Revealing and Modifying Non-Local Variations in a Single Image

Tali Dekel [1] Tomer Michaeli [2] Michal Irani[3] William T. Freeman[1]

[1] MIT CSAIL [2] Technion [3] Weizmann Institute

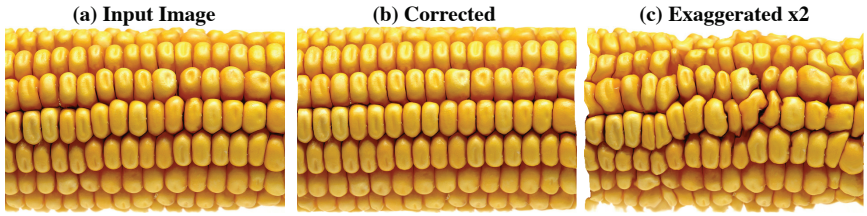

Figure 1: *An example of using our Non-Local Variations algorithm to automatically detect and visualize small deformations between repeating structures in a single image (a). Our method can be used to identify and correct these variations, thus producing an 'idealized' version of the image (b). Alternatively, our method can be used to exaggerate these variations (c). In the corrected output, the variability in the shape of the corn's kernels is reduced, and the misalignment of rows is corrected. In the exaggerated output, the subtle differences between the kernels and the row misalignment are highlighted. Photo courtesy of Giandomenico Pozz.*

Abstract

We present an algorithm for automatically detecting and visualizing small non-local variations between repeating structures in a single image. Our method allows to automatically *correct* these variations, thus producing an 'idealized' version of the image in which the resemblance between recurring structures is stronger. Alternatively, it can be used to *magnify* these variations, thus producing an exaggerated image which highlights the various variations that are difficult to spot in the input image. We formulate the estimation of deviations from perfect recurrence as a general optimization problem, and demonstrate it in the particular cases of *geometric deformations* and *color variations*.

CR Categories: I.4.3 [Image Processing and Computer Vision]: Enhancement—Geometric correction; deviations;

Keywords: Internal patch recurrence, irregularities, distortions, deformations, defects.

1 Introduction

Our visual world is comprised of small structures that recur abundantly. From cells under the microscope to the leaves of a tree or a crowd of people in a stadium, recurrences dominate our visual environment. However, even when the resemblance between repeating structures is strong, it is often not perfect. For example, the hexagonal cells of a honey hive might slightly differ in their shape, and the tomatoes in a super-market stand might slightly differ in their color. In many situations, these deviations from ideal recurrence may reveal interesting and useful information about the underlying objects.

We propose a method for detecting and visualizing *Non-Local Variations* (NLV) between repeating structures in a single natural image. For example, in Fig. 1(a), our algorithm automatically detects the repetitions and estimates the *nonparametric* variations between the corn's kernels. Our method is then used to visually explore these variations by rendering a new image in which these variations are modified. Specifically, reducing these variations results in an 'idealized' version of the image in which the kernels are more similar to one another and the rows are more aligned (see Fig. 1(b)). By exaggerating these variations, our method reveals various subtle differences between the kernels, which are hard to spot in the input image, such as changes in their shape, size, alignment in rows (see Fig. 1(c)).

Our approach is based on searching for repeated patches inside the entire image, and estimating their deviations from ideal recurrence. The redundancy of small patches in natural images has been used in many low-level vision tasks. Examples include image compression [Barnsley and Sloan 1990], image summarization [Jojic et al. 2003; Simakov et al. 2008], texture synthesis [Efros and Freeman 2001], inpainting [Criminisi et al. 2004], denoising [Buades et al. 2005; Dabov et al. 2007], deblurring [Danielyan et al. 2012] and super-resolution [Glasner et al. 2009; Freedman and Fattal 2011]. These methods often ignore slight deviations between the repeating patches (*i.e.,* treat them as noise) whereas our goal is to estimate and visualize these variations.

Our key idea is that by applying some simple transformation on the input image, we can make it closer to a *model image* in which the resemblance between recurring patches is maximal. The choice of type of transformation depends on the assumed underlying variations. In this paper, we focus on two types of variations: *geometric deformations* for which the transformation is a dense flow field, and *color variations* for which the transformation is a local color mapping. However, the framework we propose is more general and can be easily adapted to other types of transformations. We formulate an optimization problem, from which we derive an iterative algorithm for estimating both the idealized version of the image and the transformation which turns the image closest to its ideal version.

Our method allows each patch to find its nearest-neighbors at arbitrary locations in the image. This is in contrast to texture manipulation methods (*e.g.,* [Liu et al. 2004; Eisenacher et al. 2008; Liu et al. 2012; Hays et al. 2006; Park et al. 2009; Kim et al. 2012]), which often assume a single texton repeating on a lattice. The combination

of non-local repetitions together with a nonparametric transformation, makes our technique applicable to *unconstrained natural images* in which both the arrangement of structures and the variations between them are complex (*e.g.,* see Fig. 2 and Fig. 5). Therefore, our algorithm has a wide range of applications. In this paper, we specifically demonstrate its use for: (i) Idealizing images (making them more visually pleasing). (ii) Revealing properties of objects, which are hard to visually perceive by the naked eye (*e.g.,* which tomatoes in a market stand are riper than others). (iii) Visualizing defects in materials inspection. (iv) Generating humoristic/artistic images.

The rest of the paper is organized as follows. We survey related work in Sec. 2. We provide a mathematical formulation of the NLV problem in Sec. 3. In Sec. 4, we give an overview of our algorithm and in Sec. 5 we explain each of its steps in more detail. In Sec. 6, we discuss the visualization of the algorithm's output. Finally, we present synthetic validation experiments as well as results on real-world images in Sec. 7.

2 Related Work

Texture Manipulation Estimating geometric and photometric deformations has attracted attention mainly in the context of *texture images*. Methods in this category are used for texture symmetrization [Liu et al. 2004; Kim et al. 2012], texture synthesis [Eisenacher et al. 2008; Kim et al. 2012], texture replacement [Liu et al. 2004; Eisenacher et al. 2008; Liu et al. 2012] and shape from texture [Malik and Rosenholtz 1997; Clerc and Mallat 2002]. In this paper, however, we do not restrict ourselves to texture images but rather consider *unconstrained natural images*.

In particular, our approach bears three fundamental differences w.r.t. the texture manipulation line of work: (i) Most existing methods are designed for near-regular textures (*i.e.,* a single texton recurring on a lattice) and explicitly rely on the detection of a deformed lattice (which is done either manually [Liu et al. 2004; Eisenacher et al. 2008] or automatically [Hays et al. 2006; Park et al. 2009]). In our setting, structures may repeat at arbitrary locations, where no lattice can be defined (see *e.g.,* Figs. 5 and 9). (ii) While some methods treat "stochastic" textures without a well-defined lattice [Kim et al. 2012], they still cannot handle images with multiple different textures (see *e.g.,* Fig. 5(d)). (iii) Our method captures complex deformations at multiple scales. Thus, even when operating on near-regular textures, we recover not only the lattice deformation, but also smaller-scale variations. These differences render texture based methods inapplicable in our setting.

Motion Magnification The problem of revealing and magnifying small variations, has also been addressed in the past in the context of video-sequences [Liu et al. 2005; Wu et al. 2012; Wadhwa et al. 2013]. It has been shown that by temporally processing a video, it is possible to amplify tiny movements between consecutive frames, which are unnoticeable in the raw image sequence. The variations we analyze in this paper, however, occur within a *single image*. That is, rather than observing the same location at different frames of a movie, we focus on revealing the *non-local variations* between structures that occur at different general locations in a single image.

