Efficient dense blur map estimation for automatic 2D-to-3D conversion

Efficient Dense Blur Map Estimation for Automatic 2D-to-3D Conversion

L.P.J. Vosters[a,b] and G. de Haan[a]

[a]Eindhoven University of Technology, Den Dolech 2, Eindhoven, the Netherlands
[b]Axon Digital Design, Lange Wagenstraat 55, Gilze, the Netherlands

ABSTRACT

Focus is an important depth cue for 2D-to-3D conversion of low depth-of-field images and video. However, focus can be only reliably estimated on edges. Therefore, Bea et al. [1] first proposed an optimization based approach to propagate focus to non-edge image portions, for single image focus editing. While their approach produces accurate dense blur maps, the computational complexity and memory requirements for solving the resulting sparse linear system with standard multigrid or (multilevel) preconditioning techniques, are infeasible within the stringent requirements of the consumer electronics and broadcast industry. In this paper we propose fast, efficient, low latency, line scanning based focus propagation, which mitigates the need for complex multigrid or (multilevel) preconditioning techniques. In addition we propose facial blur compensation to compensate for false shading edges that cause incorrect blur estimates in people's faces. In general shading leads to incorrect focus estimates, which may lead to unnatural 3D and visual discomfort. Since visual attention mostly tends to faces, our solution solves the most distracting errors. A subjective assessment by paired comparison on a set of challenging low-depth-of-field images shows that the proposed approach achieves equal 3D image quality as optimization based approaches, and that facial blur compensation results in a significant improvement.

Keywords: 2D-to-3D conversion, focus/defocus, sparse-to-dense, propagation, depth map, optimization

1. INTRODUCTION

Following the successful introduction of color 4 decades ago (CTV), and high definition (HDTV) in the last decade, many researchers are currently exploring opportunities for 3-dimensional television (3DTV). As was the case for the earlier revolutions, rapid introduction of 3DTV is hindered by the chicken-and-egg problem of the industry [1]; lacking content prevents customers from buying sets, while absence of receivers prevents broadcasters from investment. In this paper, we look at the common solution [1] to break this chicken-and-egg problem, i.e. real-time 2D-to-3D conversion.

The main challenge in 2D-to-3D conversion lies in estimating depth in a monoscopic video. In state-of-the-art 2D-to-3D conversion both motion based and pictorial cue based methods can be distinguished. In motion based algorithms camera and object motion are used to estimate depth. Pictorial-cue based methods use depth cues, like focal-blur, perspective, texture-density, occlusion, etc. An excellent overview on state-of-the-art 2D-to-3D conversion techniques can be found in [2].

In photography and cinematography and live broadcast many images and image sequences are recorded with a low depth-of-field (DOF) camera. A low DOF draws the viewer's attention in the image to objects which appear sharp, while distracting cluttered backgrounds appear blurred. Since the amount of camera out-of-focus blur is directly related to the distance of an object to the camera, it is an important depth cue in monoscopic low depth-of-field video and images.

Since out-of-focus blur can only be reliably estimated on edges, most methods estimate focus on edges and propagate the blur by solving a quadratic optimization problem [3, 4, 5], which corresponds to solving a sparse system of linear equations. The idea for blur propagation originated from Levin et al. [6] who posed the

Further author information: (Send correspondence to)
L.P.J. Vosters: E-mail: l.p.j.vosters@tue.nl, Telephone: +31 40 247 3614

colorization of a gray scale image as a similar optimization problem. In their method scribbles, i.e. sparse color indicators, are propagated to non-scribbled image areas based on the luminance channel.

Numerical optimization techniques for solving a large sparse systems of linear equations include geometric multigrid (GMG) [7, 8], algebraic multigrid (AMG) [8, 9] and (multilevel) preconditioning [10]. While these techniques have O(N) complexity they remain computationally expensive. GMG requires several pre and post smoothing iterations of a relaxation method (i.e. Jacobi, Gauss-Seidel or Conjugate Gradient) on each grid. Preconditioning techniques require calculating an approximation to the inverse of the coefficient matrix, to speed up the convergence of an iterated relaxation method, prior to relaxation. For fast convergence the preconditioner (approximate inverse) should be as close as possible to the true inverse of the coefficient matrix. However, calculating a more accurate inverse comes with increased computational cost. Furthermore the application of the preconditioner, which might not be sparse, in each relaxation iteration significantly contributes to the computational cost. Due to their high computational complexity, optimization based solvers often solve the sparse system on a low resolution grid followed by (joint) upsampling.

To mitigate the need for complex multigrid or (multilevel preconditioning) we propose in this paper an efficient recursive low cost, line scanning based focus propagation method, which estimates the dense blur map on a low resolution grid, followed by joint bilateral upsampling (JBU). In contrast with traditional upsampling methods like bilinear, bicubic and Lanczos interpolation, JBU uses the available high resolution input image to output a high quality dense blur map that features sharp edges. Furthermore, we show that JBU suppresses inaccurate blur estimates in the low resolution sparse blur map, from appearing in the high resolution dense blur map. Additionally we show how JBU can be implemented into the real-time O(1) bilateral filtering framework of Yang et al. [11].

The quality of the propagated blur map strongly depends on the accuracy of the estimated edge blur. However, state-of-the-art (sparse) blur estimators cannot distinguish between camera out-of-focus blur on the one hand and motion, shading or penumbral blur on the other [12], since they do not accurately model intrinsic scene geometry and illumination which underlie the blur formation process in an image. Hence, when the blur in a foreground object is caused by one of the latter cases, the dense blur map will contain areas in which the blur and thus the depth estimate is too high. Shading and penumbral blur particularly occur in people's faces (Fig. 1), and can cause a person's body to appear closer than the face, cause significant depth changes between hair and face, or cause the eyes, mouth and face contours to have a different depth value. In addition inconsistent facial blur leads to inconsistent depth which causes annoying facial deformations in the rendered stereoscopic image pair. Since the visual attention of people mostly tends towards faces this leads to unnatural 3D effect and significantly degraded image quality, in the rendered 3D image pair. To solve this problem, we propose a facial blur compensation scheme. This scheme applies face detection and skin tone detection to remove erroneous blur estimates from the facial region in the sparse blur map. Our subjective evaluation in Section 4 shows this facial blur compensation solves the most distracting errors.

In the remainder of the paper we briefly introduce blur estimation and blur propagation by optimization in Section 2. Section 3 introduces the proposed blur propagation, efficient O(1) JBU, and facial blur compensation method. Finally, in Section 4 we present the results of a subjective evaluation by paired comparison in which we compare the proposed blur propagation method with an optimization based propagation method, and we demonstrate the effectiveness of the facial blur compensation.

2. OPTIMIZATION BASED DENSE BLUR MAP ESTIMATION

2.1 Blur Estimation

The amount of defocus is related to the distance of an imaged point to the camera by

$$D_{cam} = Fv_0/(v_0 - F - uf), \tag{1}$$

where D_{cam} is the distance of an imaged point to the camera, F is the focal length of the camera, v_0 the distance between the lens and the image plane, f the f-number of the lens system and u the blur induced by the point spread function [13]. The out-of-focus image can be modeled as a convolution between the in focus image and a Gaussian point spread function. Moreover, by estimating blur, the relative depth in the scene can be obtained.

Most parametric blur estimators assume a step edge model consisting of a contrast, edge offset and edge center parameter. Edge blur is modeled by convolving the edge with a Gaussian filter of standard deviation u. Therefore, parametric blur estimators can only retrieve blur on edges.

