The Domain Transform Solver

Akash Bapat and Jan-Michael Frahm
Department of Computer Science, The University of North Carolina at Chapel Hill
{akash, jmf}@cs.unc.edu

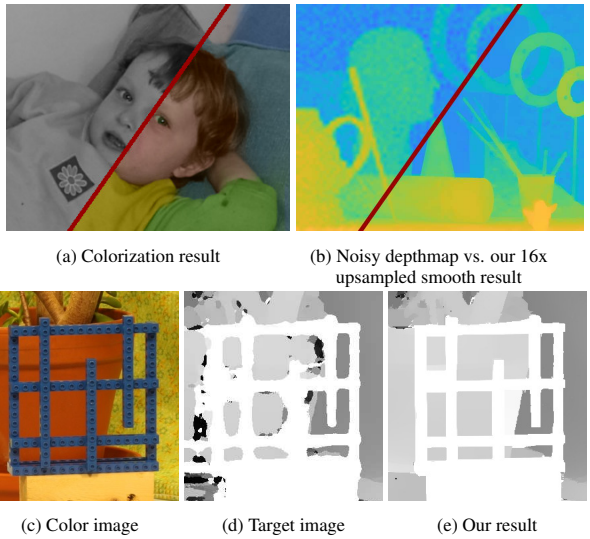
Abstract

We present a novel framework for edge-aware optimization that is an order of magnitude faster than the state of the art while maintaining comparable results. Our key insight is that the optimization can be formulated by leveraging properties of the domain transform [17], a method for edge-aware filtering that defines a distance-preserving 1D mapping of the input space. This enables our method to improve performance for a wide variety of problems including stereo, depth super-resolution, render from defocus, colorization, and especially high-resolution depth filtering, while keeping the computational complexity linear in the number of pixels. Our method is highly parallelizable and adaptable, and it has demonstrable linear scalability with respect to image resolutions. We provide a comprehensive evaluation of our method w.r.t speed and accuracy for a variety of tasks.

1. Introduction

Edge-aware optimization is a widely utilized tool in computer vision. It has been applied to a large variety of tasks, including semantic segmentation [21], stereo [5], colorization [22], and optical flow [32]. Edge-awareness is motivated by the intuition that similar-looking pixels often have similar properties. For this reason, a wide variety of edge-aware filtering algorithms have been developed, including the bilateral filter [37], anisotropic diffusion [29], and edge-avoiding wavelets [14], all of which identify similar-looking pixels. However, using such filters in optimization frameworks typically leads to computationally expensive algorithms. While high-level groupings like super-pixels can be used to compensate for this sluggishness [25], the color-space clusterings of such approaches are not guaranteed to respect the semantics of the underlying domain, often leading to artifacts.

We propose a general optimization framework that directly operates in the pixel space while maintaining distances in the combined color and pixel space with an edge-aware regularizer. The framework can be applied for a variety of optimization problems, as we demonstrate in Fig. 1 and Sec. 3. Our method achieves competitive accuracy in applications like stereo optimization (Sec. 4.1), rendering from defocus (Sec. 4.2), depth super-resolution (Sec. 4.3), and high resolution depth filtering for multi-view stereo (Sec. 5). While being extremely fast, our framework excels in two aspects that are not jointly achieved by any prior edge-aware optimization algorithms: 1) With increasing image resolution, as well as a growing number of image dimensions/channels, our method scales linearly and is more than twice as fast as existing parallel methods. 2) Our approach is independent of blur kernel sizes. Considering the increasingly high resolutions of cameras in consumer devices such as smartphones, and the desire for on-device computing for tasks like automatic photo enhancement, our contributions are poised to enable crucial advances for up-



(a) Colorization result (b) Noisy depthmap vs. our 16x upsampled smooth result

(c) Color image (d) Target image (e) Our result

Figure 1: Our domain transform solver can tackle a variety of problems, including (a) colorization, (b) depth super-resolution using the color image as reference, and (c-e) depth map refinement. The second row shows a color image from the Middlebury dataset [33] with an initialization (target) obtained from MC-CNN [44], which is then refined in an edge-aware sense to obtain our result (e).

coming technologies. To foster broad use of our framework, we will release the source code upon publication.

The remainder of the paper is organized as follows: Sec. 2 describes the related approaches for edge-aware filtering and optimization. Sec. 3 derives our domain transform solver (DTS) optimization framework and highlights its similarities as well as dissimilarities with previous work. We also illustrate how our framework can be leveraged for various vision tasks like stereo, depth super-resolution, colorization, high-resolution filtering, and synthetic defocus, in Sec. 4. In Sec. 5, we provide a quantitative evaluation as well as validation of the timing performance. Finally, we conclude in Sec. 6 and provide some future directions for expanding our framework.

2. Related Work

We briefly review the most relevant prior work related to edge-based filtering and optimization, namely: bilateral filters and their variants, optimizations leveraging superpixels, machine learning for edge-aware filtering, the domain transform and its filtering applications, and bilateral solvers. We consider an algorithm a *filtering technique* when a filter is applied on the input image to produce an output image. On the other hand, we consider an algorithm a *solver* when it uses one or more input images and optimizes towards a goal defined by a cost/loss function.