Deviations From Perfect Patch Recurrence Examining *deviations* from ideal patch recurrences in a natural image has been used to recover properties of the imaging device. Specifically, this idea was used for recovering the camera's blur kernel in the context of blind deblurring [Michaeli and Irani 2014] and blind super-resolution [Michaeli and Irani 2013], and for estimating the noise

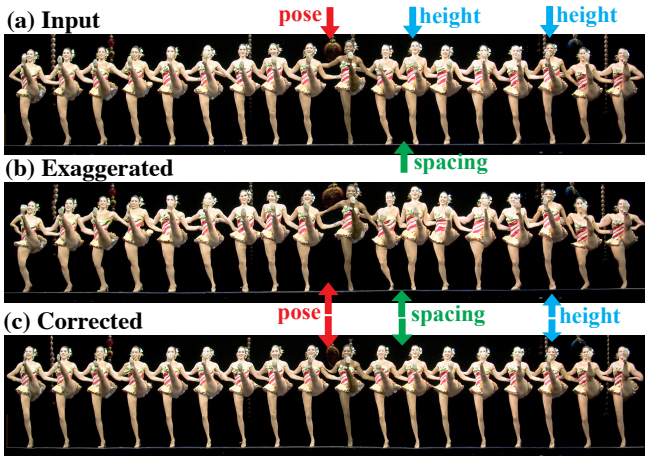

Figure 2: Dancers. *Our algorithm reveals and corrects slight differences in the poses, heights, and spacings between a line of dancers performing the same high leg-kick routine (a). In our exaggerated output (b), these various differences clearly stand out. Our corrected output (c) provides an idealized version of the image, in which the non-local variations are reduced. Photo courtesy of Andy Colwell, Penn State.*

statistics in the context of blind denoising [Lebrun et al. 2014]. In sharp contrast to these works, the deviations from ideal patch recurrences in our case, are *not caused by the imaging process* and are not related to degradations like noise or blur. Rather, these deviations are associated to inherent differences between the objects in the scene, and thus occur also in clean sharp images. Moreover, as opposed to the global models used by those methods, the variations we consider here are spatially varying and thus have many more degrees of freedom. This makes our setting much more challenging. Finally, none of the aforementioned methods considers the visualization of the variations between repeating structures.

Saliency Detecting irregularities in recurring structures is also used for defect detection on semiconductor wafers [Shankar and Zhong 2005; Zontak and Cohen 2010], target detection in sonar and multispectral images [Mishne and Cohen 2013] and saliency detection [Boiman and Irani 2007; Seo and Milanfar 2009; Goferman et al. 2012]. In these works, the main goal is to detect outliers, *i.e.,* structures that significantly differ from their surrounding. Our goal is different as we are interested in variations between structures that are not necessarily salient, but rather very similar to one another.

Parametric Deformations A closely related work is [Wadhwa et al. 2015], which presents a *local parametric* method for revealing tiny deformations. This is done by fitting a perfect *parametric model* (*e.g.,* a circle or a straight line) to an object of interest in the image, and computing the local geometric deviations from that parametric model. In contrast, our approach is *non-parametric*, hence can handle a wide variety of complex geometric deformations which cannot be modeled parametrically (*e.g.,* deviations in pose between the different dancers in Fig. 2). Furthermore, our algorithm does not assume a model, but rather implicitly infers it automatically from the data.

3 Problem Formulation

The input to our method is a still image, I, which contains repeating structures. Our key assumption is that the variations between these

repeating structures can be reduced by applying a transformation $\mathcal{T}$ on the input image. That is,

$$\mathcal{T}\{I\} = J + N, \tag{1}$$

where J is an 'ideal' image, in which the repeating structures are identical, and N is noise. This formulation is rather general and can account for various types of variations. In this paper, we focus on *geometric variations* and *color variations*.

In the case of geometric variations, $\mathcal{T}\{I\}$ backward warps the input image I with a dense deformation field $(u(x,y), v(x,y))$:

$$\mathcal{T}\{I\}(x,y) = I(x + u(x,y), y + v(x,y)). \tag{2}$$

For color variations, $\mathcal{T}$ is associated with a spatially-varying 3×3 color-correction matrix $\boldsymbol{C}(x,y)$, so that $\mathcal{T}\{I\}$ is the color-corrected image:

$$\mathcal{T}\{I\}(x,y) = \boldsymbol{C}(x,y)I(x,y). \tag{3}$$

Our goal is to recover the idealized image J, as well as the transformation $\mathcal{T}$. By doing so, we can visually modify the internal non-local variations within the input image. Specifically, applying $\mathcal{T}$ on the input image brings it closer to J, *i.e.,* **reduces** the variations between repeating structures. In this case, the transformed image serves as a correction to the input image, which may be more visually pleasing. Alternatively, by inverting the transformation, we can **exaggerate** those variations, thus highlighting and revealing subtle differences which are hard to visually perceive in the input image.

Clearly, determining both J and $\mathcal{T}$ is an ill posed problem (there are various combinations of J and $\mathcal{T}$ that are consistent with the input image). However, the desired internal recurrence of structures provides us with a strong prior on J. We formulate the estimation of nonlocal variations as the following optimization problem

$$\arg\min_{J,\mathcal{T}} \underbrace{E_{\text{rec}}(J, \text{DB})}_{\text{recurrence}} + \underbrace{\lambda E_{\text{data}}(J, \mathcal{T}\{I\})}_{\text{data fidelity}} + \underbrace{\alpha E_{\text{reg}}(\mathcal{T})}_{\text{regularization}}, \tag{4}$$

where DB denotes the database of all patches (with overlaps) extracted from J. The term $E_{\text{rec}}(J, \text{DB})$ penalizes for dissimilarities between each patch $p_j \in J$ and its nearest neighbors $\{q_i\} \in \text{DB}$. We define this term to be

$$E_{\text{rec}} = -\sum_{p_j \in J} \log\left(\sum_{q_i \in \text{DB}} \exp\left\{-\frac{1}{2h^2}\|p_j - q_i\|^2\right\}\right), \tag{5}$$

where, h is a bandwidth parameter (see [Michaeli and Irani 2014] for a similar prior and a Bayesian interpretation of this term). The role of the fidelity term in (4) is to force the ideal image J and the corrected image $\mathcal{T}\{I\}$ to be similar. We define it as

$$E_{\text{data}} = \iint \psi\left(\|J(x,y) - \mathcal{T}\{I\}(x,y)\|^2\right) dxdy, \tag{6}$$

where $\psi(a^2) = \sqrt{a^2 + \varepsilon^2}$ with a small constant ε. This penalty constitutes an approximation to the L_1 distance, which is known to be robust to outliers. The last term in (4) regularizes the transformation to be piecewise spatially smooth, and depends on the type of transformation used. Its definition for geometric and color variations is given in Sections 5.2 and 5.3, respectively.

4 Overview of the NLV algorithm

The optimization problem (4) is non-convex (due to the recurrence term (5)). To solve it, we use alternating minimization. That is, we iterate between minimizing the objective with respect to J (increasing the internal recurrence within it) and updating $\mathcal{T}$ (determining the $\mathcal{T}$ which maps I to the new J). Our method is summarized in Algorithm 1 and is illustrated in Fig. 3.

Input: Nonideal image I.
Output: Ideal image J, idealizing transformation $\mathcal{T}$.

Initialize $\mathcal{T}$ to be the identity mapping and J to be I.
for $t = 1, \ldots, T$ **do**
1. **Database Update**:
   Set DB to hold all overlapping patches in J.
2. **Image Update**:
   Minimize (4) w.r.t. J, holding DB and $\mathcal{T}$ fixed.
3. **Transformation Update**:
   Minimize (4) w.r.t. $\mathcal{T}$, holding DB and J fixed.

end

Algorithm 1: Non-Local Variations (NLV).

We first explain the intuition behind the steps of Algorithm 1 in a single iteration. Each of these steps is described in Sec. 5 in more detail.

Image Update The goal of this step is to increase the internal recurrence within J. Only the first two terms in (4) depend on J. Therefore, this step constructs a new image J which, on one hand, is close to the current transformed input $\mathcal{T}\{I\}$, but at the same time, the recurrence of patches within it is stronger. In practice, this is done by assembling J from the database patches while constraining it to be similar to $\mathcal{T}\{I\}$. This step is described in more detail in Sec. 5.1.

Transformation Update Having constructed a new J, the purpose of this step is to update the transformation $\mathcal{T}$ (last two terms in (4)), such that $\mathcal{T}\{I\}$ will be similar to J and the transformation is piecewise spatially smooth. For example, for geometric deformations, this step boils down to optical flow estimation between the images I and J. This step is described in more detail in Sections 5.2 and 5.3.

Database Update In this step we update the database to contain all patches (with overlaps) from the new J. By doing so, we reduce the diversity (or entropy) of the patches in DB from iteration to iteration. This allows the patches in J at the next iteration to become even more similar to one another.

To be able to capture variations between structures of various sizes, we run our algorithm coarse-to-fine in a pyramid structure. That is, at each pyramid level, we run Algorithm 1 and then upscale the recovered 'ideal' image J and the transformation $\mathcal{T}$ to constitute initializations for the next pyramid level. This speeds up the process, and prevents the algorithm from getting stuck in a local minimum.

