Journal of Real-Time Image Processing
https://doi.org/10.1007/s11554-018-0810-z

ORIGINAL RESEARCH PAPER

Color HDR video processing architecture for smart camera

How to capture the HDR video in real-time

Svetozar Nosko[1] · Martin Musil[1] · Pavel Zemcik[1] · Roman Juranek[1]

Received: 6 March 2018 / Accepted: 21 July 2018

Abstract
This paper presents a novel FPGA architecture of high dynamic range (HDR) video processing pipeline, based on the capturing of a sequence of differently exposed images. An acquisition process enabling multi-exposure HDR as well as fast implementation of local tone mapping operator involving bilateral filtering is proposed. The HDR acquisition process is enhanced by the application of novel deghosting method, which is dedicated for hardware implementation and proposed in this paper. The hardware processing pipeline is designed with regards to efficiency and performance and the calculations are performed in fixed point arithmetic. The pipeline is suitable for programmable hardware (FPGA—Field Programmable Gate Arrays) implementation and it achieves real-time performance on full HD HDR video which overcomes state-of-the-art solutions that use local tone mapping and deghosting algorithm.

Keywords HDR · HDR camera · Tone mapping · Real-time HDR processing · HDR deghosting

1 Introduction

Standard video cameras are unable to capture the dynamic range of visual information the human eye is capable of. The dynamic range is the variation of luminance within a given scene and the obvious goal of image and video acquisition is to capture the whole luminance range of the scene into the captured image. The current sensors are very limited and capable of capturing variations within two or three orders of luminance magnitude, while some scenes contains variations over the five orders. The video cameras are able to select which part of dynamic range is captured and which is lost as under/over exposed, e.g., by selection of the aperture and shutter speed. An example of such a scene capture is shown in Fig. 1.

Acquisition of high dynamic range (HDR) images is a very popular topic in professional photography and film industry. HDR is also increasingly getting established in industrial applications and many other areas, especially those where the automatic processing of image input is required and where good light conditions (for standard image acquisition) cannot be expected.

Two main approaches to HDR image capture exist. The first approach assumes production of specialized HDR sensors, e.g., [35] but this approach requires specialized sensors; however, they are generally expensive and technologically demanding. The second approach is based on standard sensors and on acquisition of image sequence with different exposure times [4, 22, 24, 29].

This paper presents a novel FPGA-applicable architecture of color HDR video pipeline. The HDR video is obtained through the acquisition of a sequence of differently exposed frames and their merging. The resulting HDR image is post-processed using local tone mapping operator based on bilateral filter.

The main contribution of this paper is the complete design of HDR pipeline suitable for implementation in FPGA, which achieves theoretical performance of

✉ Svetozar Nosko
inosko@fit.vutbr.cz

Martin Musil
imusil@fit.vutbr.cz

Pavel Zemcik
zemcik@fit.vutbr.cz

Roman Juranek
ijuranek@fit.vutbr.cz

[1] Brno University of Technology, FIT, Brno, Czech Republic

Published online: 30 July 2018

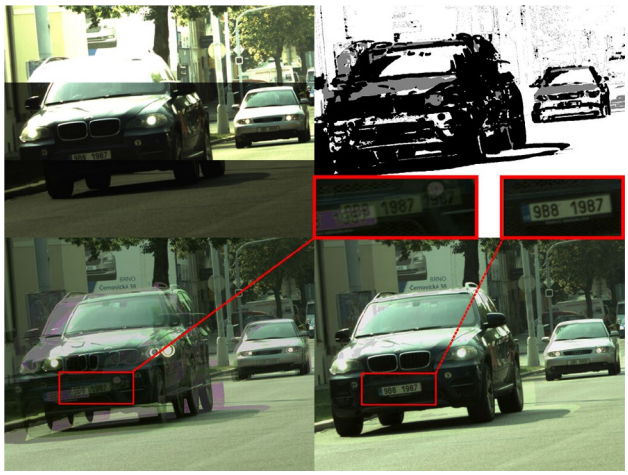

Fig. 1 Real HDR output of the proposed pipeline. Top left quarter contains stripes of original images, where significant car motion could be observed. The bottom left image reflects motion without our deghosting technique. Top right image contains Ghostmap used for image recovery (gray signs the under/overexposed patches, see Sect. 4.2). The right bottom image shows the resulting deghosted image. The upscaled license plates are shown directly under the Ghostmap

100 FPS on full HD video and thus it overcomes state-of-the-art in performance. The competitive advantages include deghosting method for simple ghost effect removal and novel architecture of local tone mapping operator. All parts of the proposed solution are suitable for hardware implementation and optimized for FPGA-based platforms and they were experimentally implemented on Xilinx Zynq-based smart camera platform.

2 High dynamic range imaging

Two main approaches to HDR image capture exist. The first approach assumes existence and exploitation of specialized HDR sensors, e.g., [35]; however, such sensors are generally expensive and technologically demanding. The second approach is based on standard sensors and on acquisition of image sequence with different exposure times [4, 22, 24, 29]. The sequence is merged into HDR images and the video composed from such images can be processed and eventually, for example, tone mapped for rendering on standard display devices. The images can be captured simultaneously, e.g., through beam splitter on several CCD/CMOS sensors [32] but, more often, they are captured sequentially, which is also our intended approach.