The multipoint method in [14, 15] first detects edges with a Canny edge detector [16], that convolves the image with the first derivative of a Gaussian function. Then for each edge point the response is sampled on 3 equidistant points in the direction of the edge gradient, and the blur parameter is calculated in closed form. Mislocalization of edges can lead to inaccurate blur estimates. The multipoint method neither relies on the maximum of a ratio at an edge position [17, 18], nor on the location of the extrema of the second derivative [12], which estimation is affected by noise. Instead, the blur is calculated in closed form from three equidistant sample points, resulting in more accurate blur estimates, also for edges at off grid positions. The multipoint method has low computational complexity compared to [12, 17], because only two separable derivative of Gaussian filter kernels with small kernel sizes* need to be applied to the image. Therefore we select the multipoint method as blur estimator.

2.2 Optimization based blur propagation to non-edge image portions

From the previous section it is clear that blur can only be reliably estimated on edges. Therefore, blur estimation should be followed by a blur propagation step, to propagate the estimated blur into the interior of objects. Inspired by the colorization paper of Levin et al. [6] blur propagation is posed as the following optimization problem [3, 4]:

$$\min_{\vec{u}} J(\vec{u}) = \min_{\vec{u}} \sum_{i,j \in I} w_{ij}(u_i - u_j)^2 + 2\lambda \sum_i (u_i - u_i^c)^2 b_i, \tag{2}$$

where u_i is the blur to be estimated at pixel i, $\vec{u}$ is a vector containing the lexicographical ordering of u_i, u_i^c the estimated blur at pixel i, b_i a binary indicator that is 0 when no blur is estimated at pixel i and 1 otherwise, I the set of all pixels in the image, w_{ij} the weight between pixel i and j, and λ a constant. The first term on the right hand side of Eq.2 enforces that neighboring pixels with similar color have similar blur. The second term is a soft constraint that enforces u_i to have a blur close to u_i^c depending on the value of λ. This optimization problem is solved by setting the gradient of Eq.2 with respect to $\vec{u}$, to $\vec{0}$. This results in solving the following sparse system of linear equations

$$\begin{array}{lcl} B & = & diag(b_1, ..., b_N), \\ (D - W + \lambda B)\vec{u} & = & \lambda B \vec{u_C}, \\ D & = & diag(\sum_{j \in \Omega_1} w_{1j}, ..., \sum_{j \in \Omega_N} w_{Nj}), \\ W & = & \{w_{ij}\}, \forall (i,j) \in I, \\ B & = & diag(b_1, ..., b_N), \end{array} \tag{3}$$

where u_C is a vector containing u_i^c, Ω_i is the 2D square neighborhood centered around pixel i and N is the number of pixels in the image. Additionally, the weighting function can be calculated by [3, 4]

$$w_{ij} = exp(-\frac{||\vec{I_i} - \vec{I_j}||_2^2}{2\sigma_r^2}) / \sum_{j \in \Omega_i} exp(-\frac{||\vec{I_i} - \vec{I_j}||_2^2}{2\sigma_r^2}), \tag{4}$$

where, $\vec{I_i}$ is a vector containing the rgb color components at pixel i. The weights in Eq.4 correspond exactly to the weights of the edge preserving bilateral filter in [19]. Another edge preserving weighting function is

$$w_{ij} = \frac{1}{|\Omega_i|^2} \sum_{k:(i,j) \in \Omega_i} (1 + (\vec{I_i} - \vec{\mu}_k)^T (\Sigma_k + \epsilon U_3)^{-1} (\vec{I_i} - \vec{\mu}_k)), \tag{5}$$

where μ_k and Σ_k correspond to the mean and variance in the neighborhood of k, U_3 is the 3 by 3 identity matrix, ϵ is a smoothing parameter similar to σ_r in Eq. 4. Applying this weighting function corresponds to substituting

*The filter kernel is approximated by truncating the Gaussian kernel at 3 times it standard deviation. The length of a typical filter kernel in the multipoint method ranges from 7 to 13 pixels, for a derivative of Gaussian function with a standard deviation of 1.0 to 2.0 pixels respectively.

the matrix $D - W$ in Eq. 3 by the matting Laplacian matrix L of [20], which assumes a local linear model between blur and input image. For blur propagation this approach has been used in [18]. Direct calculation of the matting Laplacian's weights is computationally expensive. Instead efficient linear time algorithms exist, which efficiently calculate the matrix vector product Lu[†] with box filters [23] when solving the sparse system $L\vec{u} = \lambda\vec{u_C}$. In addition Lu can be regarded as the output of the guided filter applied to $\vec{u}$ [24].

Two efficient classes of iterative techniques exist for solving large sparse systems of linear equations like Eq. 3. The first are multigrid techniques, which recursively alternate between solving the sparse system on low and high resolution grids, to reduce the low frequency and high frequency error in the solution respectively [7, 10]. An example of a popular geometric multigrid (GMG) approach to solve optimization problems arising in computer vision problems is [7, 8]. An algebraic multigrid (AMG) approach, in which the interpolation and restriction functions are adapted to local structure of the sparse coefficient matrix, can be found in [8, 9]. The second class of techniques use (multilevel) preconditioning on the sparse coefficient matrix, to effectively reduce its condition number. The preconditioner accelerates the convergence of a relaxation method like Jacobi, Gauss-Seidel (GS) or CG descent, by making the descent direction more independent and better scaled [10].

The difference between GMG and AMG lies in the definition of the grids and the interpolation and restriction operators. In geometric multigrid the unknown variables of $\vec{u}$ are defined at known spatial locations. Restriction and interpolation operators are comparable to image upscaling and downscaling with fixed weights that are constant among grids, respectively. However in AMG the weights for interpolation and restriction adapt to the local structure of the matrix A and vary among grids. AMG computes the coarse grid as a subset of $\vec{u}$, by coarsening in the direction of geometrically smooth error [8, 9]. Thus the computational complexity and memory storage capacity are predictable for GMG. This is not the case for AMG, since the number of grid points in the coarse grid is not known in advance, and depends on the problem under consideration. This unpredictable nature of AMG is undesirable for real-time systems.

Instead of solving the original linear equation $A\vec{v} = \vec{f}$ multilevel preconditioning techniques solve $P^{-1}(A\vec{v} - \vec{f}) = 0$, which has the same solution as long as P is nonsingular. If $P^{-1}A$ has a smaller condition number than A, the preconditioned system will converge faster in an iterative solver. Since P^{-1} has to be applied to a vector at each iteration, P^{-1} should be sparse. However, for fast convergence P^{-1} should be close to the inverse of A, which might not be sparse. These are two conflicting demands. Computing the preconditioner can be computationally expensive [21]. Also the addition of an extra matrix vector product in each iteration, significantly contributes to the computational complexity. A multilevel preconditioner that achieves rapid convergence on the colorization problem (1 to 2 iterations of CG), is given in [10]. However, the computation of the preconditioner requires slightly longer than 1 iteration of CG in the solver. In addition an iteration in this preconditioned CG is twice as slow as one in basic CG.

For multigrid the coefficient matrix A (i.e. 9 weights per pixel for an 8-connected neighborhood) and vector f need to be stored into memory. In addition to A for multilevel preconditioning also the preconditioner P needs to be stored. Furthermore these techniques have a latency that is larger than one frame, since the (multigrid) relaxation iterations can start only when the last pixel of the image is fetched. This last pixel is required for the solution on the coarsest grid in multigrid, and for calculating the preconditioner in (multilevel) preconditioning.

