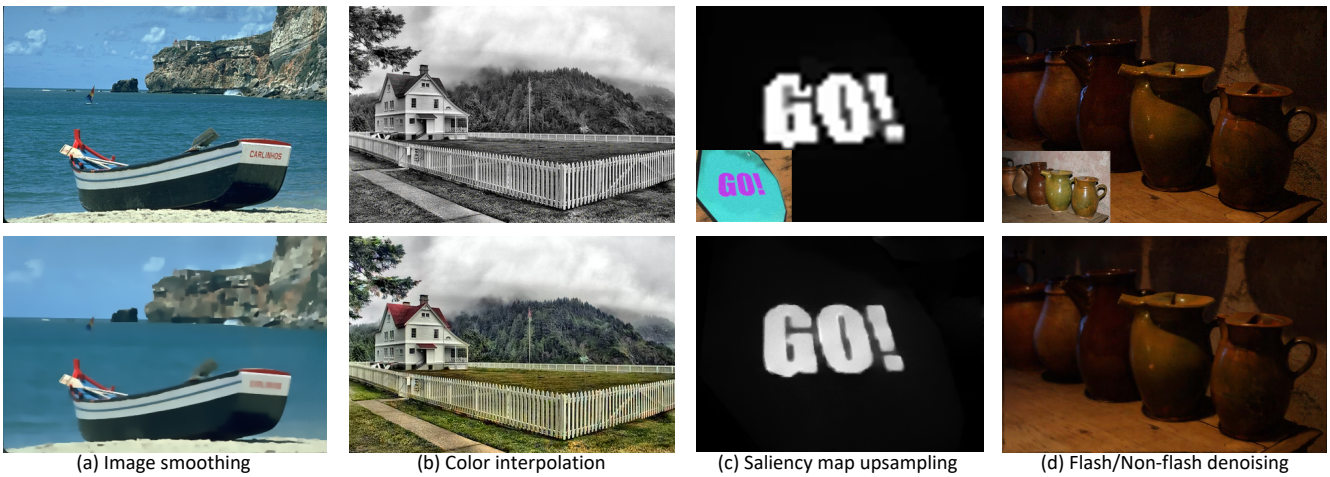
DOI: 10.1111/cgf.13827
Pacific Graphics 2019
C. Theobalt, J. Lee, and G. Wetzstein
(Guest Editors)

Volume 38 (2019), Number 7

Learning Explicit Smoothing Kernels for Joint Image Filtering

Xiaonan Fang[1], Miao Wang[2], Ariel Shamir[3] and Shi-Min Hu[1]

[1]BNRist, Department of Computer Science and Technology, Tsinghua University, Beijing
[2]State Key Laboratory of Virtual Reality Technology and Systems, Beihang University
[3]The Interdisciplinary Center, Herzliya



(a) Image smoothing (b) Color interpolation (c) Saliency map upsampling (d) Flash/Non-flash denoising

Figure 1: *We introduce a deep learning model that learns explicit image smoothing kernels from paired unsmoothed and smoothed images. With the learned image smoothing kernels, our method can perform image smoothing (a). Moreover, the merit of the proposed method lies in joint image filtering for various image processing tasks such as color interpolation (b), saliency map upsampling (c) and flash/non-flash image denoising (d) using the model learned from (a).*

Abstract

Smoothing noises while preserving strong edges in images is an important problem in image processing. Image smoothing filters can be either explicit (based on local weighted average) or implicit (based on global optimization). Implicit methods are usually time-consuming and cannot be applied to joint image filtering tasks, i.e., leveraging the structural information of a guidance image to filter a target image.Previous deep learning based image smoothing filters are all implicit and unavailable for joint filtering. In this paper, we propose to learn explicit guidance feature maps as well as offset maps from the guidance image and smoothing parameter that can be utilized to smooth the input itself or to filter images in other target domains. We design a deep convolutional neural network consisting of a fully-convolution block for guidance and offset maps extraction together with a stacked spatially varying deformable convolution block for joint image filtering. Our models can approximate several representative image smoothing filters with high accuracy comparable to state-of-the-art methods, and serve as general tools for other joint image filtering tasks, such as color interpolation, depth map upsampling, saliency map upsampling, flash/non-flash image denoising and RGB/NIR image denoising.

CCS Concepts

• ***Computing methodologies*** → ***Image processing;***

1. Introduction

Edge-preserving image smoothing aims to remove noises and small variation in images while retaining the strong edges and structures. Edge-preserving filters have been extensively studied and have a large variety of applications such as image denoising, cartoonization, detail enhancement, HDR tone mapping and joint image filtering. Joint image filtering uses a guidance map to filter a target image, for example, filtering a low resolution depth image to make its edge aligned with the high resolution color image, or propagating sparse data into the whole image domain according to the boundary of the guidance image. The structural information of the guidance map is transferred to the target image by a joint filter. In this paper, we propose a method to use deep learning to learn image smoothing kernels with controllable parameters that are also capable of joint image filtering.

Edge-preserving image smoothing filters can be divided into two categories: explicit methods and implicit methods. Given an input image I, explicit methods compute the smoothed image S by weighted averaging pixel values in the neighborhood of each pixel p:

$$S(p) = \sum_{q \in N(p)} w(p,q) I(q), \tag{1}$$

where $N(p)$ represents the neighborhood of pixel p and the weights are normalized, i.e., $\sum_{q \in N(p)} w(p,q) = 1$. The weight $w(p,q)$ evaluates the affinity between pixel p and q. Typical explicit methods include bilateral filter [TM98], region covariance method [KEE13], and bilateral texture filter [CLKL14]. Other methods such as guided filter [HST10], domain transform [GO11] and tree filter [BSY*14; ZDXZ15] may have different computational strategies, but they could be viewed as weighted average methods in essence. Explicit methods can be applied to joint image filtering if the affinity term $w(\cdot,\cdot)$ is computed on a guidance image G different from I.

Implicit methods formulate image smoothing as a global optimization problem that minimizes

$$E(S) = E_d(S,I) + \lambda E_v(S), \tag{2}$$

where $E_d(S,I)$ is a data term measuring the difference between input I and output S, while the smooth term $E_v(S)$ penalizes the color variations in output S with parameter λ balancing these two terms. Representative methods include total variation-L_1 [ROF92], weighted least square (WLS) [FFLS08], L_0 optimization [XLXJ11], relative total variation (RTV) [XYXJ12] and L_1 optimization [BHY15]. Note that although WLS is a global optimization based method, it has a closed-form solution using matrix multiplication, so it is also available for joint filtering. However, most optimization based methods can only smooth the input image itself because there is not an explicit form for the solution.

