Published in Image Processing On Line on 2018–11–17.
Submitted on 2018–08–04, accepted on 2018–10–24.
ISSN 2105–1232 © 2018 IPOL & the authors CC–BY–NC–SA
This article is available online with supplementary materials,
software, datasets and online demo at
https://doi.org/10.5201/ipol.2018.230

An Implementation of the Exposure Fusion Algorithm

Charles Hessel

CMLA, ENS Cachan, CNRS,
Université Paris-Saclay, 94235 Cachan, France
charles.hessel@cmla.ens-cachan.fr

Abstract

Exposure Fusion is a high dynamic range imaging technique to fuse a bracketed exposure sequence into a high quality image, introduced in 2009 by Mertens et al. Contrarily to most HDR imaging methods, exposure fusion does not construct an intermediate HDR image but directly constructs the final LDR one by seamlessly fusing the best regions of the input sequence, using the Laplacian pyramid. Since its publication, this method received considerable attention, being both effective and efficient. We propose in this paper a precise description of the method and an analysis of its main limitation, an out-of-range artifact.

Source Code

The source code, the code documentation, and the online demo are accessible at the web page of this article[1]. It uses the Matlab implementation provided by T. Mertens[2], merely adapted for its execution with Octave on the IPOL platform.

Keywords: high dynamic range; image fusion; tone-mapping; Laplacian pyramids

1 Introduction

The dynamic range of real scenes is generally higher than the one of our camera sensors. To capture the entire dynamic range, photographers are led to acquire a sequence of images with different exposure times: long times capture information in dark parts of the scene and saturate the bright ones, while short exposure times capture relevant information in the bright parts. The result of this acquisition is called a bracketed exposure sequence. This sequence must then be merged into a high dynamic range (HDR) image, which gets a far higher number of bits than those that can be displayed on normal screens. Thus the HDR image needs to be remapped to the low dynamic range (LDR) of most displays through a tone-mapping operator, which alters the colors to make them fit all in the 8 bits Procrustean bed.

[1] https://doi.org/10.5201/ipol.2018.230
[2] https://github.com/Mericam/exposure-fusion

Exposure Fusion [10, 11] was introduced by T. Mertens, J. Kautz and F. Van Reeth in 2009 as an alternative way of constructing an LDR image from a bracketed exposure sequence. This method does not build an intermediate HDR picture. In a nutshell, it directly selects for each pixel the values, among the provided pictures, which should be kept in the final image. As a result, the fused image combines the best areas of the various input images. Although similar techniques already existed [5], this technique has brought interesting and successful answers to two crucial questions: how to detect the best pixel from the provided set of images, and how to seamlessly merge those pixels in the final image.

We shall first give a precise description of the bracketed exposure sequence fusion algorithm in Section 2, and of the Laplacian pyramid construction in Section 3. In Section 4 we explore the effect of the parameters on the fused image. Then, we present a simple algorithm for the robust normalization of the output image in Section 5. We propose in the demo to register the input sequence before applying the exposure fusion, so as to avoid ghosts on sequences with misalignment. We describe the implemented registration algorithm in Section 6. Next, in Section 7 we discuss some results of exposure fusion and carry out an experiment allowing to easily visualize the participation of the input images in the final result. In Section 8 we discuss and explain the saturation that generally occurs in the fused images, and conclude in Section 9.

2 Exposure Fusion

Exposure fusion first measures the perceptual quality of each pixel in each image of the input sequence. Three pixel-wise metrics are used: the contrast C, saturation S and well-exposedness E. We will denote in the following by $\mathbf{x} = (x_1, x_2)$ the position of the pixel in a image, by c the color channel, and by k the position of the image in the input sequence. We shall use in the paper the convention that the intensity range of the images is $[0, 1]$ for the computations (floating point numbers with high precision), but $[0, 255]$ for the display (8-bits representation).

The *contrast metric* uses the absolute value of a discrete Laplacian filter applied to the grayscale version of the image. Denoting by $K_{\text{Laplacian}}$ a Laplacian kernel, we set

$$C_k(\mathbf{x}) = \left| \left(\frac{1}{3} \sum_{c=1}^{3} u_{c,k} \right) * K_{\text{Laplacian}} \right| (\mathbf{x}). \tag{1}$$

The authors use for $K_{\text{Laplacian}}$ the sum of differences over the four nearest neighbors, i.e.

$$K_{\text{Laplacian}} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}.$$

The *saturation metric* is the standard-deviation of the pixel's color,

$$S_k(\mathbf{x}) = \sqrt{\frac{1}{3} \sum_{c'=1}^{3} \left(u_{c',k}(\mathbf{x}) - \frac{1}{3} \sum_{c=1}^{3} u_{c,k}(\mathbf{x}) \right)^2}. \tag{2}$$

Finally, the *well-exposedness* metric measures how close each pixel's color channel is to the median value 0.5 using a Gauss curve

$$E_k(\mathbf{x}) = \prod_{c=1}^{3} \exp \frac{-\left(u_{c,k}(\mathbf{x}) - 0.5\right)^2}{2\sigma^2}, \tag{3}$$

with $\sigma = 0.2$.

The quality measure of each pixel is finally obtained as a product of these three metrics. By using the product, the authors force their method to only keep pixels which are acceptable for the three qualities simultaneously. To allow the user to choose the importance given to each quality measure, they added a power function to each one, with parameters ω_c, ω_s and ω_e (by default equal to 1)

$$w_k(\mathbf{x}) = C_k(\mathbf{x})^{\omega_c} \times S_k(\mathbf{x})^{\omega_s} \times E_k(\mathbf{x})^{\omega_e}. \tag{4}$$

For the blending process, the resulting weights need to be normalized as

$$\widehat{w}_k(\mathbf{x}) = \frac{w_k(\mathbf{x})}{\sum_{k'=1}^{N} w_{k'}(\mathbf{x})}, \tag{5}$$

where N is the number of images in the sequence.

At this point, each input image has its normalized weight map. As the authors explain, one could directly use them to fuse the images. But such an operation would lead to strong seams due to the sharp variations in the weights. They instead propose a multiscale fusion, using the method introduced by Ogden et al. [9]. This technique builds the Laplacian pyramid [4] of the output image by blending the Laplacian pyramids of the input images according to the Gaussian pyramid of the weight maps. The fused image is obtained by collapsing the constructed pyramid. We will denote $L\{u\}$ the Laplacian pyramid of the input image u, $G\{w\}$ the Gaussian pyramid of the weights, and l the scale. The blending operation is then

$$L\{v\}^l(\mathbf{x}) = \sum_{k=1}^{N} G\{\widehat{w}\}_k^l(\mathbf{x}) L\{u\}_k^l(\mathbf{x}). \tag{6}$$

Algorithm 1 describes the whole process, from the quality measurements to the multiscale fusion.