Bilateral filters The bilateral filter was introduced by Tomasi and Manduchi [37] and is one of the initial edge-aware blurring techniques. The major bottleneck for bilateral filtering is that it is costly to compute, especially for large blur windows. For N pixels in an image, filter radius r in each dimension, and d dimensions, its complexity is $\mathcal{O}(Nr^d)$ [30]. Since its invention, there have been multiple approaches proposed to speed up the bilateral filter [38, 1, 8, 41, 12]: Durand and Dorsey approximate the bilateral filter by a piece-wise linear function [11]. Pham and van Vliet proposed to use two 1-D bilateral kernels, reducing the complexity to $\mathcal{O}(Nrd)$ [30]. Paris and Durand treat the image as a 5-D function of color and pixel space and then apply 1-D blur kernels in this high-dimensional space [27], leading to complexity of $\mathcal{O}(N + N/r^2 \times \prod_{i=3}^{d}(D_i/r))$, where the N/r^2 term is due to the 2-D pixel space and the remainder is the contribution of dimensions $i = 3 \ldots d$, each with size D_i, *e.g.* $D = 255$ for an 8-bit grayscale image [28]. Note that this is still exponential in the dimensionality [1]. These approximations decouple the 2-D adaptive bilateral kernel into a 1-D kernel, reducing the computational cost significantly.

When used as a post-processing step, the bilateral filter removes noise in homogeneous regions but is sensitive to artifacts such as salt and pepper noise [45]. Our method emphasizes edge-aware concepts in the same spirit as bilateral filters, and our formulation yields a generalized optimization framework that is efficient and accurate. When using robust cost functions, our method remains resistant to outliers.

Machine learning for edge-awareness Porikli [31] achieves independence to kernel sizes, but is inaccurate for low color variances. To remedy this, Yan *et al*. [42] use support vector machines (SVM) to mimic a bilateral filter by using the exponential of spatial and color distances as feature vectors to represent each pixel. Their approach supports varying intra-image color variance while remaining efficient. Traditionally, conditional random fields (CRFs) are used for enforcing pair-wise pixel smoothness via the Potts potential. For efficiency, superpixels are used to reduce the number of optimization variables in the CRF [6]. Unfortunately, this leads to a loss in resolution, as an entire superpixel is assigned one estimate; *e.g*., in stereo, this leads to fronto-parallel depths. Alternatively, Krähenbühl and Koltun [21] proposed to use the permutohedral lattice data structure [1], which is typically used in fast bilateral implementations, to accelerate inference in a fully connected CRF by using Gaussian distances in space and color. With the recent explosion of compute capacity and convolutional models in the vision community, there are also deep-learning methods that attempt to achieve edge-aware filtering. Chen *et al*. [10] presented DeepLab to perform semantic segmentation; there, they use the fully connected CRF from Krähenbühl and Koltun [21] on top of their convolutional neural network (CNN) to improve the localization of object boundaries. Xu *et al*. [40] learn edge-aware operators from the data to mimic various traditional handcrafted filters like the bilateral, weighted median, and weighted least squares filters [13]. However, machine learning approaches require large amounts of training data specific to a task, plus significant compute power, while our approach works without any task-dependent training and runs efficiently on a single GPU. More recently, deep learning methods have been introduced to learn optimal bilateral weights [18, 23, 39] instead of using the traditional Gaussian color weights.

The domain transform Gastal and Oliveira [17] introduced the domain transform, a novel and efficient method for edge-aware filtering that is akin to bilateral filters. The domain transform is defined as a 1-D isometric transformation of a multi-valued 1-D function such that the distances in the range and domain are preserved. (See Sec. 3.2 for more details.) When applied to a 1-D image with multiple color channels, the transformation maps the distances in color and pixel space into a 1-D distance in the transformed space. When the *scalar* distance is measured in the transformed space, it is equivalent to measuring the *vector* distance in [R,G,B,X] space. This has the benefit of dimensionality reduction, leading to a fast edge-aware filtering technique which respects edges in color while blurring similar pixels.

To apply the domain transform to a 2-D image, the authors apply two passes, once in the X direction and once in Y. This has a complexity of $\mathcal{O}(Nd)$, and as a result scales well with dimensionality. Applying the domain transform to an image results in a filtering effect, while in our case we optimize according to an objective function. Chen *et al*. [9] proposed to perform edge-aware semantic segmentation using deep learning and use a domain transform filter in their end-to-end training of their deep-learning framework. They also alter the definition of what is considered as 'edge' by learning an edge prediction network, and they then use the learned edge-map in a domain transform. Their application of the domain transform is in the form of a *filter*, and hence is similar to one iteration of our method. We use the domain transform in our method in an iterative fashion within our optimization framework because it provides an efficient way to compute the local edge-aware mean. The application of the domain transform as a filter similar to Chen *et al*. [9] produces less accurate results than our framework; as we show in Sec. 5.

Bilateral solvers Recently, Barron *et al*. [3] suggested to view a color image as a function of the 5-D space [Y,U,V,X,Y], which they call the 'bilateral space', to estimate stereo for rendering defocus blur. They proposed to transform the stereo optimization problem by expressing the problem variables in the bilateral space and then optimizing in this new space. We will refer to this method as BL-Stereo, or BLS in short. Barron and Poole's Fast Bilateral Solver (FBS) [4], on the other hand, solves a linear optimization problem in the bilateral space, which is different from BLS. In this setting, they require a target map to enhance, as well as a confidence map for the target. The linearization of the problem allows them to converge to the solution faster. (See Sec 3.1 for more details.) Both of these approaches quantize the 5-D space into a grid, where the grid size is governed by the blur kernel size. This reduces the number of optimization variables and hence the complexity, leading to low runtimes. Mazumdar *et al*. [26] use a dense grayscale-space 3-D grid (instead of a 5-D color-space grid) and the Heavy-Ball algorithm to make FBS faster, but at the cost of accuracy.