Many image smoothing filters are time-consuming because of the complexity of optimization or the feature extraction step. With the development of deep neural networks, researchers have proposed several strategies to approximate existing smoothing filters with acceleration [XRY*15; LPY16; FYH*17], or to directly learn new deep filters [LYB18; FYW*18]. Nevertheless, current deep neural networks based filters are totally end-to-end black-box models, making it impossible to jointly filter images in other domains. The motivation of this work is to exploit the power of image smoothing algorithms for joint filtering applications under a deep learning framework. Gharbi et al. [GCB*17] proposed an architecture that can approximate many kinds of image operators, but it requires training on every specific task.

Some approaches proposed to directly learn a joint filter using convolutional neural networks (CNNs). Li et al. [LHAY19] trained CNN models for joint depth map upsampling and joint depth map denoising respectively. However, these models are limited to a specific scenario, e.g., $8\times$ upsampling, and hence cannot well generalize to other tasks. Moreover, their method cannot be adapted to perform sparse data interpolation.

In this work we do not learn the smoothed output directly, but rather hypothesize that the output can be obtained according to Equ. (1) and learn the kernel weights $w(\cdot,\cdot)$ from a CNN model. Our strategy is similar to the kernel prediction method [JDTG16], which has been applied to denoising [MBC*18; BVM*17; VRM*18]. However, directly applying Equ. (1) requires a large neighborhood, so we instead filter the image by stacked deformable convolution layers [DQX*17; ZHLD18] with small kernel sizes. We extract feature maps and offset maps from the guidance image with a CNN, and define the kernel weight $w(p,q)$ as the affinity between feature vectors of pixel p and q. These weights can then be applied to filter other target images. Following Fan et al. [FCY*18], we learn image filters with adjustable input parameters that control the strength of smoothing. Figure 2 illustrates the pipeline of our approach. At the training stage the target and guidance images are identical while for usage they might differ. Our model can not only smooth the guidance image itself but also filter images from other target domains using the extracted feature maps and offset maps.

We choose 3 representative image smoothing methods for evaluation: weighted least square [FFLS08], L_0 optimization [XLXJ11], relative total variation [XYXJ12], and using the same network we can learn a model for each of these methods. Experiments show that our method can achieve comparable approximation accuracy with state-of-the-art approaches. We also apply the method to other joint image filtering tasks: color interpolation, depth map upsampling, saliency map upsampling, flash/non-flash denoising and RGB/NIR image denoising. Results demonstrate that the learned image smoothing filters are useful for these joint filtering tasks.

In summary, the main contributions of this paper are:

- Proposing a CNN framework to learn explicit kernels of image smoothing filter consisting of a sampling strategy and a weighting function that can replicate the effects of existing optimization based image smoothing filters;
- Exploiting the learned explicit kernels for joint image filtering tasks using the same model and parameters, which previous deep image smoothing filters could not do.

2. Related Work

In this section, we will introduce the representative explicit and implicit image smoothing methods. Then we briefly review joint image filtering algorithms and spatially variant filters in deep neural

networks. Finally we illustrate the development of image smoothing filters learning with convolutional neural networks.

2.1. Explicit Image Smoothing Methods

Filtering an image with Gaussian kernel can smooth the image content with spatially invariant strength, which might leads to the destruction of visually prominent structures. Bilateral filter (BF) [TM98] preserves the strong edges by combining spatial distance and range distance when applying Equ. (1):

$$w(p,q) = \exp(-||p-q||^2/\sigma_s^2) \cdot \exp(-|||I(p)-I(q)||^2/\sigma_r^2). \quad (3)$$

Rolling guidance filter [ZSXJ14] applies bilateral filter recursively by using the output of the latest iteration as guidance image. Researchers have designed other hand-crafted features to compute the weights $w(\cdot,\cdot)$, such as region covariance method [KEE13] and bilateral texture filter [CLKL14]. Anisotropic diffusion [PM90] is another spatially varying filter that propagates pixel values according to local gradients. Guided filter [HST10] estimates a linear transformation on local patches and then averages them. Domain transform [GO11] maps the image manifold into a planar space and then smooths the region within small geodesic distance. The tree filter [BSY*14] uses a minimal spanning tree to define the inter-pixel distance. The segment graph based filter [ZDXZ15] improves the tree filter by constructing a local MST on each superpixel. Since the above methods have an explicit form to directly compute the weighted average of input image, they are available for joint image filtering.

2.2. Implicit Image Smoothing Methods

Image smoothing can also be formalized as a global optimization problem, as described in Equ. (2). Total variation method penalizes the L_1 norm of the gradient on the output image [ROF92]. Weighted least square (WLS) method [FFLS08] defines both the data term and smooth term in quadratic form so that the global minimum could be found by solving a linear system. Efforts have been made to reduce the complexity of WLS by solving the linear system on one or a few rows/columns [MCL*14; LCS*17]. Least-squares images [WCL*15] have zero inhomogeneous Laplacian and approximate the original image in a least-squares sense. L_0 method [XLXJ11] defines the smooth term as the L_0 norm of image gradients, providing results with sharp edges. A non-local concentration regularization term could be added to improve the quality of L_0 optimization [LZGZ15]. Relative total variation (RTV) method [XYXJ12] was designed to separate textures from structures, which is useful to smooth the textures for mosaic images. L_1 optimization [BHY15] can be used to smooth the image and to estimate the reflectance and shading map. Most implicit methods do not have a closed-form solution, and hence are not available for joint image filtering. Besides, the optimization-based methods are usually time-consuming.

2.3. Joint Image Filtering

Filtering a target image with guidance from another image is known as joint image filtering. The explicit image smoothing filters can be applied for joint image filtering. Among them the most popular methods are joint bilateral upsampling [KCLU07] and guided filter [HST10]. Joint filtering can also be formulated as an optimization problem, as shown in [DT06; PKT*11]. Ham et al. [HCP15] proposed to use both the static and dynamic guidance for filtering. The mutual structures between target and guidance image could be utilized for filtering [SZXJ15]. Li et al. [LHAY19] proposed deep joint filter (DJF), a CNN framework that extracts structures from both the target and guidance image for joint upsampling and joint denoising tasks. Our work differs from Li et al. [LHAY19] in two aspects. Firstly our goal is to approximate existing image smoothing filters rather than to directly learn a joint filtering model for one specific task. Secondly we learn controllable parameters so that our model is a general tool available for a large variety of tasks including color interpolation, upsampling and denoising, while Li et al. [LHAY19] learned one model for upsampling, one model for denoising, and neither of them can perform color interpolation.

2.4. Spatially Variant Filters in Deep Neural Networks

Traditional deep neural networks use spatially invariant convolution layers. Jia et al. [JDTG16] proposed to predict a dynamic convolution layer or a dynamic local filtering layer from input data and apply the layer on the input. The latter strategy, namely kernel prediction method, has been used for denoising bursts of images [MBC*18] and denoising Monte Carlo renderings [BVM*17; VRM*18]. Deformable convolutional network [DQX*17] predicts the sampling locations of the convolution operator and it was further improved by modulating the input feature amplitudes from different spatial locations [ZHLD18]. We adopt deformable convolution in this work.

