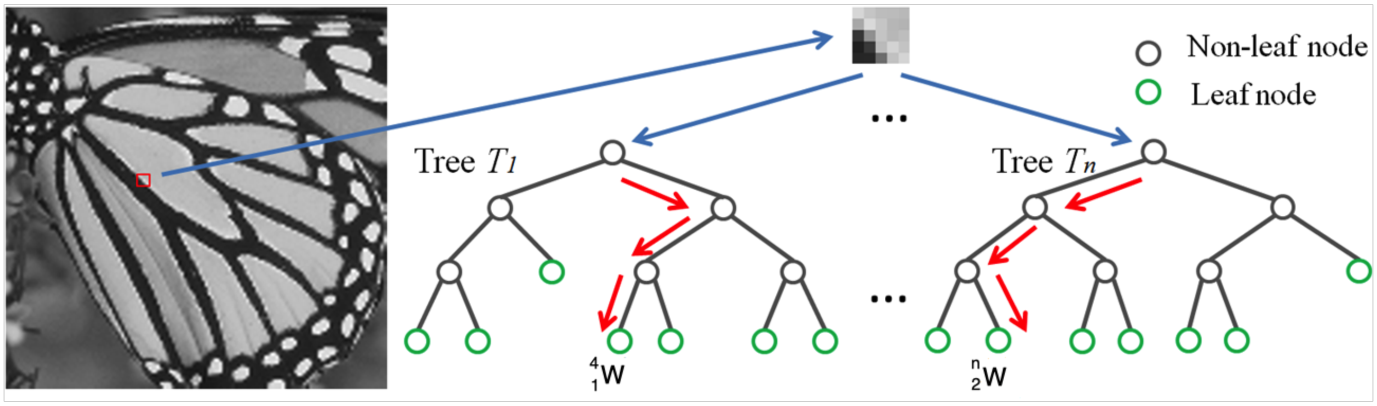


Fig. 11: A random forests which consists of a set of decision trees. Each decision tree recursively classifies the input LR patch into left or right child node, until a leaf node is reached. By using the linear regression models stored in the reach leaf node, the input LR patch can be projected to the HR patch space[35]

References

[1] M. Elad and Y. Hel-Or, "A fast super-resolution reconstruction algorithm for pure translational motion and common space-invariant blur," *IEEE Transactions on Image Processing*, vol. 10, no. 8, pp. 1187–1193, Aug. 2001. DOI: 10.1109/83.935034.

[2] S.-C. Lin and C.-T. Chen, " Reconstructing Vehicle License Plate Image from Low Resolution Images using Nonuniform Interpolation Method," Tech. Rep. 1, p. 21.

[3] M. S. Alam, J. G. Bognar, R. C. Hardie, and B. J. Yasuda, " Infrared Image Registration and High-Resolution Reconstruction Using Multiple Translationally Shifted Aliased Video Frames," Tech. Rep. 5, 2000.

[4] S. Lertrattanapanich and N. K. Bose, "High resolution image formation from low resolution frames using delaunay triangulation," *IEEE Transactions on Image Processing*, vol. 11, no. 12, pp. 1427–1441, Dec. 2002. DOI: 10.1109/TIP.2002.806234.

[5] R. Hardie, "A fast image super-resolution algorithm using an adaptive wiener filter," *IEEE Transactions on Image Processing*, vol. 16, no. 12, pp. 2953–2964, Dec. 2007. DOI: 10.1109/TIP.2007.909416.

[6] E. Nadaraya, " On Estimating Regression," *Theory of Probability & Its Applications*, vol. 9, no. 1, pp. 141–142, 1964. DOI: 10.1137/1109020. [Online]. Available: https://doi.org/10.1137/1109020.

[7] H. Takeda, S. Member, S. Farsiu, P. Milanfar, and S. Member, " Kernel Regression for Image Processing and Reconstruction," *IEEE TRANSACTIONS ON IMAGE PROCESSING*, vol. 16, no. 2, p. 349, 2007. DOI: 10.1109/TIP.2006.888330. [Online]. Available: http://www.soe.ucsc.edu/%20%7B%20~%20%7D%20htakeda..

[8] M. Wand and M. Jones, *Kernel Smoothing*, ser. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. Taylor & Francis, 1994, ISBN: 9780412552700. [Online]. Available: https://books.google.com/books?id=GTOOi5yE008C.

[9] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proceedings of the Sixth International Conference on Computer Vision*, ser. ICCV '98, Washington, DC, USA: IEEE Computer Society, 1998, pp. 839–, ISBN: 81-7319-221-9. [Online]. Available: http://dl.acm.org/citation.cfm?id=938978.939190.

[10] M. Irani and S. Peleg, "Improving resolution by image registration," *CVGIP: Graphical Model and Image Processing*, vol. 53, pp. 231–239, 1991.

[11] M. Elad and A. Feuer, "Restoration of a single superresolution image from several blurred, noisy, and undersampled measured images," *IEEE Transactions on Image Processing*, vol. 6, no. 12, pp. 1646–1658, Dec. 1997. DOI: 10.1109/83.650118.

[12] R. S. Anderssen and G. H. Golub, "Richardson"s non-stationary matrix iterative procedure.," Stanford, CA, USA, Tech. Rep., 1972.

[13] G. Casella and R. Berger, *Statistical Inference*. Duxbury Resource Center, Jun. 2001, ISBN: 0534243126.

[14] M. Haris, G. Shakhnarovich, and N. Ukita, "Deep back-projection networks for super-resolution," *CoRR*, vol. abs/1803.02735, 2018. arXiv: 1803.02735. [Online]. Available: http://arxiv.org/abs/1803.02735.