GMG has constant $\mathcal{O}(1)$ runtime. Multilevel preconditioners have near constant runtime since they converge within very few iterations [10]. However, calculating the preconditioner, or alternating between resolution grids requires considerable computational effort. Furthermore these techniques require at least one frame latency as setup time to compute the coarse grids (e.g. multigrid) or the preconditioner (e.g. preconditioned CG). They also require a significant amount of memory since the coefficient matrix A and f need to be stored in memory and be accessible at all times during processing. In addition the number of grids or preconditioning levels are often determined heuristically and are problem dependent. Consequently, multigrid and multilevel solvers are particularly attractive for user interactive applications like colorization [6, 25] or semi-automatic depth assignment [26] where they are solved at low resolution.

Multigrid and multilevel preconditioning solvers require very few iterations to converge. However, calculating

[†]In most of the iterative techniques used (e.g. conjugate gradient (CG) [21, 22]), these matrix vector products are the main computational bottleneck for solving large sparse systems of linear equations.

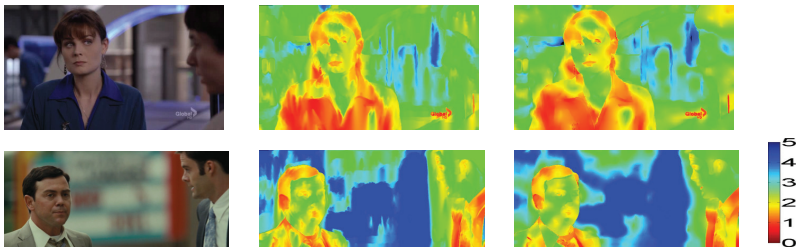


Figure 1. Dense blur estimates for low-depth-of-field images obtained by sweeper and CG on a single image. Input image (1st column), sweeper (2nd column), CG 300 iterations (3th column). The color bar encodes blur ranging from 0 to 5.

the preconditioner, or alternating between resolution grids requires an infeasible amount of computational effort and memory for real-time video processing. Furthermore the number of grids or preconditioning levels are often determined heuristically and are problem dependent. In the next section we present an efficient low cost, low latency, low memory, line scanning based focus propagation approach, that achieves equal visual quality compared to propagation by optimization. In contrast to multigrid or preconditioning techniques, it requires only a single iteration of Gauss-Seidel on the finest grid, and a very small amount of memory since it operates on only 3 scan lines at the same time. In addition no coarse grid selection, coarse grid correction, restriction, interpolation, post relaxation or preconditioning is required.

3. PROPOSED BLUR PROPAGATION

In this section we present the proposed blur propagation method and its efficient implementation. In addition we introduce facial blur compensation to remove shading and penumbral blur from people's faces. We obtain the sparse blur map by detecting edges and consequently calculating the amount of blur for each edge point with the method of van Beek [14].

3.1 Sweeper

Sweeper is inspired by GS relaxation [21]. For the sparse linear system in Eq. 3, the GS forward relaxation can be formulated as [21]

$$u_i^{k+1} = \frac{1}{\sum\limits_{j\in\Omega_i, j\neq i} w_{ij} + \lambda b_i} \left(- \sum_{j\in\Omega_i, j<i} w_{ij} u_j^{(k+1)} - \sum_{j\in\Omega_i, j>i} w_{ij} u_j^{(k)} + \lambda u_i^c \right), \tag{6}$$

which updates blur values in the order of $i = 1, ..., N$. Here k denotes the iteration number and i and j denote pixel position. Similarly the backward GS relaxation is given by

$$u_i^{k+1} = \frac{1}{\sum\limits_{j\in\Omega_i, j\neq i} w_{ij} + \lambda b_i} \left(- \sum_{j\in\Omega_i, j<i} w_{ij} u_j^{(k)} - \sum_{j\in\Omega_i, j>i} w_{ij} u_j^{(k+1)} + \lambda u_i^c \right) \tag{7}$$

which updates the blur values in the order of $i = N, ..., 1$. In symmetric Gauss-Seidel these recursions are alternately applied. Note that when the weights are chosen as in Eq. 4 and $\lambda = 0$, we obtain a spatially recursive bilateral filter. Note that λ determines the tradeoff between smoothing and the boundary condition.

The advantage of using the above relaxation method is that it can be implemented as a local filter on a scan line. Furthermore a single picture memory suffices, since the old blur values are overwritten by the updated ones. This makes Gauss-Seidel relaxation particularly well-suited to process videos from live cameras in an FPGA. Multigrid or multilevel relaxation methods have to delay processing until the whole frame is retrieved and loaded into memory.

Unfortunately GS, like any other relaxation method, converges slowly. This is because each iteration strongly smooths high frequency components in the error while it reduces low frequency components only slightly in

amplitude [7]. This problem is typically solved by applying multigrid or multilevel preconditioning techniques, however these techniques are not well suited for live video (See Section 2). We solve this problem by adapting Eq. 6 as follows

$$u_i^{k+1} = \frac{1}{\sum_{j \in \Omega_i, j \neq i} w_{ij} b_j + \lambda b_i} \left(- \sum_{j \in \Omega_i, j < i} w_{ij} u_j^{(k+1)} b_j - \sum_{j \in \Omega_i, j > i} w_{ij} u_j^{(k)} b_j + \lambda u_i^c \right), \tag{8}$$

We selected the weights according to Eq. 4. Then only pixels for which a blur value has been estimated or has already been calculated previously, are averaged with bilateral weights. Due to its recursive update strategy, which is closely related to 3D recursive search motion estimation techniques [27], the filter aggressively fills in homogeneous and textured regions while edges are preserved (see Fig. 1). In order not to favor blur estimates on either the left or right side of the image, we apply meander scanning similar to what was proposed by de Haan et al. [27], in which the even lines are processed from left to right, and the odd lines in opposite direction.

3.2 Efficient implementation of Sweeper

To reduce computational complexity we apply sparse blur propagation on a low resolution grid that is downscaled by $D_{fac} = 8$. To get a smoothed high resolution dense blur map in which edges are well aligned with object borders we apply JBU [28]. Since JBU already takes care of smoothing the propagated sparse feature map, we can choose a small window size ($s = 3$) for the propagation filter in Eq. 8.

We obtain a low resolution sparse blur map, by performing edge detection and blur estimation on the high resolution input image. Then we downscale the sparse blur map by $D_{fac} = 8$ in 3 step with a box filter that only averages the sparse blur values. Performing edge detection and blur estimation on the low resolution grid requires proper downscaling of the input image with a strong prefilter. Such filters are computationally expensive, and in addition they can cause ringing artifacts on edges which leads to inaccurate blur estimates. Furthermore, since the prefilter is not a brick wall filter, additional non-Gaussian blur is introduced on edges.

We propose an efficient low cost technique for JBU, that is based on a fast $\mathcal{O}(1)$ bilateral filtering framework. Recently developed, fast approximations of the (joint) bilateral filter include [11, 29, 30, 31, 32]. All these methods have $\mathcal{O}(1)$ runtime (except [29] which runtime is $\mathcal{O}(\log(2R+1))$), meaning that the number of computations per output pixel is constant and independent of the kernel radius. Yang et al. [11] approximate the bilateral filter with a set of filtered image components, from which the output is linearly interpolated. We choose this approximation because it has a lower computational complexity than the other approximations, and it is suitable for parallel implementation. Furthermore Yang et al.'s approximation is more accurate than the approximation of Porikli et al. [30] and Paris et al. [29], since it only quantizes the range function, while [30] also quantizes the range function and image intensities, and in addition [29] also reduces the spatial resolution of the input image. Although the approximation with trigonometric range kernels of Chaudhury et al. [32], is very accurate, computational complexity significantly increases for narrow Gaussian range kernels (i.e. $\sigma_t < 50$). Igarashi et al.'s $\mathcal{O}(1)$ approach [31], features low memory usage. However, when it is applied to JBU, it becomes $\mathcal{O}(2R+1)$ and thus computationally very demanding. To reduce the computational complexity of JBU, Riemens et al. upsample a low resolution depth map by a factor 8 in 3 intermediate steps. In each step the result of the previous step is upsampled by a factor 2 with a JBU filter which kernel size decreases after each step.