5 Detailed Description of the Algorithm

We now provide the mathematical details of our algorithm. To simplify the notation, we denote the corrected image $\mathcal{T}\{I\}$ by I^{c}.

5.1 Image Update

Minimizing the objective (4) with respect to J requires setting the gradient of $E_{\text{rec}} + \lambda E_{\text{data}}$ with respect to J to zero. As we show in

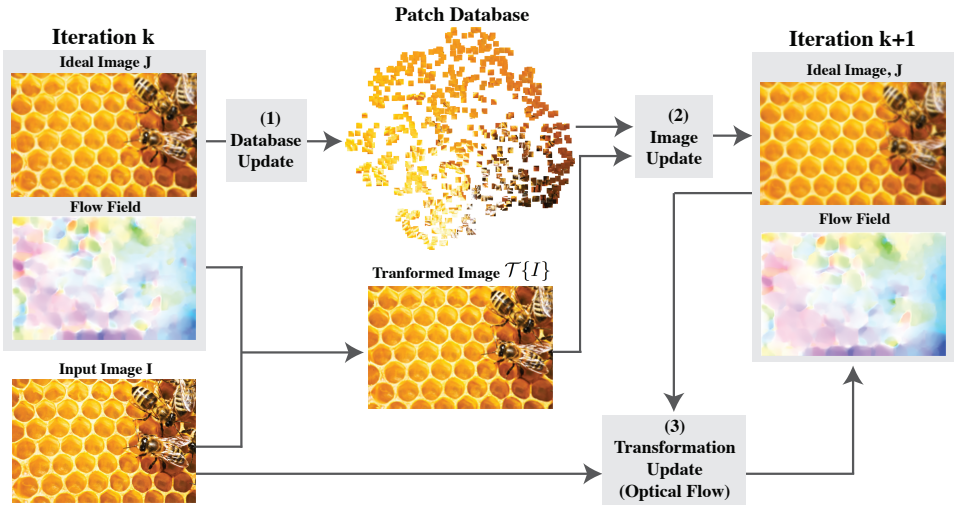

Figure 3: A single iteration of NLV (illustration for geometric mode). *At each iteration, the 'ideal' image J and the flow field $\mathcal{T}$ from the previous iteration (left) are updated using three steps: (i) Update the patch database by extracting all patches from the current J. (ii) Given the new database and the warped input image $\mathcal{T}\{I\}$, update the 'ideal' image J. This step encourages each patch in J to be as similar as possible to its NNs in the database, while also keeping J similar to $\mathcal{T}\{I\}$. (iii) Given the new J and the input I, update the transformation $\mathcal{T}$, by computing the optical flow between J and I.*

Appendix A, this leads to the equation

$$J(x,y) = \beta(x,y)Z(x,y) + (1-\beta(x,y))I^{\mathrm{c}}(x,y), \qquad (7)$$

where Z is an image obtained by replacing each patch in J by a weighted combination of its Nearest Neighbors (**NNs**) from the database. That is, J is a linear combination of Z and I^{c}, with a spatially-varying weight β that determines to which of the two images J is closer. More concretely (see derivation in Appendix A),

$$\beta(x,y) = \frac{W_{\mathrm{data}}(x,y)}{W_{\mathrm{data}}(x,y) + \frac{h^2}{M^2}}, \qquad (8)$$

$$W_{\mathrm{data}}(x,y) = \frac{1}{\lambda}\psi\left(\|J(x,y) - I^{\mathrm{c}}(x,y)\|^2\right), \qquad (9)$$

where M is the patch width and ψ is as in (6).

Since both the image Z and the weight map β are nonlinear functions of the unknown J, Eq. (7) admits no closed form solution. To solve it, we iterate between computing Z and β from our current estimate of J, and updating J according to Eq. (7). These two phases can be viewed as the iteratively reweighted least-squares (IRLS) method.

The intuition here is simple. Constructing Z amounts to applying nonlocal means "denoising" on J [Buades et al. 2005]. That is, replacing each patch by a weighted average of all its NNs, where the NNs can be found in any location in the image. This step essentially creates an image with strong patch recurrences. However, Z may not necessarily be close to I^{c}, as required by our data-term. Therefore, it is blended with I^{c} according to (7). Since this blending might reduce the internal recurrences in Z, the entire process is repeated (applying nonlocal means "denoising" on the new J, blending with I^{c}, and so on). This process converges to an image J that has strong recurrences, yet is still similar to I^{c}.

The effect of λ The parameter λ controls the relative weight of the data fidelity term w.r.t. the recurrence term in Eq. (4). That is, the smaller λ is, the stronger the patch recurrence in J will become. To understand how this is achieved, note that regions in J that are not repetitive (*i.e.,* do not have good NNs), are altered by the nonlocal means method more than repetitive regions. Thus, at some point, they may start to significantly deviate from the corrected image I^{c}. When λ is small, this will increase the weights $\beta(x,y)$ towards 1 (Eq. (8),(9)). Hence, the image J will be almost entirely copied from Z at those locations, so that J will exhibit an 'inpainting' effect. That is, these non-repetitive regions will be filled-in with some structure that *does* recur at other parts of the image (see Fig. 7).

5.2 Transformation Update: Geometric Deformations

For geometric deformations, the transformation $\mathcal{T}$ is given by (2) and we define the regularization term E_{reg} to be the robust penalty

$$E_{\mathrm{reg}} = \iint \psi\left(\|\nabla u(x,y)\|^2 + \|\nabla v(x,y)\|^2\right) dxdy, \qquad (10)$$

where $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y})$ and ψ is as in (6). This term is used in the optical flow literature [Brox et al. 2004], and is known to promote piecewise smooth flow fields.

Minimizing the objective (4) with respect to the flow (u,v), boils down to minimizing $\lambda E_{\mathrm{data}} + \alpha E_{\mathrm{reg}}$ (Eqs. (6),(10)). This corresponds to estimating the optical flow[1] between the images J and I. To solve this problem, we use the IRLS method of [Liu 2009].

5.3 Transformation Update: Color Deformations

Recall that in the case of color variations, J is an 'ideal' image in which the repeating structures have the same color and $\mathcal{T}$ corresponds to a local color correction matrix $\boldsymbol{C}(x,y)$, which maps the color of each pixel in I to the color of the corresponding pixel in J (Eq. (3)). The only change in the objective with respect to the geometric case is the regularization term, which we now define as

$$E_{\mathrm{reg}} = \iint \psi\left(\|\boldsymbol{C}_x(x,y)\|_{\mathrm{F}}^2 + \|\boldsymbol{C}_y(x,y)\|_{\mathrm{F}}^2\right) dxdy. \qquad (11)$$

[1] Note that the relative weight of the smoothness term in our optical flow estimation is α/λ.

Here $\|\cdot\|_{\text{F}}$ denotes the Frobenius norm, ψ is as in (6), and $\boldsymbol{C}_x$ and $\boldsymbol{C}_y$ are the x and y derivatives of the color mapping $\boldsymbol{C}(x,y)$.

As in Sec. 5.2, determining the color transformation requires minimizing $\lambda E_{\text{data}} + \alpha E_{\text{reg}}$ (now given by Eqs. (6) and (11)). This minimization can be solved using the Euler-Lagrange equations, which lead to the condition (see Appendix B for full derivations):

$$\boldsymbol{C}(x,y) = \bar{\boldsymbol{C}}(x,y) + \gamma(x,y)\left(J(x,y) - \bar{\boldsymbol{C}}(x,y)I(x,y)\right)I(x,y)^T, \quad (12)$$

where $\bar{C}(x,y)$ is a smoothed version of $\boldsymbol{C}(x,y)$ and

$$\gamma(x,y) = \frac{W_{\text{reg}}(x,y)}{W_{\text{data}}(x,y) + W_{\text{reg}}(x,y)\|I(x,y)\|^2}, \quad (13)$$

$$W_{\text{reg}}(x,y) = \frac{1}{\alpha}\psi\left(\|\boldsymbol{C}_x(x,y)\|_{\text{F}}^2 + \|\boldsymbol{C}_y(x,y)\|_{\text{F}}^2\right), \quad (14)$$

with W_{data} of (9).

Here too, there is no closed form solution to Eq. (12). We thus follow again an iterative approach. That is, in each iteration we compute $\bar{\boldsymbol{C}}$ and γ based on the current $\boldsymbol{C}$, and then use them to update $\boldsymbol{C}$ according to (12).

