Learning Factorized Weight Matrix for Joint Filtering

Xiangyu Xu [1] **Yongrui Ma** [2] **Wenxiu Sun** [3]

Abstract

Joint filtering is a fundamental problem in computer vision with applications in many different areas. Most existing algorithms solve this problem with a weighted averaging process to aggregate input pixels. However, the weight matrix of this process is often empirically designed and not robust to complex input. In this work, we propose to learn the weight matrix for joint image filtering. This is a challenging problem, as directly learning a large weight matrix is computationally intractable. To address this issue, we introduce the correlation of deep features to approximate the aggregation weights. However, this strategy only uses inner product for the weight matrix estimation, which limits the performance of the proposed algorithm. Therefore, we further propose to learn a nonlinear function to predict sparse residuals of the feature correlation matrix. Note that the proposed method essentially factorizes the weight matrix into a low-rank and a sparse matrix and then learn both of them simultaneously with deep neural networks. Extensive experiments show that the proposed algorithm compares favorably against the state-of-the-art approaches on a wide variety of joint filtering tasks.

1. Introduction

Joint image filtering is a fundamental problem in computer vision which enhances an input image (*e.g.*, a noisy depth map) by exploiting the information from a paired guidance image (*e.g.*, a clear RGB image). It has broad applications in different areas, such as depth restoration (Ham et al., 2018), depth upsampling (Park et al., 2011), image matting (Levin et al., 2007), image colorization (Levin et al., 2004), natural image denoising (Buades et al., 2005), human segmentation (Xu et al., 2018), optical flow estimation (Sun et al., 2010; Su et al., 2019), texture removal (Xu et al., 2011; Li et al., 2019), super-resolution (Xu et al., 2019a), cross-domain restoration (Yan et al., 2013), and light field reconstruction (Zheng et al., 2018).

While existing algorithms use different tools to solve the joint image filtering problem, they mostly share the same basic remark that the filtering is achieved in a weighted averaging process (Tomasi & Manduchi, 1998; Buades et al., 2005; Kopf et al., 2007; Park et al., 2011; He et al., 2013; Zhang et al., 2014a;b; Ham et al., 2018): $\mathbf{y} = W(\mathbf{x}, \mathbf{g})\mathbf{x}$, where $\mathbf{x} \in \mathbb{R}^m$ and $\mathbf{g} \in \mathbb{R}^m$ respectively represent the input and guidance images that both have m pixels and are reshaped into vectors column-wisely. $\mathbf{x}$ and $\mathbf{g}$ are both given beforehand according to the application and can be identical (He et al., 2013; Buades et al., 2005). $\mathbf{y} \in \mathbb{R}^m$ is the filtered output image. $W \in \mathbb{R}^{m\times m}$ is the weight matrix of the image filtering process, which is a function of the input and the guidance image (Ham et al., 2018; Li et al., 2019; Park et al., 2011; He et al., 2013). The i-th row of W represents the weights for aggregating all the pixels in the input image $\mathbf{x}$ to generate the i-th pixel of the output $\mathbf{y}$.

The strategies of deciding W are the key factors to distinguish different joint filtering approaches. For example, the bilateral filter (Tomasi & Manduchi, 1998; Kopf et al., 2007) constructs a weight matrix using spatially-variant Gaussian kernels. The non-local means algorithm (Buades et al., 2005; Zhang et al., 2014a) aggregates global information of the guidance image and derives a dense weight matrix for filtering. In addition, the optimization-based methods (Ferstl et al., 2013; Ham et al., 2018) exploits the global structures by minimizing a fidelity function, which involves solving a large linear system and can also be seen as weighted averaging with an inverse weight matrix. Although achieving impressive results, existing approaches design the weight matrix W with hand-crafted features (Park et al., 2011; Kopf et al., 2007) or empirical priors (Ferstl et al., 2013; Ham et al., 2018), which are not robust in complex scenarios and can lead to artifacts when the features are not effective or the priors are violated for certain samples.

With the rapid advances of deep learning, convolutional neural networks (CNNs) have been used for joint image filtering (Xu et al., 2015; Hui et al., 2016; Li et al., 2019), which can learn to regress the desired output by absorbing

[1]Carnegie Mellon University, Pittsburgh, PA, USA [2]SenseTime, Beijing, China [3]SenseTime, Hong Kong. Correspondence to: Xiangyu Xu <xuxiangyu2014@gmail.com>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

knowledge from a large amount of training data and outperform the classical algorithms. These models can also be seen as an implicit form of weighted averaging as they usually stack a couple of convolution layers, which essentially combines several Toeplitz matrices to approximate the latent weight matrix (Gray, 2006). As the learned convolution kernels are fixed and not dependent on $\mathbf{x}$ and $\mathbf{g}$, nonlinear activation functions are applied between convolution layers such that the learned deep network can achieve desirable effects for different pixels of different input images.

In this paper, we use the data-driven models in a different way and propose to explicitly learn sample-dependent weight matrix for joint image filtering. However, this is a challenging task as the weight matrix has a dimension of m^2, and the ultra-high dimensionality makes the estimation computationally intractable. For example, the weight matrix of a 256 $\times$ 256 input image has more than 6e4 $\times$ 6e4 entries, which is too large to be directly generated by a neural network. To alleviate this problem, we introduce the feature correlation (Gan et al., 2018) to estimate the relationship between different pixels to approximate the weight matrix W. However, this strategy only uses a simple and fixed function (*i.e.*, inner product) for computing the aggregation weights. For better approximating W, we further propose to learn a nonlinear function with neural networks to predict sparse residuals for improving the weights of the feature correlation matrix.

The proposed method is in spirit similar to the Robust PCA (Candès et al., 2011), which factorizes W into a low-rank matrix and a sparse matrix. The low-rank matrix is able to capture global information of the image, and the sparse matrix can better exploit local image structures. In this manner, the proposed method can effectively learn the latent weight matrix W and achieve high-quality results for joint image filtering. Different from previous approaches (Candès et al., 2011; Chen et al., 2011) which achieve the factorization by minimizing the nuclear norm and the ℓ_1 norm of factorized matrices, we encode the low-rank and sparse constraints in a specifically-designed neural network.

