Hindawi Publishing Corporation
EURASIP Journal on Advances in Signal Processing
Volume 2007, Article ID 80971, 9 pages
doi:10.1155/2007/80971

Research Article

Video Enhancement and Dynamic Range Control of HDR Sequences for Automotive Applications

Stefano Marsi, Gaetano Impoco, Anna Ukovich, Sergio Carrato, and Giovanni Ramponi

Image Processing Laboratory (IPL), Department of Electrical and Electronics Engineering (DEEI), University of Trieste, Via A. Valerio 10, 34127 Trieste, Italy

Received 16 March 2006; Revised 12 March 2007; Accepted 13 May 2007

Recommended by Yap-Peng Tan

CMOS video cameras with high dynamic range (HDR) output are particularly suitable for driving assistance applications, where lighting conditions can strongly vary, going from direct sunlight to dark areas in tunnels. However, common visualization devices can only handle a low dynamic range, and thus a dynamic range reduction is needed. Many algorithms have been proposed in the literature to reduce the dynamic range of still pictures. Anyway, extending the available methods to video is not straightforward, due to the peculiar nature of video data. We propose an algorithm for both reducing the dynamic range of video sequences and enhancing its appearance, thus improving visual quality and reducing temporal artifacts. We also provide an optimized version of our algorithm for a viable hardware implementation on an FPGA. The feasibility of this implementation is demonstrated by means of a case study.

1. INTRODUCTION

The human visual system (HVS) can handle dynamic ranges that are several orders of magnitude larger than those of conventional acquisition and visualization devices. In order to fill the gap between the direct observation of a scene and its digital representation, high dynamic range (HDR) sensors have recently been devised, mainly based on CMOS sensors with logarithmic [1] or piecewise-linear response [2]. Moreover, some authors have recently tried to extend the dynamic range of current visualization devices [3–5]. However, the problem is far from being well investigated. As a consequence, the dynamic range of HDR images must be reduced to fit the one of the visualization device at hand.

Unfortunately, a simple mapping from the original signal range to the display range generally provides somehow poorly contrasted "flat" images, while an overstretching of the range inevitably leads to signal saturation. Hence, we need more sophisticated algorithms that can preserve local contrast, while reducing the dynamic range of the scene. If this is not trivial for still images, handling HDR video sequences is even more challenging, due to the temporal nature of the data. In particular, *real-time* video processing has applications in many different fields, such as video surveillance, traffic monitoring, and driving assistance systems. In all these applications the sensor operates in challenging outdoor environments. Lighting conditions can change significantly because of weather, presence of light sources in the scene, and so on. In this paper, we will focus on a driving assistance scenario.

Algorithms devised to reduce the dynamic range of still pictures cannot be simply extended to video. Moreover, in driving assistance applications, video processing is usually performed on low-cost hardware; these devices are often embedded in the camera box itself (e.g., smart cameras). In this paper, we propose an algorithm which reduces the dynamic range of video sequences to fit the one of the display [6]. In order to cover the applications mentioned above, we propose some simplifications and optimizations of our algorithm that make it suitable for the implementation on low-cost hardware. A case study is also presented.

2. RELATED WORK

The problem of reducing the dynamic range of still pictures has drawn the attention of many authors. Since the Retinex model [7] was proposed, a number of different methods have been devised [8–16]. All these methods are tailored to still pictures. Video sequences coming from driving assistance applications present further problems to be addressed.

Namely, abrupt changes in the global illumination of the scene may occur between frames, and the processing has to be accomplished in real time.

Actually, a straightforward extension of the previous approaches to video sequences is to reduce the dynamic range of the scene frame by frame: this is the case of Hines et al. [17], Monobe et al. [18], Artusi et al. [19]. In particular, the first one proposes a DSP implementation of the single scale Retinex algorithm which is suitable for real-time applications. However, there are some parameters to be tuned by hand in order to control the output visual appearance and it is likely that in the presence of large illumination variations in the video sequence, the same parameter values are not suitable for all the frames.

Hence, an automatic temporal adaptation should be better introduced. Pattanaik et al. in [20], as well as in other works related to visualization for computer graphics applications [21–24], a time-varying parameter is exploited, mimicking the HVS temporal adaptation to illumination changes. When we go from a bright place to a dark one, it takes a few minutes to adapt to the new luminosity. The same happens when going from a dark place to a bright one, though in this case the adaptation is faster and lasts few seconds. However, this is exactly the opposite of what we want to obtain. Indeed, when a car enters a tunnel, the processing should make the driver see very well from the first instant, rather than smoothly and slowly adapt to the new illumination.

Another solution is to filter the frames in the time domain, averaging the current frame with the previous ones if certain conditions occur. Wang et al. [25] recognize the importance of performing temporal filtering to avoid flash and flickering effects as well as color shifts. Bennett and McMillan [26] also use a temporal filtering to enhance dark videos, extending the bilateral filtering of Durand and Dorsey [9] in the temporal direction. In both approaches, motion detection precedes the temporal filtering, in order to avoid motion blurring. However, both algorithms require a high computational time and are not suitable for implementation on low-cost hardware for real-time applications.