JBU can be mathematically expressed as

$$\tilde{u}_H(x,y) = \frac{\sum_{i_\downarrow, j_\downarrow = -R}^{R} f_S\left(\vec{0}, [i_\downarrow, j_\downarrow]^T\right) f_R\left(\vec{I}_H(x,y), \vec{I}(x_\downarrow + i_\downarrow, y_\downarrow + j_\downarrow)\right) \tilde{u}(x_\downarrow + i_\downarrow, y_\downarrow + j_\downarrow)}{\sum_{i_\downarrow, j_\downarrow = -R}^{R} f_S\left(\vec{0}, [i_\downarrow, j_\downarrow]^T\right) f_R\left(\vec{I}_H(x,y), \vec{I}(x_\downarrow + i_\downarrow, y_\downarrow + j_\downarrow)\right)}, \tag{9}$$

where, $\tilde{u}_H$ and $\tilde{u}$ denote the upsampled and low resolution dense blur map respectively, R is the kernel radius for joint upsampling, $\vec{I}_H(x,y)$ and $\vec{I}(x,y)$ denote pixels of the high and low resolution input guidance images

respectively, x, y, i and j denote integer high resolution pixel coordinates, $x_\downarrow$, $y_\downarrow$, $i_\downarrow$ and $j_\downarrow$ denote their corresponding low resolution, possibly fractional counterparts, and f_R and f_S denote the range and domain filter kernel's respectively. $\vec{I_H}(x,y)$ and $\vec{I}(x,y)$ are a vector for color and scalar for luminance images and $f_R(\vec{a},\vec{b}) = f_S(\vec{a},\vec{b}) = exp(-||\vec{a}-\vec{b}||_2^2 \cdot (2\sigma_{JBU}^2 f)^{-1})$. Because we cannot index $\tilde{u}$ with fractional coordinates, we round them to the nearest integer pixel value. This filter samples from a high and a low resolution grid at the same time. In contrast with Kopf et al., who sparsely sample the high resolution guidance image in the range kernel $f_R(\cdot)$ without proper prefiltering, we, like in multistep joint bilateral upsampling (MJBU) of Riemens et al. [33], use a prefiltered downscaled guidance image $\vec{I}(x,y)$ in the range kernel $f_R(\cdot)$ (Eq. 9), to avoid alias in the upsampled result. Note that the guidance image is already available because it is also used in spatial sparse feature propagation (Eq. 8).

We obtain the low resolution input guidance image, by downscaling the input image by $D_{fac} = 8$ in 3 steps. Similar to Riemens et al. [33], we first apply the filter kernel [1 3 3 1] on the rows and downsample the rows with a factor of 2. Then we filter the columns with the same filter and downsample them with a factor of 2. This procedure is repeated 3 times for each color channel.

We now show how Eq. 9 can be approximated by the fast bilateral filtering framework of Yang et al. [11]. Assuming a single channel luminance input image I for $\vec{I}$, we can express Eq. 9 by

$$\tilde{u}_H(x,y) = \begin{cases} (L_{k+1} - I_H(x,y))\, P^k(x_\downarrow, y_\downarrow) + (I_H(x,y) - L_k)\, P^{k+1}(x_\downarrow, y_\downarrow), \\ if \quad (I_H(x,y) > L_k) \wedge (I_H(x,y) < L_{k+1}) \end{cases} \tag{10}$$

$$L_k = \tfrac{I_{max} - I_{min}}{L-1} k, \quad k \in \{0, 1, ..., K-1\}, \tag{11}$$

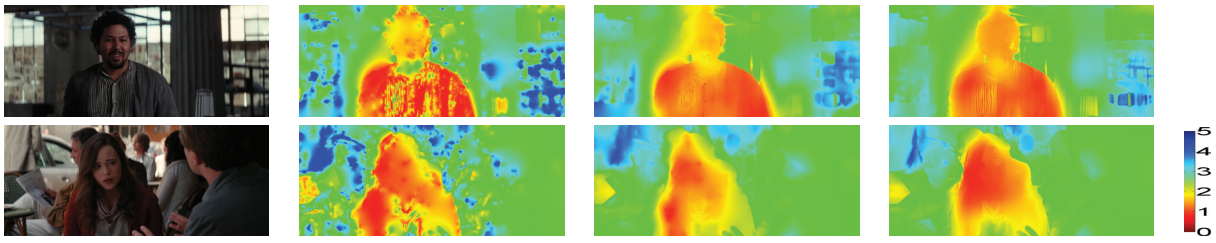
where, $I_{max} = 255$ and $I_{min} = 0$ are the maximum and minimum intensity values for 8 bits pixel depth. Furthermore $P^k(x,y)$ is the k^{th} out of K principle bilateral filtered image components (PBFIC), which can be computed by

$$P^k(x,y) = \left(\sum_{i=-R}^{R} \sum_{j=-R}^{R} f_R\left(k, I^{(t)}(x+i, y+j)\right) u^{(t)}(x+i, y+j) \right) \left(\sum_{i=-R}^{R} \sum_{j=-R}^{R} f_R\left(k, I^{(t)}(x+i, y+j)\right) \right)^{-1}. \tag{12}$$

Here we selected a box filter kernel for $f_S(\cdot)$. However, instead of a spatial box filter kernel, we can also approximate the the Gaussian spatial filter kernel of Eq. 9 in constant time, by filtering the PBFIC's with an infinite impulse response (IIR) approximation of the Gaussian filter similar to what Yang et al. [11] did.

Experimental results in [11] demonstrate that with only $K = 8$ PBFIC's, an accurate approximation of the bilateral filter is obtained. Practically for joint upsampling this corresponds to filtering the numerator and denominator of Eq.12 with a box filter for $k \in 0, 1..K-1$. Such box filter can be calculated efficiently in constant time, with just 4 additions per pixel [34]. Instead of a 3 channel (e.g. RGB) vector we use only the luminance channel in the range filter kernel in JBU. This is justified by the fact that most natural scenes do not contain very saturated colors. The use of a 3 channel color vector in Yang et al.'s approximation is prohibitive, because the number of PBFIC's will explode when more channels are added. E.g. a 3 channel color image with $K = 8$ PBFIC's per channel requires $8 \cdot 8 \cdot 8 = 512$ PBFIC's. We argue that this is a reasonable assumption since natural video often does not contain much (saturated) color. On the webpage [35], we provide a qualitative comparison between defocus sequences upsampled by Riemens et al. and the proposed JBU. This comparison shows that the proposed JBU unlike MJBU exhibits no edge serrations, while blur in objects is slightly less uniform.