This update step, which was derived analytically, has a simple intuitive explanation. Recall that we want that $J \approx \boldsymbol{C}I$ (Eqs. (1) and (3)). We also want the transformation $\boldsymbol{C}(x,y)$ to be smooth (Eq. (11)). Thus, in each iteration we compute the "error" $J - \bar{\boldsymbol{C}}I$, and add this difference to the smoothed mapping $\bar{\boldsymbol{C}}$ from the previous iteration. This back-propagation step reduces the error at the next iteration. The procedure converges once $\bar{\boldsymbol{C}}$ maps I into J.

Note that the trivial solution to (4) is the black image $J(x,y) = 0$ and a transformation $\mathcal{T}$ which maps all pixels to black (namely, $\boldsymbol{C}(x,y) = 0$, which is also a smooth transformation). To avoid this undesired trivial solution, we constrain the intensity of the 'ideal' image J to be equal to the intensity of the input image I, and allow only the two chromatic values to change. Namely, $\boldsymbol{C}$ is a 2×2 matrix. We solve for this in the YCbCr color space.

6 Visualization and User Interaction

Once the idealizing transformation $\mathcal{T}$ is determined, it can be used to visualize the non-local variations in the input image. Specifically, we generate a new image by

$$I_{\text{out}} = \mathcal{T}^{\mu}\{I\}, \quad (15)$$

where $\mathcal{T}^{\mu}$ denotes the μ-th power of $\mathcal{T}$. The sign of μ determines whether to apply the transformation or its inverse, and the absolute value of μ determines the extent of magnification ($\mu = 0$ leaves the input image unchanged). For example, for $\mu = 1$, the image is corrected and I_{out} becomes similar to the ideal image J, whereas $\mu = -1$ exaggerates the variations by a factor of one. For $\mu > 1$, the nonlocal variations are inverted (*e.g.,* the tall dancers in Fig. 2 will become shorter than the rest, and vice versa).

In the case of *geometric deformations*, we warp the input image with the scaled flow field $(\tilde{u}, \tilde{v}) = \mu \cdot (u, v)$. In the case of *color variations*, applying $\mathcal{T}^{\mu}$ amounts to applying the matrix $\boldsymbol{C}^{\mu}(x,y)$ on each of the pixels of the input image.

When exaggerating geometric deformations, care must be taken in order to produce visually pleasing results. In particular, the exaggerated field may fold the image onto itself at certain locations. This effect occurs wherever the determinant of the Jacobian of the transformation, which is given by $|\mathbb{J}(\mathcal{T}^{\mu})| = 1 + \tilde{u}_x + \tilde{v}_y + \tilde{u}_x\tilde{v}_y - \tilde{u}_y\tilde{v}_x$, is negative. To prevent this undesired phenomenon, we smooth the flow at the locations in which the folding occurs and their surroundings. We repeatedly apply this smoothing and recompute the Jacobian, until $|\mathbb{J}(\mathcal{T}^{\mu})| > 0$ everywhere in the image.

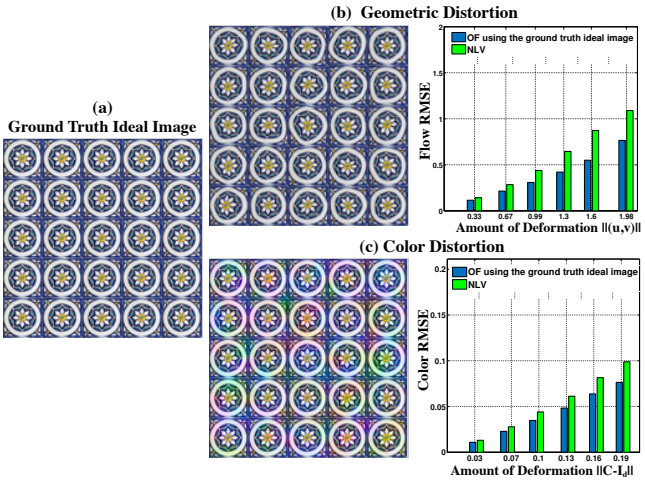

Figure 4: Synthetic Evaluation. *A ground truth ideal image consisting of 25 identical tiles (a), was distorted geometrically (b), and photometrically (c). On the right, plots show the root mean square error (RMSE) of the transformation recovered by our method (green) and of the transformation recovered by using the ground truth ideal image, which is not available to our method (blue). Both errors are plotted as a function of the average deformation magnitude (see text for more details). Note that the errors of our method are very close to those obtained using the ground truth.*

To aid the visualization and grant the user more control over the final result, we incorporated the following options (see Supplementary Material for demonstrations):

1. *User Interaction:* The user can interactively adjust the amount of exaggeration/correction with a knob (controlling μ). In addition, the user can control the spatial scale of the explored variations. This is done by filtering the flow field (u, v). In particular, removing the low frequencies (coarse scales) of the flow leaves only the small-scale variations (*e.g.,* the dancers in Fig. 2 become more similar but the spacings between them are not corrected).
2. *Regions of interest:* The user can apply exaggeration/correction only on regions of interest (ROI). Different exaggerations can be applied to different ROIs.
3. *Animation:* A video clip can be generated, in which the exaggeration is gradually increased or decreased (by applying the transformation $\mathcal{T}^{\mu}$ with varying μ).

7 Results

We evaluated our method both for color and for geometric variations, using synthetic and real data. We use a pyramid with 0.75 scale gaps, and apply 20 iterations of NLV (Alg. 1) at the coarsest level, and 5 iterations at the rest of the levels. In each such iteration, the image update step (7) is iterated 5 times, using 20 nearest-neighbors. For geometric deformations, we use the optical flow implementation of [Liu 2009]. For color variations, the transformation update step (12) is iterated 300 times.