We make the following contributions in this work. First, we explicitly learn the weighted averaging model of joint image filtering and solve the high-dimensionality problem by using the correlation of deep features. We also introduce an efficient way for its computation. Second, we propose to learn a nonlinear function to estimate residuals for a subset of the entries in the weight matrix. We use neural networks to predict the locations of the non-zero entries of the sparse residual matrix, which is able to better exploit the local structures and thus distribute the learned sparse residuals more effectively. Third, we show that the proposed algorithm is essentially similar to the Robust PCA which factorizes a large matrix into a low-rank and a sparse matrix which could be more easily handled due to the special matrix structures. Extensive experiments on different benchmark datasets demonstrate that the proposed method compares favorably against the state-of-the-art approaches on a wide variety of joint image filtering tasks.

2. Related Work

Most classical joint image filtering algorithms use heuristic strategies to construct the weight matrix for pixel aggregation (Tomasi & Manduchi, 1998; Kopf et al., 2007; Buades et al., 2005; He et al., 2013; Zhang et al., 2014a). As a typical example, the bilateral filtering (Tomasi & Manduchi, 1998; Kopf et al., 2007) uses a weight matrix by designing spatially-variant Gaussian kernels, which can reduce noise and remain edges in the output. However, it only captures local information and cannot exploit global structures of the image. To solve this problem, the non-local means (Buades et al., 2005; Zhang et al., 2014a) uses a dense weight matrix to aggregate pixels globally for better filtering performance. Nevertheless, these methods often use simple and hand-crafted features to decide the weight, such as color similarity, spatial location, and super-pixel. In contrast, we propose to learn a dense weight matrix from large amount of image data, which can benefit from more powerful deep features.

On the other hand, optimization-based methods have been proposed for joint image filtering, which mostly rely on empirical smoothness priors and can be seen as an implicit weighted averaging process with a inverse weight matrix (Park et al., 2011; Ferstl et al., 2013; Ham et al., 2018). To properly use the smoothness prior, Park *et al*. propose a large-neighborhood regularization term to protect the thin structures of the filtered image (Park et al., 2011). However, the regularization function only constrains the first-order derivative of the output and thus favors constant results in smooth image regions. To solve this problem, Ferstl *et al*. (Ferstl et al., 2013) apply the second-order total generalized variation prior for piece-wise smoothness of the output. Further, Ham *et al*. (Ham et al., 2018) introduce the dynamic guidance (*i.e.*, the intermediate result) in the regularization term, such that the smoothness constraint can be relaxed for some outliers. However, these methods are not robust for complex scenarios and tend to generate artifacts when the empirical priors are violated.

Recently, deep CNNs have also been used for joint image filtering (Xu et al., 2015; Hui et al., 2016; Li et al., 2019; Pan et al., 2019; Su et al., 2019). Most of this kind of methods (Xu et al., 2015; Hui et al., 2016; Li et al., 2019) treat the problem as a general regression task similar to other computer vision problems, such as monocular depth (Eigen et al., 2014) and optical flow estimation (Dosovitskiy et al., 2015). These approaches learn to regress the desired output with stacked convolutional layers, which essentially combines several Toeplitz matrices with nonlinear activation functions to approximate the latent weight matrix (Gray, 2006). However, these methods use the same convolution kernels for different spatial locations and different input images, which does not well meet the requirement of the problem for spatially-variance and input-dependence. Different from the above algorithms, most recent works (Su et al., 2019; Pan et al., 2019; Mildenhall et al., 2018; Xu et al., 2020) directly learn spatially-variant kernels for joint image filtering. However, these methods only consider local information, and the global structures are largely neglected. By contrast, we propose to explicitly learn the weighted averaging process which is both spatially-variant and input-dependent and can effectively exploit both local and global information.

Closely related to our work, the Robust PCA factorizes an observable matrix into a low-rank and a sparse matrix by minimizing the nuclear norm and the ℓ_1 norm of the targets with Principal Component Pursuit (Candès et al., 2011). Different from it, we achieve the factorization of a latent weight matrix by learning with deep neural networks.

3. Algorithm

In this work, we propose to explicitly learn the weight matrix W for joint image filtering. Since the latent matrix is ultra-high dimensional, it is difficult to directly predict it with a neural network. To solve this problem, we introduce the feature correlation (Gan et al., 2018; Fu et al., 2019) to estimate the relationship between different pixels to approximate the weight matrix W. While this feature correlation strategy is able to achieve impressive results for joint image filtering, it only uses a simple and fixed function (*i.e.*, inner product) for computing the aggregation weights. For better approximating W, we further propose to learn a nonlinear function to predict residuals to improve the weights of the feature correlation matrix. Since it is computationally intractable to predict residuals for all pixels with a neural network, we only estimate the residuals for a small set of pixels, and the estimated values are added to the feature correlation matrix for final image filtering. The above process is in spirit similar to the Robust PCA (Candès et al., 2011) which factorizes W into a low-rank matrix L and a sparse matrix S. As the latent matrix W cannot be observed, we propose to learn both L and S from data to approximate it. Detailed explanations about the learning process are presented as follows.

3.1. Feature correlation

As explained in Section 1, the weight W_{ij} represents the relationship between two pixels i and j, and decides how much the pixel j contributes to the output at pixel i. And a simple and effective method to represent the relationship between two pixels is the feature correlation (Gan et al., 2018): $\phi(\mathbf{x},\mathbf{g})_i \cdot \phi(\mathbf{x},\mathbf{g})_j$, where "$\cdot$" denotes inner product, and ϕ is a feature extractor implemented as a CNN in our model. $\phi(\mathbf{x},\mathbf{g}) \in \mathbb{R}^{m\times d}$ represents the features extracted from the input and the guidance image, where ϕ_i (the i-th row of $\phi(\mathbf{x},\mathbf{g})$) represents a d-dimensional feature vector of pixel i. Since $d \ll m$, ϕ can be efficiently learned with modern deep learning tools (Abadi et al., 2016).