Our work is closely related to Barron *et al*. [3] and Barron and Poole [4] in that we are targeting the same goal of developing general solvers that are edge-aware and efficient. The gridding strategy of the previous methods scales well with higher blur amounts and larger spatial windows. However, using higher blur windows is not a viable option, especially in high-resolution imagery where it is important to maintain fine details, such as multi-camera capture for virtual reality and satellite imagery. In contrast, our method does not require large blur kernels to be efficient. Our method operates on the pixels themselves. Our approach is inherently parallelizable, and hence can greatly benefit from GPU processing. Hence, it scales well with higher image resolutions irregardless of blur kernel size. As we show in Sec. 5, the high level of parallelism especially outshines prior methods for high-resolution imagery.

We present our general optimization framework and demonstrate its performance on a variety of problems. We also present quantitative evaluations to show the competitive accuracy of our method, as well as the significant speed-up that it delivers.

3. Approach

Edge-aware filtering techniques smooth similar-looking regions of the image while preserving crisp edges. Filtering techniques can often be formulated as single iterations in a solver. For instance, the bilateral filter is equivalent to a single step in minimizing a weighted least squares energy function[12]. In the following, we introduce the domain transform solver (DTS), which leverages repeated applications of the domain transform (DT) filter. We first present an optimization framework that is analogous to the existing bilateral solver formulations and then explain how the DT significantly boosts our framework's speed and scalability.

3.1. Optimization framework

Our DTS solves the following optimization problem:

$$\min_{z} \lambda \sum_{i} \underbrace{\left(z_i - \bar{z}_{N_i}\right)^2}_{=e(z_i,N_i)} + \omega_i c_i \left(z_i - t_i\right)^2 + \sum_{m} \lambda_m \Phi_m\left(z\right) \tag{1}$$

Here, the z_i are the values we want to estimate, *e.g*. disparity in stereo, at the i^{th} pixel of an image. The initial target estimate t_i with a confidence c_i is also given for the i^{th} pixel; ω_i is an edge-aware normalization term described below. This optimization objective has an edge-aware regularizer $e\left(z_i, N_i\right)$, which forces the z_i to be similar to the mean $\bar{z}_{N_i}$ of its neighborhood N_i, computed in an edge-aware sense. This edge-aware mean is $\bar{z}_{N_i} = \left(\sum_j W_{i,j} * z_j\right) / \sum_j W_{i,j}$, where $W_{i,j}$ takes into account the pixel color similarity as well as the distance between pixels i and j. We derive $W_{i,j}$ in Sec. 3.2. The term $\omega_i = 1/\sum_j W_{i,j}$ scales the confidence of each pixel, such that pixels with similar neighbors are influenced less by their target value, and vice versa. We compute $\bar{z}_{N_i}$ using the domain transform, which enables us to evaluate our pairwise regularizer for any blur kernel size and dimensionality, faster than traditional approaches [5, 37, 28, 15, 27]. $\Phi_m\left(z\right)$ is an application-dependent term with a weighting factor of λ_m. For example, $\Phi_m\left(z\right)$ could be the photometric matching cost for the left-right image pair for stereo.

In all applications, our framework aims to solve Eq. (1). The minimum at the point of the solution necessarily has a zero derivative. Hence, we next seek to characterize this

minimum in order to leverage it later in our proposed approach. For simplicity, we first only investigate a simplified version of Eq. (1) that does not contain the problem-specific term $\Phi_m(z)$. This simplified version can be written as :

$$\min_z F(z) = \min_z \lambda \sum_i \underbrace{(z_i - \bar{z}_{N_i})^2}_{=e(z_i,N_i)} + \sum_i \omega_i c_i (z_i - t_i)^2 . \tag{2}$$

Taking the gradient of Eq. (2) with respect to z_i and setting it to zero, at the minima of Eq. (2) we have

$$z_i = \frac{\lambda \bar{z}_{N_i} + \omega_i c_i t_i}{\lambda + \omega_i c_i}. \tag{3}$$

In our supplementary material, we show that this optimal solution closely approximates the optimization function of the FBS [4]. The key distinction for our method lies in the explicit computation of $\bar{z}_{N_i}$ via $W_{i,j}$. Whereas the FBS algorithmically depends on a kernel-size-dependent domain gridding/tiling in the computation of $W_{i,j}$, our DTS instead maps the z_i values into a 1-D space using the domain transform, allowing pixel affinities to be calculated with a runtime independent of blur kernel size.

3.2. Domain transform with the L_2 norm

Gastal and Oliveira [17] define an isometric transformation, which they call the domain transform (DT), for a 1-D multi-valued function $I : \Omega \rightarrow \mathbb{R}^c$, $\Omega = [0, \infty)$ by treating $C = (x, I(x))$ as a curve in $\mathbb{R}^{c+1}$. The domain transform $DT : \mathbb{R}^{c+1} \rightarrow \mathbb{R}$ is such that it preserves distances between two points on the curve C under a given norm. In contrast to Gastal and Oliveira [17], we use the L_2 norm to define the distances which results in higher accuracy as shown in Table 2. Hence, we derive the domain transform here, which satisfies the constraint $\|DT(x_i, I(x_i)) - DT(x_j, I(x_j))\|_2 = \|(x_i, I(x_i)) - (x_j, I(x_j))\|_2$ for the nearest neighbors x_i and x_{i+1}. Here, only x_i and x_{i+1} are considered because the following derivation is exact only when we are close to x_i. Using a shorthand notation $DT(x) = DT(x_i, I(x_i))$ and assuming a small shift h in x, we can express the distance in pixels and color equal to the distance of the transform as follows:

$$(DT(x+h) - DT(x))^2 = h^2 + \sum_{k=1}^{c} (I(x+h) - I(x))^2 . \tag{4}$$

Taking the limit as $h \rightarrow 0$, followed by taking the square root and constraining $DT(x)$ to be monotonic to avoid negative roots, then integrating both sides, we obtain

$$DT(u) = \int_0^u \sqrt{1 + \sum_{k=1}^{c} I_k'^2(x)}\, dx, \quad u \in \Omega. \tag{5}$$

More detailed derivation is available in the supplementary material. Using this definition of the domain transform of the 4-D space $[X, R, G, B]$ with the curve C defined by RGB color and the spatial domain X, we choose to express the edge-aware weights as follows:

$$W_{i,j} = \delta_r (|DT(z_i) - DT(z_j)|) \tag{6}$$

with indicator function $\delta_r(d) = (d \leq r)$.