2.5. Learning Image Smoothing Filters with CNN

With the development of deep convolutional neural networks, researchers have been engaged in learning image smoothing filters with CNN to approximate previous filters, to improve time efficiency or to enhance the smoothing quality. Xu et al. [XRY*15] proposed to learn the output gradient field for reconstruction. Liu et al. [LPY16] designed a linear RNN architecture to learn a recursive filter on internal feature maps. Fan et al. [FYH*17] proposed a cascaded CNN that estimates the edge map of the output before predicting the output itself, which could improve the accuracy. They subsequently extended the network structure such that it can receive variable smoothing parameters as input [FCY*18]. They also developed an unsupervised method to optimize an extremely complex energy function using CNN [FYW*18]. In addition, Lu et al. [LYB18] generated artificial images by mixing cartoon images and pure textures to learn a model for texture removal. Though current deep learning methods have achieved impressive approximation accuracy, they all perform image smoothing in a black box that are neither explainable nor available for joint filtering, meaning that the applications of the learned filters are highly limited: any learned filter is implicit, no matter whether its original form is explicit or implicit. Besides, Gharbi et al. proposed Deep Bilateral Learning [GCB*17] to accelerate a wider range of single image enhancement operators, but in that work one model can only work on one task, so their model has limited capability. The goal

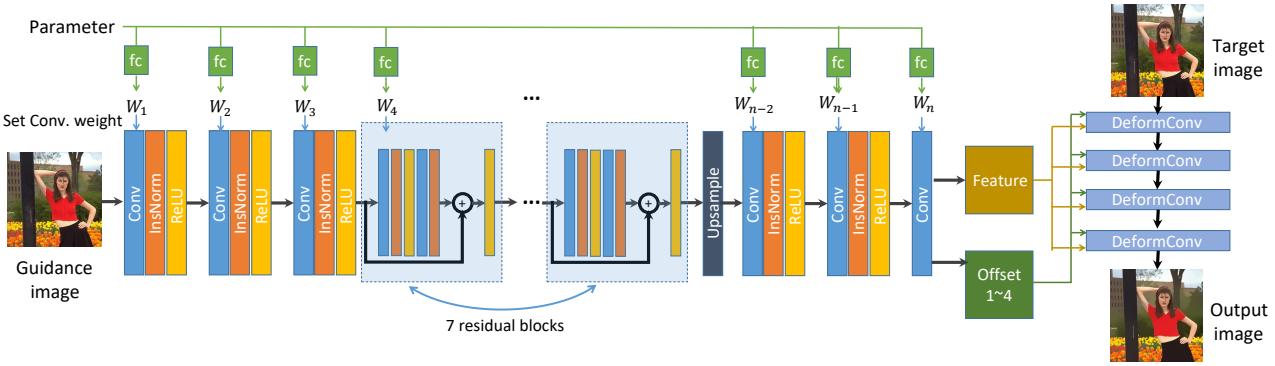


Figure 2: *Pipeline of our method. First, we use a fully convolutional block to extract features and offsets from the guidance image. The convolution weights are obtained from a fully connected layer that receives smoothing parameter as input. The second stage, composed of 4 deformable convolution layers, has no learnable parameters but utilizes the feature maps and offset maps given by previous step to filter the target image.*

The sampling locations are determined by offset maps and convolution weights are defined as the normalized inter-pixel affinity on the feature maps. The model can perform joint filtering tasks when the target image and guidance image come from different domains.

of this work is to build the deep learning model in an explicit form, making it a more powerful and general tool.

3. Learning Explicit Image Smoothing Kernels

This work borrows the idea from deep learning for parameterized image operators [FCY*18] as well as deformable CNN [DQX*17]. As illustrated in Figure 2, our model has two parts. The first part is a feature extractor that receives a guidance image together with smoothing parameters and outputs a series of feature maps and offset maps with the same resolution as input. The second part performs deformable convolution [DQX*17] on the target image 4 times with weights defined by the affinity between the feature vectors and the sampling locations offered by 4 groups of offset maps. We train the model on image smoothing task, in which the guidance image and the target image are identical. But for usage we can feed data from different domains to achieve joint filtering.

3.1. Feature Extractor

The feature extractor is supposed to predict feature maps and offset maps from the guidance image and input parameter. It begins with three convolution layers, followed by 7 residual blocks and ends with another three convolution layers. Before entering the residual blocks the image resolution is reduced with a convolution stride of 2 and after the residual blocks the feature maps are upsampled back to the original resolution using nearest interpolation. Inside the residual blocks the dilation varies from 1 to 16 to enlarge the reception field. We use instance normalization and ReLU activation after every convolution layer except the last one.

The guidance input has $C+2$ channels consisting of C color channels and 2 channels of x, y coordinates. For example, since WLS [FFLS08] method processes each color channel separately, we set $C=1$ when learning this filter, and set $C=3$ when learning L_0 optimization [XLXJ11] and RTV filter [XYXJ12]. All of the intermediate layers have 64 channels. Unlike conventional convolutional neural networks, the weights in each convolution layer are predicted by a linear layer using the smoothing parameter as input, as illustrated at the top of Figure 2. The weights of the k-th convolution layer, W_k, is obtained by:

$$W_k = A_k P + B_k, \tag{4}$$

where P is the input parameter and A_k, B_k are learnable parameters. In all of the previous image-smoothing algorithms mentioned above, there is a parameter λ to balance the data term and smoothness term, as described in Equ. (2). In this work, we only test the case of 1 variable parameter, though it is possible to extend our model for multiple input parameters within our framework. A larger input parameter indicates higher smoothing strength in the original smoothing algorithms. We use both the parameter and its reciprocal as the input for the linear layer, i.e., $P = (\lambda, 1/\lambda)^T$. Under this setting the convolution kernels can adapt to variable smoothing strength according to the input parameter. The last convolution layer outputs $d_F = 5$ feature maps F and 72 offset maps O, which will be used in the following filtering process. Both of them have the same resolution as the guidance image.

3.2. Stacked Deformable Convolution Block

We implement the weighted average operator (Equ. (1)) with deformable convolution layers at this stage. Image filtering usually requires computation in a large neighborhood. Most explicit image smoothing methods use a large square patch, which our deep learning framework cannot afford. Instead, we use a few small kernels to filter the image step by step. To further enlarge the sample range, we leverage the idea of deformable convolution from [DQX*17; ZHLD18]. Different from regular convolution that samples values from a regular grid, a deformable convolution determines the sampling locations according to offset maps. Let O^x_{ij}, O^y_{ij} be the offset maps on x and y dimension. For a 3×3 kernel, the sampling loca-

Figure 3: *Results of image smoothing. (a) Input image. (b, d, f) output of original method. (c, e, g) output of our network. Top row:* L_0, $\lambda = 0.005, 0.01, 0.02$*; second row: RTV,* $\lambda = 0.0045, 0.0100, 0.0224$*; third row: WLS,* $\lambda = 0.45, 1.00, 2.24$.