Increasing the kernel size in Eq. 8, leads to smoother dense blur maps, because more noisy sparse estimates are averaged. Unfortunately, the computational cost of Sweeper increases quadratically with the window size. Instead of increasing the neighborhood Ω in Eq. 4, Bae et al. [3] and Zhang et al. [4] refine inaccurate sparse blur estimates, with a joint bilateral filter which has a kernel size of more than 10% of the image width. This filter significantly improves the dense blur estimates (Fig. 2), and like JBU it can also be implemented in the bilateral filtering framework of Yang et al. However JBU, which is applied directly after Sweeper, will in addition to upsampling also strongly suppress noisy sparse blur estimates. Hence the additional bilateral filter applied by



(a) (b) (c) (d)

Figure 2. Input image (column a), propagation result on half the original resolution without post processing of the sparse blur map (column b), propagation result on the original resolution where the sparse blur map was post-processed with a joint bilateral filter as in [3, 4, 18] (column c), propagation result on a low resolution followed by JBU without any post-processing on the sparse blur map (column d). For all images the blur was propagated with CG. The color bar encodes blur ranging from 0 to 5.

Bae et al. and Zhang et al. is redundant. Moreover the JBU filter allows a small neighborhood Ω for both CG and Sweeper, hence we select an 8-connected neighborhood, i.e. square 3x3 kernel, in our implementation. Fig. 2 shows that this approach gives equal results with significantly lower computational cost, since blur propagation and PBFIC calculation are restricted to the low resolution grid.

3.3 Face compensation

The underlying blur formation process in an image is very complex. This process is not taken into account in state-of-the-art blur estimators, that model camera-out-of-focus blur, as the convolution of a step edge with a Gaussian blur kernel [12], [17], [5], [18], [14]. This approximation only holds for edges in a constant depth plane. However, the blur formation process at depth contrast defining contours, and edges in an object's interior surface depend on scene and object geometry, illumination and motion. Consequently motion, shading and penumbral[‡] blur are not accounted for and can cause (false) edges with a blur value that is too high. Liu et al. [36] propose a blur classification method that can distinguish camera-out-of-focus blur from motion blur, however, in general state-of-the-art blur estimators cannot distinguish among camera-out-of-focus, shading, penumbral and motion blur.

In low-depth-of-field photo or video material, shading, penumbral and motion blur can cause false edges in people's faces that lead to blur discontinuities within the face and between the face, hair and body of a person in the propagated sparse blur map. Consequently when the estimated (8 bit) blur value is directly used as (8 bit) depth value, shading, penumbral and motion blur cause a person's body to appear closer than the face, cause significant depth changes between hair and face, or cause the eyes, mouth and face contours to have a different blur value. In addition inconsistent facial blur leads to inconsistent depth that causes unnatural facial deformations in the rendered stereoscopic image pair (Fig. 3). Since the visual attention of people mostly tends towards faces this leads to an unnatural 3D effects, eyestrain and a significantly degraded 3D image quality. To solve this problem, we propose a facial blur compensation scheme. This scheme applies face detection and skin tone detection to remove erroneous blur estimates from the facial region in the sparse blur map.

First we detect faces in a low resolution input image[§] with the Viola Jones face detector [37]. While this detector accurately detects faces, its detection window does not enclose all skin pixels and thus not all false edges of the facial region (Fig. 4). To detect all skin pixels in a facial region we build a skin tone color model. Therefore, we convert the RGB input image into the HSV colorspace and compute a 1D, 30 bin, histogram of the hue, sampled only from pixels in the detection windows designated as facial region. Hue is used because it is more insensitive to changes in lighting than RGB [38]. Then by calculating the backprojection of the histogram onto the Hue of each pixel, we obtain a skin probability image. This skin probability image is then used by CAMSHIFT [38] to further refine the size and location of the detection window of each face. We use the location

[‡]penumbras are caused by shadows of light sources with non-zero radius

[§]The input image has already been downscaled with D_{fac} in the blur propagation step. For face detection we use this downscaled input image.

Figure 3. Result of facial blur compensation on stereoscopic images. (Left) no facial blur compensation. (Right) proposed facial blur compensation. See Fig. 5 for the blur maps of these test images. We recommend to view this figure in the paper's electronic version.

and size of each detection window containing a face, as initial search window location and size in CAMSHIFT [38] respectively. For each face CAMSHIFT outputs the local skin distribution's length, width, centroid and rotation angle, which define an ellipse, that tightly encloses (most part of) that face (Fig. 4). Finally for each ellipse E_j we discard all sparse blur estimates u_i^c, which have a value higher than $U_T^{E_j}$, where

$$\begin{aligned} U_T^{E_j} &= S^{E_j}(\lfloor |S^{E_j}| \cdot T \rfloor), \\ S^{E_j} &= sort\left(\{u_i^c | \forall i \in E_j\}\right), \end{aligned} \tag{13}$$

where i indexes all sparse blur estimates u_i^c in ellipse E_j, j indexes all ellipses in the image, u denotes the value of the sparse blur estimate, the function $sort(\cdot)$ sorts all elements in the set $\{u_i^c | \forall i \in E_j\}$, in ascending order by blur value, $|\cdot|$ returns the number of elements in a set, and $T \in [0, 1]$ is a constant. If for example $T = 0.3$ than Eq. 13 removes all sparse blur estimates that are higher than the 30% lowest sparse blur estimates. Hence Eq. 13 removes most sparse blur estimates that are caused by shading and penumbral blur within each ellipse (Fig. 4).

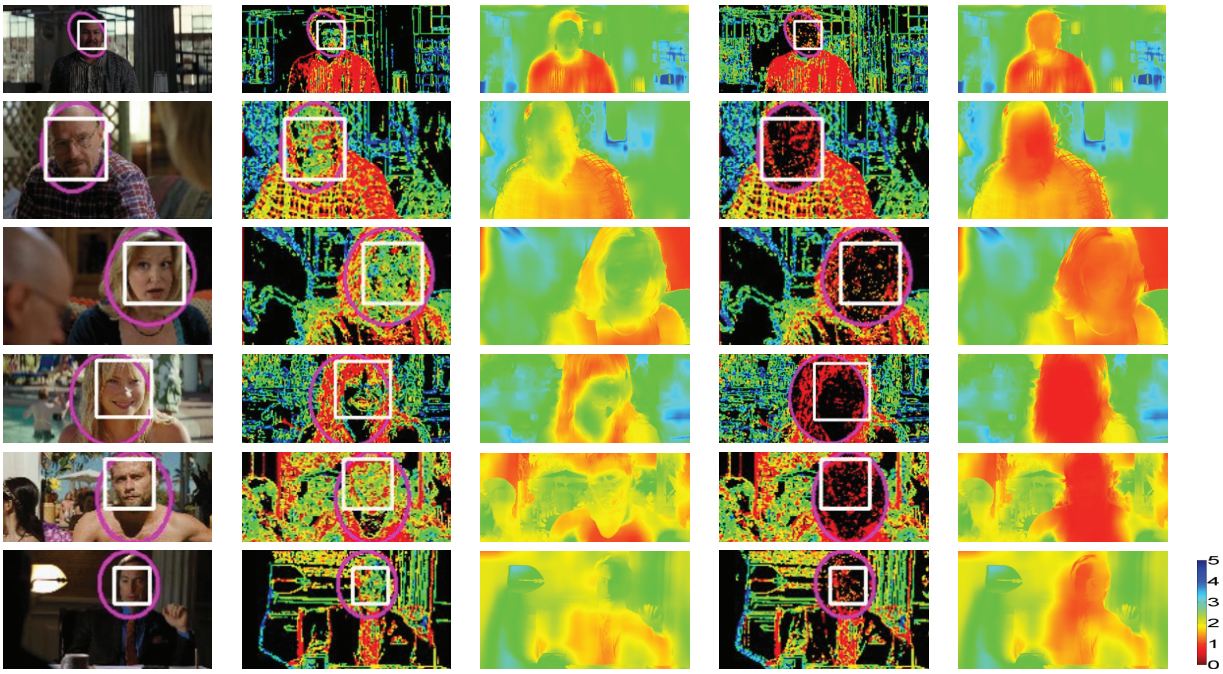
Face detection is cheap through the use of integral images and the cascade classifier [37]. The calculation of the 30-bin Hue histogram and its backprojection to the rest of the image in CAMSHIFT are low cost operations. In addition the first and second moment calculation in meanshift's iterations is independent of the window size when using integral images. Furthermore, like blur propagation, face detection and CAMSHIFT are applied to the input image that is downscaled by D_{fac}. As a result the proposed facial compensation approach features very low additional computational complexity, and can be performed even on a modern CPU in the order of a few milliseconds.