The relation with the simple domain transform blurring [17] can be seen by setting the confidence scores c_i to zero in Eq. (1). This will lead to the same solution as the domain transform filtering. Similarly, setting c_i to zero and $W_{i.j}$ to Gaussian weights in color and space will lead to bilateral filtering. Note that the above derivation is isometric since the function I is multi-valued but with a *1-dimensional* domain Ω. By extending the domain to 2-D, the exact isometry is not valid, and following Gastal and Oliveira [17] we use alternating passes by separately considering the image as a function of X and then Y.

We compute the confidence scores c_i by estimating the variance of the refined variable z in an edge-aware sense using the domain transform as suggested in [4], and normalizing the variance into [0,1] range by taking negative exponential scaled by σ_c.

4. Applications

Now, all the terms except the application-specific terms in Eq. (1) are defined. In the following, we present the application of our proposed framework to a variety of scenarios where we adapt Eq. (1) by changing function Φ_m.

4.1. Stereo optimization

Stereo depth estimation is a well-studied problem [34, 19] in which the task is to estimate a matching correspondence of pixels in the left image to the pixels in the right image. This matching correspondence defines the disparity of the pixels and in turn the depth, and when done for each pixel provides us with a disparity map. Typically, a dense search is done along the row of a rectified pair by matching the pixel color similarity, *i.e.*, applying a photometric matching cost. In the following, we refine a disparity map. We obtain the disparity map from MC-CNN [44], which acts as the target (Fig. 2c) for which we calculate a confidence score (Fig. 2d). We use the left color image to define and compute the edge-aware mean and optimize the disparities to obtain a disparity map that is smooth at homogeneous regions but has sharp edges (Fig. 2e). Similar to our proposed solver, Barron and Poole [4] show that the FBS works well for a wide variety of optimization problems including stereo. When they apply the FBS to the stereo problem, they achieve faster convergence compared to BL-Stereo because they neglect the physical implication of changing the disparity. In other words, if an optimizer changes the estimate of

disparity at a point in the left image, this results in a change in the matching pixel in the right image and accordingly carries the potential of a change in the corresponding color. Here, we present a method for solving for the disparity in an edge-aware sense while having a photometric penalty for the left-right matching. Our loss for stereo optimization is

$$\min_z \lambda \sum_i \left(z_i - \bar{z}_{N_i}\right)^2 + \sum_i \omega_i c_i \underbrace{\rho\left(z_i - t_i\right)}_{target\ term} + \sum_i \underbrace{\gamma \left(I_L(i) - I_R(i - z_i)\right)^2}_{=\lambda_m \Phi_m(z_i)}, \tag{7}$$

where I_L and I_R are the left and right image of the stereo pair respectively. As disparity can be viewed as 1-D flow, for robust optimization, we use a Charbonnier loss $\rho\left(r\right) = \sqrt{r^2 + \varepsilon^2}$ with $\varepsilon = 0.001$ on the target term, which has been shown to be effective for optical flow [36]. Next, we will detail the application of our method to the problem of rendering defocus from depth, which is another application heavily relying on accurate depth edges.

4.2. Synthetic defocus from depth

Interest in creating synthetic defocus from depth is growing, with phones like the Google Pixel 3 and the OnePlus 6 providing a portrait mode where the shallow depth of field effect is mimicked through the estimation of depth. In fact, BL-Stereo's synthetic defocus method is used as part of the Lens Blur feature on Google's phones [3]. We use our stereo optimization from Sec. 4.1 to estimate depth maps, which retain sharp discontinuities at color edges. Fig. 3 shows the original color image and the defocus rendering produced by using our estimated depthmaps for scenes in the Middlebury dataset [33]. As our stereo optimization is edge-aware, the defocus rendering maintains high quality even at the edges. In the Jadeplant scene shown in Fig. 3a, the background is in focus, and for the same scene the blue block in the front is kept in focus (Fig. 3b). In the Playroom scene in Fig. 3c, the front chairs are chosen to be in focus. To render the synthetic defocus, we used the algorithm described in Sec. 6 of the supplementary material of Barron *et al*. [3]. See supplementary material for additional results.

4.3. Depth super-resolution

The availability of cheap commodity depth sensors like the Microsoft's Kinect, Asus Xtion, and Intel RealSense has spurred many avenues of research, including depth super-resolution. Depth super-resolution is important for sensors like the above because, often, the color camera is of high resolution, but the depth camera/projector has low-resolution, which leads to crude depth maps [20]. Ferstl *et al*. [16] adapted the Middlebury dataset for the depth super-resolution task to create a benchmark, which we use to evaluate our method. For this task, we use a simple bicubic interpolation to upsample the low-resolution depth map and use this map as a target in our optimization; we then use the high-resolution color image to compute the domain transform based on an edge-aware mean and obtain our optimized result (Fig. 4c). We follow Barron and Poole [4] by setting the confidence scores using a Gaussian bump model to represent the contribution of each pixel to the nearby upsampled pixels. We do not use additional penalties in Eq. (1) for this task in the form of Φ_m.