Focusing on the application, as far as we know, there is few literature about video enhancement for driving assistance. Andrade et al. [27] develop an algorithm for night driving assistance, but they tackle the problem of reducing the effect of glares (e.g., caused by the lights of an incoming car) rather than the general enhancement of the original video.

We propose an algorithm for dynamic range reduction which accounts for global illumination changes and preserves the temporal consistency, similar to the works of Wang et al. and Bennett and McMillan. In addition to this, we propose a fast and low-cost implementation for real-time driving assistance applications. Like in the two approaches mentioned above, motion blurring should be prevented. However, motion estimation would require excessive computational time and resources. Thus, we model motion as a local illumination change, and we temporarily smooth out only the global illumination changes. Moreover, in the hardware implementation, with the aim of using less memory and of speeding up the computation, we use a subsampled version of the frame, following the idea of Artusi et al. [19] and Durand and Dorsey [9].

3. THE ALGORITHM

In this section, we introduce the algorithm we designed for the control of HDR video sequences. Moreover, we discuss the tuning of its parameters. We show that once the camera is chosen, the parameters can be set according to its characteristics and do not need to be tuned by the user during the processing.

3.1. Algorithm description

Similar to many other dynamic range reduction techniques, our algorithm is based on the Retinex theory [7]. The theory states that an input image $\mathbf{I}(x, y)$ can be considered as the result of the point-by-point product of the illumination $\mathbf{L}(x, y)$ of the scene and the reflectance $\mathbf{R}(x, y)$ of the objects:

$$\mathbf{I}(x, y) = \mathbf{L}(x, y)\mathbf{R}(x, y). \tag{1}$$

In Retinex-like approaches, the $\mathbf{L}(x, y)$ and $\mathbf{R}(x, y)$ components are estimated from the available $\mathbf{I}(x, y)$ data. Then, they are suitably modified (in the HDR case, the dynamic range of the illumination component is usually reduced, while the reflectance is enhanced). Finally, the components are reassembled to yield the output image $\mathbf{I}'(x, y)$. Equation (1) is intended for a linear input. However, some sensors (mainly with CMOS technology) have a logarithmic output [28], that is, the output signal is proportional to the logarithm of the incident light. Hence, they can provide an extremely high dynamic range. Since in our application these sensors are used, the hypotheses of (1) are replaced by

$$\log \mathbf{I}(\mathbf{x}, \mathbf{y}) = \log \mathbf{L}(\mathbf{x}, \mathbf{y}) + \log \mathbf{R}(\mathbf{x}, \mathbf{y}). \tag{2}$$

When dealing with video sequences, several further issues arise. In particular, we have to take into account large variations in the global illumination of the scene between consecutive frames. In order to obtain a more uniform appearance of the sequence, we extract the global illumination from the scene and smooth out its abrupt temporal variations.

A block scheme of our complete algorithm is shown in Figure 1. We estimate the illumination component $L(x, y) = \widehat{\log \mathbf{L}(\mathbf{x}, \mathbf{y})}$ using an edge-preserving lowpass filter. The reflectance $R(x, y)$ is obtained by difference between the input $I(x, y) = \log \mathbf{I}(\mathbf{x}, \mathbf{y})$ and the illumination $R(x, y) = I(x, y) - L(x, y)$. The illumination component for the tth frame $L(x, y)$ is separated into local $(L_L(x, y))$ and global $(L_G(x, y))$ illuminations. $L_L(x, y)$ is intended to contain the local illumination variations in the scene (due to objects in motion, e.g., the lights of a car traveling in the opposite direction). $L_G(x, y)$ should represent the global sensation of illumination, that is, the "measure" that human beings use to judge if a picture is lighter or darker than another one.

In more detail, with this objective in mind we first compute $L(x, y)$ as Marsi et al. in [11] using a recursive rational filter:

$$\begin{aligned} L(x,y) = {} & \frac{1}{S_v(x,y) + S_h(x,y) + 1} \\ & \cdot \{\kappa[L(x,y-1)S_v + L(x-1,y)S_h] \\ & + [(S_v(x,y) + S_h(x,y))(1-\kappa) + 1] \\ & \cdot I(x,y)\}, \end{aligned} \tag{3}$$

where $I(x, y)$ is the value of the pixel in position (x, y) in the input image I, κ is the recursion coefficient. S_h and S_v are the edge sensors in the horizontal and vertical directions, respectively,

$$\begin{aligned} S_h(x,y) &= T_s \Big/ \left[\left(\log \frac{\delta_1 + I(x-1,y)}{\delta_1 + I(x+1,y)} \right)^2 + \delta_2 \right], \\ S_v(x,y) &= T_s \Big/ \left[\left(\log \frac{\delta_1 + I(x,y-1)}{\delta_1 + I(x,y+1)} \right)^2 + \delta_2 \right], \end{aligned} \tag{4}$$

where δ_1, δ_2 are two small constants that prevent illegal operations, and T_s is a coefficient used to trigger the sensor response. The *edge-preserving* feature of the filter is important to avoid halo artifacts, as already noticed [9, 10].