		WLS					L_0					RTV		
λ	Liu	Gharbi	Fan	Ours	λ	Liu	Gharbi	Fan	Ours	λ	Liu	Gharbi	Fan	Ours
0.20	36.05	31.17	45.19	44.17	0.002	35.65	38.17	43.22	44.61	0.0020	35.13	31.57	42.21	40.40
0.45	34.45	29.36	44.61	44.67	0.005	32.98	33.28	41.04	40.98	0.0045	33.56	28.42	42.50	41.01
1.00	32.87	28.24	43.48	44.45	0.010	31.31	30.57	39.05	38.67	0.0100	31.38	26.88	42.51	41.45
2.24	30.89	27.59	42.18	44.26	0.020	30.90	28.03	37.05	36.58	0.0224	29.05	25.97	41.67	41.06
5.00	29.70	27.65	40.57	43.42	0.050	26.17	25.60	34.12	34.16	0.0500	27.19	26.17	39.38	39.29
Avg.	32.79	28.80	43.20	44.19	Avg.	31.40	31.13	38.90	39.00	Avg.	31.26	27.80	41.65	40.64

Table 1: *Average PSNR of the learned filter tested on BSDS dataset [MFTM01]. We selected five input parameters* λ *in the training range of each filter for testing. Our method is compared with Liu et al. [LPY16], Gharbi et al. [GCB*17] and Fan et al. [FCY*18].*

tions q_{ij} for pixel $p = (x, y)$ are:

$$q_{ij} = (x + i + O^X_{ij}(x,y), y + j + O^Y_{ij}(x,y)), i, j \in \{\pm 1, 0\}. \quad (5)$$

Given an input image I, the output Z of one deformable convolution layer is computed by

$$Z(p) = \sum_{i,j} w(p, q_{ij}) I(q_{ij}), \quad (6)$$

where the weights are defined as:

$$w(p, q_{ij}) = \frac{\exp(-||F(p) - F(q_{ij})||^2)}{\sum_{i,j} \exp(-||F(p) - F(q_{ij})||^2)}, i, j \in \{\pm 1, 0\}. \quad (7)$$

The weight increases when the feature vector $F(q_{ij})$ is close to $F(p)$, i.e., the pixel q_{ij} is similar to pixel p in the image. Note that the sampling coordinates may not be integer values, so we use bilinear interpolation to obtain the sampled value $I(q_{ij})$ and $F(q_{ij})$, making the whole process differentiable for back propagation. Our definition of the weights in Equ. (7) is similar to that of bilateral filter in Equ. (3). The difference lies in that we learn a network to map the RGBXY coordinates into another feature space. In Equ. (7) we do not introduce the variance term because we expect the network to learn the scale factor itself given the input parameter.

In this work we employ 4 deformable convolution layers so that a large reception field could be approximated. In each layer we refer to the same feature maps F, which contains 5 channels, but change the sampling offsets O. To define the offsets for one 3×3 kernel on x, y direction, we need $3 \times 3 \times 2 = 18$ offset maps. For 4 layers of deformable convolution $18 \times 4 = 72$ offset maps are required in total. The deformable convolution block takes the target image as input and uses the feature maps as well as offset maps to compute the weighted average output.

3.3. Loss Function

The basic function of our model is to approximate an existing image smoothing algorithm, so it is trained on image smoothing task. Provided the input image I and the parameter λ, the model M predicts a filtered output $M(I, \lambda)$. Since we can obtain the ground truth output from existing image smoothing algorithms, denoted by $GT(I, \lambda)$, we directly use the mean squared error (MSE) as the loss function:

$$L(I, \lambda) = \frac{1}{N} ||M(I, \lambda) - GT(I, \lambda)||^2, \quad (8)$$

where N is the pixel numbers times the image channels.

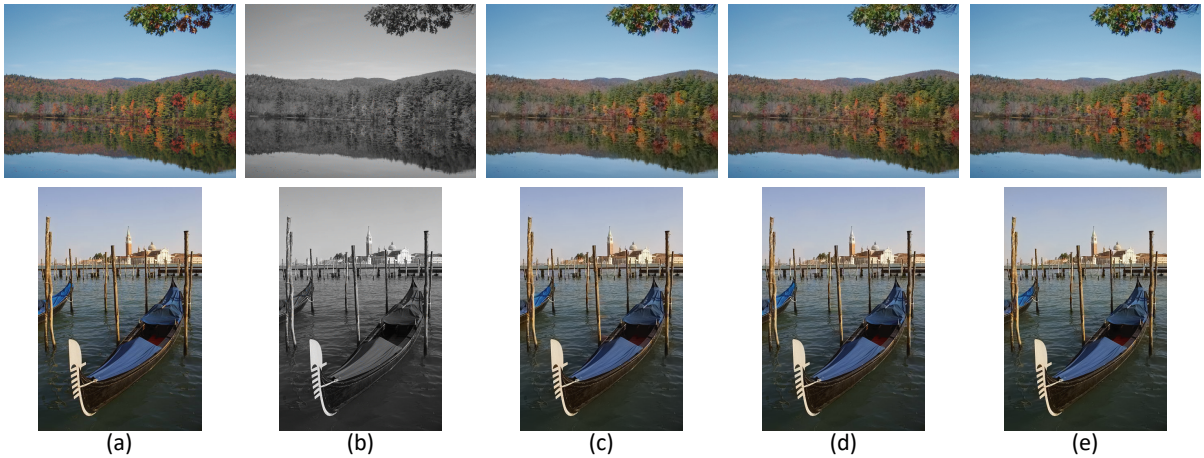


Figure 4: *Results of color interpolation. (a) Original image. (b) Grayscale image with 3% color pixels. (c) DT [GO11], PSNR: 36.32 (top), 39.56 (bottom). (d) Levin [LLW04], PSNR: 36.95 (top), 39.61 (bottom). (e) Our method, PSNR: 37.12 (top), 40.51 (bottom).*

4. Experiment

We trained the proposed model for image smoothing task, where the target image and guidance image are identical. We selected three representative image smoothing filters: weighted least square (WLS) [FFLS08], L_0 optimization [XRY*15] and relative total variation (RTV) [XYXJ12], and trained our proposed model to approximate them respectively, obtaining three deep filters (WLS model, L_0 model and RTV model). We adopted the PASCAL VOC 2012 dataset [EVW*10] for training and the BSDS500 test set [MFTM01] for testing. The training set consists of nearly 17000 images and the test set consists of 200 images. We randomly sampled 7 smoothing parameters λ for each training image and generated the ground truth on randomly cropped 256×256 patches using existing smoothing methods. For testing we selected 5 parameters λ in the training range to filter images in the test set. The learnable parameters are A_k and B_k illustrated in Equ. (4). A_k is initialized with Gaussian distribution and B_k is initialized as zero. We set batch size=8 in the experiment. We used the Adam optimizer [KB14] and set the learning rate to be 0.01 at the beginning, which would decrease to 0.001 during the training process. We found that the model converged after around 10 epoches.