Different from (Gan et al., 2018) which only correlates features in a local region, we compute the feature correlation globally, and the obtained correlation matrix forms the low-rank part of our model:

$$L = [\phi(\mathbf{x},\mathbf{g})U][\phi(\mathbf{x},\mathbf{g})V]^\top, \tag{1}$$

where $U \in \mathbb{R}^{d\times d}$ and $V \in \mathbb{R}^{d\times d}$ are two learnable matrices that transform the original vectors to new feature spaces for more flexible filtering effects. $[\cdot]$ represents ReLU activation function (Nair & Hinton, 2010) which ensures the weights are nonnegative. We also normalize the matrix L such that each row sums to one. Note that (1) is in spirit similar to the self-attention model of (Zhang et al., 2019; Wang et al., 2018; Fu et al., 2019) where the softmax function is used to estimate dense affinity matrices, and thereby the low-rank property cannot be guaranteed.

3.2. Learning sparse residuals

While the learned feature correlation matrix L can capture the relationship between different pixels, the entries of L are computed by a simple and fixed function, *i.e.*, the inner product of feature vectors. To more accurately approximate the latent aggregation weights, we can learn a nonlinear function $\tilde{\eta}$ to predict weight residuals for improving the entries of L:

$$S_{ij} = \tilde{\eta}(\varphi_i, \varphi_j), \tag{2}$$

where φ is also a feature extractor similar to ϕ, and $\tilde{\eta}$ can be modeled with a multi-layer neural network taking the feature vectors as input.

However, to estimate the residuals for all weights in L is computationally intractable. Therefore, for each pixel i (or the i-th row of S), we only predict residuals for a subset of the pixels denoted as $\mathcal{D}(i)$, and for any pixel $j \notin \mathcal{D}(i)$, $S_{ij} = 0$. That is to say, we can use the learned function $\tilde{\eta}$ to improve a subset of the entries of the low-rank matrix L, which correspond to the non-zero entries of S.

Since the local structures (He et al., 2013; Kopf et al., 2007) are critical in joint image filtering, a straightforward way to decide $\mathcal{D}(i)$ is to sample a rigid neighborhood of pixel i, *e.g.*, a 3×3 patch centered at i, and then the residuals of the aggregation weights can be estimated for the pixels in the rigid region.

Nevertheless, the neighboring pixels relevant to image filtering often lies in irregular shape along edges and image structures, and the rigid sampling strategy does not exploit this information, which thereby cannot effectively obtain the most informative pixels. Instead of using a rigid image patch, we learn an adaptive sampling strategy with a CNN κ to search for the most informative pixels for joint image filtering. In other words, we use κ to learn the locations of the non-zero entries of the sparse matrix S.

The network also takes the image features φ as input, and the output $\kappa(\varphi) \in \mathbb{R}^{m \times 2k}$ is the predicted sampling locations for all the non-zero entries in S, where $k \ll m$. Specifically, we can rewrite the i-th row of the matrix $\kappa_i \in \mathbb{R}^{2k}$ as $\{(x_{it}, y_{it}) : t = 1, 2, \ldots, k)\}$ which represents the k sampling locations of pixel i in the image coordinate system. Since we column-wisely reshape the images into vectors, a pixel (x_{it}, y_{it}) in the image coordinate system corresponds to $(x_{it} - 1)h + y_{it}$ in the vector where h denotes the height of the image. Thus, the pixel subset $\mathcal{D}(i)$, where the weight residuals are estimated for pixel i, can be formulated as $\mathcal{D}(i) = \{(x_{it} - 1)h + y_{it} : t = 1, 2, \ldots, k\}$.

Note that x_{it} and y_{it} are not integers. Thus, we do not directly compute $S_{i,(x_{it}-1)h+y_{it}}$ for the sparse matrix S. Instead, we need to estimate the integer entries of S surrounding (x_{it}, y_{it}):

$$\begin{aligned}\mathcal{N}(i,t) = \{&S_{i,\lfloor x_{it}-1\rfloor h+\lfloor y_{it}\rfloor}, \quad S_{i,\lfloor x_{it}-1\rfloor h+\lfloor y_{it}+1\rfloor}, \\ &S_{i,\lfloor x_{it}\rfloor h+\lfloor y_{it}\rfloor}, \qquad S_{i,\lfloor x_{it}\rfloor h+\lfloor y_{it}+1\rfloor}\}.\end{aligned}$$

However, we cannot simply use (2) to estimate the above entries in $\mathcal{N}(i,t)$, as this function (*e.g.*, $S_{i,\lfloor x_{it}-1\rfloor h+\lfloor y_{it}\rfloor} = \tilde{\eta}(\phi_i, \phi_{\lfloor x_{it}-1\rfloor h+\lfloor y_{it}\rfloor})$) is not differentiable with respect to x_{it} and y_{it}, and thereby cannot be used to train the network κ for learning the suitable locations of the non-zero entries.

Instead, we introduce the bilinear-weight method which has been used in the optical flow literature (Sun et al., 2018; Xu et al., 2019b) to transform geometrical locations into coefficients of the weight values. Specifically, we replace $\tilde{\eta}$ with a new function η which takes the whole feature matrix φ as input, and the output $\eta(\varphi) \in \mathbb{R}^{m \times k}$ is used to predict the latent values of non-interger locations of S, *i.e.*, $S_{i,(x_{it}-1)h+y_{it}} = \eta_{it}$. Then we distribute this latent value to all the surrounding entries in $\mathcal{N}(i,t)$ with bilinear weights. For example, for the first integer location $\lfloor x_{it} - 1\rfloor h + \lfloor y_{it}\rfloor$, we have:

$$\begin{aligned}S_{i,\lfloor x_{it}-1\rfloor h+\lfloor y_{it}\rfloor} = &(1 - (x_{it} - \lfloor x_{it}\rfloor))\cdot \\ &(1 - (y_{it} - \lfloor y_{it}\rfloor))S_{i,(x_{it}-1)h+y_{it}}, \end{aligned} \tag{3}$$

where the entry is given larger weight when it is closer to the predicted sampling location $(x_{it}-1)h+y_{it}$. The estimation of the other entries in $\mathcal{N}(i,t)$ can be performed similarly. Note that (3) is differentiable to the predicted locations x_{it} and y_{it}, and thus κ can also be trained together with η.