Then, we extract the global illumination $L_G(x, y)$ applying a linear narrowband lowpass filter to $L(x, y)$. The local illumination is computed as $L_L(x, y) = L(x, y) - L_G(x, y)$. We use only the global illumination channel $L_G(x, y)$ for temporal filtering.

The amount of temporal smoothing is controlled by a parameter α, in the range $[0, \alpha_{\max}]$. It determines the influence of the previous frames on the current one:

$$\widetilde{L}_G(t) = (1 - \alpha(t)) \cdot L_G(t) + \alpha(t) \cdot L_G(t-1), \tag{5}$$

where $\alpha(t)$ denotes α at the tth frame. Here, $L_G(t)$ and $L_G(t-1)$ are the global illuminations of the current and of the previous frames, respectively. Although they are functions of (x, y) and not only of (t), we omit it in the notation of (5) for the sake of simplicity. At the beginning of the sequence, we set $\alpha(0) = 0$. When a sharp variation occurs, $\alpha(t)$ is set to a high value. Conversely, if there is small variation between $L_G(t)$ and $L_G(t-1)$, $\alpha(t)$ becomes smaller and smaller. We use the following formula:

$$\alpha(t) = \begin{cases} \alpha_{\max} & \text{if } (\mu(t) - \mu(t-1))^2 > \tau, \\ \dfrac{\alpha(t-1)}{\rho} & \text{otherwise,} \end{cases} \tag{6}$$

where $\mu(t)$ and $\mu(t-1)$ are the mean gray values of the current and previous frames, respectively. The effects of the difference between neighboring frames can be tuned by means of the threshold τ. The parameter $\rho > 1$ is related to the speed of adaptation to the current illumination.

The corrected global illumination, $\widetilde{L}_G$ in (5), is added back to the local illumination L_L. The resulting illumination

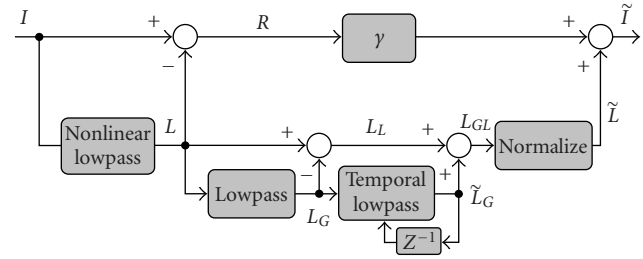

Figure 1: Block diagram of the proposed algorithm.

channel L_{GL} (the sum of the global and local illuminations, as shown in Figure 1) is remapped by:

$$\widetilde{L}(x,y) = \frac{L_{GL}(x,y) - \mu}{\sigma} \cdot r + m, \tag{7}$$

where μ and σ are the mean and standard deviation of the pixel distribution in L_{GL}. The parameters r and m are determined experimentally to fit the display range and mean values. Finally, the corrected illumination $\widetilde{L}(x, y)$ and reflectance channels are recombined as

$$\widetilde{I}(x,y) = \widetilde{L}(x,y) + \gamma R(x,y), \tag{8}$$

where γ is a constant which provides an enhancement of the details.

3.2. Color processing

The algorithms till now proposed in this paper are used to process gray-level images; however it is possible to extend them to color images too. It is well known that a color image needs at least 3 values to univocally define every pixel, and several color spaces have been proposed in the literature (RGB, YCbCr, HSV, YUV, Lab, etc.) for this purpose. Each of these color spaces presents different characteristics and peculiarities and it is far from being trivial to select the most suitable one for our purposes. Actually, it is not even obvious which kind of processing is required in the case of color images. Without claiming to have an answer to this point, we address some issues and hypotheses, trying to suggest some solutions.

The main difficulty is that like with many other enhancement algorithms, the final goal cannot be formally defined. Actually, even in monochromatic images, the final target is not objectively defined; in such a case, however, usually it has been assumed that the aim is to improve the subjective quality, that is, the ability to distinguish the image details without altering any other possible information. Extending this approach to color images, we can assume that a constraint is to avoid any alteration in the color domain; for instance, in the case of the RGB space, the constancy of the proportion between the three channels should be guaranteed. Moreover, it is mandatory that none of the processed signals exceeds its regular range, to avoid generating a saturation, and consequently a partial information leakage. Furthermore, a less

important constraint could be to limit the computational effort avoiding to replicate the same processing on each color component, rather processing just a single channel.

Within the mentioned constraints, the solution we propose as an extension of the previous algorithms to color images is quite simple, but effective. Assuming to work in the well-known RGB space, we first define a monochromatic channel V_i:

$$V_i = \max(R_i, G_i, B_i), \tag{9}$$

where R_i, G_i, B_i are the three input RGB components, respectively. The algorithms proposed in the previous section are applied to V_i, obtaining as a result the output V_o. To convert this information back into the RGB space, we apply the following equations:

$$R_o = V_o \frac{R_i}{V_i}, \qquad G_o = V_o \frac{G_i}{V_i}, \qquad B_o = V_o \frac{B_i}{V_i}, \tag{10}$$

where R_o, G_o, B_o are, respectively, the three output values in the RGB color space.