4.1. Filter Approximation

The statistics for filter approximation are exhibited in Table 1. We use PSNR to measure the similarity to the ground truth output. We compare our method with Fan et al. [FCY*18], Liu et al. [LPY16] and Gharbi et al. [GCB*17]. Note that Liu et al. and Gharbi et al. learned the smoothing model for fixed input parameter, while ours and Fan et al. can learn variable smoothing parameter in one network. Our model surpasses Liu et al. as well as Gharbi et al. and has performance comparable to Fan et al. We attribute the difficulty of learning L_0 filter to its piecewise constant effect and the sharpened edges that could not be expressed in a weighted average form. Moreover, we find that as the smoothing strength increases, the filter is harder to learn because the sampling locations spread further. Figure 3 provides qualitative examples for each filter.

Method	λ 0.002	0.005	0.01	0.02	0.05	Avg.
REGULAR	44.86	40.71	37.83	35.06	31.19	37.93
CONCAT	41.09	38.76	37.01	35.12	32.78	36.95
OURS-D3	43.85	40.34	38.05	35.93	33.54	38.34
OURS-D5	44.61	40.98	38.67	36.58	34.16	**39.00**
OURS-D7	44.03	40.85	38.72	36.68	34.28	38.91

Table 2: *Average PSNR of different network structures evaluated on L_0 smoothing. See text for details.*

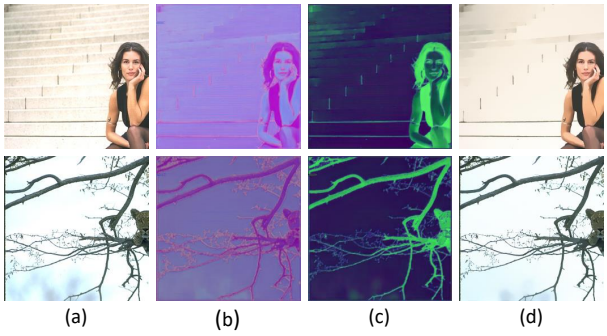


Figure 5: *Feature maps visualization. We show the output feature maps of our L_0 model. The values were scaled into range $[0, 1]$ and encoded in BGR channels. (a) Input image. (b) Feature channel 1-3. (c) Feature channel 3-5. (d) Corresponding smoothed image.*

We also evaluated other network structures: (a) replacing the deformable kernels with one regular 9×9 kernel (denoted as REGULAR); (b) directly concatenating the algorithm parameter with input image and removing the weights prediction layer described in Equ. (4) (denoted as CONCAT); (c) Using our pipeline described in

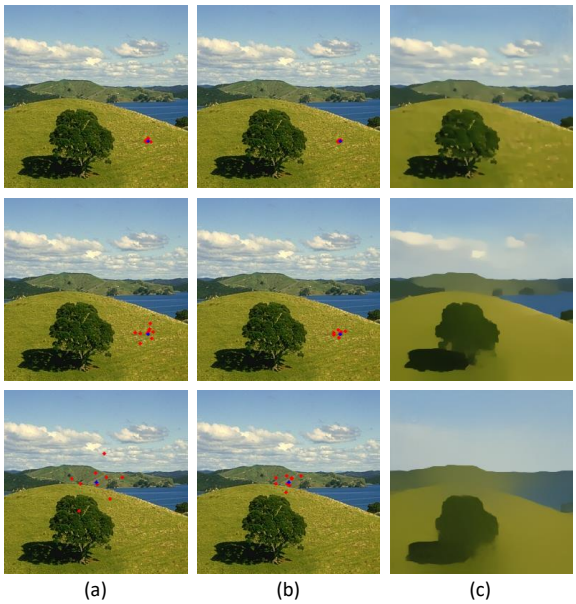


Figure 6: *Sample points visualization. (a, b) Input image and sample points for different smoothing strength. The red dots indicate the sampling points for averaging and the blue dots indicate the origin points. (c) Smoothed results of the RTV model. Top row:* $\lambda = 0.002$*; second row:* $\lambda = 0.01$*; third row:* $\lambda = 0.05$

Sec. 2.5 but varying the dimensionality of feature map $d_F = 3, 5, 7$ (denoted as OURS-D3, OURS-D5, OURS-D7 respectively). These models were trained and evaluated on L_0 optimization [XLXJ11] and the result is provided in Table 2. Using deformable kernels and weights prediction is superior to other alternatives and the choice of feature dimensionality $d_F = 5$ is sufficient.

4.2. Feature Visualization

To observe the feature maps learned by our model, we visualize the output of the feature extraction block with BGR color channels. The feature values are scaled to range $[0, 1]$. Figure 5 shows the 1-3 channels, 3-5 channels of the feature maps. The feature maps are well aligned with the edges in the output image, and the smoothed region is dominated by one specific color, meaning that our network can learn where to smooth and what to preserve, and assign appropriate feature vectors to each pixel. Figure 6 exhibits the sampling locations for one pixel in the image. As the smoothing parameter increases, the sample points move further from the origin, leading to a smoother output. Moreover, most of the sample points lie in the boundary of a smooth region.

4.3. Applications

Image smoothing filters have a wide range of applications. Problems requiring single image input can be directly tackled by our model as well as previous deep implicit models. In this section we select a few joint image filtering applications for illustration, including color interpolation, depth map upsampling, saliency map upsampling, flash/non-flash denoising and RGB/NIR denoising. Note that Li et al. [LHAY19] trained two models for upsampling and denoising respectively, and neither of them is capable of interpolating sparse data. Different from prior work, our method can tackle all of the tasks above by one model.

Color interpolation. Our model is capable of interpolating sparse data because the data source can spread information to its neighboring pixels during the sampling and averaging process. We considered two scenarios in this experiment. The first task is to restore the color channels from grayscale with randomly sampled color pixels. We converted the image into YUV format and retained the UV channels with 3% probability. We fed the grayscale image into our model as guidance to obtain the feature maps and offset maps, and then used them to filter the sparse color channels and a mask identifying the sampling pixels. The output color map was constructed by dividing the filtered color channel by the filtered mask. We compared our method with the classical optimization based colorization method [LLW04] and domain transform [GO11]. The results are showed in Figure 4. In this experiment we use the learned *WLS* model with input parameter $\lambda = 0.3$.