Algorithm 1: Exposure Fusion

input : input sequence of images u
input : $\omega_s, \omega_c, \omega_e$: weights for saturation, contrast and well-exposedness measures, respectively
output: fused image v

foreach *image at position* $k \in \{1, 2, \ldots, N\}$ *in the input sequence* **do**
    Compute contrast metric C using Equation (1)
    Compute saturation metric S using Equation (2)
    Compute well-exposedness metric E using Equation (3)
    Compute weight map W_k of the current image using Equation (4)
Normalize weights using Equation (5)
Initialize output pyramid $L\{v\}$ with zeros
foreach *image at position* $k \in \{1, 2, \ldots, N\}$ *in the input sequence* **do**
    Compute Gaussian pyramid of weights $G\{\widehat{w}\}_k$
    Compute Laplacian pyramid of input image $L\{u\}_k$
    foreach *coefficient at position* $\mathbf{x}$ *and scale* l **do**
        Update Laplacian pyramid of the output image:
        $L\{v\}^l(\mathbf{x}) \leftarrow L\{v\}^l(\mathbf{x}) + G\{\widehat{w}\}_k^l(\mathbf{x}) L\{u\}_k^l(\mathbf{x})$
$v \leftarrow$ collapse Laplacian pyramid $L\{v\}$

While the sum of the weights is guaranteed for every pixel to be equal to 1, this does not imply that the reconstructed image belongs to the initial interval. In fact it may well happen that saturation

occurs in the dark or bright part. Avoiding them is possible by applying an affine rescaling of the image's dynamic to fit it to the standard interval $[0, 1]$. In our experiments, the resulting image generally presented no artifacts. The authors however present a case where the output image suffers from a very low frequency halo, giving an unnatural sensation (see fig. 6 of their paper [11]). We describe and explain this effect in Section 8.

3 Laplacian Pyramid Blending

The Burt et al. [4] Gaussian pyramid of u is constructed by recursively downsampling the image by factors of two until its size is only one pixel. The last scale, the coarser one, will be denoted by $l_{\max}$. The Laplacian pyramid at scale l corresponds to the difference between two scales l and $l+1$ of the Gaussian pyramid, the second one being upsampled by a factor two. The last scale of the Laplacian pyramid is called the residual, and is equal to the coarsest scale of the Gaussian pyramid. Formally,

$$G\{u\}^l(\mathbf{x}) = \begin{cases} u(\mathbf{x}) & \text{if} \quad l = 0 \\ \mathit{Downsample}\big(G\{u\}^{l-1}\big)(\mathbf{x}) & \text{if} \quad l > 0 \end{cases}, \tag{7}$$

$$L\{u\}^l(\mathbf{x}) = \begin{cases} G\{u\}^l(\mathbf{x}) - \mathit{Upsample}\big(G\{u\}^{l+1}\big)(\mathbf{x}) & \text{if} \quad l < l_{\max} \\ G\{u\}^l(\mathbf{x}) & \text{if} \quad l = l_{\max} \end{cases}, \tag{8}$$

where the *Downsample* and *Upsample* operators are defined in Algorithm 2 and Algorithm 3, respectively. The filter K used for downsampling and upsampling is the one defined by Burt et al. in 1983 [4]

$$\begin{aligned} k &= [.05, .25, .4, .25, .05] \text{ (in 1D)} , \\ K &= k^{\mathrm{T}} k \qquad\qquad\quad \text{(in 2D)}. \end{aligned} \tag{9}$$

The half-symmetric boundary extension [3] is used for the convolutions.

The input image can be recovered by "collapsing" the Laplacian pyramid, that is, recursively upsampling and adding the Laplacian coefficients, starting from the residual. Indeed, $G\{u\}^l = L\{u\}^l + \mathit{Upsample}\big(G\{u\}^{l+1}\big)$ and $G\{u\}^0 = u$. The *Collapse* operator is presented in Algorithm 4. In order to handle images with arbitrary height and width, *Upsample* adds a line and/or a column when needed so that the height and width of the upsampled image are the same than before downsampling (parameters odd_h and odd_w at line 8). When performing the convolution in the downsampling procedure, the borders are replicated. In the upsampling procedure, border handling is made explicit at lines 2 and 8.

In the authors' implementation of exposure fusion, the number of scales is fixed to

$$l_{\max} = \lfloor \log(\min\{\text{height}, \text{width}\}) / \log(2) \rfloor, \tag{10}$$

where $\lfloor \cdot \rfloor$ is nearest inferior integer operator. This leads to the deepest possible pyramid when $\min\{\text{height}, \text{width}\}$ is a power of two, but this seldom happens. Because a column or a line is added before downsampling when the size is odd, the image at scale $l_{\max}$ is generally a few pixels wide.

4 Exploring the Method's Parameters

In Figure 1 we explore the influence of the parameters ω_c, ω_s, and ω_e. We shall consider only two values: 1 and 0, that is, "on" or "off". The first remark is that giving equal weights to all images is not viable: the image with all metrics to "off" gives the worst result (top left). Further, we observe

Algorithm 2: *Downsample*

input : image u with height H and width W
output: v the downsampled image

$K \leftarrow$ Burt and Adelson's kernel defined in (9)
$\bar{u} \leftarrow u * K$ // convolve the image using half-symmetric boundary extension
for $x_1 = 1$ **to** $\lfloor H/2 \rfloor - 1$ **and** $x_2 = 1$ **to** $\lfloor W/2 \rfloor - 1$ **do**
    $v(x_1, x_2) \leftarrow \bar{u}(2x_1, 2x_2)$ // re-sample
return v

Algorithm 3: *Upsample*

input : image u with height H and width W
input : parameters odd_h, odd_w
output: $\bar{u}^\uparrow$ the upsampled image of size $(2H + \text{odd}_\text{h}, 2W + \text{odd}_\text{w})$.