In such a way, all the three assumptions adopted before are guaranteed. However, there could be some drawbacks. For example, if the input image is not well white-balanced, as it often happens especially in dark images, the proposed solution emphasizes the dominant color with a consequently loss of pleasantness in the final image. A solution to such a problem has been addressed by Fattal et al. [10]: in their paper, they propose a similar approach, but the equations which map the output signal to the RGB space are a generalization of (10), that is,

$$R_o = V_o\left(\frac{R_i}{V_i}\right)^s, \qquad G_o = V_o\left(\frac{G_i}{V_i}\right)^s, \qquad B_o = V_o\left(\frac{B_i}{V_i}\right)^s, \tag{11}$$

where s is a suitable value in the range $[0, 1]$ (they propose 0.5). This solution is useful to desaturate the dominant color, but may alter the original hues; indeed, when V_i coincides with V_o, the RGB output components differ from the input ones. A most straightforward solution, useful to compensate a badly white-balanced image, is to apply the algorithm separately to each RGB channel. In such a case the output image appears quite natural and pleasant, but in fact the original color has been modified, and consequently a part of the original information is altered.

A different solution, very simple to implement, comes when the input video is coded through its luminance and crominance components. In such a case, the emphasis could be applied only to the luminance signal while the crominance could be maintained inhaltered. Even if this solution is very straightforward, it presents many negative aspects: the hue of the original colors is altered, and in particular under certain conditions, a saturation of the primary RGB component can occur. Moreover, in the case the original image is dark and the crominance signal is weak, the processed image will also be poor in chrominance and will appear grayish and unpleasant.

3.3. Algorithm parameters

With reference to the different blocks in Figure 1, the parameters involved in the algorithm are the following.

(i) κ, T_s are coefficients for the illumination estimation [11] in blocks "nonlinear lowpass;" the first parameter, which has values in $[0, 1]$, defines the amount of recursion of the filter: the lowpass effect is strong when κ is close to 1. The second one is a threshold for the edge sensors, which is responsible for the edge-preserving feature of the filter. Our experiments showed that κ and T_s depend only on the resolution of the acquired video: a small frame size will usually present sharp edges, and on the other hand will not require a strong lowpass effect, thus the smaller the frame size is the higher the T_s constant and the farther from 1 is κ.

(ii) α_{max} is parameter for the temporal filter in block "temporal lowpass;" this is usually set to a value close to 1, in order to have a strong influence of the previous frames in case a change in the global illumination is detected.

(iii) τ is threshold for α in block "temporal lowpass": it determines the amount of change in illumination which activates the temporal filtering.

(iv) ρ is in block "temporal lowpass;" this parameter determines how fast the temporal filtering ends its effect after a global illumination change is detected; it can be set by taking into account the camera frame rate.

(v) r, m are in block "normalize;" they are related to the display; in practice, good results are obtained with most displays by setting m to the mean luminance value and r to half the range of the display.

(vi) γ is multiplicative coefficient for detail enhancement in block "gamma;" it has integer values in $[1, 10]$, it can be set according to the camera characteristics; it is set to 1 in case of strongly noisy camera, in order to avoid enhancing the noise contained in the reflectance component; otherwise it can be set to higher values.

4. HARDWARE IMPLEMENTATION

We chose to tailor our implementation to FPGAs since they allow a flexibility close to that of DSPs, while guaranteeing adequate performances compared to ASICs. In this section, we present a number of simplifications to our algorithm that make it more suitable for the implementation on an FPGA.

As a first simplification, the background illumination is decimated before temporal filtering. We observed indeed that full resolution is not needed, since the background illumination contains only the lowest frequencies of the input frame. By working at low resolution, we can store the background illumination channel in its downsampled version. This turns out to be a significant memory saving. After temporal filtering, the signal is interpolated.

The algorithm for estimating the illumination of the scene that was presented in Section 3 is rather burdensome to be implemented on the FPGA. In real-time video applications, where the quality of the image sequence is usually low,

the horizontal and vertical edge sensors can be replaced by two binary operators with the following expressions:

$$S_h(x,y) \longrightarrow \begin{cases} \infty & \text{if } |I(x-1,y) - I(x+1,y)| < \varepsilon, \\ 0 & \text{otherwise,} \end{cases}$$
$$S_v(x,y) \longrightarrow \begin{cases} \infty & \text{if } |I(x,y-1) - I(x,y+1)| < \varepsilon, \\ 0 & \text{otherwise,} \end{cases} \tag{12}$$

where ε is a threshold parameter, to be set according to the environment where the CMOS sensor camera will operate. According to the resulting binary values of S_h and S_v, we perform different operations to estimate the illumination:

(i) vertical smoothing (H_v) if

$$S_v(x,y) \longrightarrow \infty \wedge S_h(x,y) \longrightarrow 0; \tag{13}$$

(ii) horizontal smoothing (H_h) if

$$S_h(x,y) \longrightarrow \infty \wedge S_v(x,y) \longrightarrow 0; \tag{14}$$

(iii) plus-shaped smoothing (H_p) if

$$S_h(x,y) \longrightarrow \infty \wedge S_v(x,y) \longrightarrow \infty; \tag{15}$$

(iv) no operation otherwise.