The overall computation time is mostly (75%) spent on the near-

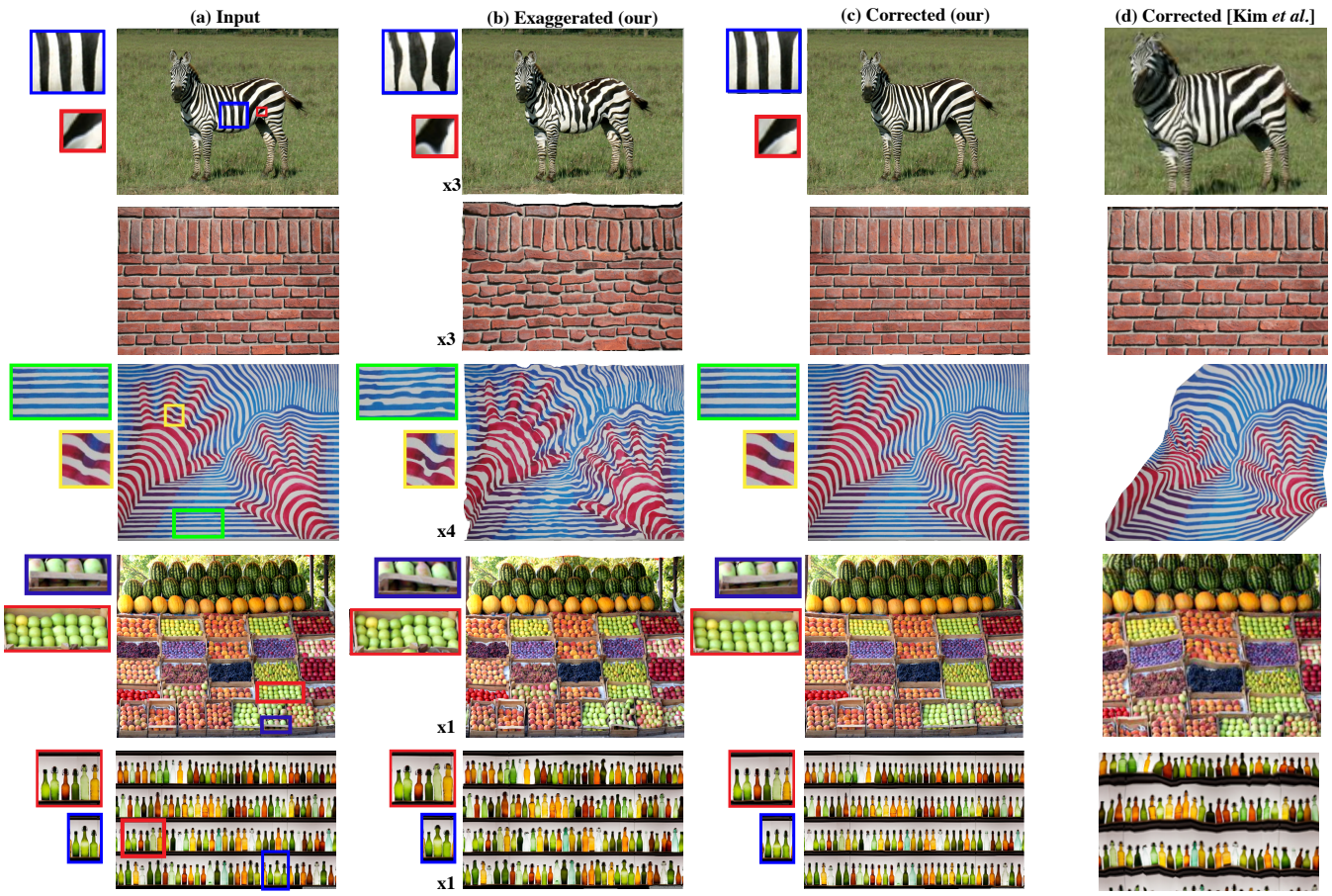

Figure 5: Revealing and correcting non-local geometric variations. *(a) Input images for Zebra, Bricks, Opt Art, Fruit, and Bottles. (b) Our exaggeration results (by a factor of μ), reveal the variety of geometric variations hiding in the input images. (c) The corrected version of (a) obtained by our method. In each of the examples, the internal variability between the repeating structures is reduced. The insets depict zoomed-in versions of the regions marked in (a). (d) The correction of the input images in (a) obtained by the texture symmetrization method of [Kim et al. 2012], which is designed for images containing a single texture. Photos courtesy of Angela Sevin (Zebra), and Betty Londergan (Fruit).*

est neighbor search (using PatchMatch [Barnes et al. 2010]). The entire run takes between 3-8 minutes for a 500×500 image, depending on the number of pyramid levels and the size of the image at the finest pyramid level. The performance was measured using our non-optimized MATLAB code[2] on a quad-core machine with 16GB RAM.

7.1 Synthetic Validation

To quantitatively evaluate our method, we performed the following experiments. A ground-truth ideal image J_{GT} was generated by placing the same 32×32 pixel tile in a 5×5 arrangement (see Fig. 4(a)). This image was once geometrically distorted and once color distorted to obtain an 'imperfect' image I. For the geometric experiment, we randomly generated a ground-truth smooth deformation field by low-pass filtering white Gaussian noise. For the color experiment, we randomly generated a smoothly-varying color deformation matrix by adding filtered noise to the identity matrix. Examples of such distorted images are shown in Figs. 4(b),(c), for geometric and color deformations, respectively.

We applied our algorithm on the distorted image I and measured the error between the estimated and the ground truth transformations. As a baseline, we also directly estimated the transformation between J_{GT} and I (note that our algorithm does not have access to J_{GT}). Figure 4 shows the root mean square errors (RMSEs) obtained by both approaches as a function of the amount of deformation[3]. As expected, the error increases with the amount of deformation. Yet, the errors of our method are very close to those obtained by using the ground truth ideal image.

7.2 Real Data

We tested our method on a variety of challenging natural images, which contain multiple textures of different sorts, as well as non-repetitive regions. In all the experiments below, we ran the algorithm on the entire image, without any ROI.

Geometry In geometry mode, we empirically found that our algorithm works well using 3-4 pyramid levels, where the image size

[2]The code and data are publicly available at the project webpage: `http://people.csail.mit.edu/taldekel/NonLocalVariations`.

[3]Averages were computed by repeating the experiment five times for each deformation level (each time randomly generating a different ground-truth deformation).

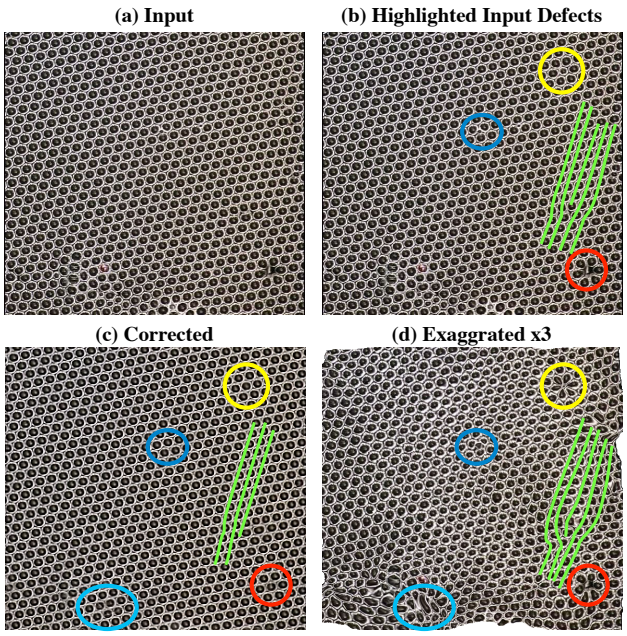

Figure 6: Raft. *An image of a bubble-raft (a), which is an array of bubbles assembled on a surface, used for studying atomic crystalline structures. Some of the various defects are marked in (b), including point-defects (blue, red, and yellow), and line defects (green); see text for more details. (c) The ideal image estimated by our method in which the various irregularities are corrected. (d) Our exaggerated image reveals lattice deformations that are hard to perceive in the input, such as tension and compression. Color markings are for illustration only. Photo courtesy of DoITPoMS [DoI].*

at the finest level is typically half the size of the input image (the estimated flow is up-sampled to the full resolution). We use patches of size 15×15, which remains fixed at all pyramid levels. The choice of patch size stems from our desire that at the coarsest pyramid level, a patch would cover a substantial portion of the largest repeating structure. At this level, the variability between the repeating structures should be still meaningful (the variability is reduced by the down-sampling), which is the reason we work with relatively large patches. This parameter can be modified by the user.

Figure 2(a) shows a line of dancers, performing the same leg-kick. At a first glance, the dancers look very similar. A closer look can reveal some differences between them, such as the pose of the hand shown by the red arrow. Yet, finer geometric variations, such as the spacing between the dancers, their exact poses, and the thickness of their legs, are harder to spot. By using our method, we can exaggerate these variations and make them much more visible, as shown in Fig. 2(b). Alternately, we can correct the variations, thus producing an image in which the dancers are more similar to one another, as shown in Fig. 2(c). These subtle variations become very pronounced when viewing the images of Figs. 2(a),(b),(c) in movie mode (please see the Supplementary Material).

Figures 5(a)–(c) show several additional examples of geometric correction and exaggeration. In the *Zebra* and the *Optical Art* images, our algorithm captures the subtle variability between the stripes. By correcting these variations, we produce stripes with more uniform widths and curvatures, while by exaggerating them we highlight the deviations from equal widths and curvatures. Note that the grass in the Zebra image is not modified, as its recurrence is quite strong to begin with. In the *Bricks*, *Fruit* and *Bottles* images, the effect of our multi-scale framework can be clearly seen. Our method captures variations at fine scales, such as the differences between the shapes of the bricks, the fruits and the bottles' corks. At the same time, it captures variations at coarser scales, such as irregularities in the global arrangement of the bricks and the fruit crates, as well as irregularities in the bottles' widths and heights.

Figure 6(a) shows an image of a bubble-raft, which is an array of soap bubbles assembled on a surface. Bubble rafts are used for modelling and studying the atomic structure of *crystalline solids*. Crystalline materials have a distinctive highly ordered internal structure. However, in reality, most crystalline materials have a variety of defects that can have a tremendous effect on their behavior. Some of these defects are marked in color over the input image in Fig. 6(b), and include *line defects* marked in green and *point defects* such as the absence of an atom (red), the presence of an extra atom (blue), or the replacement of one atom by a different type of atom (yellow). These defects cause global deformations of the lattice (*e.g.,* tension and compression), which are hard to see in the input image. Our method successfully detects, corrects and magnifies these various irregularities, as can be seen in Figs. 6(b),(d). An animation of this result is included in the supplementary materials.