3.3. More details and explanations

With the estimated matrices L and S, the final filtered result can be easily obtained by $\mathbf{y} = L\mathbf{x} + S\mathbf{x}$.

While the estimation of image features can be fast and computationally efficient with modern deep learning tools (Abadi et al., 2016), directly performing the image filtering with L (*i.e.*, $L\mathbf{x}$) still requires the computation and storage of a large matrix of size m^2, and the computational complexity is $(d+1)m^2 + 2d^2m$ float-point operations (FLOPs)[1]. However, as the dimension of the feature vector is usually much smaller than the image resolution (*i.e.*, $d \ll m$), we can significantly reduce the complexity of this problem by alternatively changing the order of the matrix multiplications. According to (1), we can compute the lower-dimensional matrix multiplications first and then the complexity of $L\mathbf{x}$ will become as small as $2dm + 2d^2$ FLOPs. This computation reduction essentially relies on the low-rank property of L.

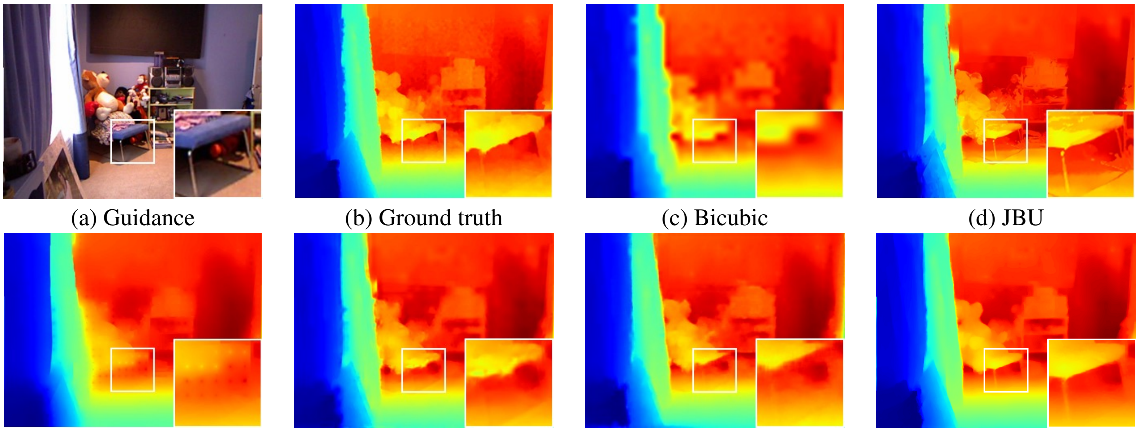
Since for each row of S the number of non-zero entries is no larger than $4k$, we have support$(S) \leq 4km$. To estimate $S\mathbf{x}$, we only need to consider the locations with non-zero residuals, and thus the computational complexity for the sparse part is no larger than $4km$ FLOPs.

As we have rank$(L) \leq d$ and support$(S) \leq 4km$, the proposed algorithm is essentially similar to the Robust PCA, where W is factorized into a low-rank and a sparse matrix. Eventually, the low-rank matrix aggregates global information of the image for joint image filtering, and the sparse matrix further learns a nonlinear function to improve the inner product and refine the neighboring weights for better exploiting local image structures.

Network structure. We use the encoder-decoder structure (Ronneberger et al., 2015; Mildenhall et al., 2018) for extracting image features, where the batch normalization (Ioffe & Szegedy, 2015) and ReLU activation (Nair & Hinton, 2010) are applied between convolutional layers similar to (He et al., 2016). We ensure the output has the same spatial resolution as the input by modifying the padding parameter of the convolutional layer. Note that we do not use batch normalization for φ as we empirically find it leads to unstable performance in training. We also use convolutional layers with ReLU for η and κ. We use (Glorot & Bengio, 2010) for initializing the networks.

Suppose we have an image dataset $\{\mathbf{x}^{(i)}, \mathbf{g}^{(i)}, \tilde{\mathbf{y}}^{(i)}\}_{i=1}^N$ where $\tilde{\mathbf{y}}^{(i)}$ is the ground truth, and the output of our network is $\mathbf{y}^{(i)}$. The proposed network is trained with the following mean squared error (MSE) loss: $\frac{1}{N}\sum_i \|\mathbf{y}^{(i)} - \tilde{\mathbf{y}}^{(i)}\|_2^2$. $\mathbf{x}, \mathbf{g}, \mathbf{y}$ can be different for different tasks, which are respec-

[1] The definition of FLOPs follows (Zhang et al., 2018), *i.e.*, the number of multiply-adds.



(a) Guidance (b) Ground truth (c) Bicubic (d) JBU

(e) Ham (f) DJF (g) PAC (h) Ours

Figure 1. Qualitative comparison for depth upsampling. Example from the test set of NYUv2 (Nathan Silberman & Fergus, 2012) with an upsampling factor of $8\times$.

tively specified in each experiment below.

4. Experiments

We evaluate the proposed method on four different tasks of joints image filtering, *i.e.*, depth upsampling, optical flow upsampling, depth denoising, and natural image denoising.

4.1. Implementation

In all the experiments, we set the feature dimension as $d = 64$ for the low-rank matrix, and the number of sampling locations as $k = 9$ for the sparse matrix. During training, we use the Adam optimizer (Kingma & Ba, 2015) with learning rate 1e-4. We randomly crop 256×256 patches from the input and the guidance images, and use a batch size of 20. During the test phase, we chop the whole input into overlapped patches and process each patch separately to save memory usage. The reconstructed patches are then placed back to the corresponding locations and averaged in overlapped regions similar to (Schuler et al., 2013).

4.2. Depth map upsampling

The main problem of existing depth sensors is that they often capture the depth map at a low resolution, which is caused by chip size limitations, such as the Time-of-Flight (ToF) camera (Park et al., 2011). Joint depth upsampling aims to super-resolve the low-resolution depth map $\mathbf{x}$ to obtain a high-resolution one $\mathbf{y}$ with the guidance of a paired high-resolution intensity image $\mathbf{g}$.

