A generic tool for interactive complex image editing

Ana B. Cambra · Ana C. Murillo · Adolfo Muñoz

Abstract Plenty of complex image editing techniques require certain per-pixel property or magnitude to be known, e.g., simulating depth of field effects requires a depth map. This work presents an efficient interaction paradigm that approximates any per-pixel magnitude from a few user strokes by propagating the sparse user input to each pixel of the image. The propagation scheme is based on a linear least squares system of equations which represents local and neighbouring restrictions over superpixels. After each user input, the system responds immediately, propagating the values and applying the corresponding filter. Our interaction paradigm is generic, enabling image editing applications to run at interactive rates by changing just the image processing algorithm, but keeping our proposed propagation scheme. We illustrate this through three interactive applications: depth of field simulation, dehazing and tone mapping.

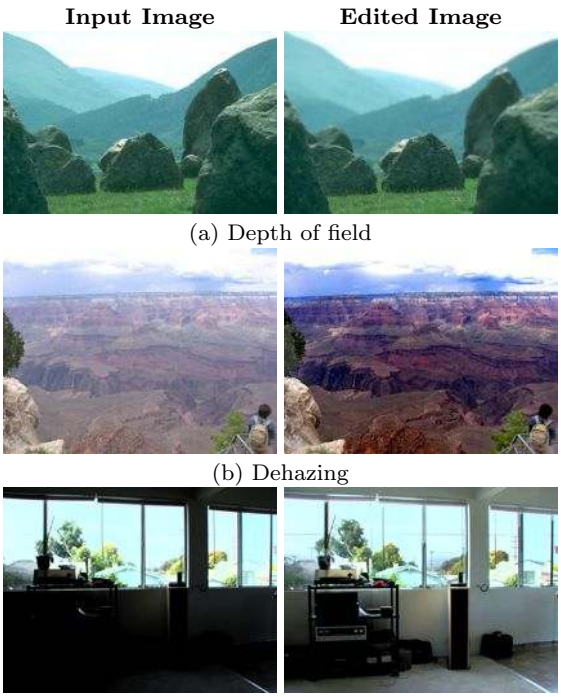
Keywords User interaction, image processing, computer vision, dense label propagation

1 Introduction

Advanced image editing techniques often rely on specific per-pixel information about certain magnitudes, in addition to each pixel intrinsic RGB coordinates.

A. B. Cambra · A. C. Murillo · A. Muñoz
Instituto de Investigación en Ingeniería de Aragón, Universidad de Zaragoza, 50009 Zaragoza, Spain

A. B. Cambra
E-mail: acambra@unizar.es
A. C. Murillo
E-mail: acm@unizar.es
A. Muñoz
E-mail: adolfo@unizar.es



(a) Depth of field

(b) Dehazing

(c) Tone mapping

Fig. 1 Image editing examples using the proposed generic interaction paradigm. From a few user strokes our approach estimates a per-pixel magnitude that enables the advanced editing of (a) depth for a *depth of field* effect (focus point is at the middle scene elements); (b) transmittance for a *dehazing* effect; (c) luminance and brightness for *tone mapping*.

For instance, simulating blur due to depth of field [23] requires a per-pixel estimation of depth information, while dehazing (removal of fog, water, smoke or any participating medium) requires per-pixel information about the transmittance or density of the medium [16].

This extra per-pixel information may come from additional knowledge or control over the capture process.

For instance, depth can be estimated from sensor motion, either intended [43] or accidental [46]. Alternatively, advanced heuristics and priors may lead to a plausible (but not necessarily accurate) additional information that can be used for filtering purposes. Examples of these include heuristics to obtain depth from a single image [26], or transmittance priors for dehazing [16]. These heuristics and priors typically involve assumptions that work very well in situations that meet them. However, we typically find specific cases when they are broken, which leads to poor performance on the editing results.

Taking advantage of user interaction is another option for gathering additional per-pixel information, which does not impose any restriction regarding control of the image capture or advanced heuristics that could not be generic enough. On one hand, traditional generic user interaction paradigms (such as geometric primitives, *lassos* or color-aware *magic wands*) are not practical for complex information editing. On the other hand, other specialized editing tools define interaction paradigms that are very practical but tailored to specific applications, such as light field editing [21], intrinsic image decomposition [9] or intrinsic video decomposition [8].

We work towards a more general approach for advanced user interaction paradigms. Whenever certain filter needs additional per-pixel information, it is densely generated by our approach from very sparse and intuitive user interaction. Our paradigm is oblivious to the internal design of the filter, i.e., we treat each filter as a black box, with two simple requirements:
1. The filter must rely on additional information per-pixel which is a continuous magnitude (such as depth).
2. The filter must run at interactive speed (preferably in less than one second) so that the interaction paradigm runs at practical frame rates.

Our approach covers a wide range of applications. Some of these applications, as demonstrated in this work (see Fig. 1), include synthetic depth of field simulation, dehazing and tone mapping from HDR images. Our paradigm is modeled as a propagation problem, in which the user sets some seed values, with one or more strokes, which are propagated throughout all the pixels. This propagation results on a per-pixel estimation of the target continuous magnitude (continuous in value, not in image space). The propagation is formally represented as a least-squares linear system of equations over a set of superpixels. This provides us a great control over the accuracy vs. propagation time trade-off.

Contributions

1. A very simple and intuitive interaction paradigm, inspired and distilled from previous specific image editing applications, which is generic and oblivious to the underlying image operator. It is based on simple user strokes that provide seed values and value relations, which can be interactively refined.
2. A propagation strategy that involves superpixels and a linear system of equations, which enables propagation of continuous magnitudes at interactive speed rates.
3. A set of applications, extensions of several state-of-the-art image filters, that now benefit from our interaction scheme.

2 Related Work

In order to achieve interactive versions of sophisticated image editing filters, our work combines two common steps in image processing and computer vision: superpixel image segmentation and label propagation. We briefly describe the state of the art of these common steps, as well as the image editing techniques we make interactive with our approach.

Superpixels. Previous research has shown how grouping pixels into superpixels can turn computationally complex problems into tractable ones. They have been used for estimating depth from multiple panoramic images [30], decomposing images into their intrinsic shading and reflectance components [18] or assigning semantic and geometric labels in conventional images [41]. A more extensive survey on applications using superpixel segmentation can be found here [38]. While using superpixels (rather than individual pixels) implicitly introduces an accuracy penalty due to suboptimal superpixel segmentations, it also reduces computational complexity. In our work, this is key for keeping interactive frame rates regardless of image resolution.

Propagation. Certain applications rely on value propagation across pixels, i.e., they spread a sparse set of seed values from certain pixels to the rest of the image pixels. Markov Random Fields (MRF) are a common ingredient on solutions for propagation problems [39] but they still present high computational cost. To reduce the execution time of these solutions, we find MRF superpixel-based approaches [30, 34, 41] or strategies to optimize MRF solver optimization [13]. Reducing the execution time is the goal as well of our work. However, our formulation is less restrictive than related works and our experiments prove that we achieve a faster response.

Efficient approaches to solve propagation problems model the propagation scheme as a linear optimization, as we do, which can be solved through linear systems of

equations. Random Walker (RW) algorithm [19] computes the probability of each pixel in the image for each seed value given by the user. Unfortunately, when there is a high number of seed values and high resolution images are used, this algorithm can become slow. We find multiple improvement towards more efficient RW propagation. These improvements can be based on offline pre-computation [4], or, as our work, on the use of superpixels [17]. The RW algorithm has been adapted to multi-view depth estimation for image based navigation [12].