Effect of Parameters In Fig. 7 we demonstrate the effect of the parameter $\boldsymbol{\lambda}$ in Eq. (4) on a *Honey Hive* image. This image contains a repetitive structure that is partly obscured by the bees on the right. When λ is large the 'ideal' image J is constrained to be similar to the deformed input image $\mathcal{T}\{I\}$, which comes at the cost of less repetitiveness in J. Thus, although the bees do not have good NNs, they remain intact when λ is large (*e.g.,* $\lambda = 30$), as can be seen in Fig. 7(b). As λ is decreased, the fidelity to the input image is reduced and the repetitiveness of J is increased. This causes the bees to be washed out ($\lambda = 5$), or even to be completely inpainted by the surrounding repetitive structure (at $\lambda = 1$). Typically, we use $\lambda = 30$ in our experiments in order to avoid the inpainting effect.

In Fig. 8, we demonstrate the effect of the pyramid level on our results. We fixed the patch size to 16×16, and ran our algorithm on a single scale. We then up-sampled the deformation field to produce an exaggerated image at the original resolution. The results obtained for different scales are shown in Fig. 8(b). For each result, we also show the relative size of the 16×16 patch with respect to the pyramid scale (*i.e.,* the size of the recurring structures). At coarse scales, our algorithm captures the deviations of the corn kernels and the bricks from a global regular arrangement. At finer scales, the algorithm captures variations between smaller structures, such as the spacing between corn kernels and the bumps on the bricks.

Comparison to Texture Manipulation Methods Near-regular texture manipulation methods rely on a preliminary lattice detection step [Liu et al. 2004; Eisenacher et al. 2008]. Figure 9 depicts the results of the state-of-art deformed lattice detection method of [Park et al. 2009]. As can be seen, this method fails to detect a global lattice even in images with a clear repetitive pattern. Stochastic texture manipulation methods, such as [Kim et al. 2012], are somewhat less constrained, but still require a single texture throughout the image. Figure 5(d) shows the results attained by the stochastic texture 'symmetrization' algorithm of [Kim et al. 2012]. This method aims to increase the symmetry of the image by 'sharpening' the peaks in its auto-correlation function. As can be seen, this method is not suitable for natural images which contain a variety of textures with complex deformations.

Color When running in color mode, we use small patches (5×5 or 7×7) and operate only on the full resolution image (without

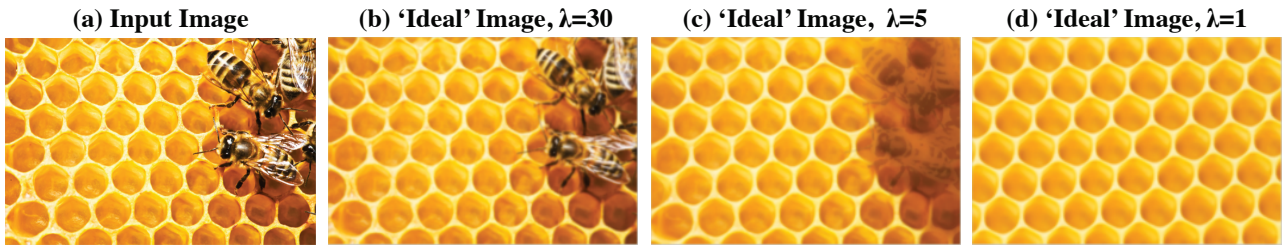

Figure 7: The effect of λ. *(a) Input image. (b),(c),(d) The recovered 'ideal' image J with $\lambda = 30, 5, 1$, respectively. When λ is large, J is constrained to be similar to the transformed input image. Thus, the bees (which do not have good NNs) remain intact in (b). Decreasing the value of λ lowers the weight of the fidelity term with respect to the recurrence term. This causes the bees to be washed out (c) or even to be completely inpainted by the surrounding repetitive structure (d).*

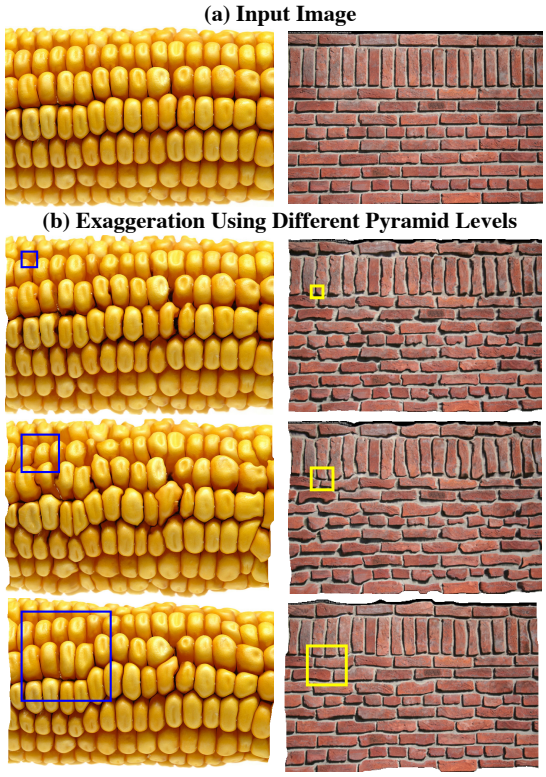

Figure 8: The effect of pyramid level. *(a) Input images. (b) Exaggerated images, each obtained by applying NLV to a* ***different single individual scale****. For all scales,* ***the same patch size*** *(16×16) was used; the relative size of the patch w.r.t. pyramid scale, is marked on the output image (at the original image resolution). Coarser scales capture global deformations in the arrangement of the corn kernels and the bricks. Finer scales capture the variability in the shapes of the individual corn kernels and the bricks (*e.g., *highlighting small bumps on the bricks' contours).*

using a pyramid). We performed the following controlled experiment, shown in Fig. 10. We took a picture of 9 tomatoes, 3 of which were bought the same day, and 6 were bought four days earlier (all from the same supermarket). While it is impossible to see the color differences between the tomatoes in the input image, when we exaggerate the color variations by a factor of 35, the three newer tomatoes become more greenish and the 6 older (riper) tomatoes turn purplish (see Fig. 10(b)). The reason for that is that patches find their NNs non-locally across different tomatoes. An example of a reference patch and the spatial distribution of its NNs is shown in Fig. 10(c).

In the *Lizard* image shown in Fig. 11(a), the lizard and the background contain small repetitive structures that are very similar to one another. Thus, many of the small patches that belong to the lizard find their neighbors in the background, as shown in Fig. 11(c). A close look at these patches reveals slight color differences between the lizard and the background. Our method is able to capture those differences. Then, by magnifying them, we can highlight the camouflaged lizard, revealing that it is more blueish than the background (see Fig. 11(b)). Moreover, the subtle color variations in the background are revealed as well. In the corrected image shown in Fig. 11(d), the color variations between the lizard and the background are reduced.

Note that our color mapping is spatially varying and structure dependent. That is, two patches that have the same color may undergo different color mappings, if their structure is different. Such results could not be obtained by off-the-shelf image processing methods. In particular, we checked techniques such as histogram stretching, histogram equalization, increasing saturation or contrast, increasing deviations from global color-mean, increasing deviations from local color-mean. None of these methods could highlight the color variations spotted by our method.

8 Discussion and Conclusions

We presented a method for revealing and exploring small variations between recurring structures within a single image. Our algorithm recovers a transformation which maximizes the patch recurrences. Once this idealizing transformation is recovered, it can be used to either reduce or exaggerate the nonlocal variations in the image. We specifically focused on recovering geometric deformations and color variations.