We also evaluated our models' performance of propagating user-specified color strokes to the whole image. Under this setting it is necessary to filter the color map for a few iterations. Our model can produce feasible colorization results by propagating the sparse input iteratively, as exhibited in Figure 9.

Depth map upsampling. Depth data collected from depth sensors might have lower resolution than images captured by RGB camera. It is necessary to upsample the depth map in many applications. We utilized our model to perform depth map upsampling. The offset maps for smoothing usually lead to a large neighborhood, which is unnecessary in this task. Therefore we only used the feature maps and interpolated the unknown pixels by its neighbors on the downsampled grid. We followed the idea of joint bilateral upsampling [KCLU07] while replacing the guidance image with the feature maps predicted by the network.

Method	Bicubic	GF	JBF	DJF	Ours
4×	5.10	5.25	3.34	1.96	2.17
8×	8.26	8.15	5.50	3.60	3.56

Table 3: *Average RMSE of depth map upsampling on the Middlebury dataset [HS07]. Our method is compared with guided filter (GF) [HST10], joint bilateral filter (JBF) [KCLU07] and deep joint filter (DJF) [LHAY19]. We used the RTV model with* $\lambda = 0.003, 0.004$ *for* $4\times$ *and* $8\times$ *upsampling respectively.*

We evaluated our method on the Middlebury dataset [HS07]. The missing depth was completed using method in [LRL14]. Our method was compared with bicubic interpolation, guided filter [HST10], joint bilateral upsampling [KCLU07] and the deep joint filter (DJF) [LHAY19]. Table 3 illustrates the statistics of $4\times$ and $8\times$ upsampling. Note that the deep joint filter was trained directly for this task, but our model has comparable performance with it. Besides, our method is superior to traditional explicit image filters. Figure 7 gives a qualitative comparison.

Saliency map upsampling. We evaluated our model on the task

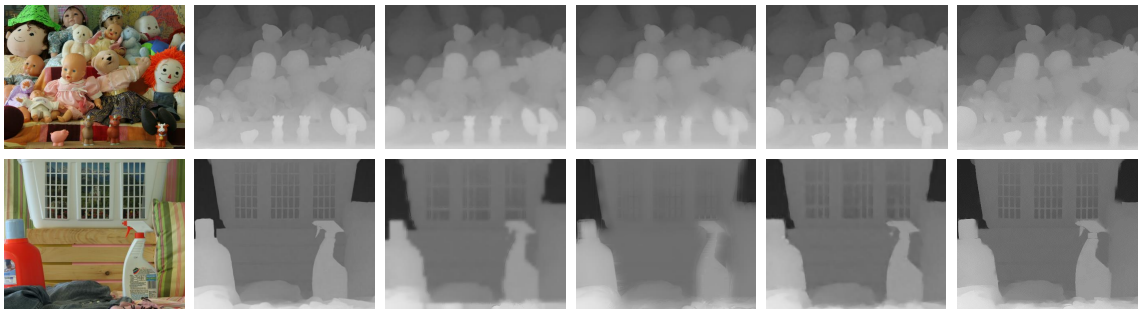


Figure 7: *Results of depth map upsampling. (a) Guidance image. (b) Ground truth. (c) JBF [KCLU07]. (d) GF [HST10]. (e) DJF [LHAY19]. (f) Our method. The first row shows* $4\times$ *upsampling results while the second row shows* $8\times$ *upsampling.*

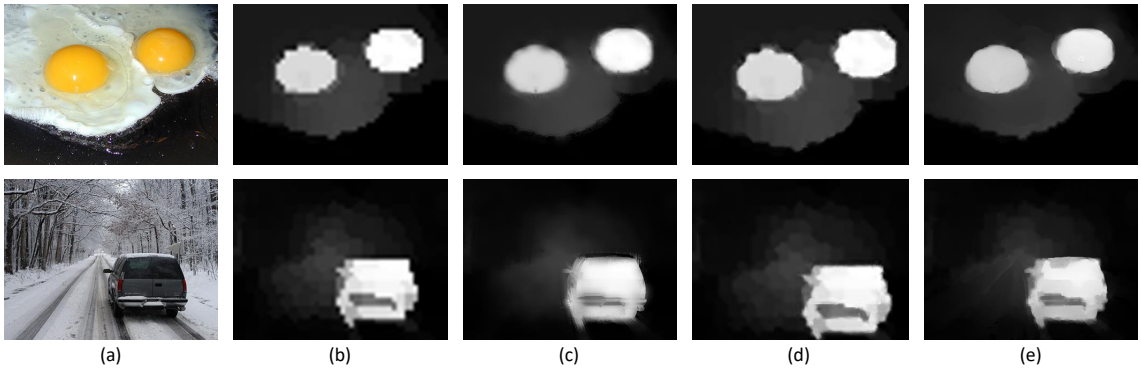


Figure 8: *Example of saliency map upsampling. We estimated the saliency using method proposed in [YZL*13] on the* $8\times$ *downsampled images and upsampled them to the original resolution. (a) Input image. (b) Bicubic interpolation. (c) GF [HST10]. (d) DJF [LHAY19]. (e) Our method.*

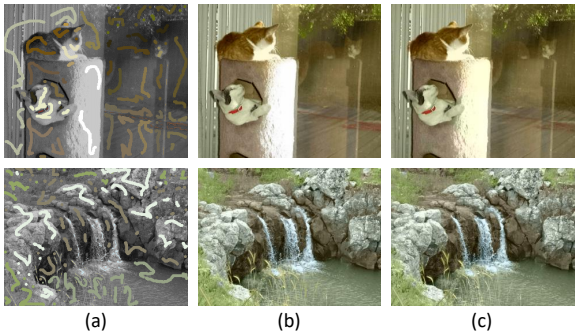


Figure 9: *Colorization using strokes. (a) Input grayscale with color strokes. (b) Levin's method [LLW04]. (c) our method.*

of saliency map upsampling. We applied the method proposed in [YZL*13] to estimate the saliency map on the downsampled images ($8\times$). We randomly select 165 images from the MSRA1000 dataset [AHES09]. Then we used our model to filter the saliency map upsampled by bicubic interpolation. We applied the L_0 model with $\lambda = 0.01$ in this task. Figure 8 shows two examples of the filtering results and Table 4 reports the F1-score of the output saliency map using optimal dataset scale (ODS) and optimal image scale (OIS). Compared with guided filter [HST10] and deep joint filter [LHAY19], our method achieved higher accuracy.