2.1 HDR merging

The most common approach for HDR acquisition is based on merging in radiance domain. This approach requires knowledge of the camera response function (CRF) [4, 24, 29] which is the function of the sensor output to the incident light. In general, RAW images are preferable for HDR composition, assuming that they have a linear or close to linear response function. Debevec and Malik [4] proposed an algorithm which fuses multiple images into a high dynamic range radiance map, whose pixel values are proportional to the true radiance values in the scene. The contribution of each pixel is determined from the weight function w, which reflects the probability that pixel value reflects the real luminance incident to the camera sensor. HDR image H is then calculated as a weighted average of the individual exposures L:

$$H = \frac{\sum_{i=1}^{N} w(L_i)\frac{L_i}{t_i}}{\sum_{i=1}^{N} w(L_i)}, \tag{1}$$

where N is the number of exposures and t_i exposure times.

2.2 HDR deghosting

The merging algorithms [4, 24, 29] are suitable for static scenes. Motion of objects during the image sequence capture causes adverse effects which are called as ghosting. To reduce such effects, various methods that can be used to detect and remove ghosting from HDR images have been developed.

As we focus on embedded solution with FPGA, we reviewed only those existing methods that are feasible for FPGA implementation. Sidibe et al. [30] suggests detection of ghost region using the pixel order relation. The approach is simple; if the scene does not contain motion, the pixel intensities have to follow the order of exposure times. This approach is very sketchy and detects motion only under very specific conditions. Grosh [12] proposes a simple method based on pixel value prediction. The pixel value can be derived from the known exposure time, assuming the linear response of CMOS values to the incident light. If the values do not fulfill the prediction, the region is marked as ghosted. Pece et al. [26] proposes extraction of threshold maps created using segmentation by median pixel value. Any difference between the threshold maps of input images and the reference image map is marked as a motion region. The pixels in the motion region are assigned smaller weights used in the HDR merge. Min et al. [23] improves method of Pece et al. [26] and introduces multi-level threshold map, where thresholds are selected to divide the image into multiple

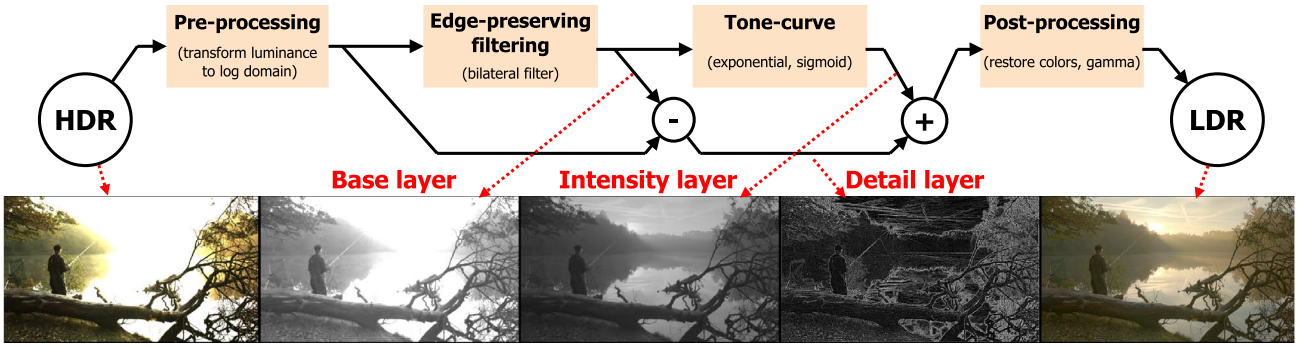

Fig. 2 Illustration and principled scheme of local tone mapping algorithms (Image retrieved from dataset [11])

regions with the same number of pixels. Both methods [23, 26] are very dependent on the histogram equalization since the region thresholds located close to each other may result in a large number of false ghost detections. Bouderbane et al. [2] proposes a simple ghost removing algorithm based on modification of weighting functions. Despite the comparison and experiments, the algorithm very significantly reduces the dynamic contrast in the whole image, independently on motion presence. Bouderbane et al. [2] implemented this method on FPGA into the HDR pipeline of Lapray et al. [15–17]. Most of the advanced and precise deghosting methods are based on optical flow, that is very memory and computationally demanding and thus it is not feasible for FPGA implementation or real-time processing.

2.3 Tone mapping

Displaying HDR content is still a challenging topic as standard displaying devices have a limited dynamic range, typically 8 or 10 bits per channel. To display image with bigger bit depth, the HDR images have to be post-processed by algorithms commonly called as tone mapping operator (TMO) which reduce the bit depth of HDR image so that it can be displayed using standard devices while preserving all the important details.