$K \leftarrow$ Burt and Adelson's kernel defined in Equation (9)
$u_\text{pad} \leftarrow$ increase size of u by replicating its first and last lines and columns
$u^\uparrow_\text{pad} \leftarrow$ initialize with zeros an image of size $(H', W') = (2H + 4, 2W + 4)$
for $x_1 = 0$ **to** $H + 1$ **and** $x_2 = 0$ **to** $W + 1$ **do**
    $u^\uparrow_\text{pad}(2x_1, 2x_2) \leftarrow 4 \times u_\text{pad}(x_1, x_2)$ // factor 4 for normalization
$\bar{u}^\uparrow_\text{pad} \leftarrow u^\uparrow_\text{pad} * K$ // interpolate with the same filter K
$\bar{u}^\uparrow \leftarrow$ remove the 2 first lines and 2 first columns from $\bar{u}^\uparrow_\text{pad}$ // remove padding
$\bar{u}^\uparrow \leftarrow$ remove the $(2 - \text{odd}_\text{h})$ and $(2 - \text{odd}_\text{w})$ last lines and columns from $\bar{u}^\uparrow$ // remove padding

Algorithm 4: *Collapse*

input : Laplacian pyramid $L\{u\}$
output: image u

$u^{l_\text{max}} \leftarrow L\{u\}^{l_\text{max}}$ // residual
for $l = l_\text{max} - 1$ **to** 0 **do**
    $\text{odd}_\text{h} \leftarrow \text{height}(L\{u\}^l) - 2 \times \text{height}(u^{l+1})$
    $\text{odd}_\text{w} \leftarrow \text{width}(L\{u\}^l) - 2 \times \text{width}(u^{l+1})$
    $u^l \leftarrow L\{u\}^l + \mathit{Upsample}\big(u^{l+1}, \text{odd}_\text{h}, \text{odd}_\text{w}\big)$
return u^0

that once the well-exposedness is "on" (right column), the other two parameters seem to have only little influence on the result. In the column with $\omega_e = 0$, we can compare the respective influence of the contrast and saturation metrics. It appears that the saturation is useful to find the best parts to fuse from the input sequence, whereas the contrast measure does not help much. Indeed, the image with $\omega_c = 1$ and $\omega_s = \omega_e = 0$ is close to the worst one (top left), whereas the image with $\omega_s = 1$ and $\omega_c = \omega_e = 0$ is close to the "optimal" one, with all the parameters activated. To conclude, on this sequence the well-exposedness metric is clearly the more useful, the saturation metric helps but slightly (the difference is small between $\omega_c = \omega_s = 0$, $\omega_e = 1$ and $\omega_c = 0$, $\omega_s = \omega_e = 1$), and the contrast metric helps only when the other two are not used.

In Figure 2 we compare the results obtained using only the well-exposedness metric ($\omega_c = \omega_s = 0$, $\omega_e = 1$), but with different values for its σ. The values between 0.1 and 0.2 seem to be the optimal ones for this sequence (we remind that the dynamic range is $[0, 1]$). Above and below those values the result degrades. This is easily explainable for values $\sigma > 0.2$: with such values every image tends to be a good candidate according to the metric, so the blending does not favor a particular image at each pixel, which results in a blending of all the images similar to the case $\omega_c = \omega_s = \omega_e = 0$ (see Figure 1, top left image). For values $\sigma < 0.10$, the problem is practically the same: since for certain pixels none of the input images have a value close enough to the mid-histogram value 0.5, the weights get equally distributed for all input images, which does not lead to optimal results (see Figure 1, top left image). This happens for example for the pixels of the left chair when $\sigma = 0.05$. This can be verified in Figure 3, where we display the weight maps obtained with $\sigma = 0.50$ and $\sigma = 0.05$.

5 Clipping and Robust Normalization

The images obtained with exposure fusion do not always fit the standard $[0, 1]$ dynamic. One can thus either clip the values or normalize the fused image. Both options are proposed in the demo associated to this paper. We shall describe now the robust normalization we suggest to use.

Algorithm 5: Robust Normalization

input : color image u with N pixels
input : percentage of saturation in the white and the black, respectively S_{white} and S_{black}
output: image v

1 $u_{\max} \leftarrow \max\{u_R, u_G, u_B\}$ // compute max channel
2 $u_{\min} \leftarrow \min\{u_R, u_G, u_B\}$ // compute min channel
3 $\overline{u}_{\max} \leftarrow \mathit{Sort}(u_{\max})$
4 $\overline{u}_{\min} \leftarrow \mathit{Sort}(u_{\min})$
5 $\text{val}_{\max} \leftarrow \overline{u}_{\max}(\lceil (1 - \frac{S_{\text{white}}}{100})N - 1 \rceil)$ // find maximal value
6 $\text{val}_{\min} \leftarrow \overline{u}_{\min}(\lfloor \frac{S_{\text{white}}}{100} N \rfloor)$ // find minimal value
7 $v_{R,G,B} \leftarrow (u_{R,G,B} - \text{val}_{\min})/(\text{val}_{\max} - \text{val}_{\min})$ // rescale the intensities
8 $v_{R,G,B} \leftarrow \max\{0, \min\{1, v_{R,G,B}\}\}$ // clip out-of-range values

In Algorithm 5 we describe the implementation of the robust normalization of color images. First, the max and min channels of the image are computed (Lines 1 and 2). This avoids counting the same pixel several times when finding the value at which saturation will occur. Then, the min and max values $v_{\min}$, $v_{\max}$ of the output image are found, using the user-set parameters S_{black} and S_{white} respectively (Lines 5 and 6). These values then serve to rescale the intensities (Line 7), then the rescaled image is clipped in $[0, 1]$ (Line 8).