3.4 Performance

A prototype of Sweeper was implemented in unoptimized C++ code on a CPU. We took a GPU implementation of Jacobi preconditioned CG from [39], and ran the CG solver on the GPU. Both Sweeper and CG were run on a laptop with an Nvidia Quadro NVS 160M and an Intel Core 2 Duo 2.80GHz. We used only a single core of the CPU. We obtain frame rates of 1.3 frames per second (fps) for Sweeper and 0.2 fps for CG on a progressive 1920x800 image sequence.

4. RESULTS

In this section we qualitatively compare Sweeper with optimization based blur propagation and the proposed facial blur compensation on a set of 10 low depth-of-field test images, that were extracted from the movies "Inception" (1920x800) and "The ruins"(1280x544) and from the TV-series "Breaking Bad" (1280x720).



(a) (b) (c) (d) (e)

Figure 4. Facial blur compensation. (column a) Low resolution input image. (column b) downscaled sparse blur map. (column c) Propagation result for column b. (column d) downscaled sparse blur map with facial compensation, (column e) propagation result for column d. The dense blur map is obtained from Sweeper (Section 3.1). The bounding box from face detection and the ellipse from CAMSHIFT are shown in white and cyan respectively. The color bar encodes blur ranging from 0 to 5. We recommend to view the blur maps electronically.

We did not have the implementation of a multigrid solver at our disposal, instead we apply jacobi preconditioned CG [23], until the 2-norm of the residue $\lambda B\vec{u_C} - (D - W + \lambda B)\vec{u}$ of Eq. 3, is below a pre-defined threshold of 10^{-6}. We refer to this approach by *CG*. Convergence of CG typically occurs within 300 iterations. This approach should give similar results as multilevel preconditioning, due to the large number of iterations, and possibly better results than the geometric multigrid approach of [7], because we do not downscale $\vec{u}_C$, which can lead to inaccuracies in fine details [23].

We estimate blur in the range $u \in [0, 5]$. For both CG and Sweeper we set $D_{fac} = 8$, $\sigma_C = 2.0$, $Th_u = 2$ and $Th_l = 2$ (the scale of the derivative of Gaussian filter, and the hysteresis thresholds on the gradient magnitude in the Canny edge detector respectively). The sparse blur map is calculated at the original image resolution. For JBU and joint propagation we use a box and Gaussian kernel for $f_S(\cdot)$ and $f_R(\cdot)$ in Eq. 9 respectively. Also we select $R_{JBU} = 8$, $\sigma_{JBU} = 18$ and $L = 8$, which are the window radius, standard deviation of the Gaussian in $f_R(\cdot)$ and the number of PBFIC's respectively. Furthermore we set $\sigma_r = 30$ in Sweeper and CG. The window radius for both Sweeper and CG is set to 3x3. For CG we set $\lambda = 0.5$. Finally for facial blur compensation we set $T = 0.35$.

Similar to Zhang et al. [40], we apply a 2D asymmetric Gaussian smoothing filter on the depth maps prior to rendering the stereoscopic 3D images with depth image based rendering (DIBR). This filter reduce the size of disocclusion areas, removes geometric distortion, and prevents that video quality is influenced by DIBR-artefacts in the subjective assessment of Section 4.3. We adopt a mild filter setting with a horizontal and vertical scale of $\sigma_h = 4$ and $\sigma_v = 4$, respectively. In addition we only allow zero and positive parallax.

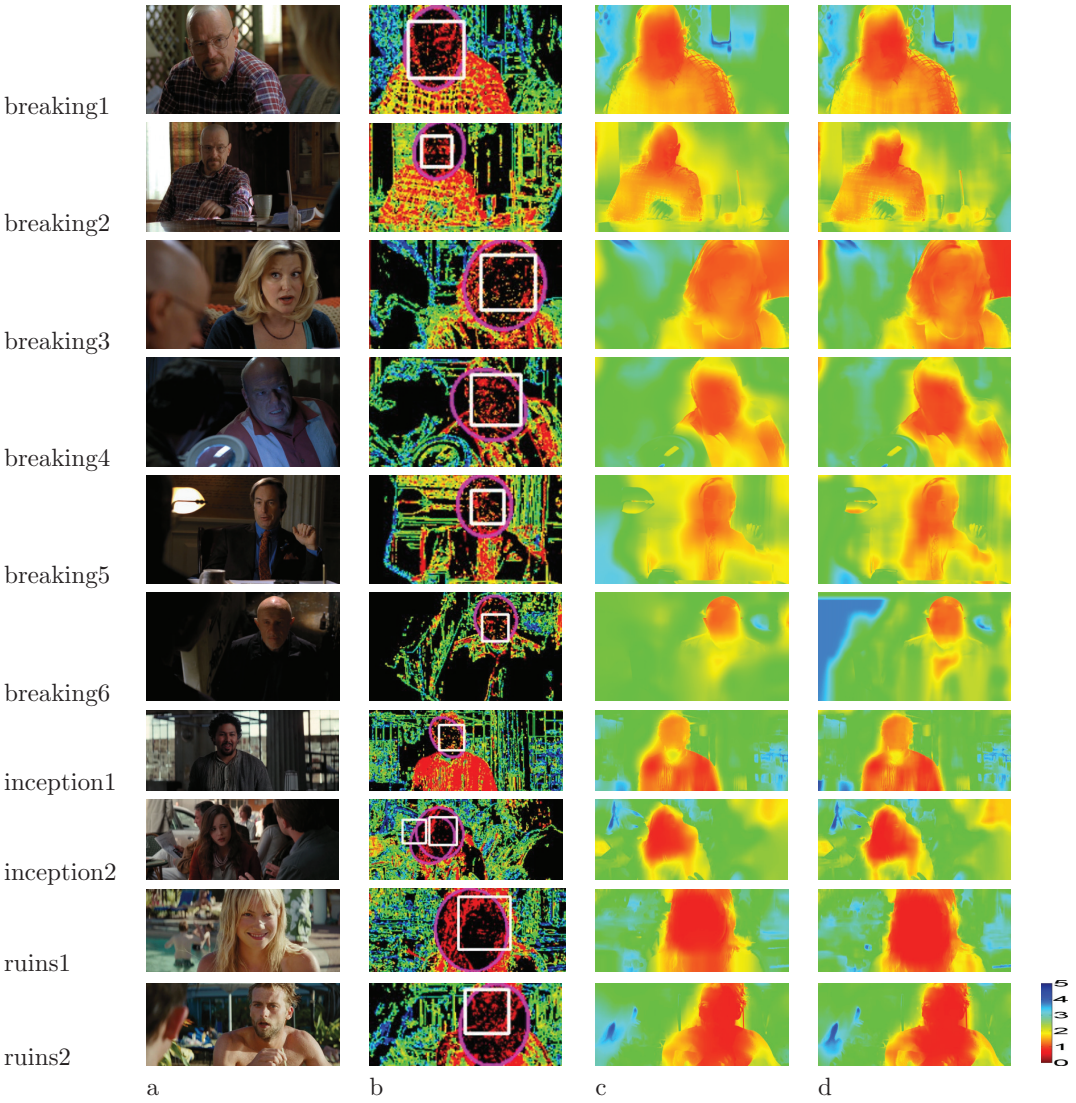


Figure 5. Comparison of propagation methods. From left to right: Input image, downscaled sparse blur map with face compensation, blur propagated by CG, and blur propagated by Sweeper. The bounding box from face detection and the ellipse from CAMSHIFT are shown in white and cyan respectively. The color bar encodes blur ranging from 0 to 5.