[15] M. Elad and A. Feuer, "Super-resolution reconstruction of image sequences," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 9, pp. 817–834, Sep. 1999. DOI: 10.1109/34.790425.

[16] R. C. Hardie, K. J. Barnard, and E. E. Armstrong, " Joint MAP Registration and High-Resolution Image Estimation Using a Sequence of Undersampled Images," Tech. Rep. 12, 1997.

[17] Nhat Nguyen, P. Milanfar, and G. Golub, "A computationally efficient superresolution image reconstruction algorithm," *IEEE Transactions on Image Processing*, vol. 10, no. 4, pp. 573–583, Apr. 2001. DOI: 10.1109/83.913592.

[18] A. van der Sluis and H. A. van der Vorst, " The rate of convergence of Conjugate Gradients," *Numerische Mathematik*, vol. 48, no. 5, pp. 543–560, Sep. 1986,

ISSN: 0945-3245. DOI: 10.1007/BF01389450. [Online]. Available: https://doi.org/10.1007/BF01389450.

[19] P. J. Huber, "Robust estimation of a location parameter," *Ann. Math. Statist.*, vol. 35, no. 1, pp. 73–101, Mar. 1964. DOI: 10 . 1214 / aoms / 1177703732. [Online]. Available: https://doi.org/10.1214/aoms/1177703732.

[20] D. Capel and A. Zisserman, "Super-resolution enhancement of text image sequences," in *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, vol. 1, Sep. 2000, 600–605 vol.1. DOI: 10.1109/ICPR.2000.905409.

[21] L. I. Rudin, S. Osher, and E. Fatemi, " Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, no. 1, pp. 259–268, 1992, ISSN: 0167-2789. DOI: https : / / doi . org / 10 . 1016 / 0167 - 2789(92) 90242 - F. [Online]. Available: http : / / www . sciencedirect . com / science / article / pii / 016727899290242F.

[22] S. Farsiu, M. D. Robinson, M. Elad, and P. Milanfar, "Fast and robust multiframe super resolution," *IEEE Transactions on Image Processing*, vol. 13, no. 10, pp. 1327–1344, Oct. 2004. DOI: 10 . 1109 / TIP. 2004 . 834669.

[23] M. Elad, "On the origin of the bilateral filter and ways to improve it," *IEEE Transactions on Image Processing*, vol. 11, no. 10, pp. 1141–1151, Oct. 2002. DOI: 10.1109/TIP.2002.801126.

[24] W. T. Freeman, T. R. Jones, and E. C. Pasztor, "Example-based super-resolution," *IEEE Computer graphics and Applications*, no. 2, pp. 56–65, 2002.

[25] J. Yedidia, W. Freeman, and Y. Weiss, "Understanding belief propagation and its generalizations," in *Exploring Artificial Intelligence in the New Millennium*, G. Lakemeyer and B. Nebel, Eds., Morgan Kaufmann Publishers, Jan. 2003, ch. 8, pp. 239–236, ISBN: 1-55860-811-7. [Online]. Available: https://www.merl.com/publications/TR2001-22.

[26] G. Donatti, "Memory organization for invariant object recognition and categorization," PhD thesis, Jun. 2016. DOI: 10.13140/RG.2.1.2880.7921.

[27] L. K. Saul and S. T. Roweis, "An introduction to locally linear embedding," *unpublished*, 2000. [Online]. Available: https://cs.nyu.edu/~roweis/lle/papers/lleintro.pdf.

[28] Jianchao Yang, J. Wright, T. Huang, and Yi Ma, "Image super-resolution as sparse representation of raw image patches," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2008, pp. 1–8. DOI: 10.1109/CVPR.2008.4587647.

[29] A. M. Tillmann, "On the computational intractability of exact and approximate dictionary learning," *IEEE Signal Processing Letters*, vol. 22, no. 1, pp. 45–49, Jan. 2015. DOI: 10.1109/LSP.2014.2345761.

[30] D. L. Donoho and J. Tanner, " Sparse nonnegative solution of underdetermined linear equations by linear programming," *Proceedings of the National Academy of Sciences*, vol. 102, no. 27, pp. 9446–9451, 2005, ISSN: 0027-8424. DOI: 10.1073/pnas.0502269102. [Online]. Available: https://www.pnas.org/content/102/27/9446.

[31] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Transactions on Image Processing*, vol. 19, no. 11, pp. 2861–2873, Nov. 2010. DOI: 10 . 1109 / TIP. 2010 . 2050625.

[32] R. Timofte, V. De, and L. V. Gool, "Anchored neighborhood regression for fast example-based super-resolution," in *2013 IEEE International Conference on Computer Vision*, Dec. 2013, pp. 1920–1927. DOI: 10.1109/ICCV.2013.241.

[33] R. Zeyde, M. Elad, and M. Protter, " On single image scale-up using sparse-representations," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6920 LNCS, no. 1, pp. 711–730, 2012, ISSN: 03029743. DOI: 10.1007/978-3-642-27413-8_47.

[34] S. Schulter, C. Leistner, and H. Bischof, "Fast and accurate image upscaling with super-resolution forests," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, pp. 3791–3799. DOI: 10.1109/CVPR.2015.7299003.

[35] J. Huang, W. Siu, and T. Liu, "Fast image interpolation via random forests," *IEEE Transactions on Image Processing*, vol. 24, no. 10, pp. 3232–3245, Oct. 2015. DOI: 10.1109/TIP.2015.2440751.