To generate training data, we use the NYUv2 depth dataset (Nathan Silberman & Fergus, 2012) which consists of 1449 image/depth pairs. Following the protocols of (Li et al., 2019), we use 1000 data pairs for training and the rest for testing. Similar to (Li et al., 2019), we generate the low-resolution depth maps from the ground truth using nearest-neighbor downsampling with a factor of $4\times$, $8\times$, and $16\times$ respectively. The low-resolution map is first upsampled with bicubic interpolation before fed into the neural network.

We provide both quantitative and qualitative evaluations of the proposed algorithm against the state-of-the-art joint depth upsampling approaches, including MRF (Diebel & Thrun, 2006), GF (He et al., 2013), JBU (Kopf et al., 2007), Ham (Ham et al., 2018), DMSG (Hui et al., 2016), FBS (Barron & Poole, 2016), DJF (Li et al., 2019), and PAC (Su et al., 2019).

Table 1 shows the quantitative results in terms of the root mean squared errors (RMSE). For the baseline methods, we use the default parameters in the original implementations. The proposed algorithm performs well against the state-of-the-art methods across all three upsampling factors.

For more intuitive study, we present a visual example from the test dataset in Figure 1. The JBU (Kopf et al., 2007) uses hand-crafted features which are agnostic to structural consistency between the target and guidance images, and thus transfers erroneous details as shown in Figure 1(d). The Ham (Ham et al., 2018) algorithm relies on empirical prior to build a non-convex model, where the prior can be violated in complex scenarios and the non-convex optimization can result in local minimum solution. As shown in Figure 1(d), it produces oversmoothing artifacts and inaccurate depth. The deep learning based methods, *i.e.*, DJF (Li et al., 2019) and PAC (Su et al., 2019), can exploit large amount of training data and generate much better results as shown in Figure 1(f) and (g). However, they still cannot effectively restore high-quality depth maps, especially around the edges and fine details, as they do not explicitly learn the spatially-variant weighted averaging process and cannot exploit the

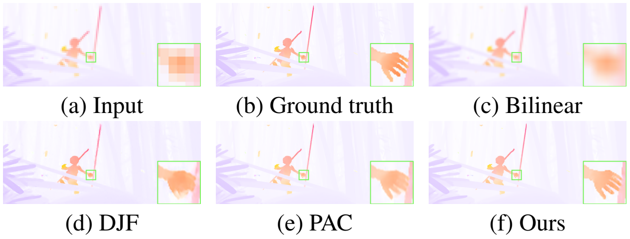


Figure 2. Qualitative evaluation for optical flow upsampling. Example from the Sintel dataset (Butler et al., 2012) with an upsampling factor of $8\times$.

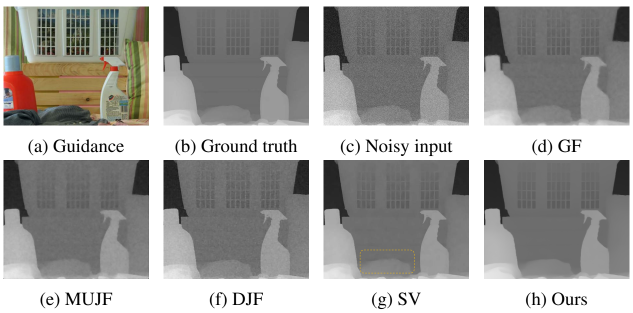


Figure 3. Qualitative comparison for depth denoising. Example from the benchmark dataset (Lu et al., 2014).

global structures which are important for reducing depth ambiguity (Eigen et al., 2014). In contrast, the results of our algorithm are smoother, sharper and more accurate with respect to the ground truth.

4.3. Optical flow upsampling

Optical flow estimation is an important task in computer vision, and it is often time-consuming and memory-intensive to compute the high-resolution flow map directly. Most existing approaches (Sun et al., 2018; Dosovitskiy et al., 2015) perform the optical flow estimation at a low resolution and then upsample the flow map with simple interpolation functions, such as bilinear and bicubic upsampling. As the intensity images can provide important structure information for the optical flow, especially around motion boundaries, it is desirable to apply joint image filtering to the task of optical flow upsampling. The input $\mathbf{x}$ here is the low-resolution optical flow map, and the guidance $\mathbf{g}$ is the high-resolution image.

Table 1. Quantitative evaluations of the joint depth upsampling task for upsampling factors $4\times$, $8\times$ and $16\times$ in terms of RMSE.

Methods	$4\times$	$8\times$	$16\times$
Bicubic	8.16	14.22	22.32
MRF (Diebel & Thrun, 2006)	7.84	13.98	22.20
GF (He et al., 2013)	7.32	13.62	22.03
JBU (Kopf et al., 2007)	4.07	8.29	13.35
Ham (Ham et al., 2018)	5.27	12.31	19.24
DMSG (Hui et al., 2016)	3.78	6.37	11.16
FBS (Barron & Poole, 2016)	4.29	8.94	14.59
DJF (Li et al., 2019)	3.38	5.86	10.11
PAC (Su et al., 2019)	2.39	4.59	8.09
Ours	**2.16**	**4.32**	**7.66**

For this task, we experiment with the Sintel dataset (Butler et al., 2012). The quantitative result in Table 2 indicates that our method is effective for joint optical flow upsampling and can achieve state-of-the-art performance in terms of the End-Point-Error (EPE). We also qualitatively compare our method against the baselines in Figure 2. While the PAC (Su et al., 2019) algorithm can learn spatially-variant and data-dependent kernels, it does not generate as good flow fields as ours due to that it uses fixed simple functions (*e.g.*, Gaussian kernel) to estimate the relationship between different pixels and only considers information within a limited local regions. In contrast, we learn a non-linear function η to improve the weight matrix and can effectively exploit both global and local structures for processing the motion fields. As shown in Figure 2(f), the optical flow produced by our method is more accurate than the baselines.

4.4. Joint depth denoising

The depth map obtained by ranging sensors can be affected by acquisition noise, *e.g.*, when the active illumination energy is limited for ToF cameras. Similar to the joint depth upsampling, the joint image filtering techniques can also be used for depth denoising, where the input $\mathbf{x}$ is the noisy depth map, and the guidance $\mathbf{g}$ is the clear intensity image.