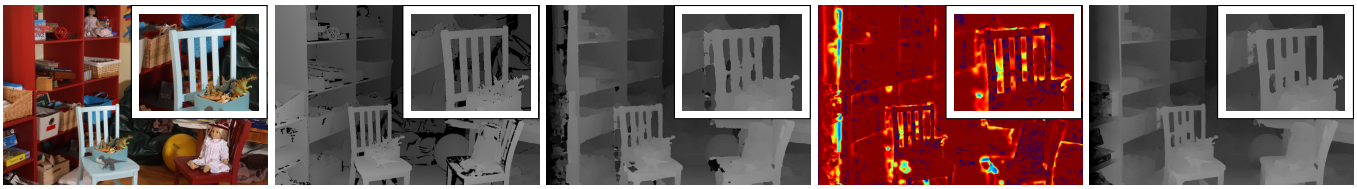
4.4. Colorization

Levin *et al*. [22] presented a method to convert a grayscale image into a color image using a few color strokes as input; see Fig. 1a for our result. Instead of an approach specifically developed for this task [43], we show the generalizability of our efficient framework to the colorization task achieving similar results. We convert the input grayscale image into the YCbCr color space to extract the Y channel, which is used to compute the edge-aware weights $W_{i,j}$. The input strokes act as the target with a confidence of one, and all other confidences are zero. See Fig. 6 for an example input with color strokes.

5. Experiments

We now detail our quantitative evaluation of our framework as well as its run time performance.

Stereo Optimization For the quantitative evaluation of our method, we use the Middlebury dataset [33]. Barron and Poole [4] used MC-CNN [44] as their initialization. For a fair comparison, we also use it as our target disparity map. Table 1 shows our results for the training and testing set, where we present the mean absolute error (MAE), root mean square error (RMSE), time per megapixel (sec/MP), and time normalized by number of disparity hypotheses (sec/GD) for non-occluded regions and for all pixels. All of these values were determined by the Middlebury evaluation website, and all of our times include the time to calculate MC-CNN on the target disparity maps. The timings for FBS and our method show the additional time spent in processing MC-CNN, with the total value in parentheses. Note that we obtain a large performance boost compared to MC-CNN with a marginal overhead in time. We compare favorably to Barron and Poole [4], especially for non-occluded pixels, while having significant computational savings. We used $\sigma_x = \sigma_y = 64px$, $\sigma_r = 0.25$ with RGB colors normalized to a range of [0,1], $\lambda = 0.99$, and $\gamma = 0.001$. These parameters were found to work best via a grid search strategy on the Middlebury training data. We ran a gradient descent algorithm for 3000 iterations in this experiment with a step size of 0.99 times the gradient to avoid numerical precision errors. Fig. 1 (c-e) and Fig. 5 show zoomed regions from



(a) Color image (b) GT disparity (c) Target image (d) Confidence (e) Our result

Figure 2: Stereo Optimization: The top row shows our result in (e) which is computed using the color image (a) used to define color distance in the domain transform, and target (c) disparity obtained from MC-CNN [44]. The confidence map (d) is used to weigh the target disparity in the optimization (Eq. (1)). Notice in the zoomed regions that our results are aligned to the edges of the color image.

(a) Jadeplant: back carton in focus (b) Jadeplant: blue block in focus (c) Playroom: chairs in focus

Figure 3: Render from defocus for the Middlebury scenes. The original is all-in-focus. (a) Original vs. ours, background in focus. (b) Original vs. ours, foreground in focus. (c) Original vs. ours, with the chairs in the front in focus.

	Algorithm	MAE (px) no occ | all	RMSE (px) no occ | all	sec/MP	sec/GD
Training	[44]	3.81 | 11.8	18.0 | 36.6	83.3	259
	[44] + FBS [4]	2.60 | 6.66	10.2 | 20.9	42.7 (126)	153 (412)
	[44] + DTS (ours)	3.02 | 9.12	10.8 | 27.4	5.9 (89.2)	19 (278)
Testing	[44]	3.82 | 17.9	21.3 | 55.0	112	254
	[44] + FBS [4]	2.67 | 8.19	15.0 | 29.9	28 (140)	91 (345)
	[44] + DTS (ours)	3.78 | 14.6	17.6 | 43.4	10 (122)	23 (277)

Table 1: Performance comparison on images from the Middlebury dataset. The timing for FBS and our method shows the additional time spent in processing MC-CNN [44], and the total value in the parentheses. Our method takes a fraction of time as compared to FBS [4] to obtain a significant reduction in error versus MC-CNN.

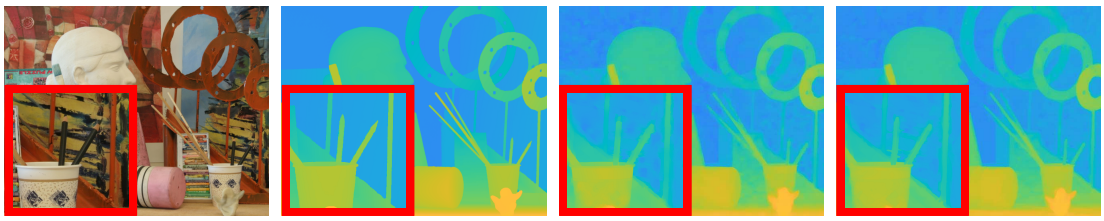
the Jadeplant and Pipes scenes to highlight that we improve the target disparity maps from MC-CNN [44] to estimate sharp depth edges.

Depth super-resolution We use the dataset introduced by Ferstl *et al*. [16] to evaluate our method for depth super-resolution. This dataset consists of three scenes (Art, Books, and Moebius) with added noise at 2, 4, 8, and 16x levels of upsampling. We used $\sigma_x = \sigma_y = 8 \times f \; px$ where f is the level of upsampling. We used $\sigma_r = 0.1$ with YCbCr colors normalized to a range of $[0, 1]$, $\lambda = 0.99$, and 10 iterations of the gradient descent with a step size of 0.99.