F1-score	Yang	Bicubic	GF	DJF	Ours
ODS	0.8012	0.8004	0.8097	0.7568	0.8115
OIS	0.9007	0.8808	0.8897	0.8095	0.8964

Table 4: *F1-score of upsampled saliency maps evaluated on the MSRA1000 dataset [AHES09]. We compared with GF [HST10] and DJF [LHAY19] on both the optimal dataset scale (ODS) and optimal image scale (OIS). The low resolution saliency map* ($8\times$) *was generated by Yang's method [YZL*13]. We also listed the performance of Yang's method applied to full resolution (Column 2).*

Flash/Non-flash and RGB/NIR denoising. Our model can perform cross-modality filtering, in which a clean image in one domain guides the filtering on a noisy image in another domain. We adopted our model to filter non-flash image with flash guidance image and to filter RGB image with NIR guidance image. As shown in Figure 10 and Figure 11, deep joint filter [LHAY19] cannot handle varying noise level while ours can be adjusted to different scenarios.

Figure 10: *Example of flash/non-flash denoising. (a) Guidance flash image. (b) Non-flash image. (c) Result of DJF [LHAY19]. (d) Result of our method.*

Figure 11: *Example of NIR/RGB denoising. (a) Guidance NIR image. (b) RGB image. (c) Result of DJF [LHAY19]. (d) Result of our method.*

5. Conclusion

In this paper we propose a novel deep learning framework to approximate existing image smoothing filters. Different from previous work, we learn explicit kernels for weighted average filtering, which could be adapted to joint image filtering applications. The feature maps and offset maps are predicted by a fully convolutional network. The filtering weights are defined by the affinity between per-pixel feature vectors and the sampling locations are obtained from the offset maps. We use stacked deformable convolution layers to filter the input image with the predicted weights and sampling locations. Our model achieves high approximation accuracy and serves as a general tool for other joint image filtering tasks such as color interpolation, depth map/saliency map upsampling and cross-modality denoising.

Our method also has a few limitations. Firstly, due to the limit of sample numbers, our model's performance degrades when smoothing the image with extremely high strength. Secondly, as a tool for joint image filtering, our model only utilizes the structural information from the guidance image but neglects valuable information from the target image. We desire to develop an architecture that can switch between single image input and multiple images input. Besides, Fan et al. proposed an unsupervised method to learn an image smoothing filter [FYW*18], and we suggest that it is also possible to learn an explicit form for their filter.

6. Acknowledgement

We thank the anonymous reviewers for their valuable comments and Yukuo Cen for helpful discussions. This work was supported by the Natural Science Foundation of China (Project Number 61521002), the Joint NSFC-ISF Research Program (project number 61561146393), Research Grant of Beijing Higher Institution Engineering Research Center and Tsinghua-Tencent Joint Laboratory for Internet Innovation Technology.

References

[AHES09] ACHANTA, RADHAKRISHNA, HEMAMI, SHEILA, ESTRADA, FRANCISCO, and SÜSSTRUNK, SABINE. "Frequency-tuned salient region detection". *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR 2009)*. CONF. 2009, 1597–1604 8.

[BHY15] BI, SAI, HAN, XIAOGUANG, and YU, YIZHOU. "An L 1 image transform for edge-preserving smoothing and scene-level intrinsic decomposition". *Acm Transactions on Graphics* 34.4 (2015), 78 2, 3.

[BSY*14] BAO, LINCHAO, SONG, YIBING, YANG, QINGXIONG, et al. "Tree Filtering: Efficient Structure-Preserving Smoothing With a Minimum Spanning Tree". *IEEE Transactions on Image Processing A Publication of the IEEE Signal Processing Society* 23.2 (2014), 555–69 2, 3.

[BVM*17] BAKO, STEVE, VOGELS, THIJS, MCWILLIAMS, BRIAN, et al. "Kernel-predicting convolutional networks for denoising Monte Carlo renderings". *ACM Transactions on Graphics (TOG)* 36.4 (2017), 97 2, 3.

[CLKL14] CHO, HOJIN, LEE, HYUNJOON, KANG, HENRY, and LEE, SEUNGYONG. "Bilateral texture filtering". *ACM Transactions on Graphics (TOG)* 33.4 (2014), 128 2, 3.

[DQX*17] DAI, JIFENG, QI, HAOZHI, XIONG, YUWEN, et al. "Deformable convolutional networks". *Proceedings of the IEEE international conference on computer vision*. 2017, 764–773 2–4.

[DT06] DIEBEL, JAMES and THRUN, SEBASTIAN. "An application of markov random fields to range sensing". *Advances in neural information processing systems*. 2006, 291–298 3.

[EVW*10] EVERINGHAM, MARK, VAN GOOL, LUC, WILLIAMS, CHRISTOPHER KI, et al. "The pascal visual object classes (voc) challenge". *International journal of computer vision* 88.2 (2010), 303–338 6.

[FCY*18] FAN, QINGNAN, CHEN, DONGDONG, YUAN, LU, et al. "Decouple learning for parameterized image operators". *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, 442–458 2–6.

[FFLS08] FARBMAN, ZEEV, FATTAL, RAANAN, LISCHINSKI, DANI, and SZELISKI, RICHARD. "Edge-preserving decompositions for multi-scale tone and detail manipulation". *ACM Transactions on Graphics (TOG)*. Vol. 27. 3. ACM. 2008, 67 2–4, 6.

[FYH*17] FAN, QINGNAN, YANG, JIAOLONG, HUA, GANG, et al. "A generic deep architecture for single image reflection removal and image smoothing". *Proceedings of the IEEE International Conference on Computer Vision*. 2017, 3238–3247 2, 3.

[FYW*18] FAN, QINGNAN, YANG, JIAOLONG, WIPF, DAVID, et al. "Image Smoothing via Unsupervised Learning". *ACM Transactions on Graphics (Proceedings of SIGGRAPH ASIA 2018)* 37.6 (2018) 2, 3, 9.

[GCB*17] GHARBI, MICHAËL, CHEN, JIAWEN, BARRON, JONATHAN T, et al. "Deep bilateral learning for real-time image enhancement". *ACM Transactions on Graphics (TOG)* 36.4 (2017), 118 2, 3, 5, 6.

[GO11] GASTAL, EDUARDO S. L and OLIVEIRA, MANUEL M. "Domain transform for edge-aware image and video processing". *Acm Transactions on Graphics* 30.4 (2011), 1–12 2, 3, 6, 7.

[HCP15] HAM, BUMSUB, CHO, MINSU, and PONCE, JEAN. "Robust image filtering using joint static and dynamic guidance". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, 4823–4831 3.

[HS07] HIRSCHMULLER, HEIKO and SCHARSTEIN, DANIEL. "Evaluation of cost functions for stereo matching". *2007 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2007, 1–8 7.