A common drawback of the techniques described so far is that none of them is able to deal with a continuous magnitude. While the continuous magnitude could be discretized into a set of potential values, some discrete propagation techniques (such as the Random Walker) propagate just the values they have been seeded with (and no other value) which works poorly with sparse inputs. Complex MRF modifications [37,45] are able to deal with continuous magnitudes, unfortunately they are still not efficient enough for interactive use. Furthermore, the Random Walker probability has been used also as a continuous magnitude [32] but can only be seeded with two probability values (0 and 1). In contrast, our approach can be seeded with any potential value but does not restrict the range of output values.

Image editing techniques. In this work we illustrate our interaction paradigm through three applications: depth of field effects [23,46], dehazing [16,6] and tone mapping [33,29]. These applications are described in Section 5.

Related to our work, there is previous research on interaction to propagate complex magnitudes: color for black and white images or video [24,28,14], light field edits [5], image segmentation [31], shading and reflectance in images [9] or video [8] and depth [27,20,48] . However, the interaction at each of these techniques is tailored to the specific application itself, and in some cases it does not reach interactive rates. In contrast, our work presents a global propagation technique that is oblivious to the underlying filter, is more flexible and works at interactive speeds.

Interactive editing. Examples of interactive image editing applications include transferring edits between different views through affine transform estimation [47], multiview depth transfer through shortest path algorithm [12], appearance transfer using simplex optimization for reducing parameter space [3], or material editing on a linear space using a GPU [15].

Our propagation paradigm is closely related to other image editing techniques which propagate continuous magnitudes. Lischinski et al. approach [25] solves systems at different image resolutions to show a progressive result (which otherwise would not be interactive). Chen et al. [14] propose a linear system over feature space. SteroBrush [42] also uses a linear system for propagating stereo disparities while preserving pleasantness, which becomes interactive by taking advantage of the GPU. Such linear optimizations are per-pixel, while our approach works at superpixel level, therefore becoming more efficient. The AppProp approach [2] propagates exposures through a per-pixel linear system, but it is done faster by identifying a subset of representative columns in the matrix. Instead of column sampling, previous work [44] has reduced the parameter space by clustering it into a KD tree. Still, in both cases matrix operations (inversions, products) are needed before solving the system. Our approach is inspired by such approaches, reducing the linear system by working in superpixel space, but improves over them by building the linear system matrix on the fly.

Iizuka et al. [20] propose propagation through optimization based on geodesic distances over superpixels (with some additional filtering), with a propagation speed that matches our work. However, their edits are limited to such distances, while our formulation is more flexible and enables more global edits (see Section 3.1.2).

Our work builds on top of a previous technique for propagating depth values throughout an image [11] by generalizing the formulation to any filter based on one (or more) magnitudes. First, we show more detailed analysis of the depth of field application suggested there. Additionally, we demonstrate two other applications requiring estimation of participating media transmittance (for dehazing) and brightness and contrast (two magnitudes for tone mapping). These magnitudes fit perfectly into our interactive editing paradigm. We also show how the editing results achieved with our approach are comparable to state-of-the-art effect-specific techniques.

3 Interactive propagation

This section describes the interactive propagation system.

3.1 System

Our proposed system is built upon prior work [11], to propagate per-pixel user information, which is summarized in Figure 2. This diagram shows how the different

steps are divided into *initialization*, if they can be computed just once when the input image is loaded, and *interactive*, if they are re-calculated every time the user performs an interaction with the system.

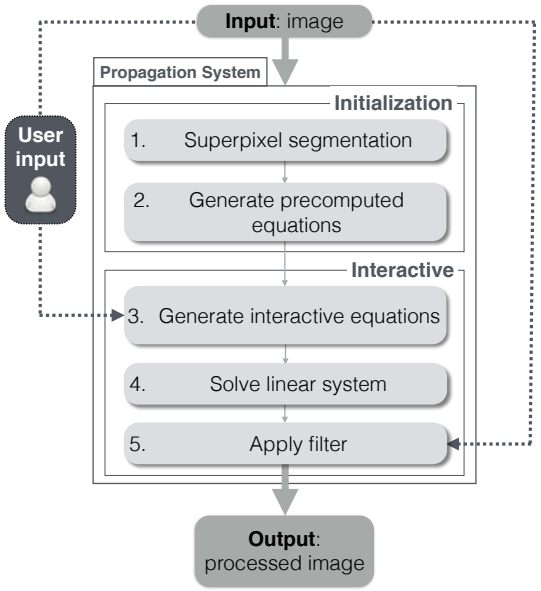


Fig. 2 Interactive propagation system. Steps 2 and 3 build the equations and step 4 solves the system to get the best propagation given the input image and the user interaction. Each user edition (user input) creates new interactive equations, and the output is updated, i.e., steps 3 to 5 are run, in less than 1 second.

3.1.1 Initialization.

When the input image is loaded, we initialize the propagation system. This involves two steps: superpixel segmentation and linear system construction. Even if this initialization involves some pre-computation, it is not computationally very expensive. It takes a few seconds at most (depending on processor speed and image resolution), which is within a reasonable time for an initialization of an interactive application. For superpixel segmentation we use the standard SLIC algorithm [1] with its single configuration parameter, *number of superpixels* which depends on the image size. We estimate a superpixel size around 10×10 but the number of superpixels is limited to 1000. This algorithm is reasonably fast, although our propagation technique could use any other segmentation algorithm. This segmentation helps us to control the efficiency of the solver independently of image resolution.

Our propagation assigns a single value to each superpixel (which is equivalent to assigning the same value to all pixels inside the superpixel). The propagation is modeled as a linear system of equations $Ax = b$ where x (the unknowns) are the superpixel propagated values, and the equations (represented by the matrix A and the vector b) are the restrictions that model the propagation. The number of equations is greater than the number of unknowns (the number of superpixels) so the system is over-constrained. Therefore our propagation actually solves the least squares minimization $\min_x ||Ax - b||$ through the equivalent linear system:

$$(A^T A)x = A^T b. \tag{1}$$

Binary equations. These equations establish binary relationship between values of connected superpixels. We consider that two superpixels are connected following 8-neighbors connectivity. Given two connected superpixels p and q, their binary relationship is represented as

$$w_b(x_p - x_q) = 0, \tag{2}$$

where x_p and x_q are the unknown values of superpixels p and q respectively. The preconditioning factor w_b prioritizes the connections whose border is similar in the CIE-Lab color space. If the distance between the frontier pixels average CIE-Lab values (of both superpixels) is greater than a threshold of 0.05, w_b is $\frac{1}{\#b}$. Otherwise, it is equal to $\frac{0.01}{\#b}$. The denominator $\#b$ is the total number of binary connections, and helps to keep a similar linear system behaviour regardless of the number of superpixels. As CIE-Lab values are normalized between 0 and 1, the threshold of 0.05 represents a percentage over Euclidean distance between those values.

Linear system construction. With all the superpixels and the corresponding connecting binary equations, we build the square matrix $(A^T A)$ and vector $A^T b$ from (1). The dimension of the square matrix and the vector is N, where N is the number of superpixels. By calculating and storing both the matrix and the vector during initialization, we avoid doing the matrix product (potentially time consuming) during user interaction.

3.1.2 Interactive propagation.

The interactive propagation includes three steps: the addition of equations that correspond to user interaction, the value propagation itself (by solving the linear system) and the application of the image filter that uses the corresponding propagated magnitude.