The illumination for the tth frame is estimated as

$$L(x,y) = I(x,y) \otimes H(x,y), \tag{16}$$

where $H(x,y)$ is chosen among H_h, H_v, and H_p according to the values of $S_v(x,y)$ and $S_h(x,y)$, and $\otimes$ denotes convolution. The mask sizes of the filters H_h, H_v, and H_p are $1 \times N_1$, $N_1 \times 1$, and $N_1 \times N_1$, respectively.

The global illumination L_G is estimated from the illumination component L using a lowpass filter with mask size N.

Prior to the temporal smoothing, the new frame is decimated in the horizontal and vertical directions by a factor s. Hence, the resized frame is $s \times s$ times smaller than the full-resolution frame. The downsampling factor s is selected with respect to the frame size of the camera. After temporal smoothing, the output frame is interpolated back using two linear interpolators, one for each direction. The mask size of the two linear interpolators is s.

5. SIMULATION RESULTS

We tested both our full algorithm and the simulation of its hardware implementation. Sequence 1 was acquired by means of a sensor belonging to the *Pupilla* family [1], while Sequence 2 by means of an Ethercam [29] with a different sensor [2]. The cameras are mounted on the rear mirror of a car. The frame sizes are 125×86 for Sequence 1 and 160×120 for Sequence 2, and the frame rate is 24 frame/s for both sequences. The input dynamic range of the cameras is 10 bits/pixel. The output dynamic range we want to obtain is 8 bits/pixel.

The sequences present some critical scenes, such as backlights and direct sunlight on the camera lens. Moreover, the sunlight is periodically obscured by trees on one side of the road. This leads to annoying flashing effects due to sudden illumination variations. Both the mean luminance and the mean contrast change abruptly.

Figure 2 shows three consecutive frames of one sequence, where the flashing effect is visible: notice the abrupt illumination change in the second frame where a tree blocks direct sunlight on the sensor. The columns show, respectively, linear remapping, the result of the frame-by-frame multiscale Retinex [30], and the result of our algorithm. Clearly, the input sequence has low contrast and presents flashes. Our algorithm remarkably reduces this effect. Illumination variations are smoother but the local contrast is still well exploited. The multiscale Retinex produces good results in the central frame, but too bright images in the first and last frames. This is due to the frame-by-frame processing, according to which the same algorithm parameters need to be used for all the frames, as noticed in Section 2.

Experiments have also been carried out for the simulation of the hardware implementation described in Section 4. Figure 3 shows a comparison between the histogram equalization and the hardware implementation of our algorithm. The quality of the latter is still better than the quality of the histogram equalization. In the hardware implementation, in order to limit the circuit complexity, we have been forced to use filters with a smaller impulse response and a larger bandpass with respect to the software version. The consequence is that the improved details in the hardware version are more concentrated in the high-frequency region. Actually, the visual quality of the processed scene is the most important result, especially in the time domain. Since this aspect cannot of course be appreciated in this paper, the sequences are available for download at the address http://www.units.it/videolab.

Figure 4 shows the results for color sequences. As noticed in Section 3.2, the processing on the RGB color space provides better results than the processing in the YCbCr space, due to the fact that the considered frame is dark and the chrominance values are low.

Table 1 shows the parameters we used in the experiments. The same parameter values are used for both Sequence 1 and Sequence 2; this fact proves that the proposed method is robust.

5.1. Hardware resources estimation

As a case study, we evaluate the feasibility of the implementation of the proposed algorithm on a commercial FPGA, the characteristics of which are reported in Table 2. Some implementation choices are strictly related to the specific FPGA employed. In case another model is used, the implementation can be further improved to fit the features of the used FPGA.

In the following, the resources needed for the implementation are discussed in some more detail.

Table 1: Parameters for the case study implementation, assuming that the input image range is between 0 and 1.

κ	0.7
T_s	$2 \cdot 10^{-4}$
ε	0.12
s	4
N	5
N_1	3
r	0.21
m	0.5
$\alpha_{\max}$	0.95
τ	0.4
ρ	1.1
γ	6