4.1 Face Compensation

Fig. 4 column b shows that false edges in people's faces, that are caused by shading or penumbras, constitute a significant problem for blur propagation. In the propagated blur of Fig. 4 column c, it can be clearly seen that the face is assigned a significantly higher blur than the hair and body. However, with the proposed facial compensation (Section 3.3) most false edges and their corresponding blur estimates are removed from the sparse blur map, resulting in a correct more uniform blur estimate in the facial region (Fig. 4 column e). Fig. 3 shows this leads to a more comfortable and natural 3D effect.

4.2 Qualitative comparison

Fig. 5 shows a qualitative comparison of CG and Sweeper with facial blur compensation, and CG without facial blur compensation. The figure shows that the proposed approach features sharper edges at object boundaries

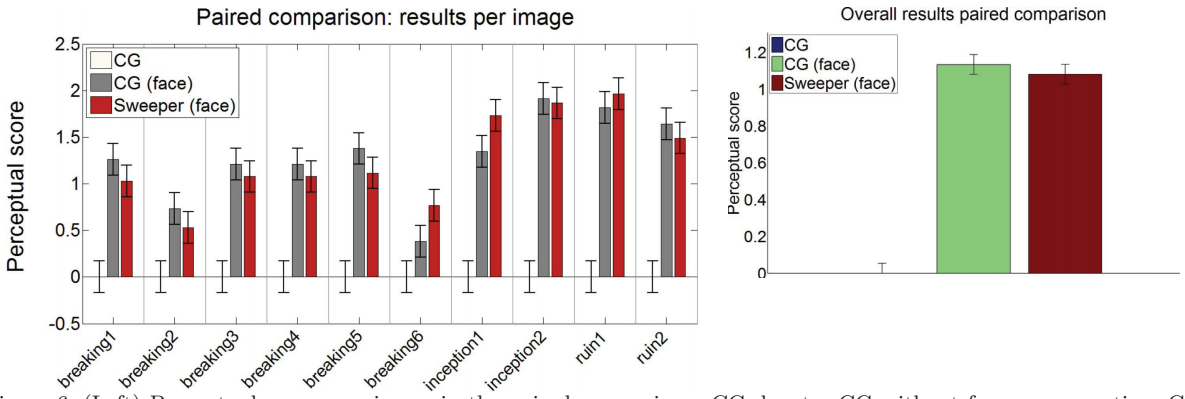


Figure 6. (Left) Perceptual scores per image in the paired comparison. CG denotes CG without face compensation, CG (face) and Sweeper (face) denote CG and Sweeper with face compensation, respectively. This figure shows that for all test images, face compensation improves the 3D image quality. In addition the difference in performance between Sweeper and CG is statistically insignificant, except for *inception1* and *breaking6* for which Sweeper achieves a higher perceptual score. (Right) Overall perceptual scores of the paired comparison. Overall face compensation significantly improves 3D image quality. Furthermore the overall difference in performance between Sweeper and CG is statistically insignificant.

compared to CG, while the interior of objects is slightly less homogenous. When an object is surrounded by a uniform image area for which no sparse blur estimates are available, the object blur is propagated into this area for both CG and Sweeper (Fig. 5 image *breaking3*). This also commonly happens in recursive search motion estimators which operate on the same principle as Sweeper. This will not cause any problems for the viewers depth perception, since an erroneous disparity in a uniform region can be hardly noticed.

4.3 Paired comparison

We compare the performance of CG, CG with facial blur compensation, and Sweeper with facial blur compensation in a paired comparison [41] on the 10 low-depth-of-field test images of Fig. 5. Seventeen participants (no experts in video quality assessment), were requested to rank 30 paired comparisons. For each comparison participants were asked: which image provides a more comfortable and natural 3D experience? In each session the images of a pair were presented sequentially in full screen on a stereoscopic display. The participants could alternate between images of a pair by pressing 'Ctrl' on the keyboard. Each image had to be viewed at least once before voting. To avoid any bias in the perceptual scores, participants could select 'equal' when they did not prefer one image over the other. Each time 'equal' was encountered in the data analysis for e.g. pair AB or BA, we select A and B alternately in the construction of the frequency matrix. In this way any bias from random voting is avoided. The paired comparison experiment was carried out on a 50inch Panasonic VT20 3D plasma television with active shutter glasses, in a room with low illumination.

Fig. 6 shows the results of the paired comparison per image, and for all sequences together, respectively. Since these results should be evaluated on the differences in perceptual score, we set the perceptual score for CG to 0. The 95% confidence intervals were calculated by the empirical formula of [41]. Both figures show that face compensation leads to a significant improvement in the overall perceptual score, and the perceptual score for each test image independently. Furthermore Sweeper and CG have equal performance. Face compensation reduces the 3D effect in faces, however, it also removes strong and unnatural facial depth discontinuities. Therefore, facial deformations, facial depth conflicts, and 3D image blur due to DIBR, are avoided. Hence, a sharper more natural and comfortable 3D image is obtained, and eyestrain is reduced. To prevent tedious and time consuming ranking of images, we did not include Sweeper without face compensation in the paired comparison. Nevertheless, we observed that Sweeper and CG both with face compensation perform significantly better than Sweeper without face compensation. In addition, we did not observe any significant differences between Sweeper and CG both without face compensation.

We made no effort to distinguish between near out-of-focus objects and objects behind the focus plane.

This would require higher level semantic analysis of the image, and we consider that to be outside the scope of this paper. However, in our experiments we perceived objects, standing in front of the plane of focus with high blur, closer than objects standing on or behind the plane of focus. We reckon this is caused by the HVS, which automatically places objects at the correct depth, which it perceives from other monocular image depth cues. Huynh-Thu et al. [42] made a similar observation for other depth ambiguities. The dense defocus map merely indicates depth contrasts among objects. Hence sharp image parts have small disparity and are effectively placed on the screen while blurred image parts are placed behind the screen. This is favorable for reducing the accommodation vergence conflict, since the limits of binocular fusion for high disparity values become less strict with decreased spatial frequency [42].

5. CONCLUSION

In this paper we proposed an efficient, low latency, line scanning based propagation method for estimating dense defocus maps for 2D-to-3D conversion, that is computationally less complex and requires less memory than optimization based approaches. The proposed method, Sweeper, is composed of two efficient bilateral filters, namely a spatially recursive, and joint upsampling filter. Furthermore we have demonstrated that incorrect and inconsistent depth estimates in faces leads to deformations, visual discomfort and significantly degraded 3D image quality, and that the proposed face compensation solves this problem.

The subjective evaluation in Section 4 shows that for low-depth-of-field images the proposed approach achieves equal 3D image quality as optimization based approaches, and that facial blur compensation results in a significant improvement in 3D image quality.

Like defocus various other important 3D depth cues such as, motion, occlusion and disparity, can only be estimated reliably at distinct sparse image locations like edges and corners. Therefore, for future work we plan to extend the proposed approach to the more general sparse-to-dense conversion of 3D depth cues for automatic 2D-to-3D conversion of video.