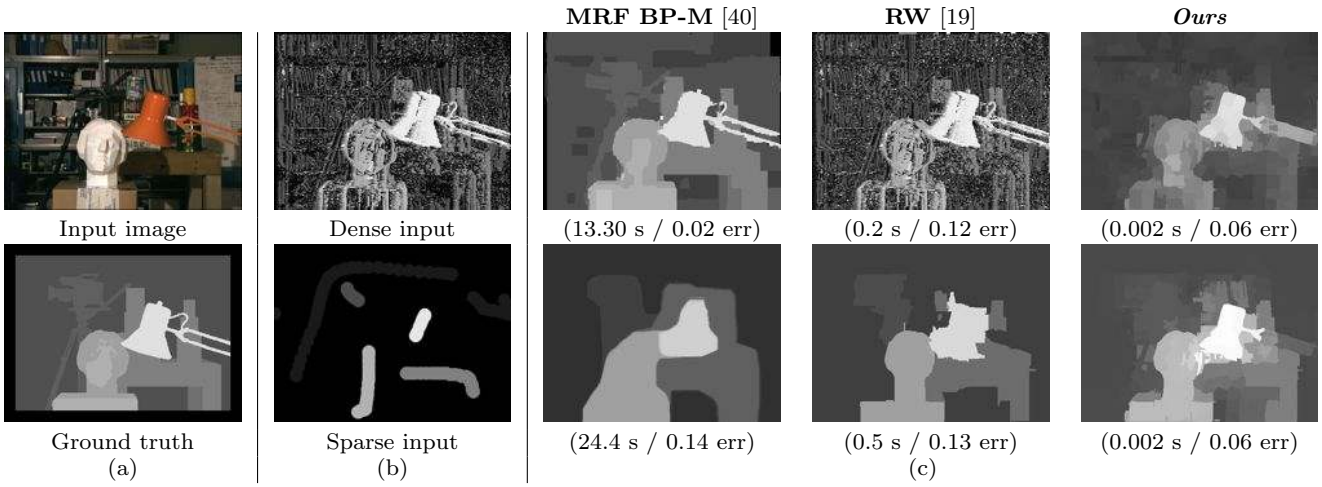


Fig. 3 (a) Input image and ground truth disparity map used in both experiments. (b) Input data used to initialize the propagation in the two experiments: *dense automatic input* (top row) or using a *sparse user input* (bottom row). (c) Dense propagation obtained in the two experiments. Each column shows the propagation obtained with a different method: two reference methods (MRF BP-M and RW) and our approach (*Ours*). Each result includes between parenthesis the execution time (s) and ratio of mislabeled pixels with each method (*err*).

Unary equations. The unary equations link user input with superpixel values and are built during the *interactive* steps. The user chooses a brush that corresponds to a specific value of the magnitude of interest v. For each pixel affected by the stroke inside superpixel p, we include the following equation into the system:

$$w_u x_p = w_u v, \tag{3}$$

where x_p is the (still) unknown magnitude value of superpixel p and $w_u = \frac{1}{\#u}$ (where $\#u$ is the number of unary equations) is a preconditioning factor that ensures stability on the behavior of the system no matter the number or length of user strokes. We interactively add (3) both into the pre-computed matrix $(A^T A)$ and the vector $(A^T b)$, as only one of the cells of the matrix and the vector is affected by the equation. Therefore, there is no need to recalculate the matrix product.

Depending on the user strokes, the user may include contradictory equations for the same superpixel. This is expected and supported by the approach, because the solver is a linear least squares minimization that will find the optimal x_p that minimizes the contradiction. Furthermore, larger strokes may include many equal unary equations that affect the same superpixel, which in practice increases the overall weight of the equation in the minimization. This is expected and desired as well.

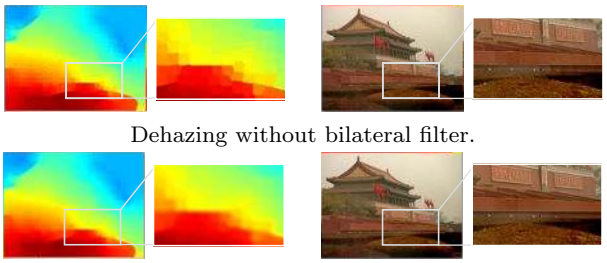
Equality equations. While unary equations set specific superpixel values, *equality* equations establish similarity relationships between superpixels through user strokes. Those superpixels are not necessarily contiguous. Such equations are set during the *interactive* step. Given two superpixels p and q, their binary relationship is represented as

$$w_e(x_p - x_q) = 0, \tag{4}$$

where x_p and x_q are the unknown values of superpixels p and q respectively and $w_e = \frac{1}{\#e}$ (where $\#e$ is the number of equality equations) is the preconditioning factor. Such equations are useful where specific superpixel values are rather unintuitive (rendering unary equations useless) but similarity can intuitively be spotted. For example, for dehazing the specific transmittance is rather confusing to identify, but it is easier to relate regions with same fog (Fig.9).

Linear system solving. After the unary and/or equality equations are stored we solve the equation system in (1). The $(A^T A)$ matrix is symmetric by definition and positive semi-definite. Therefore, the system can be solved applying Cholesky decomposition, specifically LDLT decomposition, with the advantage of being fast and numerically stable. Note that all preconditioning factors (w_b, w_u and w_e) are global parameters which help to control the effect of each equation type. In order to keep such global weights constant, each preconditioning factor is related with the number of equations, so each new equation gradually reduces the influence of all the equations of the same type. This is intended and helps to preserve the stability of the global effect of each user stroke.

For each superpixel value x_p obtained from the linear system, we propagate it to every pixel in the superpixel p. This final propagated values are used as



Dehazing without bilateral filter.

Dehazing with bilateral filter.

Fig. 4 Effect of applying a bilateral filter to the transmittance map (left column) used in our interactive dehazing application. Bilateral filter avoids artifacts caused by the superpixel segmentation in the processed image.

input to the corresponding image filter. Due to superpixel segmentation, the final propagated values could be locally coarse and cause artifacts in processed image when the corresponding image filter is applied. To avoid this possible artifacts (similarly to related work [20]), we apply a **bilateral filter** to the propagated values. The bilateral filter is controlled by three parameters: d, σ_{color} and σ_{space}. d represents the diameter of each pixel neighborhood that is used during filtering. The range parameter σ_{color} and the spatial parameter σ_{space} control edge preservation when applying the filter. In our work, they are set to 15, 100 and 100 respectively.

Figure 4 shows the effect of applying such bilateral filter in our dehazing application. Note that, the propagation technique is oblivious to the underlying semantics of the propagated magnitude, while the image filter is application-specific. In our work we test three different image filters for three different applications (see Section 5). Each of them relies on different magnitudes (depth, transmittance and local brightness and contrast) that are propagated exactly the same way, as detailed in this section, and become input of the corresponding filter to the specific application.

4 Validation.

This section presents a numerical validation of our interactive propagation system. We validate the proposed propagation strategy considering only step 4 (see Fig. 2), since it is the only common step to all the considered approaches. We perform this validation using a specific magnitude (disparity) for which there are public datasets available with ground truth [35] [36]. These datasets are designed to evaluate stereo algorithms, which obtain a disparity map that represents the disparity relation between corresponding pixels from two images.