For training the depth denoising model, we also use the NYUv2 depth dataset (Nathan Silberman & Fergus, 2012) described in Section 4.2 . We add Gaussian noise with noise level ranging from 0 to 26 to each ground truth depth map similar to (Pan et al., 2019). Following the protocols of (Pan et al., 2019), we evaluate the proposed method using the test set of (Lu et al., 2014), where Gaussian noise with noise level 20 is added to each test image.

We compare our method with the state-of-the-art depth denoising approaches: GF (He et al., 2013), JBU (Kopf et al., 2007), MUJF (Shen et al., 2017), MUGIF (Guo et al., 2018), DJF (Li et al., 2019), and SV (Pan et al., 2019).

As shown in Table 3, the proposed algorithm can achieve consistently better results than the baselines in terms of

Table 2. Quantitative evaluation for optical flow upsampling with upsampling factors $4\times$, $8\times$, and $16\times$ in terms of EPE.

Methods	$4\times$	$8\times$	$16\times$
Bilinear	0.465	0.901	1.628
DJF (Li et al., 2019)	0.176	0.438	1.043
PAC (Su et al., 2019)	0.105	0.256	0.592
Ours	**0.096**	**0.236**	**0.548**

Table 3. Quantitative evaluations for the joint depth denoising task on the benchmark dataset (Lu et al., 2014) in terms of PSNR, SSIM and RMSE.

Metrics	GF	JBU	MUJF	MUGIF	DJF	SV	Ours
PSNR	30.79	26.08	30.67	34.07	32.58	36.44	**37.74**
SSIM	0.9214	0.7820	0.9282	0.9657	0.9016	0.9762	**0.9807**
RMSE	7.75	12.96	7.76	5.26	6.14	4.02	**3.43**

all the evaluation metrics: PSNR, SSIM, and RMSE. We also present a visual example in Figure 3. The classical methods GF (He et al., 2013) and MUJF (Shen et al., 2017) use simple features to construct the weight matrix which cannot effectively remove heavy depth noise as shown in Figure 3(d) and (e). DJF (Li et al., 2019) learns fixed convolution kernels to regress the final output, which relies on very deep structures and high nonlinearity to approximate a spatially-variant and input-dependent solution. However, it may suffer from the overfitting problem which can lead to severe artifacts when the test image is significantly different from the training dataset as shown in Figure 3(f). The SV approach (Pan et al., 2019) estimates spatially-variant kernels and thus can better generalize to challenging test data and generate more accurate result in Figure 3(g). Nevertheless, the predicted kernels of SV are relatively small (1×1), which limits its ability to exploit pixels from a larger region. In contrast, our method explicitly aggregates global information by learning a spatially-variant weight matrix and achieves higher-quality results as shown in Figure 3(h).

4.5. Natural image denoising

We also apply the proposed method to natural image denoising where the input $\mathbf{x}$ and guidance $\mathbf{g}$ are identical, *i.e.*, the noisy intensity image. We adopt the MIRFLICKR 25K dataset (Huiskes & Lew, 2008) for training, which consists of 24550 images after data cleaning. We use two popular benchmarks for evaluation, *i.e.*, Set12 and BSD68 (Dabov et al., 2007) with noise levels of 15, 25, and 50.

We quantitatively and qualitatively evaluate the proposed approach against the state-of-the-art methods including BM3D (Dabov et al., 2007), WNNM (Gu et al., 2014), TNRD (Chen & Pock, 2016), DnCNN (Zhang et al., 2017), and NLRN (Liu et al., 2018). As shown in Table 4, our method compares favorably against the baselines on different noise levels, which demonstrates the effectiveness of the proposed strategy for learning the weight matrix.

For qualitative evaluation, we present an example from Set12 with noise level 50 in Figure 4. The state-of-the-art approaches are not effective in recovering image details and produce oversmoothing artifacts in Figure 4(c)-(d). In contrast, the proposed algorithm employs the low-rank matrix and the sparse matrix to jointly filter the noisy input image, which can better exploit the global and local information. Hence, the sharp edges and the fine details can be well recovered under severe image noise as shown in Figure 4(e).

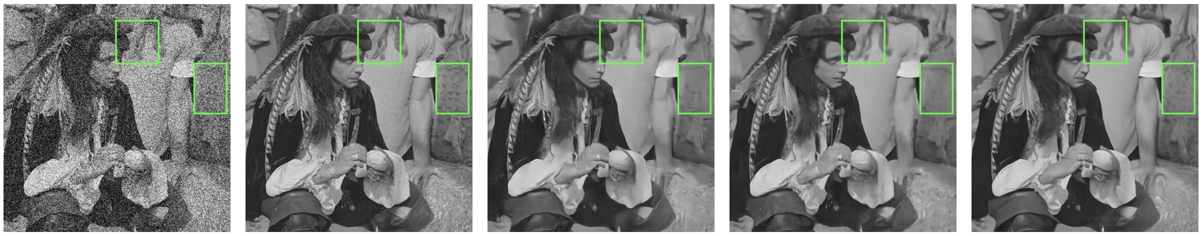
4.6. Ablation study

We conduct the ablation study on the joint depth upsampling task as shown in Table 5. A simple variant of our method is to only use the low-rank matrix L for image filtering without learning the sparse residuals. As shown in Table 5, this approach (*i.e.*, the first row) does not perform as well as the models with η (*i.e.*, the second and third rows), which demonstrates the effectiveness of learning a nonlinear function to refine the entires of the low-rank matrix. Furthermore, the proposed method learns the locations of the sparse entries of S with a neural network κ, which can more effectively exploit the local image structures such that the function η can be applied to more informative pixels. The second row of Table 5 shows that the model without learning κ is also inferior to our full model.