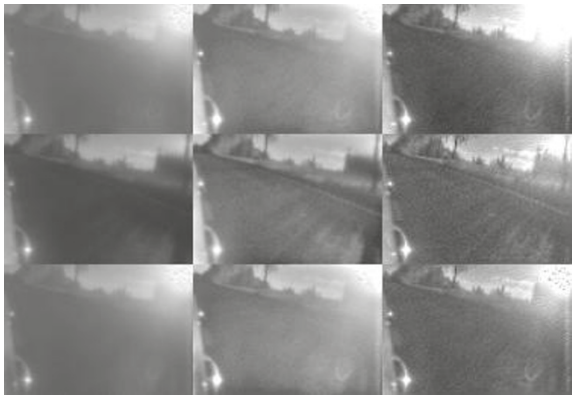

Figure 2: Three consecutive frames from Sequence 1: the high dynamic range camera is mounted on the rear mirror of a car. A part of the car can be recognized on the left. In the center, there is the road. On the upper right part, the trees at the road border can be recognized, with the sunlight passing through the trees. The scene presents difficulties related to the low quality, low resolution, and residual fixed pattern noise of the sensor (after the on-chip calibration). The left column shows a linear remapping of the input. The central column shows the results of the multiscale retinex [30] (obtained using the software PhotoFlair in the "High Contrast Mode"). The right column shows the result of the proposed algorithm. Note that the dark flashes (sequences of dark, bright, dark frames) present in the original sequence are removed by our algorithm.

The edge-preserving smoothing for the estimation of the illumination L is performed by a pair of 3-tap filters, one for each direction. A filter is activated when the corresponding edge sensor is inactive (i.e., no edge is detected). This block is implemented using 2 *MAC* 3 *tap FIR* filters. Two frame lines are stored in the block RAM (BRAM) memory.

The lowpass filter for L_G calculation is a 5×5 filter, implemented by means of 5 *MAC* 5-*tap FIR* filters. These filtering structures can perform 5 operations (sums and multiplications) per clock cycle. This is obtained thanks to an FPGA DCM that increases the filter inner clock frequency with respect to the external frequency. *MAC n-tap FIR* can be realized either using block RAMs (BRAMs) or the distributed RAM in the CLBs; we choose the latter solution. The 5 FIR filters require to store four frame lines into the BRAMs.

The previous frame must be stored for temporal filtering. It is downsampled to 1/4 in both directions (1/16 memory) and stored in a BRAM memory. We do not account for the downsampling block since its requirements are negligible. The interpolation block performs a weighted sum of four input pixels in the downsampled frame. The weights depend on the position of the pixel in the upsampled frame. This block is thus implemented by means of six multiplier blocks and some additional LUTs.

Figure 3: Single frame from Sequence 2: input frame (upper left), histogram equalization (upper right), our algorithm (bottom left), simulation of the hardware implementation of our algorithm (bottom right). Notice that our algorithm yields a better visual quality than a simple global operator as histogram equalization. In particular, the details are better rendered, and this is true even with the simplified version for the hardware implementation.

Figure 4: Single frame from Sequence 2, color results: input frame (upper left), histogram equalization (upper right), our algorithm in RGB (bottom left), our algorithm in YCbCr (bottom right).

Table 2: Features of the commercial FPGA used.

System gates	1 M
Logic cells	17 280
CLBs	1 920
Distributed RAM (bits)	120 K
Block RAM (bits)	432 K
Dedicated multipliers	24
DCMs	4
Maximum user I/O	391

Table 3: Total resources needed by the proposed algorithm and resources available on the selected hardware.

Resources	Slice	FF	BRAM	Multipliers
Total	890	1362	20	16
Available	7680	7680	24	24

The normalization block requires the evaluation of some global frame parameters, such as the mean and luminance values of the illumination channel. This requires the current (full-sized) frame to be stored in the BRAM memory. The emphasizing of the details is a simple multiplication by a constant. It is implemented using a single multiplier. Finally, adder blocks are required to recombine signals.

Table 3 shows a comparison between the overall required resources and the available resources on the FPGA. We employ less than 50% of the available flip flops and slices, and less than 85% BRAMs and multipliers.

Our input stream has a frame rate of 24 frame/s, and the frame is 125×86 pixels wide. The resulting pixel rate is then 258 K pixels/s. Taking into account that some filters, such as the *MAC FIRs* 5-*taps*, require 5 clock ticks to process a pixel, the minimum required clock frequency is $1,3$ MHz. The entire system has been developed using a pipeline architecture with a clock frequency synchronized on the input pixel rate. The bottleneck of the system is in the FIR filters implemented using the MAC 5-tap structure. The maximum clock frequency of these filters has been tested to be approximately 213 MHz in a Xilinx xc2v250-6 part [31].

In the case the frame size of the sequence to be acquired and processed in real time is larger, the most critical resource to take into account for the implementation of the algorithm is the BRAM memory, which is mostly needed by the temporal filtering and the normalization blocks. A different FPGA should thus be selected, either belonging to the same low-end family, or to a high-end family. A QCIF format sequence can be processed in real time using a higher-performance FPGA model belonging to the same commercial low-cost low-end FPGA family as the one considered here. If the most powerful FPGA belonging to the same low-cost low-end family is used, a sequence with frame size up to a quarter PAL can be processed in real time by our algorithm.

6. CONCLUSIONS

We have presented an algorithm to reduce the dynamic range of HDR video sequences while preserving local contrast. The global illumination of the previous frames is taken into account. Experimental data show that our algorithm behaves well even in extreme lighting variations.