We run the experiments with two very different inputs: a **dense** but unreliable (noisy) input labeling obtained automatically from two stereo images, and a **sparse** (but reliable) input labeling obtained from a few user strokes. We compare our results against well established propagation approaches: Markov Random Fields (MRF) strategies [39] and a Random Walk (RW) approach [19]. RW is the most different approach with respect to the way to formulate the linear equation system. The rest of methods (MRF strategies) are based on the same equations and weights than our approach. Figure 3 shows the final disparity estimation obtained by the evaluated methods for one of the test images, *Tsukuba*-test from the public dataset [35,36]. For each method evaluated, Table 1 summarizes the execution time and error for all images in the dataset. We measured the error obtained in each solution as the mean of the differences (*err*) between each pixel in the solution and the same pixel in the ground truth. Note that both MRF and RW implementations are based on pixel-wise formulations, while our work is superpixel-based. More detailed experiments about numerical comparisons of the presented approach and other propagation methods can be found in previous work [11].

Method	Dense Input		User input	
	time	**err**	**time**	**err**
Pixel-based				
ICM [7]	0.96	0.12	N/A	N/A
Expansion [10]	9.68	0.03	N/A	N/A
Swap [10]	7.28.94	0.03	N/A	N/A
TRW-S [22]	93.94	0.03	N/A	N/A
BP-S [40]	10.42	0.03	N/A	N/A
BP-M [40]	68.76	0.02	31.23	0.14
BCD [13]	1.72	0.11	-	-
RW [19]	0.40*	0.16	0.60*	0.14
Superpixel-based				
Expansion [10]	5.21	0.08	6.17	0.10
Ours	**.004**	**0.08**	**.004**	**0.05**

−: the method did not converge to a solution
*: execution time is measured in Matlab.

Table 1 Execution time (*seconds*) on a standard desktop machine (Intel Core i5 2,5 GHz) and mean error (*err*) for propagation obtained for all dataset images with automatic input and user input initialization.

Our experiments validate two main hypotheses: our approach is faster but of comparable quality to the best existing methods using dense input data; our approach is significantly better than related approaches using more realistic sparse user input data.

Using dense input, the proposed approach obtains comparable results to related approaches, while being orders of magnitude more efficient. Apparently, the use of superpixels enables extra efficiency without a remarkable penalty on accuracy. Note that the execution time comparison with RW implementation is somehow unfair as the RW available code is implemented in Mat-

(a) (b) (c)

Fig. 5 Interactive depth of field application. (a) Input image. (b) Processed image after applying the depth of field effect. (c) Depth map used to apply the effect (estimated by our application from user interactions).

(a) (b) (c)

Fig. 6 Interactive dehazing application. (a) Input image. (b) Processed image after applying the dehazing effect. (c) Transmittance map used to apply the effect (estimated by our application from user interactions).

lab, while the rest use C++. In an optimistic scenario, an optimized RW could potentially be almost as fast as our approach, but its results are significantly worse.

However, user interaction is expected to be sparse. Our experiments show that neither MRFs nor RWs are suboptimal with sparse input: MRFs are focused on a more dense noisy input, while RWs are not well suited for this problem because they only propagated the assigned input values, therefore neglecting all potential intermediate ones.

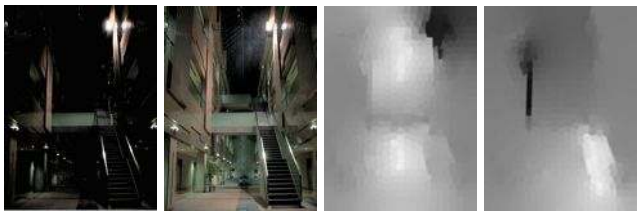
5 Applications

We illustrate the versatility of our interactive propagation scheme through several applications described next. The supplementary video (available on-line [1]) demonstrates the real time execution and the behavior of each application.

5.1 Depth of field

In photography, control over in-focus and out-of-focus elements is an important artistic and expressive tool. **Depth of field** effect consists of the blurring of out-of-focus objects on a single image from a depth map as software post-processing. This depth can be estimated from camera motion, either accidental [46] or fixed to capture two images (stereo) and then estimate depth from them [43].

[1] `https://www.youtube.com/watch?v=Fps5SasG9v4`



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

Fig. 7 Interactive HDR tone mapping application. (a) Input HDR image. (b) Output LDR image after applying the tone mapping using the estimated local brightness map (c) and local contrast map (d).

In our work, depth is obtained from the user by means of our propagation approach, as simple user strokes (using four different brushes associated with different depths in the range $[0, 1]$). Then, we obtain the depth of field effect by applying a set of convolutions (blurs) of different radius according to the difference between depth at each pixel and focal distance, following previous work [23]. The blurs are applied up to a maximum radius defined by the user. The focal distance is set by the user.

Figure 5 shows an example of this application, where the focal distance is fixed at the front scene elements. Therefore, elements from the back of the scene (according to the estimated depth map) become blurred after the processing. Figure 8 includes several intermediate outputs while processing another example image. The output is automatically refined as additional sample depth strokes are provided by the user. The user can also change the desired focal distance by clicking on the desired focus point.

5.2 Dehazing

As light interacts with small particles suspended in air (or water), fog, smoke or the aerosols in the atmosphere become visible. Repeated interactions across such media reduce the visibility by creating a translucent layer of ambient light. **Dehazing** consists on removing such ambient light from a single image. For that purpose, algorithms estimate the medium's transmittance (visibility) through heuristics and priors such as interpreting albedo as locally constant (color lines) and transmittance as smooth [16] or globally identify the haze-free colors along haze lines [6]. This previous work models haze as:

$$I(\mathbf{x}) = t(\mathbf{x})\, J(\mathbf{x}) + (1 - t(\mathbf{x}))\, A, \tag{5}$$

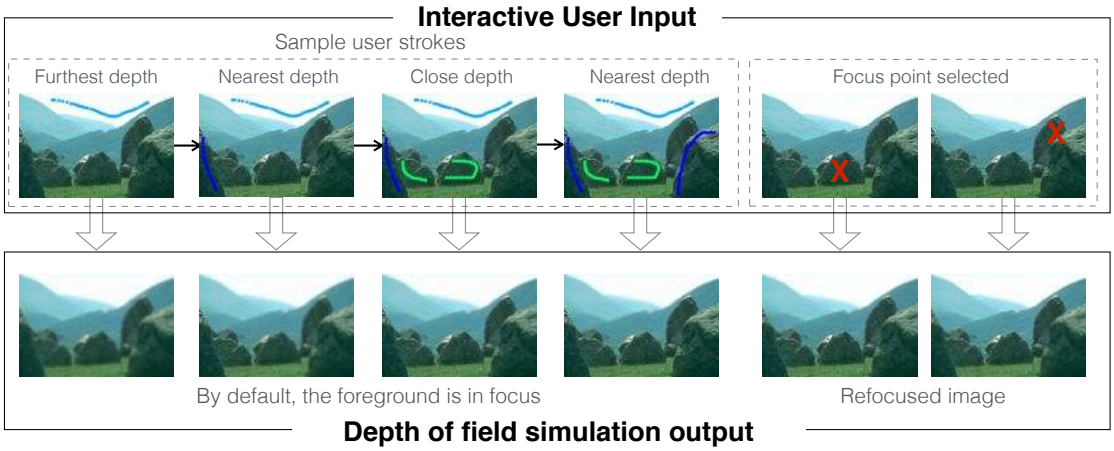


Fig. 8 Behavior of the interactive depth of field application with user interactions. Left: the user marks a few strokes (different colors for different depths) to represent object positions. Each new edition interactively propagates depth estimation and applies a depth of field effect. Right: the user changes the focus point.