In Table 2, we present the RMS for each scene and mean geometric error for the dataset. The bicubic, DT and FBS results were produced by using the data and code provided by Barron *et al*. [2] (marked with †in Table 2). We also used the same code to evaluate our method. Both our method and the FBS use the bicubic upsampling as the target image. Our time is computed as the average over all images over 10 trials and include the 7 ms required for bicubic upsampling. Our method is 12 times faster than Barron *et al*. [4] while achieving comparable performance on most images, especially for higher upsampling factors. The DTS is also more accurate and 2x faster than HFBS [26], which is one of the fastest parallel edge-aware optimizers.

We also compare against the available results for Wei *et al*. [24], who use the mean absolute difference metric and report on $2\times$, $4\times$, and $8\times$ upsampling. For $(r = 1, \tau = 1)$, their SG-WLS results in a MAD of 1.812, and for $(r = 4, \tau = 4)$, they report a MAD of 1.51. In comparison, the DTS has a MAD of 1.48.

Colorization Fig. (6) shows an example of our results, and we provide additional results in the supplementary material. To simulate infinite confidence, we overwrite the solution with the strokes at each iteration of the gradient descent. We used the following parameters for this experiment: $\sigma_x = \sigma_y = 64$, $\sigma_r = 0.25$, and 100 iterations in the gradient descent scheme. Our method performs the compu-



(a) Color image. (b) GT disparity (c) Our result (d) Barron and Poole [4].

Figure 4: Depth super-resolution: (a) original color image, (b) ground-truth disparity, (c) our optimized disparity, and (d) results using FBS obtained from the author's website [2]. The inset highlights the details and the amount of smoothness we obtain in homogeneous regions while being edge-aware.

	Art				Books				Moebius				RMS	Time
	2x	4x	8x	16x	2x	4x	8x	16x	2x	4x	8x	16x		(ms)
Bicubic	5.32	6.07	7.27	9.59	5.00	5.15	5.45	5.97	5.34	5.51	5.68	6.11	5.94	7†
BGU [7]	4.77	6.63	9.39	14.17	2.35	3.57	5.93	10.41	2.19	3.19	5.38	9.00	5.48	-
BGU-w[7]	5.06	6.94	9.42	14.21	2.57	3.83	5.95	10.41	2.36	3.39	5.40	8.96	5.67	-
DT [17]	3.95	4.91	6.33	8.78	1.80	2.40	3.23	4.43	1.83	2.44	3.41	4.70	3.60	21†
FBS [4]	3.02	3.91	5.14	7.47	1.41	1.86	2.42	3.34	1.39	1.82	2.40	3.26	2.75	234†
HFBS [26]	4.73	5.56	6.38	8.32	2.14	2.60	3.08	4.04	2.25	2.67	3.18	4.11	3.74	49.86
DTS$_{L1}$ (Ours)	3.27	4.01	5.18	7.93	1.85	2.30	3.10	4.34	1.92	2.40	3.37	4.71	3.38	18.88
DTS (Ours)	3.77	4.29	5.15	7.56	1.78	2.18	2.81	3.91	1.78	2.20	2.96	4.23	3.24	19.29

Table 2: Performance of DTS on the depth super-resolution task. Our method is 12x faster than FBS while having comparable performance in most images, especially images with higher upsampling factors. † marks results taken from [4].

(a) Color image (b) GT disparity (c) Target image (d) Our result

Figure 5: Stereo optimization closeup for Middlebury Pipes scene. (a) Color image. (b) Ground-truth disparity. (c) Target obtained using MC-CNN [44]. (d) Our result.

(a) Grayscale image with color strokes. (b) Our result. (c) Levin *et al*. [22]

Figure 6: Colorization: (a) We use the input grayscale image as our reference image and the user annotated strokes as target, (b) our result using DTS and (c) Levin *et al*. [22] result. Note that (b) and (c) are virtually identical.

tation at 0.267s/MP, which is a more than 3x speedup in comparison to Barron and Poole [4]. These colorization images are less than 1k×1k in resolution and do not fully exploit the highly parallel nature of our algorithm.

High-resolution stereo When working with high-resolution data, we can truly exploit the high level of parallelism of our method. To demonstrate the parallelism of our approach, we tested our method on high-resolution depthmaps generated using the COLMAP 3D reconstruction software [35]. For this task, we created a dataset of 6000×4000 px images using a Nikon D5300 DSLR. During COLMAP's processing, we had to downsample the image sizes to 4500×2945 px after radial undistortion due to memory limits on the GPU. We used COLMAP to generate a dense depthmap per image of the dataset, which acted as target in our method. The confidence scores were calculated according to the variance in target depth, (Sec. 3.1). We refined the raw noisy depthmaps to obtain filtered depthmaps, which are visualized in Fig. 7 in the form of point clouds. As DTS can optimize the depths in parallel, we can achieve a high throughput. In particular, we require 0.0098s/MP to process the high-resolution imagery of this dataset. In contrast, FBS is difficult to parallelize due to the use of a sparse color grid which introduces non-coherent memory access and a hierarchical preconditioner that has interdependent gradients across multiple optimization variables [26]. We use the following

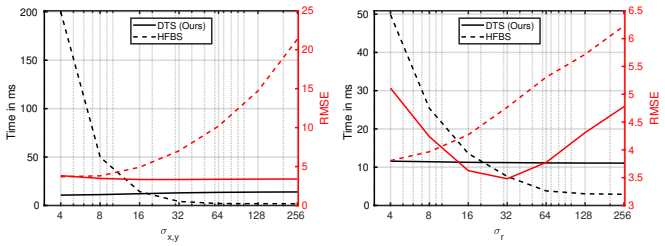
(a) Color images.

(b) Our result. (c) COLMAP raw geometric depthmap.