[HST10] HE, K., SUN, J., and TANG, X. "Guided Image Filtering". *European Conference on Computer Vision*. 2010, 1–14 2, 3, 7, 8.

[JDTG16] JIA, XU, DE BRABANDERE, BERT, TUYTELAARS, TINNE, and GOOL, LUC V. "Dynamic filter networks". *Advances in Neural Information Processing Systems*. 2016, 667–675 2, 3.

[KB14] KINGMA, DIEDERIK P and BA, JIMMY. "Adam: A method for stochastic optimization". *arXiv preprint arXiv:1412.6980* (2014) 6.

[KCLU07] KOPF, JOHANNES, COHEN, MICHAEL F, LISCHINSKI, DANI, and UYTTENDAELE, MATT. "Joint bilateral upsampling". *ACM Transactions on Graphics (ToG)*. Vol. 26. 3. ACM. 2007, 96 3, 7, 8.

[KEE13] KARACAN, LEVENT, ERDEM, ERKUT, and ERDEM, AYKUT. "Structure-preserving image smoothing via region covariances". *ACM Transactions on Graphics (TOG)* 32.6 (2013), 176 2, 3.

[LCS*17] LIU, WEI, CHEN, XIAOGANG, SHEN, CHUANHUA, et al. "Semi-global weighted least squares in image filtering". *Proceedings of the IEEE International Conference on Computer Vision*. 2017, 5861–5869 3.

[LHAY19] LI, YIJUN, HUANG, JIA-BIN, AHUJA, NARENDRA, and YANG, MING-HSUAN. "Joint Image Filtering with Deep Convolutional Networks". *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019) 2, 3, 7–9.

[LLW04] LEVIN, ANAT, LISCHINSKI, DANI, and WEISS, YAIR. "Colorization using optimization". *ACM transactions on graphics (tog)*. Vol. 23. 3. ACM. 2004, 689–694 6–8.

[LPY16] LIU, SIFEI, PAN, JINSHAN, and YANG, MING-HSUAN. "Learning recursive filters for low-level vision via a hybrid neural network". *European Conference on Computer Vision*. Springer. 2016, 560–576 2, 3, 5, 6.

[LRL14] LU, SI, REN, XIAOFENG, and LIU, FENG. "Depth enhancement via low-rank matrix completion". *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, 3390–3397 7.

[LYB18] LU, KAIYUE, YOU, SHAODI, and BARNES, NICK. "Deep Texture and Structure Aware Filtering Network for Image Smoothing". *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, 217–233 2, 3.

[LZGZ15] LIU, QIAN, ZHANG, CAIMING, GUO, QIANG, and ZHOU, YUANFENG. "A nonlocal gradient concentration method for image smoothing". *Computational Visual Media* 1.3 (2015), 197–209 3.

[MBC*18] MILDENHALL, BEN, BARRON, JONATHAN T, CHEN, JIAWEN, et al. "Burst denoising with kernel prediction networks". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, 2502–2510 2, 3.

[MCL*14] MIN, DONGBO, CHOI, SUNGHWAN, LU, JIANGBO, et al. "Fast global image smoothing based on weighted least squares". *IEEE Transactions on Image Processing* 23.12 (2014), 5638–5653 3.

[MFTM01] MARTIN, D., FOWLKES, C., TAL, D., and MALIK, J. "A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics". *Proc. 8th Int'l Conf. Computer Vision*. Vol. 2. July 2001, 416–423 5, 6.

[PKT*11] PARK, JAESIK, KIM, HYEONGWOO, TAI, YU-WING, et al. "High quality depth map upsampling for 3d-tof cameras". *2011 International Conference on Computer Vision*. IEEE. 2011, 1623–1630 3.

[PM90] PERONA, PIETRO and MALIK, JITENDRA. "Scale-space and edge detection using anisotropic diffusion". *IEEE Transactions on pattern analysis and machine intelligence* 12.7 (1990), 629–639 3.

[ROF92] RUDIN, LEONID I, OSHER, STANLEY, and FATEMI, EMAD. "Nonlinear total variation based noise removal algorithms". *Physica D: nonlinear phenomena* 60.1-4 (1992), 259–268 2, 3.

[SZXJ15] SHEN, XIAOYONG, ZHOU, CHAO, XU, LI, and JIA, JIAYA. "Mutual-structure for joint filtering". *Proceedings of the IEEE International Conference on Computer Vision*. 2015, 3406–3414 3.

[TM98] TOMASI, CARLO and MANDUCHI, ROBERTO. "Bilateral filtering for gray and color images". *Computer Vision, 1998. Sixth International Conference on*. IEEE. 1998, 839–846 2, 3.

[VRM*18] VOGELS, THIJS, ROUSSELLE, FABRICE, MCWILLIAMS, BRIAN, et al. "Denoising with kernel prediction and asymmetric loss functions". *ACM Transactions on Graphics (TOG)* 37.4 (2018), 124 2, 3.

[WCL*15] WANG, HUI, CAO, JUNJIE, LIU, XIUPING, et al. "Least-squares images for edge-preserving smoothing". *Computational Visual Media* 1.1 (2015), 27–35 3.

[XLXJ11] XU, LI, LU, CEWU, XU, YI, and JIA, JIAYA. "Image smoothing via L 0 gradient minimization". *Acm Transactions on Graphics* 30.6 (2011), 1–12 2–4, 7.

[XRY*15] XU, LI, REN, JIMMY S. J, YAN, QIONG, et al. "Deep edge-aware filters". *International Conference on International Conference on Machine Learning*. 2015, 1669–1678 2, 3, 6.

[XYXJ12] XU, LI, YAN, QIONG, XIA, YANG, and JIA, JIAYA. "Structure extraction from texture via relative total variation". *Acm Transactions on Graphics* 31.6 (2012), 1–10 2–4, 6.

[YZL*13] YANG, CHUAN, ZHANG, LIHE, LU, HUCHUAN, et al. "Saliency detection via graph-based manifold ranking". *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2013, 3166–3173 8.

[ZDXZ15] ZHANG, FEIHU, DAI, LONGQUAN, XIANG, SHIMING, and ZHANG, XIAOPENG. "Segment Graph Based Image Filtering: Fast Structure-Preserving Smoothing". *IEEE International Conference on Computer Vision*. 2015, 361–369 2, 3.

[ZHLD18] ZHU, XIZHOU, HU, HAN, LIN, STEPHEN, and DAI, JIFENG. "Deformable convnets v2: More deformable, better results". *arXiv preprint arXiv:1811.11168* (2018) 2–4.

[ZSXJ14] ZHANG, QI, SHEN, XIAOYONG, XU, LI, and JIA, JIAYA. "Rolling Guidance Filter". *European Conference on Computer Vision*. 2014, 815–830 3.