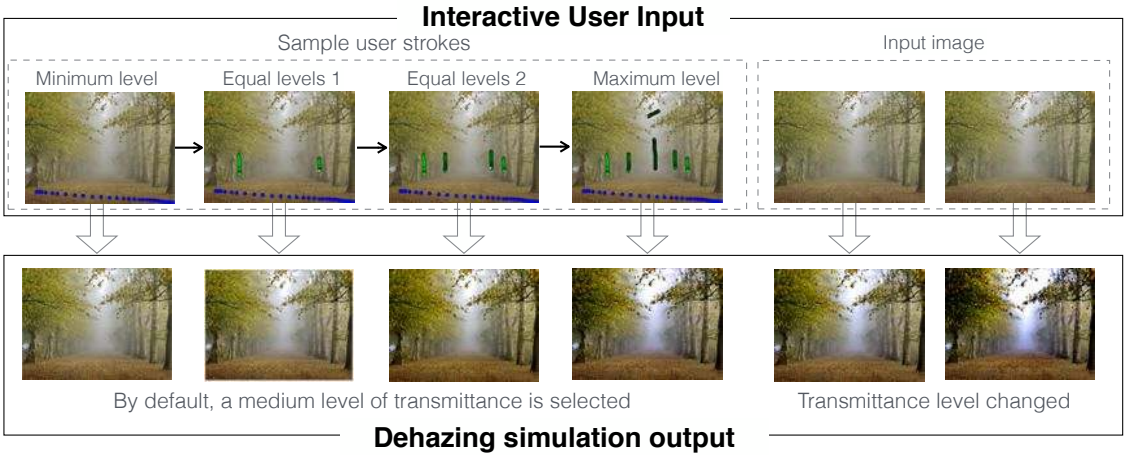


Fig. 9 Behavior of the interactive dehazing application with user interactions. Left: the user marks a few strokes (different colors for different transmittance levels) to represent the level of fog. Each new edition interactively re-estimates transmittance values and applies the dehazing effect. Right: the user can adjust the level of the dehazing effect.

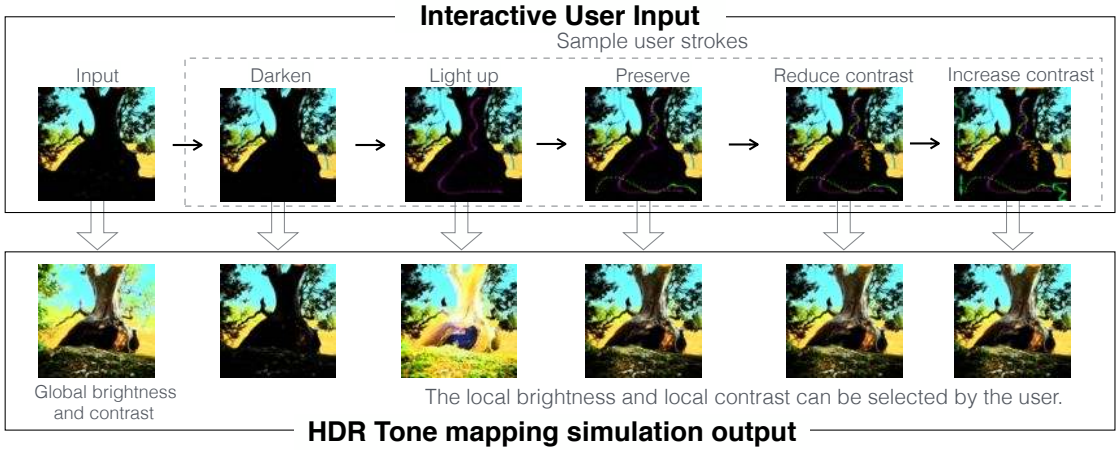


Fig. 10 Behavior of the interactive HDR tone mapping application with user interactions. Left: the user can adjust global parameters. Right: the user marks with strokes that regions need to be darken or illuminated and how the contrast of the regions should increase or reduce. Each color represents a different action.

where $\mathbf{x}$ are the 2D pixel coordinates, $I(\mathbf{x})$ is the input hazy image, $J(\mathbf{x})$ is the dehazed image (what we want to obtain), $t(\mathbf{x})$ is a scalar *transmittance map* (unknown, $0 \leq t(\mathbf{x}) \leq 1$) and A is the ambient light (a RGB vector, unknown).

In our case, the transmittance map $t(\mathbf{x})$ is propagated from user strokes using our technique, and A is approximated as the average of all pixels $\mathbf{x}$ where $t(\mathbf{x}) > 0.9$ (the most occluded pixels). Then the image is easily obtained from (5) as

$$J(\mathbf{x}) = \frac{I(\mathbf{x} + (t\,(\mathbf{x}) - 1)\,A}{t(\mathbf{x})}. \tag{6}$$

Figure 6 shows an example of this application, where the fog is removed from the scene according to the estimated transmittance map. Figure 9 illustrates incremental output results, which are refined interactively as the user provides additional strokes to mark the level of fog in different parts of the image.

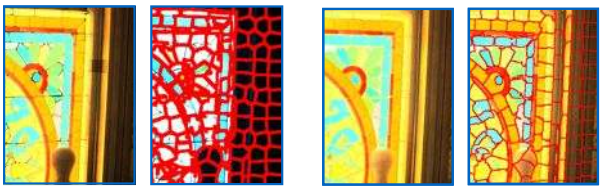
5.3 Tone mapping

The potential range of luminance in the real world is rather large, while devices can only capture or reproduce two orders of magnitude within that range. However, High Dynamic Range image formats can represent (with floating point representation) all potential luminance values, and can be composed from several images or generated synthetically. **Tone mapping** is the process of converting a High Dynamic Range (HDR) image into Low Dynamic Range (LDR), representable by any device. Their effect can be either global (equal to every pixel) or local (often adapting global algorithms with convolution kernels) [33]. For our tone mapper we follow the following tone curve [29] that compresses the luminance of the HDR image L_{HDR} into the luminance of the LDR image L_{LDR}:

$$L' = \log\left(L_{HDR}\right) - b$$

$$L_{LDR} = \begin{cases} 0 & \text{if } L' \leq -d_l \\ \frac{c}{2}\frac{L'}{1-\left(c-\frac{1}{d_l}\right)L'} + \frac{1}{2} & \text{if } -d_l < L' \leq 0 \\ \frac{c}{2}\frac{L'}{1+\left(c-\frac{1}{d_h}\right)L'} + \frac{1}{2} & \text{if } 0 < L' \leq d_h \\ 1 & \text{if } L' > d_h \end{cases} \tag{7}$$

where d_l and d_h are the lower and higher midtone ranges (set up to the recommended values of 2 and 1, respectively) and b (brightness) and c (contrast) are the tone mapper parameters of the tone curve. In our case, for the sake of simplicity, we do not apply color correction. Also, we apply this curve per-pixel, where brightness and contrast are obtained from propagation: the user can set, in the image local values for brightness and contrast, that are propagated through the whole image in two magnitudes (as opposed to previous applications, that propagated just one).



(a) (b)

Fig. 11 Superpixel segmentation of (a) HDR image and (b) LDR image.

Figure 7 shows an example of this application, where the input HDR image is converted to a LDR output image. Figure 10 illustrates incremental output results. The user can adjust the global image levels of brightness and contrast, but the user strokes define local values of brightness and contrast for different regions in the image.