Figure 7: High-resolution stereo data: (a) Color image from the toy dataset, (b) point cloud visualization of our result, and raw depth generated by COLMAP [35] in (c). The point cloud views highlight that a significant amount of noise present in the raw depthmap is removed using DTS.

hyper-parameters for this experiment: $\sigma_x = \sigma_y = 64$, $\sigma_r = 0.25$, $\sigma_c = 32$ and used 10 gradient descent iterations.

Benchmarking We benchmark our parallel approach using the depth super-resolution dataset and compare against a parallel implementation of HFBS. As HFBS relies on a grid, its speed is dependent on the grid size and, in turn, the blur kernel size used to determine the voxel sizes. For an image with N pixels and D_i possible values in each dimension i, HFBS, like any other grid based method [27, 11, 4], partitions the d-dimensional space into a grid using the blur radii $\{r_i\}$. For HFBS, $d = 3$, but in general, it spends $\mathcal{O}(\prod_i^d (D_i/r_i))$ time alone to create the grid. Because of this, we observe an exponential increase in runtime as the $\sigma_{x,y} = r_i$ decrease, see Fig. 8a. On the other hand, we are independent of r_i, and hence the time is constant for DTS. At the same time, our method is more accurate than HFBS. We evaluate the runtime and RMSE for $\sigma_{x,y} = 8, 16, 32, 64, 128, 256$, and keep the remainder of



(a) Dependence of time (black) and RMSE (red) on $\sigma_{x,y}$. (b) Dependence of time (black) and RMSE (red) on σ_r.

Figure 8: (a) Our method is independent of blur σ and remains more accurate than HFBS [26].

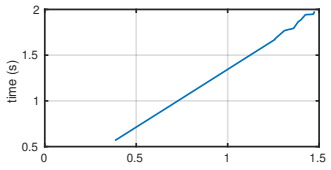


Figure 9: Dependence on image resolution for DTS.

the parameters the same as in our earlier experiments. The time is averaged over 10 trials. We observe a similar pattern by changing σ_r with $\sigma_{x,y} = 8$ (Fig. 8b).

Scale Now we present how our method scales with increasing image resolution. In practice, our method scales linearly with the number of pixels in the image. Fig. 9(a) shows the dependence of time in seconds on the number of pixels in the image. We use the training images from the Middlebury dataset and only show the time consumed by DTS for the stereo task at 3000 iterations. Due to this linear dependence, our framework is also suitable for high-resolution imagery.

6. Conclusion

We have presented a novel edge-aware solver that achieves scalable performance across a variety of applicable tasks. Our method is faster by an order of magnitude compared to the state of the art while performing at comparable accuracy. The approach is highly parallelizable and scales well w.r.t image resolution and outshines at high resolution images. Furthermore, unlike existing methods, it is independent of blurring kernel size. Our framework is faster and accurate than previous parallel edge-aware alglorithms. To extend our approach, a future step is to move to multi-GPU setting, as well as use advanced optimization methods like conjugate gradient descent to obtain faster convergence.

Acknowledgements This research was partially supported by NSF grant No. CNS-1405847.

References

[1] Andrew Adams, Jongmin Baek, and Myers Abraham Davis. Fast high-dimensional filtering using the permutohedral lattice. In *Computer Graphics Forum*, volume 29 (2), pages 753–762. Wiley Online Library, 2010. 2

[2] Jon Barron. https://jonbarron.info/. https://drive.google.com/file/d/0B4nuwEMaEsnmaDI3bm5VeDRxams/view?usp=sharing, 2008. Online; accessed 8-March-2018. 6, 7

[3] Jonathan T Barron, Andrew Adams, YiChang Shih, and Carlos Hernández. Fast bilateral-space stereo for synthetic defocus. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4466–4474, 2015. 3, 5

[4] Jonathan T Barron and Ben Poole. The fast bilateral solver. In *European Conference on Computer Vision*, pages 617–632. Springer, 2016. 3, 4, 5, 6, 7, 8

[5] Michael Bleyer, Christoph Rhemann, and Carsten Rother. Patchmatch stereo-stereo matching with slanted support windows. In *Bmvc*, volume 11, pages 1–11, 2011. 1, 3

[6] Cesar Cadena and Jana Košecká. Semantic segmentation with heterogeneous sensor coverages. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 2639–2645. IEEE, 2014. 2

[7] Jiawen Chen, Andrew Adams, Neal Wadhwa, and Samuel W Hasinoff. Bilateral guided upsampling. *ACM Transactions on Graphics (TOG)*, 35(6):203, 2016. 7

[8] Jiawen Chen, Sylvain Paris, and Frédo Durand. Real-time edge-aware image processing with the bilateral grid. In *ACM Transactions on Graphics (TOG)*, volume 26 (3), page 103. ACM, 2007. 2

[9] Liang-Chieh Chen, Jonathan T Barron, George Papandreou, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with task-specific edge detection using cnns and a discriminatively trained domain transform. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4545–4554, 2016. 3

[10] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv preprint arXiv:1606.00915*, 2016. 2

[11] Frédo Durand and Julie Dorsey. Fast bilateral filtering for the display of high-dynamic-range images. In *ACM transactions on graphics (TOG)*, volume 21 (3), pages 257–266. ACM, 2002. 2, 8

[12] Michael Elad. On the origin of the bilateral filter and ways to improve it. *IEEE Transactions on image processing*, 11(10):1141–1151, 2002. 2, 3

[13] Zeev Farbman, Raanan Fattal, Dani Lischinski, and Richard Szeliski. Edge-preserving decompositions for multi-scale tone and detail manipulation. In *ACM Transactions on Graphics (TOG)*, volume 27 (3), page 67. ACM, 2008. 2