Using the max "channel" to compute the value $\text{val}_{\max}$ and the min "channel" for $\text{val}_{\min}$ means that a pixel will be counted as clipped as soon as one of its color channels is clipped, and a pixel with

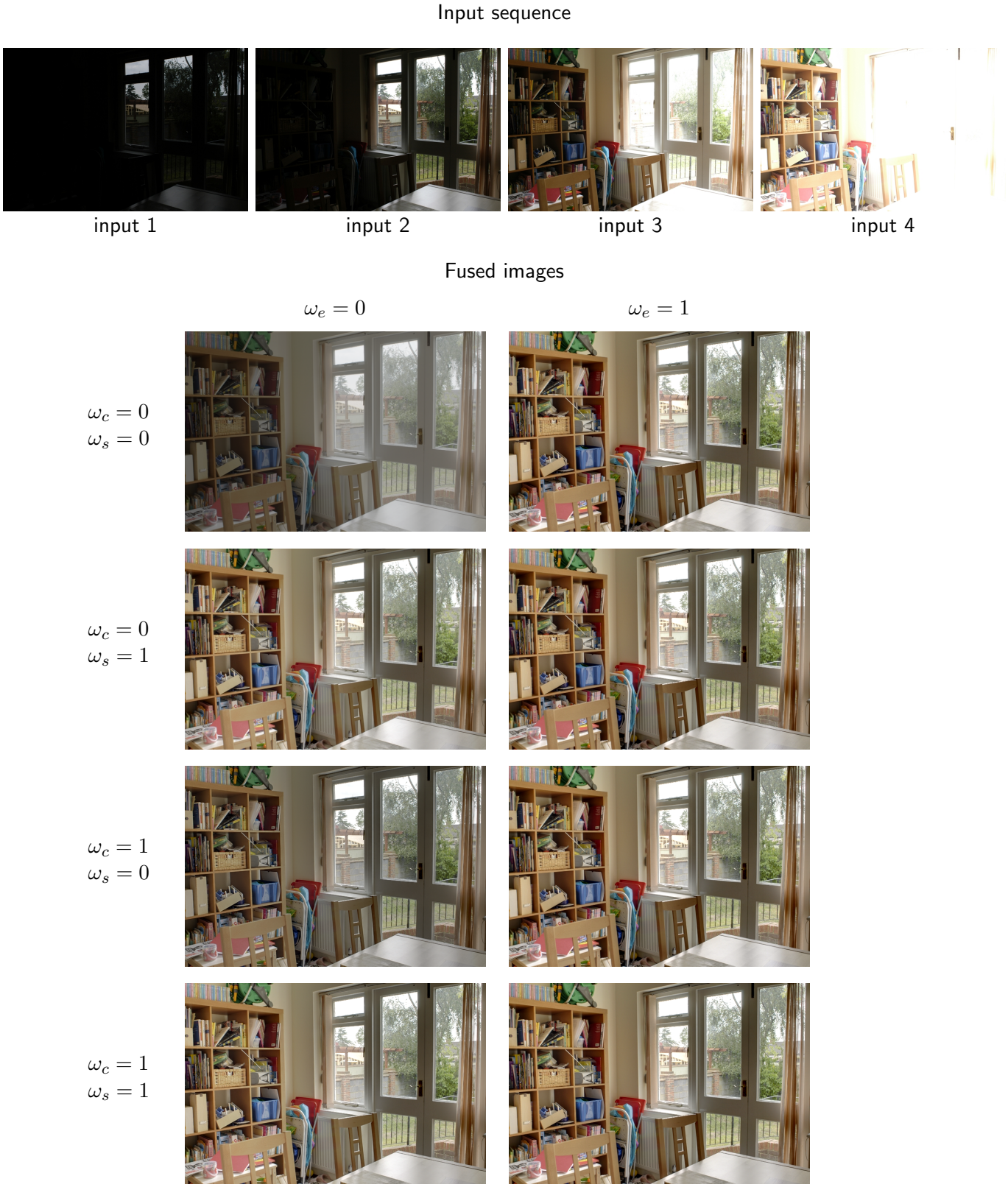


Figure 1: Influence of the parameters on the final result. We present the eight possible combinations of the parameters when they are either switched on (value $= 1$) or off (value $= 0$). The top line displays the input sequence. The table below display the fused results: in the lines we change the contrast (ω_c) and saturation (ω_s) parameters, and in the columns the well-exposedness (ω_e) parameter.

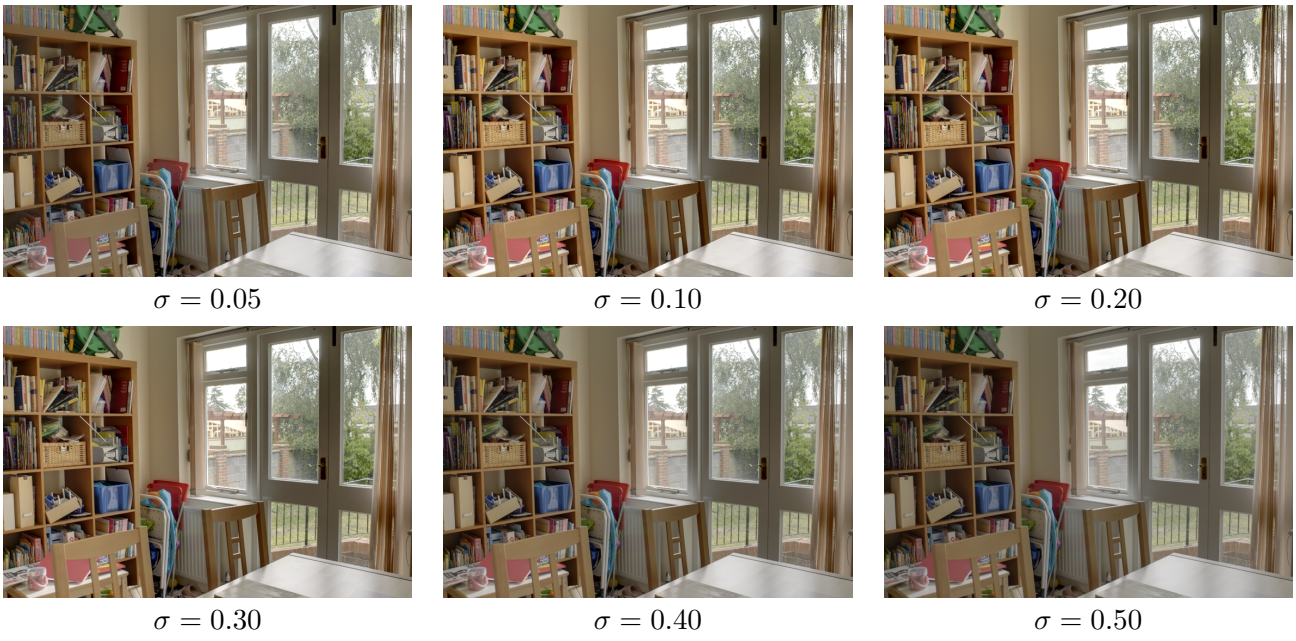


Figure 2: Influence of the parameter σ in the well-exposedness measure (Equation (3)).