Note that the algorithm we use for superpixel segmentation is not particularly well suited to work with HDR images. It does generate a superpixel segmentation, but as shown in the example from Fig. 11(a), the superpixel boundaries do not correspond well with scene objects. This limitation is accentuated in regions with very few or contradictory user input values. Then, the propagated magnitude suffers strong transitions between close superpixels creating artificial visual boundary artifacts. This could sometimes be solved with additional user input. However, a simpler solution is to apply this filter in two steps. First we get an initial (noisy) superpixel segmentation on the HDR image, used to apply an initial tone-mapping filter that generates an intermediate LDR image. Then, we propose to re-compute the superpixel segmentation over this intermediate LDR image, where we obtain a better superpixel segmentation that can be used to obtain a better final result without artifacts (see Fig. 11(b)).

6 Results

In previous Section 4, we have formally validated that our scheme can propagate interactively a few sparse sample values of a continuous magnitude to all image pixels. It is efficient while obtaining comparable quality with respect to related approaches. This section presents a set of representative results obtained using the three interactive applications presented (Sec. 5), which are based on our propagation scheme (Sec. 3), in order to analyze different situations, limitations and ad-

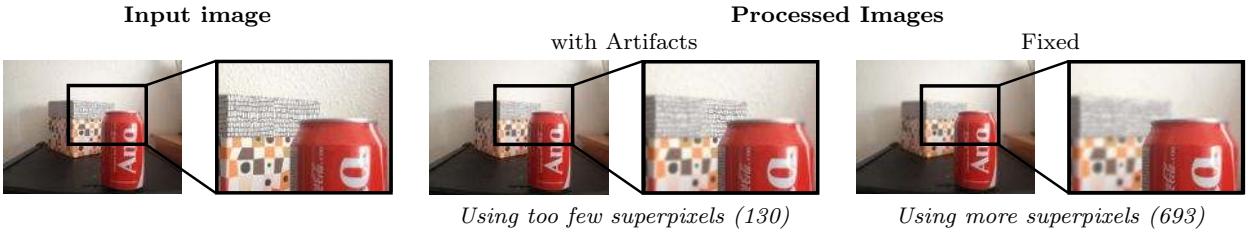


Fig. 12 Example where superpixel segmentation issues produces artifacts in the processed images of our interactive depth of field application. Artifacts are caused when the superpixel boundaries do not correspond with the object boundaries. When superpixel segmentation is improved, these artifacts are not visible in the processed image.

vantages on three real applications. The supplementary material includes more additional examples obtained with our three interactive applications.

Execution time. Efficiency is a key contribution of the proposed interaction paradigm, as it guarantees response time requirements for applications that interact with a user. The number of superpixels defines the number of unknowns in the linear system and as consequence, it determines the propagation speed. We can adjust the number of superpixels depending on the image resolution in order to keep the interactive execution time. For example, in a typical image of 2000×1340 (with 950 superpixels), our system can propagate the user information in 0.20 seconds. This time includes the time for generating the user equations, solving the system and building the dense map solution. Table 2 shows a more detailed analysis of the execution time of each of the steps.

Table 2 Typical execution time per step with an image of 2008x1340. In this particular case, the dehazing filter had been applied as image operator (step 6). Note that, other image operators may require different execution time.

Step	**Time** (seconds)
Initialization	
1. Superpixel segmentation (950)	2.39
2. Add binary equations (2725)	0.08
Interactive propagation	
3. Add all user equations (819)	0.015
4. Solve linear system	0.20
5. Bilateral filter	0.45
6. Image operator	0.53

Note that, with each user stroke, the propagation is calculated and the filter is applied. The cost of our three filter algorithms is linear with respect to the number of pixels. In typical images, it can be applied in around a second . The three implemented applications are available online [2] and work at interactive rates in a regular desktop computer (Intel Core i5 2,5 GHz).

Limitations. Superpixel based approaches are faster to compute but typically less accurate than pixel based methods. Efficiency is achieved at a small sacrifice on accuracy of the final propagation and therefore, it affects the result obtained when the filter is applied to specific images. Figure 12 shows an example of the depth of field application. The focused object is the *red can*. As we can see, in the processed image with artifacts, part of the *box* is incorrectly focused (the *box* is behind the *red can* and therefore should be defocused). This artifact is produced by insufficient or incorrect superpixel segmentation, i.e., there is a misalignment between superpixel and actual object boundaries. These artifacts can be easily eliminated increasing the number of superpixels used.

Comparison with previous work. An important advantage of our system is that the proposed paradigm is generic, i.e., the same system works for different applications. In Section 5, we have presented three interactive image processing applications that use our proposed system. This section presents a set of representative visual results from all of them, comparing our solution with well known ad-hoc methods for each application. The following results validate that our generic system can achieve comparable quality in the visual effects than specific ad-hoc methods for each of the applications.

Figure 13 shows two examples from the results of our depth of field application side by side with the results obtained with a well known approach from Yu *et al.* [46]. Both the depth map and output images are similar, but our approach works with a single image instead of a sequence.

Figure 14 compares our results for dehazing against state-of-the-art single image dehazing methods [16,6].

[2] `https://github.com/anacambra/app_lensblur/tree/TVCJ`

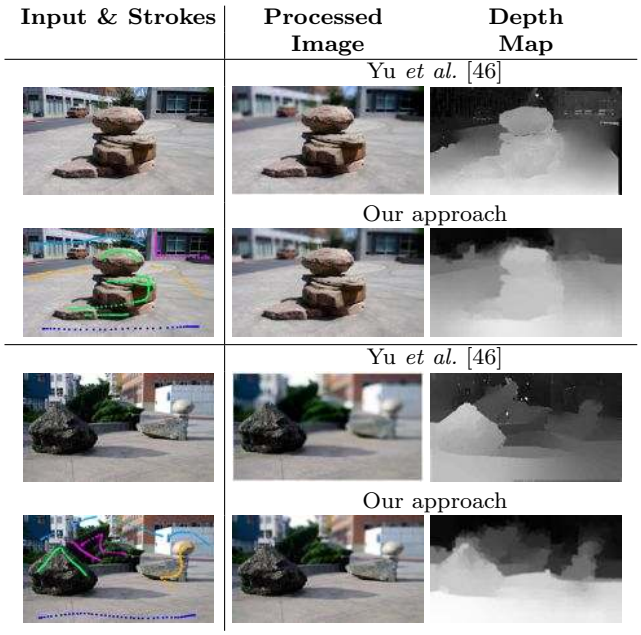


Fig. 13 Comparison of our results and a well known algorithm for depth of field effect. Depth map estimated and final processed image are similar, but our method works with a single image and very simple input (strokes), as opposed to Yu *et al.* work, which requires a sequence of images.

Note that our transmission maps may look less fine-grained than the other methods, because our propagation system is a superpixel based approach. However, the final visual effect in the dehazed images are of similar quality.

Figure 15 compares our results for tone mapping against a global tone mapping operator to adjust the luminance and contrast in the image. Our local tone mapping operators allow the user to obtain better solution than global operators. With a few strokes, the user can point which regions need to be illuminated or darken and adjust each region contrast. Figure 16 shows an example of the results of our HDR tone mapping application and the interactive tool proposed by Lischinski et al. [25]. Both provide the user an intuitive and local control of the region brightness but our method is faster and also provides the adjustment of the image contrast.