A possible hardware implementation has also been proposed. We studied the feasibility of an implementation on a low-cost FPGA architecture. The implementation on an FPGA allows to perform the compression of dynamic range on an integrated system that is embedded in the video camera box and has low power consumption. Our study shows that the resources needed by our system do not exceed the capabilities of the hardware.

ACKNOWLEDGMENTS

This work was partially supported by a grant of the Regione Friuli-Venezia Giulia. Authors would like to thank Bruno Crespi, NeuriCam S.p.A., for providing the sequences for the experiments and for his useful suggestions and discussions.

REFERENCES

[1] NeuriCam s.p.a., "NC1802 Pupilla, 640×480 CMOS high-dynamic range optical sensor," 2002.

[2] Kodak, "Kodak KAC-9619 CMOS image sensor".

[3] H. Ohtsuki, K. Nakanishi, A. Mori, S. Sakai, S. Yachi, and W. Timmers, "18.1-inch XGA TFT-LCD with wide color reproduction using high power LED-backlighting," in *Proceedings of Society for Information Display International Symposium*, pp. 1154–1157, San Jose, Calif, USA, 2002.

[4] H. Seetzen, W. Heidrich, W. Stuerzlinger, et al., "High dynamic range display systems," *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 760–768, 2004.

[5] H. Seetzen, L. Whitehead, and G. Ward, "A high dynamic range display using low and high resolution modulators," in *Proceedings of Society for Information Display International Symposium*, pp. 1450–1453, San Jose, Calif, USA, May 2003.

[6] G. Impoco, S. Marsi, and G. Ramponi, "Adaptive reduction of the dynamics of HDR video sequences," in *Proceedings of IEEE International Conference on Image Processing (ICIP '05)*, vol. 1, pp. 945–948, Genoa, Italy, September 2005.

[7] E. H. Land and J. J. McCann, "Lightness and retinex theory," *Journal of the Optical Society of America*, vol. 61, no. 1, pp. 1–11, 1971.

[8] M. Ashikhmin, "A tone mapping algorithm for high contrast images," in *Proceedings of the 13th Eurographics Workshop on Rendering (EGRW '02)*, pp. 145–156, Pisa, Italy, June 2002.

[9] F. Durand and J. Dorsey, "Fast bilateral filtering for the display of high-dynamic-range images," in *Proceedings of the 29th International Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH '02)*, pp. 257–266, San Antonio, Tex, USA, July 2002.

[10] R. Fattal, D. Lischinski, and M. Werman, "Gradient domain high dynamic range compression," *ACM Transactions on Graphics*, vol. 21, no. 3, pp. 249–256, 2002.

[11] S. Marsi, G. Ramponi, and S. Carrato, "Image contrast enhancement using a recursive rational filter," in *Proceedings of*

IEEE International Workshop on Imaging Systems and Techniques (IST '04), pp. 29–34, Stresa, Italy, May 2004.

[12] C. Pal, R. Szeliski, M. Uyttendaele, and N. Jojic, "Probability models for high dynamic range imaging," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '04)*, vol. 2, pp. 173–180, Washington, DC, USA, June-July 2004.

[13] S. N. Pattanaik and H. Yee, "Adaptive gain control for high dynamic range image display," in *Proceedings of the 18th Spring Conference on Computer Graphics (SCCG '02)*, pp. 83–87, Budmerice, Slovakia, April 2002.

[14] E. Reinhard, M. Stark, P. Shirley, and J. Ferwerda, "Photographic tone reproduction for digital images," in *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '02)*, pp. 267–276, San Antonio, Tex, USA, July 2002.

[15] A. Rizzi, C. Gatta, and D. Marini, "From Retinex to Automatic Color Equalization: issues in developing a new algorithm for unsupervised color equalization," *Journal of Electronic Imaging*, vol. 13, no. 1, pp. 75–84, 2004.

[16] J. Tumblin and G. Turk, "LCIS: a boundary hierarchy for detail-preserving contrast reduction," in *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '99)*, pp. 83–90, Los Angeles, Calif, USA, August 1999.

[17] G. Hines, Z.-U. Rahman, D. Jobson, and G. Woodell, "DSP implementation of the retinex image enhancement algorithm," in *Visual Information Processing XIII*, vol. 5438 of *Proceedings of SPIE*, pp. 13–24, Orlando, Fla, USA, April 2004.

[18] Y. Monobe, H. Yamashita, T. Kurosawa, and H. Kotera, "Dynamic range compression preserving local image contrast for digital video camera," *IEEE Transactions on Consumer Electronics*, vol. 51, no. 1, pp. 1–10, 2005.

[19] A. Artusi, J. Bittner, M. Wimmer, and A. Wilkie, "Delivering interactivity to complex tone mapping operators," in *Proceedings of the 14th Eurographics Workshop on Rendering (EGRW '03)*, pp. 38–44, Leuven, Belgium, June 2003.

[20] S. N. Pattanaik, J. Tumblin, H. Yee, and D. P. Greenberg, "Time-dependent visual adaptation for fast realistic image display," in *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '00)*, pp. 47–54, New Orleans, La, USA, July 2000.