Our approach has several limitations. First, since our method is not provided with semantic information, in some cases the 'exaggerated' image may not be visually pleasing. As an example, in

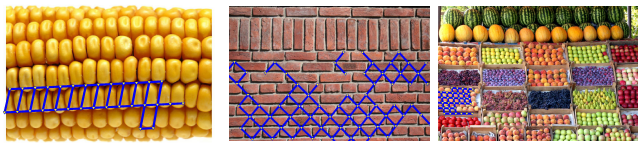

Figure 9: Lattice detection. *The results (marked in blue) obtained by applying the deformed lattice detection method of [Park et al. 2009]. In all the examples, the detected lattice covers only a small portion of the image.*

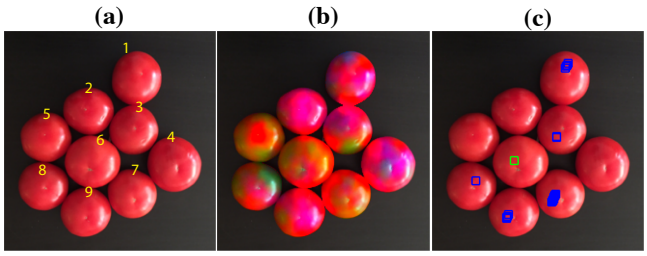

Figure 10: Tomatoes ripening– a controlled experiment. *(a) The input image contains nine tomatoes, a few of which are riper than the others. (b) Color magnification $\times 35$ using our method. Can you group the tomatoes into two groups based on color? The answer is shown below (upsidedown). (c) Example of the non-local nearest neighbors (marked in blue) of one reference patch (marked in yellow).* 7,6,5 - wəN .9,8,4,3,2,1 -plO

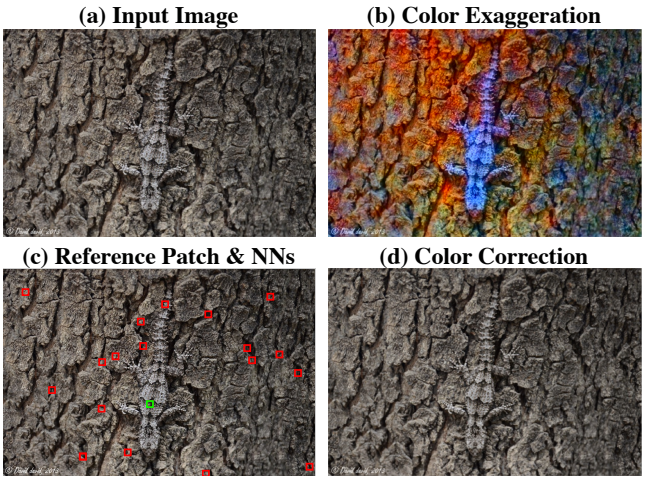

Figure 11: Correcting and magnifying nonlocal color variations. *(a) Input image. (b) The result magnifying the recovered color deviations $\times 25$. Our method captures the reputation in small structures between the lizard and background. (c) example of a reference patch (green) and its NNs (red). (d) The result correcting the color variations; the color of lizard and background is more similar.*

Fig. 2(b), some of the dancers are distorted in an unrealistic way. Therefore, our exaggerated images might be of limited use in applications which require photorealistic results. An additional limitation is that our geometric NLV approach requires strong recurrence of *large structures* (typically 10×10 to 20×20). While small patches (*e.g.,* 5×5) were shown to recur abundantly in almost *any* natural image [Glasner et al. 2009], these recurrences relate to tiny structures (*e.g.,* edges, corners) and not to the kind of structural recurrences we are interested in. In the absence of repetitions of large structures, our method may produce meaningless results. Specifically, when a patch is very distinct and has no good NNs other than itself, it remains unchanged (*e.g.,* the bees in Fig. 7(b)). But when a patch does have a few not-very-good NNs, it may deform in an undesired way. This happens for example in the lower-right hexagon in Fig. 7(b), which is stretched to the right because its color is more similar to the bees than to the rest of the hexagons.

Despite its limitations, our technique has many potential applications. In particular, we demonstrated its use for idealizing images (turning them into more visually pleasing), as well as for revealing interesting properties of the objects in the scene, which often cannot be visually perceived by the naked eye.

Acknowledgments

This work was funded in part by the Israel Science Foundation (Grant 931/14), and by Shell Research.

Appendix A: Derivation of (7)

To compute the gradient of $E_{\text{rec}} + \lambda E_{\text{data}}$ (Eqs. (5) and (6)) with respect to J, we use vector notations. Specifically, let I^{c} and J now denote the column-vector representations of the images $I^{\text{c}}(x,y)$ and $J(x,y)$, respectively. Let $\boldsymbol{Q}_j$ denote the matrix which extracts the j-th $M \times M$ patch from J, and let $\boldsymbol{R}_k$ denote the matrix which extracts the k-th pixel (all three RGB coordinates) from J. Then, writing the data term E_{data} in discrete coordinates (sums instead of integrals), our objective becomes

$$-\sum_j \log\left(\sum_i \exp\left\{-\frac{1}{2h^2}\|\boldsymbol{Q}_j J - q_i\|^2\right\}\right) + \lambda \sum_k \psi\left(\|\boldsymbol{R}_k(I^{\text{c}} - J)\|^2\right),$$

where i runs over all patches q_i in the database. Setting the gradient w.r.t. J of this objective to zero, leads to the requirement that

$$\left(\frac{1}{h^2}\sum_j \boldsymbol{Q}_j^T\boldsymbol{Q}_j + \sum_k \frac{\boldsymbol{R}_k^T\boldsymbol{R}_k}{w_k^{\text{data}}}\right) J = \frac{M^2}{h^2} Z + \sum_k \frac{\boldsymbol{R}_k^T\boldsymbol{R}_k}{w_k^{\text{data}}} I^{\text{c}}, \tag{16}$$

where $w_k^{\text{data}} = \frac{1}{\lambda}\sqrt{\|\boldsymbol{R}_k(I^{\text{c}} - J)\|^2 + \varepsilon^2}$, and

$$Z = \frac{1}{M^2}\sum_j \boldsymbol{Q}_j^T z_j$$

with

$$z_j = \sum_i w_{ij} q_i$$

and

$$w_{ij} = \frac{\exp\left\{-\frac{1}{2h^2}\|\boldsymbol{Q}_j J - q_i\|^2\right\}}{\sum_i \exp\left\{-\frac{1}{2h^2}\|\boldsymbol{Q}_j J - q_i\|^2\right\}}.$$

Up to border effects, $\frac{1}{M^2}\sum_j \boldsymbol{Q}_j^T\boldsymbol{Q}_j$ equals the identity matrix. Therefore, writing (16) in spatial coordinates again, gives

$$\left(\frac{M^2}{h^2} + \frac{1}{W_{\text{data}}(x,y)}\right) J(x,y) = \frac{M^2}{h^2} Z(x,y) + \frac{I^{\text{c}}(x,y)}{W_{\text{data}}(x,y)},$$

where W_{data} is as in (9). Isolating $J(x,y)$, leads to (7).

Note that the image Z is a denoised version of J (using nonlocal means denoising). Indeed, Z is constructed from the patches $\{z_j\}$, while averaging overlaps. Now, each patch z_j is obtained as a combination of all patches in the database, weighted according to their similarity[4] to $\boldsymbol{Q}_j J$. Since the database patches are extracted from J itself, this is precisely the nonlocal means denoising of J.

[4] In fact, most of the weights are typically close to zero, so that z_j is actually a combination of only the nearest-neighbors of $\boldsymbol{Q}_j J$. In practice, we typically work with 30 NNs.

Appendix B: Derivation of (12)

We now derive the minimization of $\lambda E_{\text{data}} + \alpha E_{\text{reg}}$, given in Eqs. (6) and (11), with respect to the color correction matrix $\boldsymbol{C}(x, y)$. This objective can be explicitly written as

$$\iint \Big[\lambda\psi\left(\|\boldsymbol{C}(x,y)I(x,y) - J(x,y)\|^2\right) + \alpha\psi\left(\|\boldsymbol{C}_x(x,y)\|_{\text{F}}^2 + \|\boldsymbol{C}_y(x,y)\|_{\text{F}}^2\right)\Big] dxdy, \quad (17)$$

The optimal $\boldsymbol{C}$ must satisfy the Euler-Lagrange equations

$$\frac{\partial L}{\partial \boldsymbol{C}_{ij}} - \frac{\partial}{\partial x}\frac{\partial L}{\partial (\boldsymbol{C}_x)_{ij}} - \frac{\partial}{\partial y}\frac{\partial L}{\partial (\boldsymbol{C}_y)_{ij}} = 0$$

for all entries i, j of $\boldsymbol{C}$, where L denotes the integrand in (17). Writing the derivatives explicitly leads to the system of partial differential equations (written in matrix form)

$$\frac{\left(\boldsymbol{C}(x,y)I(x,y) - J(x,y)\right) I(x,y)^T}{W_{\text{data}}(x,y)} - \frac{\triangle \boldsymbol{C}(x,y)}{W_{\text{reg}}(x,y)} = 0,$$

with W_{data} and W_{reg} of Eqs. (9) and (14). Omitting, for notational convenience, the coordinates (x, y), the Euler-Lagrange equations take the form

$$\boldsymbol{C}II^T - \frac{W_{\text{data}}}{W_{\text{reg}}}\triangle \boldsymbol{C} = JI^T.$$