7 Conclusions

The main contribution of this work is a generic paradigm to propagate sparse information across all image pixels. The proposed paradigm is particularly useful to model continuous magnitudes, which is not feasible with many of the existing propagation techniques. Besides, our presented implementation of this paradigm has the advantage of running at interactive rates. This means our

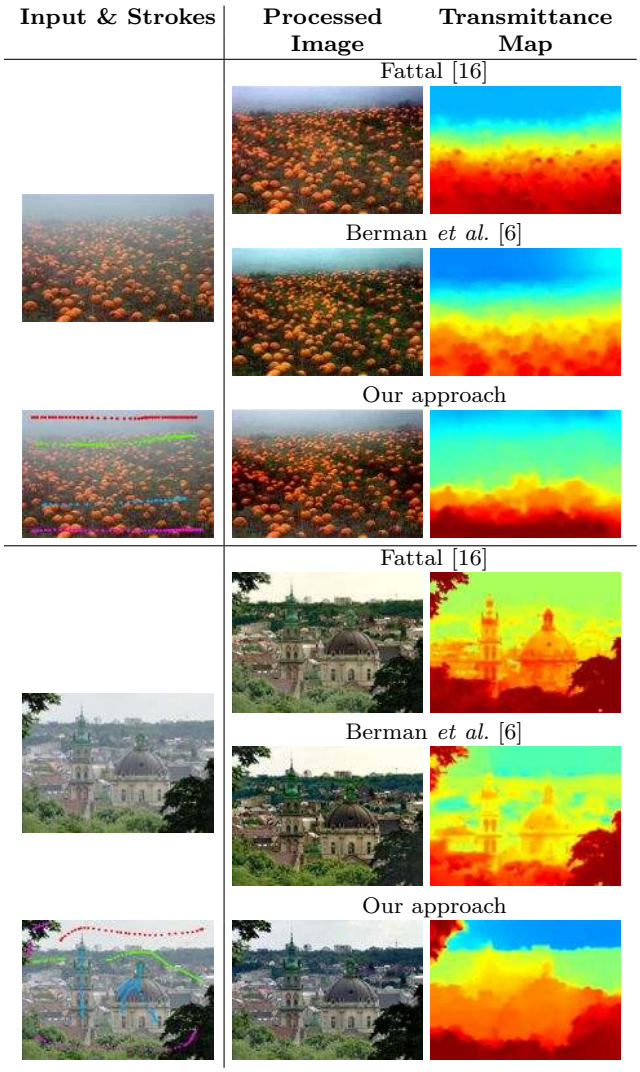


Fig. 14 Comparison of our results and state-of-the-art single image dehazing methods. Our approach obtains a less fine-grained transmittance estimation but similar dehazing visual results.

paradigm enables the development of any interactive application to apply image post processing techniques that require the estimation of continuous magnitudes per pixel.

While the interactivity of our approach is guaranteed because our paradigm is formulated as a linear system of equations over superpixels, the accuracy of our approach is also limited by the superpixels. However, we have shown that our results are accurate enough for many real applications. In particular, to demonstrate the suitability of our approach, we have implemented three interactive applications to apply complex well-known filters and effects: depth of field, dehazing and tone mapping. The three applications use the same propagation scheme presented in this work, and their results demonstrate that our generic propagation

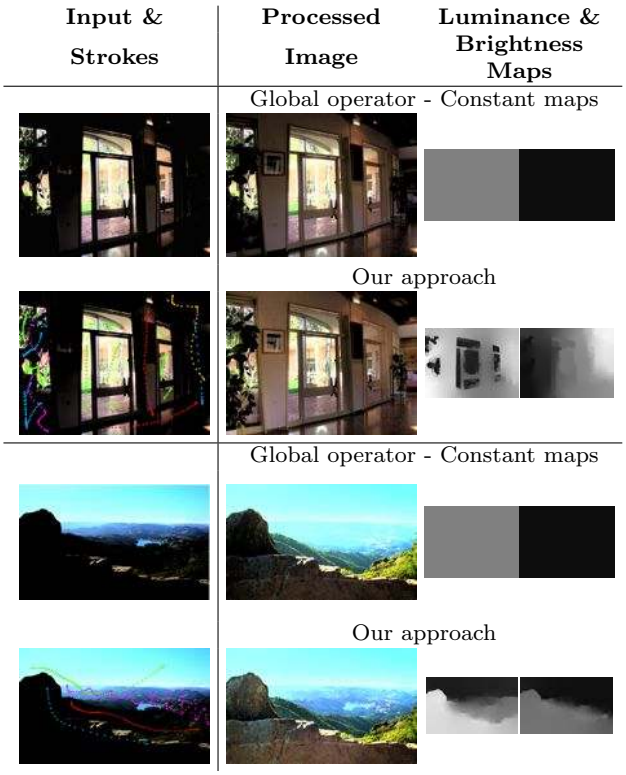


Fig. 15 Comparison of our HDR tone mapping application results. We can see that, with a few user strokes, the brightness and contrast adjusted locally work better than global operators.

(a) (b)

Fig. 16 Tone mapping produced by (a) Lischinski et al. approach [25] and by (b) our interactive application.

system achieves comparable image effects than related methods which are ad-hoc for a single specific problem. Besides, thanks our paradigm, these applications can run as interactive tools, with very low computational requirements. This opens the opportunity to bring this type of editing tools to mobile and embedded devices.

We believe our work can inspire both new interactive editing applications, as well as future research on interactive editing tools. Potential lines of future work that can be inspired from the combination of the presented work and previous work are multiview approaches for transferring edits between views [47] or for depth editing and navigation [12]. Besides, instead of plain images, our approach could be extended for material appearance [15] or even material transfer [3] applications, given the adequate adaptation and identification of the involved magnitudes to material space.

Acknowledgements This research has been partially funded by the European Research Council (ERC) under the EU Horizon 2020 program (CHAMELEON project, grant agreement No 682080), the Spanish Government (projects DPI2015-65962-R, DPI2015-69376-R) and Aragon regional government (Grupo DGA T04-FSE). The authors would like to thank D. Gutierrez and J.J. Guerrero for their support on this project.

References

1. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Susstrunk, S.: SLIC superpixels compared to state-of-the-art superpixel methods. Trans. Pattern Anal. Mach. Intell. **34**(11), 2274–2282 (2012)
2. An, X., Pellacini, F.: AppProp: all-pairs appearance-space edit propagation. In: ACM Transactions on Graphics (TOG), vol. 27, p. 40. ACM (2008)
3. An, X., Tong, X., Denning, J.D., Pellacini, F.: AppWarp: Retargeting measured materials by appearance-space warping. ACM Trans. Graph. **30**(6), 147:1–147:10 (2011)
4. Andrews, S., Hamarneh, G., Saad, A.: Fast random walker with priors using precomputation for interactive medical image segmentation. In: Medical Image Computing and Computer-Assisted Intervention, pp. 9–16. Springer (2010)
5. Ao, H., Zhang, Y., Jarabo, A., Masia, B., Liu, Y., Gutierrez, D., Dai, Q.: Light field editing based on reparameterization. In: Pacific Rim Conference on Multimedia (2015)
6. Berman, D., Treibitz, T., Avidan, S.: Non-local image dehazing. In: IEEE Conf. on Computer Vision and Pattern Recognition, pp. 1674–1682 (2016)
7. Besag, J.: On the statistical analysis of dirty pictures. Journal of the Royal Statistical Society. Series B (Methodological) pp. 259–302 (1986)
8. Bonneel, N., Sunkavalli, K., Tompkin, J., Sun, D., Paris, S., Pfister, H.: Interactive intrinsic video editing. ACM Transactions on Graphics **33**(6), 197 (2014)
9. Bousseau, A., Paris, S., Durand, F.: User assisted intrinsic images. ACM Transactions on Graphics (SIGGRAPH Asia 2009) **28**(5) (2009)
10. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. IEEE Transactions on Pattern Analysis and Machine Intelligence **23**(11), 1222–1239 (2001)
11. Cambra, A.B., Muñoz, A., Guerrero, J.J., Murillo, A.C.: Dense labeling with user interaction: an example for depth-of-field simulation. In: British Machine Vision Conference (2016)
12. Chaurasia, G., Duchene, S., Sorkine-Hornung, O., Drettakis, G.: Depth synthesis and local warps for plausible image-based navigation. ACM Transactions on Graphics **32**(3), 30 (2013)
13. Chen, Q., Koltun, V.: Fast MRF optimization with application to depth reconstruction. In: IEEE Conf. on Computer Vision and Pattern Recognition, pp. 3914–3921 (2014)