Figure 3: Weights obtained with the sequence displayed in Figure 1, with parameters $\omega_c = \omega_s = 0$ and $\omega_e = 1$. In the top row with $\sigma = 0.50$, the weights tend to be given equally in the four images. In the second row with $\sigma = 0.05$, weights are given equally to all pixels whose value is far from 0.5 in every input image.

two or more clipped color channels will still be counted as one clipped pixel. Alternatively, one could normalize the luminance channel only and re-apply the color coefficients afterwards as described in [8], but this is slightly less precise as it does not allow to control the number of saturated color pixels.

6 Image Registration

Exposure fusion assumes that the input images are perfectly aligned and do not contain moving objects. For convenience, we added an optional registration step in the demo. The registration algorithm, described in Algorithm 6, consists of three steps:

1. **Midway Image Equalization.** (Line 3 and 6 in Algorithm 6) The midway image equalization algorithm [6] aims at giving a pair of images the same histogram. This equalization is required because the estimation of the homography (step 2) is intensity-based and therefore assumes images with similar intensities. We use the implementation described in [7]. Quoting [7],

   > A satisfactory definition of the midway histogram between [the discrete histograms of images u_1 and u_2, respectively] h_1 and h_2 is the harmonic mean between their cumulative histograms H_1 and H_2, i.e. $H_{\text{Midway}} = \left(0.5(H_1^{-1} + H_2^{-1})\right)^{-1}$. The resulting contrast changes f_{12} and f_{21} applied to the images u_1 and u_2 are
   >
   > $$\begin{aligned} f_{12}(x) &= \tfrac{1}{2}(x + H_2^{-1} \circ H_1(x)), \\ f_{21}(x) &= \tfrac{1}{2}(x + H_1^{-1} \circ H_2(x)). \end{aligned} \tag{11}$$

2. **Estimation of the homography.** (Lines 4 and 7 in Algorithm 6) Then, the motion between the two images is estimated using the Briand, Facciolo and Sánchez algorithm [2]. This method is an improvement of [1]. Let ρ be an error function, $\Psi(\cdot;\mathbf{p}) : \mathbb{R}^2 \to \mathbb{R}^2$ the homography parametrized by $\mathbf{p} \in \mathbb{R}^8$, and Ω the image spatial domain. The parameters $\hat{\mathbf{p}}$ of the homography between images u_1 and u_2 are found by minimizing

   $$\sum_{\mathbf{x}\in\Omega} \rho\left(\|u_2(\Psi(\mathbf{x};\mathbf{p})) - u_1(\mathbf{x})\|^2\right), \tag{12}$$

   using an iterative scheme, with a coarse-to-fine approach.

3. **Interpolation of the registered image.** (Lines 13 in Algorithm 6) Using the estimated homography, we then interpolate the registered image using [3], which proposes an efficient algorithm for B-spline interpolation. This consists of two steps. First, a prefiltering step in which the B-spline coefficients are computed. This provides a B-spline representation of the signal. Second, the values of the signal at the new, registered positions are computed as a convolution of the B-spline coefficients with the normalized B-spline function. We use the spline of order 5. The pixels values that would be interpolated from outside the original domain are set to zero.

Limitations. The second step of the algorithm assumes images with similar intensities. However, if the images in the pair to register have not the same "valid" (non-clipped) pixels, the preliminary midway equalization step will not suffice to compensate their difference. Take for example the darkest and the brightest image of a long sequence. Trying to register them is almost bound to fail, because most of the pixels are clipped to black in the former, and most of the pixels are clipped to white

in the latter, and the non-clipped pixels in these images are not the same! In this case the midway equalization cannot help.

To reduce this problem, we estimate the homographies using only consecutive images in the sequence. Assuming that this sequence is sorted according to the exposure time, this means that we use pairs of images that are the closest in terms of intensity difference. These estimated homography matrices $H_{n\to(n+1)}$ (where n is the index of the image in the sequence) are then composed in order to obtain the homography matrix $H_{n\to n_{\text{ref}}}$ that will be used to register the image n on the reference (at index n_{ref}). This corresponds to the matrix multiplication at Lines 9 and 11 in Algorithm 6. To minimize the propagation of eventual errors of estimation, n_{ref} is defined as the center of the sequence.

Algorithm 6: Registration

input : Sequence of images $u_0, u_1, \dots, u_{(N-1)}$ ordered in function of their exposure time
output: Sequence of registered images $\tilde{u}_0, \tilde{u}_1, \dots, \tilde{u}_{(N-1)}$

1 $n_{\text{ref}} \leftarrow \lfloor (N-1)/2 \rfloor$ `// index of reference image`
2 **for** $n = 0$ **to** $n_{ref} - 1$ **do** `// from begin to reference: forward estimation`
  `// Give the pair of images the same histogram using [7]`
3 $(u_n^{\text{Midway}}, u_{n+1}^{\text{Midway}}) \leftarrow \mathit{MidwayEqualization}(u_n, u_{n+1})$
  `// Estimate the homography's matrix using Inverse Compositional Algorithm [2]`
4 $H_{n\to(n+1)} \leftarrow \mathit{HomographyEstimation}(u_n^{\text{Midway}}, u_{n+1}^{\text{Midway}})$
5 **for** $n = n_{ref} + 1$ **to** $N - 1$ **do** `// from reference to end: backward estimation`
6 $(u_n^{\text{Midway}}, u_{n-1}^{\text{Midway}}) \leftarrow \mathit{MidwayEqualization}(u_n, u_{n-1})$
7 $H_{n\to(n-1)} \leftarrow \mathit{HomographyEstimation}(u_n^{\text{Midway}}, u_{n-1}^{\text{Midway}})$
8 **for** $n = n_{ref} - 2$ **to** $n = 0$ **do** `// complete the homography list: backward`
9 $H_{n\to n_{\text{ref}}} \leftarrow H_{(n+1)\to n_{\text{ref}}} H_{n\to(n+1)}$
10 **for** $n = n_{ref} + 2$ **to** $n = N - 1$ **do** `// complete the homography list: forward`
11 $H_{n\to n_{\text{ref}}} \leftarrow H_{(n-1)\to n_{\text{ref}}} H_{n\to(n-1)}$
12 **for** $n = 0$ **to** $N - 1$ **do**
  `// Register image on the reference using [3]`
13 $\tilde{u}_n \leftarrow \mathit{Interpolate}(u_n, H_{n\to n_{\text{ref}}})$

In our implementation of Algorithm 6, most of the operations are performed in parallel. The two first loops are first executed in parallel; when finished, the two following ones are run concurrently (each one is recursive, so they cannot be parallelized, but they are independent). When all the homographies are computed, the last loop is computed in parallel.