4.7. Visualization of the learned matrix

As discussed in Section 3.3, the proposed method is similar to the Robust PCA (Candès et al., 2011), where the latent matrix W is factorized as a low-rank matrix L and a sparse matrix S. For better understanding of this process, we show an example from the joint depth upsampling and visualize the factorized matrices in Figure 5. Whereas we only show the matrices of a small image patch, the low-rank property of L can already be easily identified according to the simple patterns. This may be due to the fact that image data have low intrinsic dimensionality (Belkin & Niyogi, 2003). For the sparse matrix, the non-zero entries mostly lie around the main diagonal and other diagonals, which demonstrates that the learned sparse residuals mainly exploit the local structures for improving L. Since we reshape the images into vectors in a column-wise manner, the number of non-zero diagonals in S indicates the horizontal range that the learned sparse locations span. The vertical range can be similarly represented by row-wisely reshaping the image. As shown in Figure 5(b), the learned sparse matrix can exploit pixels from an approximately 11×11 pixels region while the number of the sampling locations is much smaller ($k = 9$ in this work). This also explains the importance and effectiveness of learning the sampling strategy η.

In addition, we also show a row of the weight matrix to vi-



(a) Input (b) Ground truth (c) DnCNN (d) NLRN (e) Ours

Figure 4. Qualitative evaluation for natural image denoising. Example from the Set12 with noise level of 50.

Table 4. Quantitative evaluation for the natural image denoising task on Set12 and BSD68 in terms of PSNR and SSIM.

Dataset	Noise level	BM3D	WNNM	TNRD	DnCNN	NLRN	Ours
Set12	15	32.37/0.8952	32.70/0.8982	32.50/0.8958	32.86/0.9031	33.16/0.9070	**33.18/0.9096**
	25	29.97/0.8504	30.28/0.8557	30.06/0.8512	30.44/0.8622	30.80/0.8689	**30.88/0.8726**
	50	26.72/0.7676	27.05/0.7775	26.81/0.7680	27.18/0.7829	27.64/0.7980	**27.83/0.8071**
BSD68	15	31.07/0.8717	31.37/0.8766	31.42/0.8769	31.73/0.8907	31.88/0.8932	**31.89/0.8958**
	25	28.57/0.8013	28.83/0.8087.	28.92/0.8093	29.23/0.8278	29.41/0.8331	**29.46/0.8377**
	50	25.62/0.6864	25.87/0.6982	25.97/0.6994	26.23/0.7189	26.47/0.7298	**26.56/0.7374**

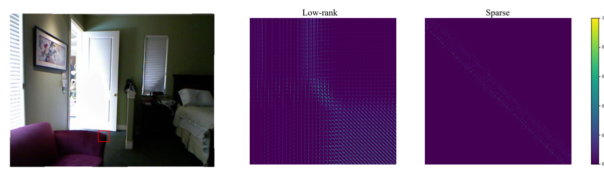
Table 5. Ablation study of the proposed model for joint depth upsampling with scale factors $4\times$, $8\times$ and $16\times$ in terms of RMSE.

Methods	$4\times$	$8\times$	$16\times$
w/o learning η and κ	2.50	4.57	7.99
w/o learning κ	2.21	4.43	7.81
ours full model	**2.16**	**4.32**	**7.66**

sualize the learned weighted averaging process in the image coordinate system. Specifically, we sample a pixel in the red square region of Figure 5(a), which is shown as the yellow point in Figure 6(b). Then we show the correlation weights between the yellow pixel and all the pixels in the image in Figure 6(a), which correspond to a row of the low-rank matrix L. Note that the locations with similar depth as the sampled yellow pixel have higher response in Figure 6(a), which demonstrates that the proposed method can effectively aggregate informative pixels globally for the image filtering. Moreover, Figure 6(b) shows the learned sampling locations and weight residuals for the yellow pixel, which represents the non-zero entries on a row of S. The sparse entries mostly lie along the object boundaries to better exploit local information and thus can effectively help improve the joint image filtering performance.

5. Conclusions

In this work, we propose to explicitly learn the weighted averaging process for joint image filtering. We first exploit the feature correlation to alleviate the ultra-high dimensionality issue of the weight matrix. We further propose to learn sparse residuals to improve the correlation matrix. The proposed learning process is similar to the Robust PCA where the weight matrix is factorized into a low-rank and a sparse matrix. We provide comprehensive evaluation and analysis



(a) guidance image (b) learned matrices

Figure 5. Visualizing the learned matrix factorization. As the weight matrix is too large, we only show the factorized matrices of a 40×40 image patch (red square in (a)). The values in the matrices are normalized for better visualization.

(a) row of L (b) row of S

Figure 6. Visualizing a row of the weight matrix. Brighter color represents higher response in (a), and darker red indicates higher weights in (b). See the text for more explanations.

of the proposed method, and demonstrate the effectiveness of our approach on a wide variety of joint filtering tasks.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, 2016.

Barron, J. T. and Poole, B. The fast bilateral solver. In

ECCV, 2016.

Belkin, M. and Niyogi, P. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.

Buades, A., Coll, B., and Morel, J.-M. A non-local algorithm for image denoising. In *CVPR*, 2005.

Butler, D. J., Wulff, J., Stanley, G. B., and Black, M. J. A naturalistic open source movie for optical flow evaluation. In *ECCV*, 2012.

Candès, E. J., Li, X., Ma, Y., and Wright, J. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3): 1–37, 2011.

Chen, Y. and Pock, T. Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. *TPAMI*, 39(6):1256–1272, 2016.

Chen, Y., Xu, H., Caramanis, C., and Sanghavi, S. Robust matrix completion and corrupted columns. In *ICML*, 2011.

Dabov, K., Foi, A., Katkovnik, V., and Egiazarian, K. Image denoising by sparse 3-d transform-domain collaborative filtering. *TIP*, 16(8):2080–2095, 2007.

Diebel, J. and Thrun, S. An application of markov random fields to range sensing. In *NeurIPS*, 2006.

Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Van Der Smagt, P., Cremers, D., and Brox, T. Flownet: Learning optical flow with convolutional networks. In *ICCV*, 2015.

Eigen, D., Puhrsch, C., and Fergus, R. Depth map prediction from a single image using a multi-scale deep network. In *NIPS*, 2014.

Ferstl, D., Reinbacher, C., Ranftl, R., Rüther, M., and Bischof, H. Image guided depth upsampling using anisotropic total generalized variation. In *ICCV*, 2013.