14. Chen, X., Zou, D., Zhao, Q., Tan, P.: Manifold preserving edit propagation. ACM Transactions on Graphics (TOG) **31**(6), 132 (2012)
15. Di Renzo, F., Calabrese, C., Pellacini, F.: AppIm: Linear spaces for image-based appearance editing. ACM Trans. Graph. **33**(6), 194:1–194:9 (2014)
16. Fattal, R.: Dehazing using color-lines. ACM Transaction on Graphics **34**(13) (2014)
17. Freedman, D.: An improved image graph for semi-automatic segmentation. Signal, Image and Video Processing **6**(4), 533–545 (2012)
18. Garces, E., Muñoz, A., Lopez-Moreno, J., Gutierrez, D.: Intrinsic images by clustering. Computer Graphics Forum **31**(4) (2012)
19. Grady, L.: Random walks for image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence **28**(11), 1768–1783 (2006)
20. Iizuka, S., Endo, Y., Kanamori, Y., Mitani, J., Fukui, Y.: Efficient depth propagation for constructing a layered depth image from a single image. Computer Graphics Forum (Proc. of Pacific Graphics 2014) **33**(7), 279–288 (2014)
21. Jarabo, A., Masia, B., Bousseau, A., Pellacini, F., Gutierrez, D.: How do people edit light fields? ACM Transactions on Graphics (SIGGRAPH 2014) **33**(4) (2014)
22. Kolmogorov, V.: Convergent tree-reweighted message passing for energy minimization. IEEE Transactions on Pattern Analysis and Machine Intelligence **28**(10), 1568–1583 (2006)
23. Kraus, M., Strengert, M.: Depth-of-field rendering by pyramidal image processing. Computer Graphics Forum **26**(3), 645–654 (2007)
24. Levin, A., Lischinski, D., Weiss, Y.: Colorization using optimization. In: ACM Transactions on Graphics, vol. 23, pp. 689–694. ACM (2004)
25. Lischinski, D., Farbman, Z., Uyttendaele, M., Szeliski, R.: Interactive local adjustment of tonal values. ACM Transactions on Graphics (TOG) **25**(3), 646–653 (2006)
26. Liu, M., Salzmann, M., He, X.: Discrete-continuous depth estimation from a single image. In: IEEE Conf. on Computer Vision and Pattern Recognition, pp. 716–723. IEEE (2014)
27. Lopez, A., Garces, E., Gutierrez, D.: Depth from a Single Image Through User Interaction. In: Spanish Computer Graphics Conf., pp. 11–20. The Eurographics Assoc. (2014)
28. Luan, Q., Wen, F., Cohen-Or, D., Liang, L., Xu, Y.Q., Shum, H.Y.: Natural image colorization. In: Proceedings of the 18th Eurographics conference on Rendering Techniques, pp. 309–320. Eurographics Association (2007)
29. Mantiuk, R., Seidel, H.P.: Modeling a generic tone-mapping operator. In: Computer Graphics Forum, vol. 27, pp. 699–708. Wiley Online Library (2008)
30. Mičušík, B., Košecká, J.: Multi-view superpixel stereo in urban environments. International Journal of Computer Vision **89**(1), 106–119 (2010)
31. Ning, J., Zhang, L., Zhang, D., Wu, C.: Interactive image segmentation by maximal similarity based region merging. Pattern Recognition **43**(2), 445–456 (2010)
32. Phan, R., Androutsos, D.: Robust semi-automatic depth map generation in unconstrained images and video sequences for 2d to stereoscopic 3d conversion. IEEE Transactions on Multimedia **16**(1), 122–136 (2014)
33. Reinhard, E., Stark, M., Shirley, P., Ferwerda, J.: Photographic tone reproduction for digital images. ACM Transactions on Graphics **21**(3), 267–276 (2002)
34. Ren, X., Bo, L., Fox, D.: Rgb-(d) scene labeling: Features and algorithms. In: IEEE Conf. on Computer Vision and Pattern Recognition, pp. 2759–2766. IEEE (2012)
35. Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. International Journal of Computer Vision **47**(1-3), 7–42 (2002)
36. Scharstein, D., Szeliski, R.: High-accuracy stereo depth maps using structured light. In: IEEE Conf. on Computer Vision and Pattern Recognition, vol. 1, pp. I–195 (2003)
37. Singaraju, D., Grady, L., Vidal, R.: P-brush: Continuous valued MRFs with normed pairwise distributions for image segmentation. In: IEEE Conf. on Computer Vision and Pattern Recognition, pp. 1303–1310. IEEE (2009)
38. Stutz, D., Hermans, A., Leibe, B.: Superpixels: An evaluation of the state-of-the-art. Computer Vision and Image Understanding (2017)
39. Szeliski, R., Zabih, R., Scharstein, D., Veksler, O., Kolmogorov, V., Agarwala, A., Tappen, M., Rother, C.: A comparative study of energy minimization methods for markov random fields with smoothness-based priors. IEEE Transactions on Pattern Analysis and Machine Intelligence **30**(6), 1068–1080 (2008)
40. Tappen, M.F., Freeman, W.T.: Comparison of graph cuts with belief propagation for stereo, using identical MRF parameters. In: IEEE International Conference on Computer Vision, pp. 900–906. IEEE (2003)
41. Tighe, J., Lazebnik, S.: Superparsing. International Journal of Computer Vision **101**(2), 329–349 (2013)
42. Wang, O., Lang, M., Frei, M., Hornung, A., Smolic, A., Gross, M.: StereoBrush: interactive 2d to 3d conversion using discontinuous warps. In: Proceedings of the Eighth Eurographics Symposium on Sketch-Based Interfaces and Modeling, pp. 47–54. ACM (2011)
43. Weinshall, D.: Qualitative depth from stereo, with applications. In: Computer Vision, Graphics, and Image Processing, pp. 222–241 (1990)
44. Xu, K., Li, Y., Ju, T., Hu, S.M., Liu, T.Q.: Efficient affinity-based edit propagation using k-d tree. ACM Trans. Graph. **28**(5), 118:1–118:6 (2009)
45. Yamaguchi, K., Hazan, T., McAllester, D., Urtasun, R.: Continuous markov random fields for robust stereo estimation. Computer Vision- ECCV pp. 45–58 (2012)
46. Yu, F., Gallup, D.: 3d reconstruction from accidental motion. In: IEEE Conf. on Computer Vision and Pattern Recognition, pp. 3986–3993 (2014)
47. Yücer, K., Jacobson, A., Hornung, A., Sorkine, O.: Transfusive image manipulation. ACM Transactions on Graphics (proceedings of ACM SIGGRAPH ASIA) **31**(6), 176:1–176:9 (2012)
48. Yücer, K., Sorkine-Hornung, A., Sorkine-Hornung, O.: Transfusive weights for content-aware image manipulation. In: Proceedings of the Symposium on Vision, Modeling and Visualization (VMV). Eurographics Association (2013)