7 Experiment with the Laplacian Pyramid Blending

We present in this section results obtained with sequences of three images. The input sequences and the weight maps, computed as explained in Section 2, are displayed respectively on the top and middle rows of Figures 4 and 6. Each weight map is associated with the image above. The result of the fusion is displayed in Figures 5 (a) and 7 (a), respectively.

In order to visualize how the images are actually blended in the final result, we make the following experiment: using the weight maps computed on the actual input sequence, we fuse another sequence whose constitutive images are still identifiable when merged together (bottom row of Figures 4 and 6). This sequence is composed of color images with only one non-zero channel, and can therefore be

created only from sequences of three images. We first compute the luminance channel of the input images using

$$u_\ell = 0.2989u_{\mathrm{R}} + 0.5870u_{\mathrm{G}} + 0.1140u_{\mathrm{B}}, \tag{13}$$

then we create three single color images by assigning the luminance of each original image to the red, green and blue channels of the new images respectively, and setting the other channels to zero. For example, in the sequence generated in Figure 4, the luminance of the correctly exposed image is assigned to the red channel of the first image, the luminance of the under-exposed image is assigned to the blue channel of the second image, and the luminance of the over-exposed image is assigned to the green channel of the third image.

The result of the fusion of these color sequences is displayed in Figure 5 (b) and Figure 7 (b). We applied a final square-root for visualization purposes (we remind that the range is $[0, 1]$). These images permit to see where the input images participate in the final result after the multiscale blending.

Observations

On these colored images one can easily identify the source pixels that participated in the fused image. In the fused image, a colored pixel means that it comes from only one input image, while a gray pixel results from the blending of the three images.

First of all, it becomes clear from these figures that almost every part of the fused image is a blend of several images. In Figure 5 (b) for example, apart from the green area on the right, nothing is saturated. The sky, in particular, should be completely blue, as the weights (second row in Figure 4) give largely this area to the underexposed image. However, it appears that much of the green image also contributes to this area. Furthermore, the dome should be completely green and it appears that the blue image contributes significantly. In Figure 7 (b), one can see that the left part of the image essentially comes from the over-exposed (green) image, which is expected given its weight map. However, this green image also contributes largely to the sky along the left and right buildings, whereas it should be essentially blue and red on the right. Furthermore, the head of the masked person should be as green as the rest of the body but contains blue. The reflection of the sky on the floor behind the person should be red yet is mainly green.

The Laplacian pyramid blending diffuses across the edges of the weight maps. This explains the appearance of a seamless result, but it also causes the out-of-range artifacts that we discuss in the next section.

8 Saturation in the exposure fusion method

Saturation occurs in the original Mertens et al. method, as shown in Figure 9 (original sequence and fusion weights shown in Figure 8) and Figure 10. Even though weights are normalized and none of the input images exceeds the final dynamic range, the fused image can inherit a larger dynamic range than any of the input images. The original exposure fusion method [10, 11] simply clips the values that exceed the dynamic range, but this results in saturated areas in the final image. The authors added in their 2009 paper [11] the following remark:

> Another issue concerns out-of-range artifacts. The pyramid reconstruction does not guarantee that the resulting intensities lie within $[0, 1]$, even if the original intensities were restricted to this domain. (...) One can simply fix this issue by shifting and scaling the intensities, at the risk of reducing contrast.

Figure 4: From top to bottom rows: Input sequence, normalized weights computed by exposure fusion, and color images that will be fused with those weights for better analysis of the merging result.

(a) fusion result with original sequence (b) fusion result with color sequence

Figure 5: Image (a): fusion of the original sequence. Image (b): fusion of the color sequence using the weights computed on the original sequence. A square-root was applied to (b) for better visualization.

Figure 6: From top to bottom rows: Input sequence, normalized weights computed by exposure fusion, and color images that will be fused with those weights for better analysis of the merging result.

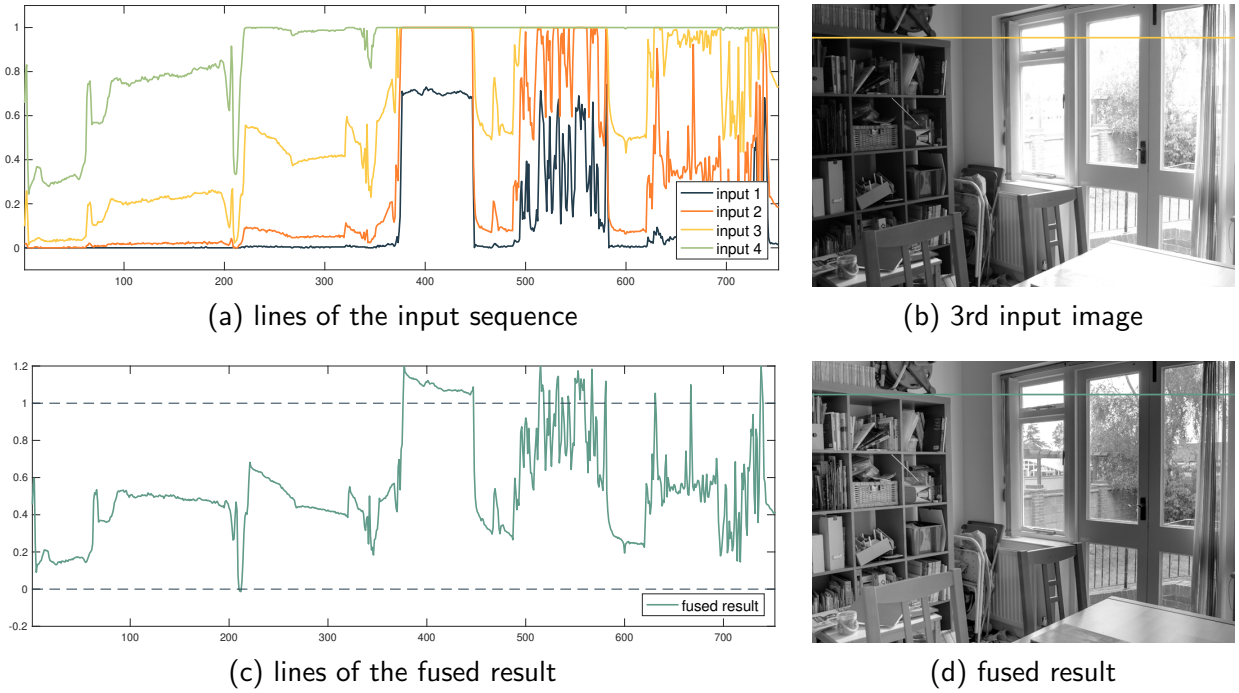
(a) fusion result with original sequence (b) fusion result with color sequence

Figure 7: Image (a): fusion of the original sequence. Image (b): fusion of the color sequence using the weights computed on the original sequence. A square-root was applied to (b) for better visualization.