The Laplace operator $\triangle$ is implemented in discrete coordinates as $\triangle \boldsymbol{C}(x, y) = \bar{\boldsymbol{C}}(x, y) - \boldsymbol{C}(x, y)$, where $\bar{\boldsymbol{C}}$ is a filtered version[5] of $\boldsymbol{C}$. Therefore,

$$\boldsymbol{C}II^T + \frac{W_{\text{data}}}{W_{\text{reg}}}\boldsymbol{C} = JI^T + \frac{W_{\text{data}}}{W_{\text{reg}}}\bar{\boldsymbol{C}}.$$

Isolating $\boldsymbol{C}$, this equation can be solved iteratively:

$$\boldsymbol{C}^{k+1} = \left(JI^T + \frac{W_{\text{data}}}{W_{\text{reg}}}\bar{\boldsymbol{C}}^k\right)\left(II^T + \frac{W_{\text{data}}}{W_{\text{reg}}}\boldsymbol{Id}\right)^{-1},$$

where $\boldsymbol{Id}$ is the identity matrix and $\bar{\boldsymbol{C}}^k$ is the smoothed version of $\boldsymbol{C}$ from the previous iteration. Using the matrix inversion lemma, this update step can be equivalently written as

$$\begin{aligned}\boldsymbol{C}^{k+1} &= \frac{W_{\text{reg}}}{W_{\text{data}}}\left(JI^T + \frac{W_{\text{data}}}{W_{\text{reg}}}\bar{\boldsymbol{C}}^k\right)\left(\boldsymbol{Id} - \frac{1}{\frac{W_{\text{data}}}{W_{\text{reg}}} + I^TI}II^T\right)\\ &= \bar{\boldsymbol{C}}^k + \frac{W_{\text{reg}}}{W_{\text{data}} + W_{\text{reg}}\|I\|^2}\left(J - \bar{\boldsymbol{C}}^k I\right)I^T,\end{aligned}$$

proving (12).

[5] We use $\bar{\boldsymbol{C}}(x, y) = \frac{1}{4}(\boldsymbol{C}(x+1, y) + \boldsymbol{C}(x-1, y) + \boldsymbol{C}(x, y+1) + \boldsymbol{C}(x, y-1))$.

References

BARNES, C., SHECHTMAN, E., GOLDMAN, D. B., AND FINKELSTEIN, A. 2010. The generalized patchmatch correspondence algorithm. In *ECCV*. Springer, 29–43.

BARNSLEY, M. F., AND SLOAN, A. D., 1990. Methods and apparatus for image compression by iterated function system. US Patent 4,941,193.

BOIMAN, O., AND IRANI, M. 2007. Detecting irregularities in images and in video. *International Journal of Computer Vision 74*, 1, 17–31.

BROX, T., BRUHN, A., PAPENBERG, N., AND WEICKERT, J. 2004. High accuracy optical flow estimation based on a theory for warping. In *ECCV*. Springer, 25–36.

BUADES, A., COLL, B., AND MOREL, J.-M. 2005. A non-local algorithm for image denoising. In *CVPR*, vol. 2, IEEE, 60–65.

CLERC, M., AND MALLAT, S. 2002. The texture gradient equation for recovering shape from texture. *IEEE Transactions on Pattern Analysis and Machine Intelligence 24*, 4, 536–549.

CRIMINISI, A., PÉREZ, P., AND TOYAMA, K. 2004. Region filling and object removal by exemplar-based image inpainting. *IEEE Transactions on Image Processing 13*, 9, 1200–1212.

DABOV, K., FOI, A., KATKOVNIK, V., AND EGIAZARIAN, K. 2007. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on Image Processing 16*, 8, 2080–2095.

DANIELYAN, A., KATKOVNIK, V., AND EGIAZARIAN, K. 2012. Bm3d frames and variational image deblurring. *IEEE Transactions on Image Processing 21*, 4, 1715–1728.

Dissemination of it for the promotion of materials science (doitpoms). `http://www.doitpoms.ac.uk/`.

EFROS, A. A., AND FREEMAN, W. T. 2001. Image quilting for texture synthesis and transfer. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM, 341–346.

EISENACHER, C., LEFEBVRE, S., AND STAMMINGER, M. 2008. Texture synthesis from photographs. In *Computer Graphics Forum*, vol. 27, Wiley Online Library, 419–428.

FREEDMAN, G., AND FATTAL, R. 2011. Image and video upscaling from local self-examples. *ACM Transactions on Graphics (TOG) 30*, 2, 12.

GLASNER, D., BAGON, S., AND IRANI, M. 2009. Super-resolution from a single image. In *ICCV*, IEEE, 349–356.

GOFERMAN, S., ZELNIK-MANOR, L., AND TAL, A. 2012. Context-aware saliency detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence 34*, 10, 1915–1926.

HAYS, J., LEORDEANU, M., EFROS, A. A., AND LIU, Y. 2006. Discovering texture regularity as a higher-order correspondence problem. In *Europian Conference on Computer Vision (ECCV)*. Springer, 522–535.

JOJIC, N., FREY, B. J., AND KANNAN, A. 2003. Epitomic analysis of appearance and shape. In *ICCV*, IEEE, 34–41.

KIM, V. G., LIPMAN, Y., AND FUNKHOUSER, T. A. 2012. Symmetry-guided texture synthesis and manipulation. *ACM Transactions on Graphics (TOG) 31*, 3, 22.

LEBRUN, M., COLOM, M., AND MOREL, J. 2014. The noise clinic: A universal blind denoising algorithm. In *ICIP*, IEEE, 2674–2678.

LIU, Y., LIN, W.-C., AND HAYS, J. 2004. Near-regular texture analysis and manipulation. In *ACM Transactions on Graphics (TOG)*, vol. 23, ACM, 368–376.

LIU, C., TORRALBA, A., FREEMAN, W. T., DURAND, F., AND ADELSON, E. H. 2005. Motion magnification. *ACM Transactions on Graphics (TOG) 24*, 3, 519–526.

LIU, X., JIANG, L., WONG, T.-T., AND FU, C.-W. 2012. Statistical invariance for texture synthesis. *IEEE Transactions on Visualization and Computer Graphics 18*, 11, 1836–1848.

LIU, C. 2009. *Beyond pixels: Exploring new representations and applications for motion analysis*. PhD thesis, Citeseer.

MALIK, J., AND ROSENHOLTZ, R. 1997. Computing local surface orientation and shape from texture for curved surfaces. *International Journal of Computer Vision 23*, 2, 149–168.

MICHAELI, T., AND IRANI, M. 2013. Nonparametric blind super-resolution. In *ICCV*, IEEE, 945–952.

MICHAELI, T., AND IRANI, M. 2014. Blind deblurring using internal patch recurrence. In *ECCV*.

MISHNE, G., AND COHEN, I. 2013. Multiscale anomaly detection using diffusion maps. *IEEE Journal of Selected Topics in Signal Processing 7*, 1, 111–123.

PARK, M., BROCKLEHURST, K., COLLINS, R. T., AND LIU, Y. 2009. Deformed lattice detection in real-world images using mean-shift belief propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence 31*, 10, 1804–1816.

SEO, H. J., AND MILANFAR, P. 2009. Static and space-time visual saliency detection by self-resemblance. *Journal of vision 9*, 12, 15.

SHANKAR, N., AND ZHONG, Z. 2005. Defect detection on semiconductor wafer surfaces. *Microelectronic Engineering 77*, 3, 337–346.

SIMAKOV, D., CASPI, Y., SHECHTMAN, E., AND IRANI, M. 2008. Summarizing visual data using bidirectional similarity. In *CVPR*, IEEE.

WADHWA, N., RUBINSTEIN, M., DURAND, F., AND FREEMAN, W. T. 2013. Phase-based video motion processing. *ACM Trans. Graph.*.

WADHWA, N., DEKEL, T., WEI, D., DURAND, F., AND FREEMAN, W. T. 2015. Deviation magnification: Revealing geometric deviation in a single image. In *SIGGRAPH Asia*.

WU, H., RUBINSTEIN, M., SHIH, E., GUTTAG, J. V., DURAND, F., AND FREEMAN, W. T. 2012. Eulerian video magnification for revealing subtle changes in the world. *ACM Trans. Graph.*.

ZONTAK, M., AND COHEN, I. 2010. Defect detection in patterned wafers using anisotropic kernels. *Machine Vision and Applications 21*, 2, 129–141.