[14] Raanan Fattal. Edge-avoiding wavelets and their applications. *ACM Transactions on Graphics (TOG)*, 28(3):22, 2009. 1

[15] Raanan Fattal, Maneesh Agrawala, and Szymon Rusinkiewicz. Multiscale shape and detail enhancement from multi-light image collections. In *ACM Transactions on Graphics (TOG)*, volume 26 (3), page 51. ACM, 2007. 3

[16] David Ferstl, Christian Reinbacher, Rene Ranftl, Matthias Rüther, and Horst Bischof. Image guided depth upsampling using anisotropic total generalized variation. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 993–1000. IEEE, 2013. 5, 6

[17] Eduardo SL Gastal and Manuel M Oliveira. Domain transform for edge-aware image and video processing. In *ACM Transactions on Graphics (ToG)*, volume 30, page 69. ACM, 2011. 1, 2, 4, 7

[18] Michaël Gharbi, Jiawen Chen, Jonathan T Barron, Samuel W Hasinoff, and Frédo Durand. Deep bilateral learning for real-time image enhancement. *ACM Transactions on Graphics (TOG)*, 36(4):118, 2017. 2

[19] Heiko Hirschmuller and Daniel Scharstein. Evaluation of cost functions for stereo matching. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007. 4

[20] Kourosh Khoshelham and Sander Oude Elberink. Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors*, 12(2):1437–1454, 2012. 5

[21] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *Advances in neural information processing systems*, pages 109–117, 2011. 1, 2

[22] Anat Levin, Dani Lischinski, and Yair Weiss. Colorization using optimization. In *ACM Transactions on Graphics (ToG)*, volume 23 (3), pages 689–694. ACM, 2004. 1, 5, 7

[23] Sifei Liu, Shalini De Mello, Jinwei Gu, Guangyu Zhong, Ming-Hsuan Yang, and Jan Kautz. Learning affinity via spatial propagation networks. In *Advances in Neural Information Processing Systems*, pages 1520–1530, 2017. 2

[24] Wei Liu, Xiaogang Chen, Chuanhua Shen, Zhi Liu, and Jie Yang. Semi-global weighted least squares in image filtering. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 6

[25] Jiangbo Lu, Hongsheng Yang, Dongbo Min, and Minh N Do. Patch match filter: Efficient edge-aware filtering meets randomized search for fast correspondence field estimation. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1854–1861. IEEE, 2013. 1

[26] Amrita Mazumdar, Armin Alaghi, Jonathan T Barron, David Gallup, Luis Ceze, Mark Oskin, and Steven M Seitz. A hardware-friendly bilateral solver for real-time virtual reality video. In *Proceedings of High Performance Graphics*, page 13. ACM, 2017. 3, 6, 7, 8

[27] Sylvain Paris and Frédo Durand. A fast approximation of the bilateral filter using a signal processing approach. In *European conference on computer vision*, pages 568–580. Springer, 2006. 2, 3, 8

[28] Sylvain Paris, Pierre Kornprobst, Jack Tumblin, Frédo Durand, et al. Bilateral filtering: Theory and applications. *Foundations and Trends® in Computer Graphics and Vision*, 4(1):1–73, 2009. 2, 3

[29] Pietro Perona and Jitendra Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on pattern analysis and machine intelligence*, 12(7):629–639, 1990. 1

[30] Tuan Q Pham and Lucas J Van Vliet. Separable bilateral filtering for fast video preprocessing. In *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*, pages 4–pp. IEEE, 2005. 2

[31] Fatih Porikli. Constant time o (1) bilateral filtering. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008. 2

[32] Jerome Revaud, Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid. Epicflow: Edge-preserving interpolation of correspondences for optical flow. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1164–1172, 2015. 1

[33] Daniel Scharstein, Heiko Hirschmüller, York Kitajima, Greg Krathwohl, Nera Nešić, Xi Wang, and Porter Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *German Conference on Pattern Recognition*, pages 31–42. Springer, 2014. 1, 5

[34] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 47(1-3):7–42, 2002. 4

[35] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 7, 8

[36] Deqing Sun, Stefan Roth, and Michael J Black. Secrets of optical flow estimation and their principles. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2432–2439. IEEE, 2010. 5

[37] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *Computer Vision, 1998. Sixth International Conference on*, pages 839–846. IEEE, 1998. 1, 2, 3

[38] Ben Weiss. Fast median and bilateral filtering. In *Acm Transactions on Graphics (TOG)*, volume 25 (3), pages 519–526. ACM, 2006. 2

[39] Huikai Wu, Shuai Zheng, Junge Zhang, and Kaiqi Huang. Fast end-to-end trainable guided filter. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1838–1847, 2018. 2

[40] Li Xu, Jimmy Ren, Qiong Yan, Renjie Liao, and Jiaya Jia. Deep edge-aware filters. In *International Conference on Machine Learning*, pages 1669–1678, 2015. 2

[41] Qingxiong Yang, Narendra Ahuja, and Kar-Han Tan. Constant time median and bilateral filtering. *International Journal of Computer Vision*, 112(3):307–318, 2015. 2

[42] Qingxiong Yang, Shengnan Wang, and Narendra Ahuja. Svm for edge-preserving filtering. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1775–1782. IEEE, 2010. 2

[43] Liron Yatziv and Guillermo Sapiro. Fast image and video colorization using chrominance blending. *IEEE transactions on image processing*, 15(5):1120–1129, 2006. 5

[44] Jure Zbontar and Yann LeCun. Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research*, 17(1-32):2, 2016. 1, 4, 5, 6, 7

[45] Ming Zhang and Bahadir K Gunturk. Multiresolution bilateral filtering for image denoising. *IEEE Transactions on image processing*, 17(12):2324–2333, 2008. 2