Figure 8: Input sequence (top row) and the corresponding normalized weights maps (bottom row). The default parameters used for this experiment are $\omega_c = 1, \omega_s = 1, \omega_e = 1$. (Images courtesy of Min H. Kim.)

(a) exposure fusion output (clipped) (b) normalized output

Figure 9: Result of the exposure fusion method (a) with the default parameters given in Figure 8: $\omega_c = 1, \omega_s = 1, \omega_e = 1$. Saturation occurs in the bright parts of the windows, despite the fact that the input images used in these areas were not saturated. The information is preserved by the fusion but the image is clipped at the end of the process, thus incurring in information loss. In this experiment, the dynamic range of the fused image is $[-0.38, 1.35]$, that is, almost $1.75\times$ larger than the input dynamic range. For comparison, we display the linearly-compressed result on the right (b). It is not saturated, yet contrast is reduced compared to the input images of the bracketed sequence.



(a) lines of the input sequence (b) 3rd input image

(c) lines of the fused result (d) fused result

Figure 10: We show here a section taken in the input sequence (represented on the images on the right column). All input images are in the correct dynamic range. The fused result however has a larger dynamic. This experiment is carried out with gray level images for the sake of clarity; we thus do not use the saturation metric: $\omega_s = 0$. The other parameters are $\omega_c = 1, \omega_e = 1$. We clipped out-of-range values in (d).

We are then stuck in the unpleasant situation where either we decide to compress the dynamic, but lose contrast (see for example Figure 9 (b)), or we apply again a tone-mapping operator, which is specifically what our method was initially designed for. As we explain below, this artifact is due to the multiscale blending.

Constructing an image that combines the most contrasted, saturated and well-exposed parts of each image of a given sequence supposes that the method is able to keep the small variations, the local contrast. Exposure fusion succeeds in selecting these best parts and to fuse them seamlessly. However, constraining the fused result to respect the initial dynamic range also requires the method to be able to reduce the edges' amplitude.

But exposure fusion is fully based on the computation of averages of Laplacian coefficients. Thus, the exposure fusion mechanism that might reduce the edges' amplitude is the blending of high amplitude Laplacian coefficients (from high amplitude edges) with lower amplitude Laplacian coefficients (from lower amplitude edges). This seldom happens because weights are designed to select the most contrasted regions. Thus, in the same way as exposure fusion preserves the local contrast of each input image, it preserves their edges.

In Figure 11 we experimentally show this effect. We designed an input sequence composed of two test-patterns. The first one has values equal to zero everywhere except in a small band in its center; this band is not saturated and has some local contrast (noise) so that exposure fusion will assign large weights to it. The second test pattern is well-exposed and contrasted for its most part, except in the same centered band where it is saturated to white. Thus, exposure fusion will fuse the center band of the input 1 with the side parts of input 2. These inputs are displayed in Figure 11(a), (b). We display the center line in the plot of the same figure. The fused image's (yellow line) edges height is the average of the two input heights. If the same or another image of the sequence has large edges in the reverse direction, then the fused image can overstep the input dynamic range: see Figure 12.

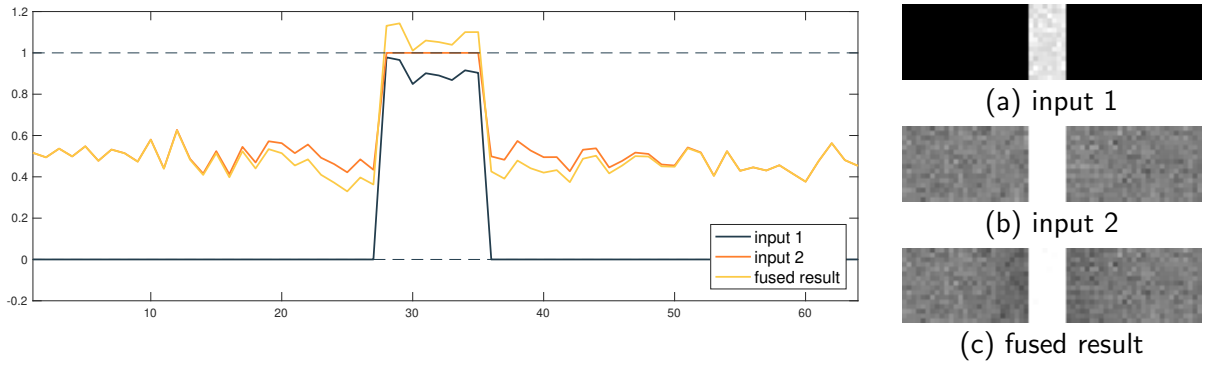


Figure 11: Fusion of input 1 and input 2 with exposure fusion. Parameters: $\omega_c = 1$, $\omega_s = 0$ (gray level images), $\omega_e = 1$. This simple experiment shows that exposure fusion cannot reduce edge amplitude at will. In fact, edge reduction is a consequence of the blending of large Laplacian coefficients (from input 1) with smaller Laplacian coefficients (from input 2). In this experiment, this is not enough to prevent a saturation of the fused result (c). Figure 12 displays a more complex case where three input images are fused.