Fu, J., Liu, J., Tian, H., Li, Y., Bao, Y., Fang, Z., and Lu, H. Dual attention network for scene segmentation. In *CVPR*, 2019.

Gan, Y., Xu, X., Sun, W., and Lin, L. Monocular depth estimation with affinity, vertical pooling, and label enhancement. In *ECCV*, 2018.

Glorot, X. and Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, 2010.

Gray, R. M. Toeplitz and circulant matrices: A review. *Foundations and Trends in Communications and Information Theory*, 2(3):155–239, 2006.

Gu, S., Zhang, L., Zuo, W., and Feng, X. Weighted nuclear norm minimization with application to image denoising. In *CVPR*, 2014.

Guo, X., Li, Y., Ma, J., and Ling, H. Mutually guided image filtering. *TPAMI*, 2018.

Ham, B., Cho, M., and Ponce, J. Robust guided image filtering using nonconvex potentials. *TPAMI*, 40(1):192–207, 2018.

He, K., Sun, J., and Tang, X. Guided image filtering. *TPAMI*, 35:1397–1409, 2013.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *CVPR*, 2016.

Hui, T.-W., Loy, C. C., and Tang, X. Depth map super-resolution by deep multi-scale guidance. In *ECCV*, 2016.

Huiskes, M. J. and Lew, M. S. The mir flickr retrieval evaluation. In *MIR '08: Proceedings of the 2008 ACM International Conference on Multimedia Information Retrieval*, 2008.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Kopf, J., Cohen, M. F., Lischinski, D., and Uyttendaele, M. Joint bilateral upsampling. *ACM Transactions on Graphics*, 26(3):96, 2007.

Levin, A., Lischinski, D., and Weiss, Y. Colorization using optimization. *ACM Transactions on Graphics (SIGGRAPH)*, 2004.

Levin, A., Lischinski, D., and Weiss, Y. A closed-form solution to natural image matting. *TPAMI*, 30(2):228–242, 2007.

Li, Y., Huang, J.-B., Ahuja, N., and Yang, M.-H. Joint image filtering with deep convolutional networks. *TPAMI*, 41 (8):1909–1923, 2019.

Liu, D., Wen, B., Fan, Y., Loy, C. C., and Huang, T. S. Non-local recurrent network for image restoration. In *NeurIPS*, 2018.

Lu, S., Ren, X., and Liu, F. Depth enhancement via low-rank matrix completion. In *CVPR*, 2014.

Mildenhall, B., Barron, J. T., Chen, J., Sharlet, D., Ng, R., and Carroll, R. Burst denoising with kernel prediction networks. In *CVPR*, 2018.

Nair, V. and Hinton, G. E. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.

Nathan Silberman, Derek Hoiem, P. K. and Fergus, R. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

Pan, J., Dong, J., Ren, J. S., Lin, L., Tang, J., and Yang, M.-H. Spatially variant linear representation models for joint filtering. In *CVPR*, 2019.

Park, J., Kim, H., Tai, Y.-W., Brown, M. S., and Kweon, I. High quality depth map upsampling for 3d-tof cameras. In *ICCV*, 2011.

Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015.

Schuler, C., Burger, H., Harmeling, S., and Scholkopf, B. A machine learning approach for non-blind image deconvolution. In *CVPR*, 2013.

Shen, X., Zhou, C., Xu, L., and Jia, J. Mutual-structure for joint filtering. *IJCV*, pp. 19–33, 2017.

Su, H., Jampani, V., Sun, D., Gallo, O., Learned-Miller, E., and Kautz, J. Pixel-adaptive convolutional neural networks. In *CVPR*, 2019.

Sun, D., Roth, S., and Black, M. J. Secrets of optical flow estimation and their principles. In *CVPR*, 2010.

Sun, D., Yang, X., Liu, M.-Y., and Kautz, J. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *CVPR*, 2018.

Tomasi, C. and Manduchi, R. Bilateral filtering for gray and color images. In *ICCV*, 1998.

Wang, X., Girshick, R., Gupta, A., and He, K. Non-local neural networks. In *CVPR*, 2018.

Xu, L., Lu, C., Xu, Y., and Jia, J. Image smoothing via L0 gradient minimization. *ACM Transactions on Graphics (SIGGRAPH)*, 30(6):174, 2011.

Xu, L., Ren, J., Yan, Q., Liao, R., and Jia, J. Deep edge-aware filters. In *ICML*, 2015.

Xu, X., Sun, D., Liu, S., Ren, W., Zhang, Y.-J., Yang, M.-H., and Sun, J. Rendering portraitures from monocular camera and beyond. In *ECCV*, 2018.

Xu, X., Ma, Y., and Sun, W. Towards real scene super-resolution with raw images. In *CVPR*, 2019a.

Xu, X., Siyao, L., Sun, W., Yin, Q., and Yang, M.-H. Quadratic video interpolation. In *NeurIPS*, 2019b.

Xu, X., Li, M., Sun, W., and Yang, M.-H. Learning spatial and spatio-temporal pixel aggregations for image and video denoising. *TIP*, 29:7153–7165, 2020.

Yan, Q., Shen, X., Xu, L., Zhuo, S., Zhang, X., Shen, L., and Jia, J. Cross-field joint image restoration via scale map. In *ICCV*, 2013.

Zhang, H., Goodfellow, I., Metaxas, D., and Odena, A. Self-attention generative adversarial networks. In *ICML*, 2019.

Zhang, J., Cao, Y., and Wang, Z. A new image filtering method: Nonlocal image guided averaging. In *ICASSP*, 2014a.

Zhang, K., Zuo, W., Chen, Y., Meng, D., and Zhang, L. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *TIP*, 26(7):3142–3155, 2017.

Zhang, Q., Shen, X., Xu, L., and Jia, J. Rolling guidance filter. In *ECCV*, 2014b.

Zhang, X., Zhou, X., Lin, M., and Sun, J. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *CVPR*, 2018.

Zheng, H., Ji, M., Wang, H., Liu, Y., and Fang, L. Crossnet: An end-to-end reference-based super resolution network using cross-scale warping. In *ECCV*, 2018.