References

[1] de Haan, G., "Television display processing: Past future," in [*Consumer Electronics, 2007. ICCE 2007. Digest of Technical Papers. International Conference on*], 1–2 (10-14 Januari 2007).

[2] Zhang, L., Vazquez, C., and Knorr, S., "3d-tv content creation: Automatic 2d-to-3d video conversion," *Broadcasting, IEEE Transactions on* **57**, 372 –383 (June 2011).

[3] Bae, S. and Durand, F., "Defocus magnification," *Computer Graphics Forum* **26**, 571–579 (September 2007).

[4] Zhang, W. and Cham, W.-K., "Single image focus editing," in [*Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*], 1947 –1954 (27 September-4 October 2009).

[5] Tai, Y.-W. and Brown, M., "Single image defocus map estimation using local contrast prior," in [*Image Processing (ICIP), 2009 16th IEEE International Conference on*], 1797–1800 (7-10 November 2009).

[6] Levin, A., Lischinski, D., and Weiss, Y., "Colorization using optimization," *ACM Transactions on Graphics* **23**, 689–694 (August 2004).

[7] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P., [*Numerical Recipes in C: The Art of Scientific Computing*], Cambridge University Press, New York, NY, USA, 2 ed. (1992). 0521431085.

[8] Briggs, W. L., Henson, V. E., and McCormick, S. F., [*A multigrid tutorial*], Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2 ed. (2000). 0898714621.

[9] Falgout, R. D., "An introduction to algebraic multigrid," *Computing in Science and Engineering* **8**, 24–33 (November 2006).

[10] Szeliski, R., "Locally adapted hierarchical basis preconditioning," *ACM Transactions on Graphics* **25**, 1135–1143 (July 2006).

[11] Yang, Q., Tan, K.-H., and Ahuja, N., "Real-time o(1) bilateral filtering," in [*Computer Vision and Pattern Recognition, 2009. IEEE Conference on*], 557–564 (June 2009).

[12] Elder, J. and Zucker, S., "Local scale control for edge detection and blur estimation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **20**, 699–716 (July 1998).

[13] Pentland, A. P., "A new sense for depth of field," *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **9**, 523–531 (July 1987).

[14] van Beek, P., *Edge-based image representation and coding*, phd, Delft University of Technology, Department of Electrical Engineering, Information Theory Group, P.O. Box 5031, 2600 GA, Delft, The Netherlands (1995).

[15] Fan, G. and Cham, W.-K., "Model-based edge reconstruction for low bit-rate wavelet-compressed images," *Circuits and Systems for Video Technology, IEEE Transactions on* **10**, 120–132 (Februari 2000).
[16] Canny, J., "A computational approach to edge detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **8**, 679–698 (November 1986).
[17] Hu, H. and de Haan, G., "Low cost robust blur estimator," in [*Image Processing, 2006 IEEE International Conference on*], 617–620 (8-11 October 2006).
[18] Zhuo, S. and Sim, T., "Defocus map estimation from a single image," *Pattern Recognition* **44**, 1852–1858 (September 2011). Computer Analysis of Images and Patterns.
[19] Tomasi, C. and Manduchi, R., "Bilateral filtering for gray and color images," in [*Computer Vision, 1998. Sixth International Conference on*], 839 –846 (4-7 Januari 1998).
[20] Levin, A., Lischinski, D., and Weiss, Y., "A closed-form solution to natural image matting," *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **30**, 228–242 (Februari 2008).
[21] Saad, Y., [*Iterative Methods for Sparse Linear Systems*], Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2 ed. (2003). 0898715342.
[22] Shewchuk, J. R., "An introduction to the conjugate gradient method without the agonizing pain," tech. rep., Pittsburgh, PA, USA (1994).
[23] He, K., Sun, J., and Tang, X., "Fast matting using large kernel matting laplacian matrices," in [*Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*], 2165–2172 (13-18 June 2010).
[24] Kaiming, H., Jian, S., and Xiaoou, T., "Guided image filtering," in [*Proceedings of the 11th European conference on Computer vision: Part I*], *ECCV'10* **1**, 1–14, Springer-Verlag, Heraklion, Crete, Greece (5-11 September 2010).
[25] Chen, J., Paris, S., and Durand, F., "Real-time edge-aware image processing with the bilateral grid," *ACM Transactions on Graphics* **26** (July 2007).
[26] Guttmann, M., Wolf, L., and Cohen-Or, D., "Semi-automatic stereo extraction from video footage," in [*Computer Vision, 2009 IEEE 12th International Conference on*], 136 –142 (29 September-2 October 2009).
[27] de Haan, G., Biezen, P., Huijgen, H., and Ojo, O., "True-motion estimation with 3-d recursive search block matching," *Circuits and Systems for Video Technology, IEEE Transactions on* **3**, 368 –379, 388 (October 1993).
[28] Kopf, J., Cohen, M. F., Lischinski, D., and Uyttendaele, M., "Joint bilateral upsampling," *ACM Transactions on Graphics* **26** (July 2007).
[29] Paris, S. and Durand, F., "A fast approximation of the bilateral filter using a signal processing approach," *Int. J. Comput. Vision* **81**, 24–52 (January 2009).
[30] Porikli, F., "Constant time o(1) bilateral filtering," in [*Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*], 1 –8 (june 2008).
[31] Igarashi, M., Ikebe, M., Shimoyama, S., Yamano, K., and Motohisa, J., "O(1) bilateral filtering with low memory usage," in [*Image Processing (ICIP), 2010 17th IEEE International Conference on*], 3301 –3304 (September 2010).
[32] Chaudhury, K., Sage, D., and Unser, M., "Fast o(1) bilateral filtering using trigonometric range kernels," *Image Processing, IEEE Transactions on* **20**(12), 3376 –3382 (2011).
[33] Riemens, A., Gangwal, O., Barenbrug, B., and Berretty, R.-P. M., "Multistep joint bilateral depth upsampling," in [*Visual Communications and Image Processing 2009, Proceedings SPIE*], 72570M (20 Januari 2009).
[34] Lukin, A., "Tips & tricks: Fast image filtering algorithms," in [*International Conference on Computer Graphics and Vision*], *Proceedings of GraphiCon* , 186189, Citeseer, Moscow, Russia (23-27 June 2007).
[35] Vosters, L., "Efficient sparse to dense conversion for automatic 2d-to-3d conversion." https://sites.google.com/site/sparsetodenseconversion/ (November 2011).
[36] Liu, R., Li, Z., and Jia, J., "Image partial blur detection and classification," in [*Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*], 1 –8 (june 2008).
[37] Viola, P. and Jones, M. J., "Robust real-time face detection," *Int. J. Comput. Vision* **57**, 137–154 (May 2004).
[38] Bradski, G. R., "Computer Vision Face Tracking For Use in a Perceptual User Interface," (1998).
[39] "Cusp library: Generic parallel algorithms for sparse matrix and graph computations." http://code.google.com/p/cusp-library/.
[40] Zhang, L. and Tam, W., "Stereoscopic image generation based on depth images for 3d tv," *Broadcasting, IEEE Transactions on* **51**, 191–199 (June 2005).
[41] Montag, E. D., "Empirical formula for creating error bars for the method of paired comparison," *Journal of Electronic Imaging* **15**, 010502 (March 2006).
[42] Huynh-Thu, Q., Barkowsky, M., and Le Callet, P., "The importance of visual attention in improving the 3d-tv viewing experience: Overview and new perspectives," *Broadcasting, IEEE Transactions on* **57**, 421 –431 (June 2011).