# Visualizing Image Priors: Supplementary Materials

Tamar Rott Shaham and Tomer Michaeli

Technion—Israel Institute of Technology
{stamarot@campus,tomer.m@ee}.technion.ac.il

1. We explain how we perform denoising with the cross-scale patch recurrence prior of [19], which was originally proposed in the context of blind deblurring.
2. We remark on how we solve the optical flow problem (5) using the algorithm proposed in [38].

## 1 Denoising Using Cross-Scale Patch Recurrence

Small patches tend to recur abundantly across scales of natural images. This property was used in [19] for performing blind deblurring. To visualize the cross-scale recurrence prior of [19], we adapt their algorithm to solving the denoising problem

$$\arg\min_{x} \|y - x\|^2 + \lambda\rho(x), \tag{1}$$

where $y$ is an input (noisy) image, and $x$ is the output denoised image. Specifically, we use the penalty term $\rho(x)$ proposed in [19], which measures the degree of dissimilarity between patches in the image $x$ and their Nearest Neighbor patches (NNs) within the $\alpha$-times smaller version of $x$, denoted $x^\alpha$. This term is defined as

$$\rho(x) = -\sum_j \log\left(\sum_i \exp\left\{-\frac{1}{2h^2}\|\mathbf{Q}_j\hat{x} - \mathbf{R}_i\hat{x}^\alpha\|^2\right\}\right). \tag{2}$$

where $\boldsymbol{Q}_j$ is the matrix which extracts the $j$-th patch from $x$, $\boldsymbol{R}_i$ is the matrix which extracts the $i$-th patch from $x^\alpha$, and $h$ is a bandwidth parameter. Following the derivation in [19], setting the gradient to zero leads to the condition

$$x = \frac{y + \beta z}{1 + \beta}, \tag{3}$$

where $\beta = \frac{\lambda M^2}{h^2}$, with $M$ being the patch width (assuming square patches), and $z$ is an image obtained by replacing each patch in $x$ by a weighted combination of its NNs from $x^\alpha$. Namely,

$$z = \frac{1}{M^2}\sum_j \mathbf{Q}_j^T \sum_i w_{i,j}\mathbf{R}_i x^\alpha \tag{4}$$

with weights

$$w_{i,j} = \frac{\exp\left\{-\frac{1}{2h^2}\|\mathbf{Q}_j x - \mathbf{R}_i x^\alpha\|^2\right\}}{\sum_m \exp\left\{-\frac{1}{2h^2}\|\mathbf{Q}_j x - \mathbf{R}_m x^\alpha\|^2\right\}}. \tag{5}$$

**Input:** Noisy image $y$
**Output:** Denoised image $x$
Initialize $x = y$
**for** $n = 1, \dots, N$ **do**
  **Image prior update**: Down-scale the image $x$ by a factor of $\alpha$ to obtain $x^{\alpha}$.
  **for** $k = 1, \dots, K$ **do**
    $z$ **step**: update the image $z$ according to (4).
    $x$ **step**: update the image $x$ according to (3).
  **end**
**end**

**Algorithm 1:** Cross-scale patch recurrence denoisng

As in [19], to solve (3), we iterate between computing $z$ based on the current $x$ and updating $x$ based on the new $z$. Once every several iterations, we update $x^{\alpha}$ to be the $\alpha$-times smaller version of the current $x$. This denoising algorithm is described in Alg. 1. As in [19], we use $\alpha = 0.75$ and one NN per patch.

## 2 Optical Flow

To solve the optical flow problem (Eq. (5) in the paper), we used the iteratively re-weighted least-squares (IRLS) algorithm proposed in [38]. We note that our problem involves an $L_2$ data fidelity term, whereas the algorithm of [38] is typically used with an $L_1$ data fidelity term. However, the derivation in [38] is actually quite general, and can be easily adapted to arbitrary data fidelity penalties. Specifically, [38] considers the minimization of the following objective

$$\begin{aligned} \arg\min_{u,v} \iint \psi\left(|x(\xi,\eta) - y(\xi + u(\xi,\eta), \eta + v(\xi,\eta))|^2\right) d\xi d\eta \\ + \alpha \iint \phi\left(\|\nabla u(\xi,\eta)\|^2 + \|\nabla v(\xi,\eta)\|^2\right) d\xi d\eta, \end{aligned} \tag{6}$$

where $x$ and $y$ are two images, $(u, v)$ is the flow field which warps $y$ into $x$, and $\alpha$ is the weight of the flow regularization term.

The algorithm proposed in [38], iteratively solves sets of linear equations to update $u$ and $v$. In [38], this approach was specifically implemented and tested with the robust functions

$$\psi(x^2) = \sqrt{x^2 + \varepsilon^2}, \quad \phi(x^2) = \sqrt{x^2 + \varepsilon^2}, \tag{7}$$

where $\varepsilon$ is some small constant. For our prior visualization algorithm, we rather need to solve (6) with an $L_2$ data fidelity (namely, where the first term in (6) is the $L_2$ distance between $x$ and the warped version of $y$). Therefore, in our implementation, we changed $\psi$ to be the $L_2$ penalty

$$\psi(x^2) = x^2. \tag{8}$$

This modification leads to a different set of linear equations, which have to be solved in each stage. But the general algorithm remains the same.