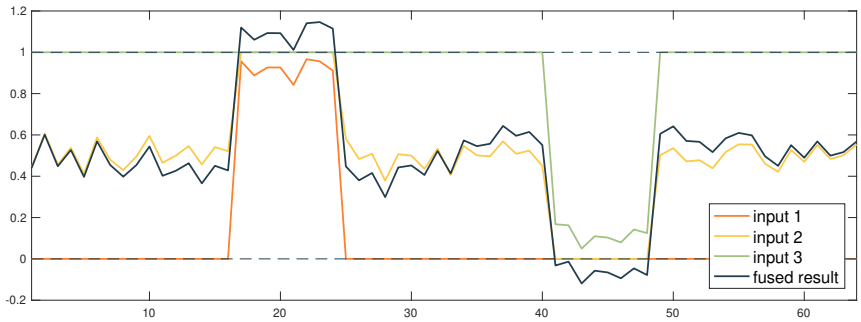


Figure 12: Edge preservation in exposure fusion and dynamic extension. In this experiment, the input sequence has three images and two contrasted bands: input 1 holds the "well-exposed" first band (saturated in the other images); input 3 holds the well-exposed second band (saturated in the other images); input 2 holds the well-exposed parts between the bands. By blending the well-exposed parts together, exposure fusion creates an image too contrasted to fit in the input dynamic range.

Halo Artifact in Exposure Fusion

The authors remark that in some cases their method can exhibit a low-frequency halo artifact. This can appear when there are large exposure differences in the bracketed sequence. Figure 13 shows this artifact. The output image (c) has a vertical low-frequency halo, visible in the sky and on the left side the wall. The result is unnatural because there is an inversion of the brightness with respect to the original scene.

This residual seam is due to the last scale of the pyramid, and can therefore be solved by using deeper pyramids. As said in Section 3, the number of scales $l_{\max}$ is computed so as to lead to a 1-pixel wide residual in the shortest dimension when its initial size is a power of two (Equation (10)). But this is not the deepest possible pyramid. First, the image's size is generally not a power of two, hence the shortest dimension is often still a few pixels wide. Furthermore, the actual limit should be on the largest dimension of the input image. Indeed, nothing prevents one from continuing downsampling when one dimension's width reaches 1 pixel: the Laplacian coefficients simply become zero in this dimension. However, deeper pyramids increase the out-of-range artifact.

In Figure 14 we present fusion results using different depths, including deeper and shallower pyramids than the reference one. For this sequence, the depth is $l_{\max} = 9$ (and the residual's shortest dimension 4 pixels wide). When adding 2 scales to the pyramid, the fused result does not have a visible halo any more. On the other hand, it produces more out-of-range artifacts, which entails a

loss of contrast due to the final normalization, even if 1% of clipping is tolerated on the black side and 1% on the white side. On another hand, using shallower pyramids leads to thinner halos, which are more visible. The halo's width is of the order of the residual's equivalent blur kernel's width.

(a) input sequence and associated weights (b) mid-sequence image (c) fused image

Figure 13: Halo in exposure fusion. The fused image (c) obtained with this sequence (in (a), top row) looks unnatural, due to an inversion of the brightness: the bottom of the fused image becomes brighter than the top (compare for example to b). Further, a vertical (low-frequency) halo is visible in the sky and on the wall. Parameters: $\omega_c = \omega_s = \omega_e = 1$, using robust normalization with 1% of clipping on both white and black sides.

mid-sequence image fused with 11 scales **fused with 9 scales** fused with 7 scales fused with 5 scales

Figure 14: Fusion results with more or fewer scales. For this sequence the algorithm normally uses 9 scales. With more scales the halo disappears but the result loses contrast due to the out-of-range artifact and final normalization. With fewer scales the halo becomes thinner and thus more visible. Parameters: $\omega_c = \omega_s = \omega_e = 1$, using robust normalization with 1% of clipping on both white and black sides.

9 Conclusion

While the principle of the method is appealing, we saw that it still has no clear mechanism to ensure that no information is lost by saturation. We tested clipping and robust normalization as a fix. But none of these solutions is quite convincing. Indeed clipping loses information in the bracketed sequence. Conversely normalization loses back the contrast we were about to gain. Hence, the job is not finished and, after fusion, we may sometimes be left with a tone mapping problem.

Acknowledgment

We thank Jean-Michel Morel for fruitful comments and discussions. Work partly financed by Office of Naval research grant N00014-17-1-2552, DGA Astrid project "filmer la Terre" n°ANR-17-ASTR-0013-01, and ANRT CIFRE Ph.D. scholarship n°2014/1323 of the French Ministry for Higher Studies, Research and Innovation.

Image Credits


Min H. Kim


Jacques Joffre


Charles Hessel

References

[1] S. Baker and I. Matthews, *Equivalence and efficiency of image alignment algorithms*, in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2001, pp. 1090–1097. https://doi.org/10.1109/CVPR.2001.990652.

[2] T. Briand, G. Facciolo, and J. Sánchez, *Improvements of the Inverse Compositional Algorithm for Parametric Motion Estimation*, Image Processing On Line. Preprint., (2018). https://www.ipol.im/pub/pre/222/.

[3] T. Briand and P. Monasse, *Theory and Practice of Image B-Spline Interpolation*, Image Processing On Line, 8 (2018), pp. 99–141. https://doi.org/10.5201/ipol.2018.221.

[4] P. Burt and E. Adelson, *The Laplacian pyramid as a compact image code*, IEEE Transactions on Communications, 31 (1983), pp. 532–540. https://doi.org/10.1109/TCOM.1983.1095851.

[5] P.J. Burt and R.J. Kolczynski, *Enhanced image capture through fusion*, in Proceedings of the IEEE International Conference on Computer Vision (ICCV), 1993, pp. 173–182.

[6] J. Delon, *Midway image equalization*, Journal of Mathematical Imaging and Vision, 21 (2004), pp. 119–134. https://doi.org/10.1023/B:JMIV.0000035178.72139.2d.

[7] T. Guillemot and J. Delon, *Implementation of the Midway Image Equalization*, Image Processing On Line, 6 (2016), pp. 114–129. https://doi.org/10.5201/ipol.2016.140.

[8] N. Limare, J.L. Lisani, J-M. Morel, A.B. Petro, and C. Sbert, *Simplest Color Balance*, Image Processing On Line, 1 (2011), pp. 297–315. https://doi.org/10.5201/ipol.2011.llmps-scb.

[9] J.M. Ogden, E.H. Adelson, J.R. Bergen, and P.J. Burt, *Pyramid-based computer graphics*, RCA Engineer, 30 (1985), pp. 4–15.

[10] J. Kautz T. Mertens and F. Van Reeth, *Exposure fusion*, in Proceedings of the Pacific Conference on Computer Graphics and Applications, 2007, pp. 382–390.

[11] ——, *Exposure fusion: A simple and practical alternative to high dynamic range photography*, Computer Graphics Forum, 28 (2009), pp. 161–171. https://doi.org/10.1111/j.1467-8659.2008.01171.x.