[21] S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "High dynamic range video," *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 319–325, 2003.

[22] G. Krawczyk, K. Myszkowski, and H.-P. Seidel, "Perceptual effects in real-time tone mapping," in *Proceedings of the 21st Spring Conference on Computer Graphics (SCCG '05)*, pp. 195–202, Budmerice, Slovakia, May 2005.

[23] P. Ledda, L. P. Santos, and A. Chalmers, "A local model of eye adaptation for high dynamic range images," in *Proceedings of the 3rd International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa (AFRIGRAPH '04)*, pp. 151–160, Stellenbosch, South Africa, November 2004.

[24] S. D. Ramsey, J. T. Johnson III, and C. Hansen, "Adaptive temporal tone mapping," in *Proceedings of the 7th IASTED International Conference on Computer Graphics and Imaging*, pp. 124–128, Kauai, Hawaii, USA, August 2004.

[25] H. Wang, R. Raskar, and N. Ahuja, "High dynamic range video using split aperture camera," in *Proceedings of the 6th IEEE Workshop on Omnidirectional Vision (OM-NIVIS '05)*, pp. 83–90, Beijing, China, October 2005.

[26] E. P. Bennett and L. McMillan, "Video enhancement using per-pixel virtual exposures," in *Proceedings of the 32nd International Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH '05)*, pp. 845–852, Los Angeles, Calif, USA, July-August 2005.

[27] L. C. G. Andrade, M. F. M. Campos, and R. L. Carceroni, "A video-based support system for nighttime navigation in semi-structured environments," in *Proceedings of the 17th Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI '04)*, pp. 178–185, Curitiba, PR, Brazil, October 2004.

[28] S. Kavadias, B. Dierickx, D. Scheffer, A. Alaerts, D. Uwaerts, and J. Bogaerts, "A logarithmic response CMOS image sensor with on-chip calibration," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 8, pp. 1146–1152, 2000.

[29] NeuriCam s.p.a. Ethercam NC51XX series.

[30] D. Jobson, Z.-U. Rahman, and G. Woodell, "A multiscale retinex for bridging the gap between color images and the human observation of scenes," *IEEE Transactions on Image Processing*, vol. 6, no. 7, pp. 965–976, 1997.

[31] http://www.xilinx.com/ise/optional_prod/system_generator.htm.

Stefano Marsi was born in Trieste, Italy, in 1963. He received the Dr. Eng. degree in electronic engineering (summa cum laude) in 1990 and the Ph.D. degree in 1994. Since 1995, he has held the position of Researcher in the Department of Electronics at the University of Trieste where he is the Teacher of some courses in electronic field. His research interests include nonlinear operators for image and video processing and their realization through application-specific electronics circuits. He is the author or coauthor of more than 40 papers in international journals, proceedings of international conferences, or contributions in books. He participated in several international projects and he is the Europractice Representative for the University of Trieste.

Gaetano Impoco graduated (summa cum laude) in computer science at the University of Catania, in 2001. He received his Ph.D. degree from the University of Pisa in 2005. During his Ph.D., he was a Member of the VCG Lab at ISTI-CNR, Pisa. In 2005, he worked as a Contract Researcher at the University of Trieste. He is currently a Contract Researcher at the University of Catania. His research interests include medical image analysis, image compression, tone mapping, color imaging, sensor planning, and applications of computer graphics and image processing techniques to cultural heritage, surgery, and food technology. He is reviewer of several international journals.

Anna Ukovich obtained her M.S. degree in electronic engineering (summa cum laude) from the University of Trieste, Italy, in 2003, and her Ph.D. degree from the same university in 2007. She has worked for one year at the Department of Security Technologies, Fraunhofer IPK, Berlin, Germany. She is currently a Contract Researcher at the University of Trieste, Italy. Her research interests include image and video processing for security applications.

Sergio Carrato graduated in electronic engineering at the University of Trieste. He then worked at Ansaldo Componenti and at Sincrotrone Trieste in the field of electronic instrumentation for applied physics, and received the Ph.D. degree in signal processing from the University of Trieste; later he joined the Department of Electrical and Electronics Engineering at the University of Trieste, where he is currently Associate Professor of electronic devices. His research interests include electronics and signal processing, and in more detail, image and video processing, multimedia applications, and the development of advanced instrumentation for experimental physics laboratories.

Giovanni Ramponi was born in Trieste, Italy, in 1956. He received the M.S. degree in electronic engineering (summa cum laude) in 1981; since 2000 he is Professor of electronics at the Department of Electrical and Electronics Engineering of the University of Trieste, Italy. His research interests include nonlinear digital signal processing, and in particular the enhancement and feature extraction in images and image sequences. Professor Ramponi has been an Associate Editor of the IEEE Signal Processing Letters and of the IEEE Transactions on Image Processing; presently he is an AE of the SPIE Journal of Electronic Imaging. He has participated in various EU and national research projects. He is the coinventor of various pending international patents and has published more than 140 papers in international journals and conference proceedings, and book chapters. Professor Ramponi contributes to several undergraduate and graduate courses on digital signal processing.