In general, two main approaches that can be used to display HDR content exist. The first approach, not used in our study, is to use specialized HDR monitors that directly render the HDR content; but such displays still have some limitations, they are expensive and they mostly must be used in very controlled environments. The second approach, that we use in our study, is based on application of dynamic range scaling with effort to reduce the dynamic range but to preserve local contrast in the scene details. This process, as mentioned above, is called tone mapping (or applying tone mapping operators). Many operators exist [5, 7, 8, 10, 19, 28]. These operators can be divided into two main categories—global and local operators. Global operators use the same mapping function for all pixels of the image. Parametrization of the function depends on global image characteristics, such as average, minimum and maximum values of luminance.

Local operators generally preserve more details than the global ones as they use information from neighboring pixels to estimate local illumination and thus adapt compression to local luminance conditions. The most frequently used approach is separation of HDR image into base and detail layers (see Fig. 2). The base layer contains large-scale variations of luminance and the detail layer contains local differences, which preserve details of the scene. The base layer can be obtained from luminance in logarithmic domain using low-pass edge-aware filtering, e.g., by bilateral filter (BF), as proposed by Durand [8], edge-avoiding wavelets [9] or by estimation from gradient domain (Gaussian pyramids [10]). The base layer, which is responsible for high dynamic range, can be compressed because fine details are preserved in detail layer.

The base layer separation is the most computationally demanding operation and it is desirable to accelerate its implementation. The BF is very computationally demanding; some approximations exist but their advantages in lower computation complexity only have some effect when very large kernels are used. For small kernels, such as the 9×9 pixel kernel which we chose to use, the original BF computation is the best choice and thus we chose to implement it.

3 Related FPGA pipeline implementations

Several papers that focused on the acceleration in FPGA and were related to our work were published [16, 17, 33, 34]. HDR image/video acquisition pipeline is summarized in Sect. 3.1 and FPGA acceleration of TMO is summarized in Sect. 3.2.

3.1 HDR acquisition pipeline

Mann [18] developed an FPGA-based wearable HDR seeing aid designed for the electric arc welding. The HDR output values are precomputed for full range of input pixel combinations and stored in lookup tables in BRAMs. Even after certain optimizations of memory consumption, the BRAM demands are very high, especially when more than two LDR images are used. They achieved very high frame rate solution—120 FPS.

Popadic et al. [27] presented an FPGA-based preprocessing unit for standard industrial cameras. The FPGA unit is independent and intended to be connected transparently between CPU and CMOS chip of camera. The unit obtains a sequence of images, merges them into HDR, and sends it to CPU in the same manner as the typical CMOS sensor.

Lapray et al. [15–17] introduced a custom-made FPGA-based platform equipped with color CMOS for HDR image acquisition and tone mapping. They published several papers regarding the same hardware platform and using various global tone mapping operators [6, 28]. They also proposed a HDR pipeline with special memory management unit [15] which allows for generation of HDR images at the same frame rate as the camera output, generating a new HDR from the currently captured frame and two previous frames. The proposed design achieves 60 FPS at resolution of 1280×1024. Bouderbane [2], inspired by work of Sidibe et al. [30], implemented a deghosting algorithm using this platform.

3.2 HDR tone mapping

Hassan and Carletta [13] presented a novel FPGA architecture for tone mapping of grayscale HDR images. Their solution is based on Reinhard operator with approximation of Gaussian pyramids. This implementation achieves 60 FPS on 1024×768 images. However, it does not involve HDR merging.

Marsi et al. [21] proposed HDR tone mapping algorithm and its FPGA implementation focused on driving assistance applications. This TMO uses low-pass filtering instead of BF to estimate the base layer and the detail layer. This approach is similar to Durand TMO [8], but the authors included temporal smoothing to prevent flickering and color shifting. Previous frame, used in temporal smoothing, is down-sampled to 1/4 and stored directly in FPGA. This operator achieves 24 FPS on 125×86 image.

Urena et al. [33] implemented their own optimized TMO. The algorithm starts with transformation into HSV color space and histogram equalization of the brightness (V) channel. This is followed by local details enhancement using convolution (window 7×7). The final brightness is a linear combination of convolution result and histogram equalization. This operator was evaluated on FPGA and GPU and it achieved real-time performance of 30 FPS on GPU (NVIDIA ION 2) and 60 FPS on FPGA, both for 640×480 pixels.

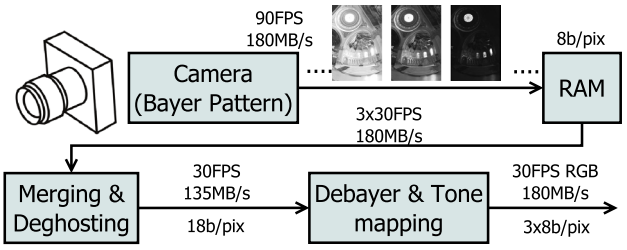

Fig. 3 Block scheme of proposed FPGA HDR pipeline with overall image throughput between individual blocks

Vytla et al. [34] developed hardware implementation of gradient domain HDR tone mapping using the Poisson equation solver inspired by Fattals operator [10]. The modified Poisson solver uses local information from pixel and its neighbors (with no global information). Each tone mapped pixel is computed using an independent application of Fattals operator on all pixel locations in 3×3 window. This architecture is implemented on FPGA and produces only grayscale tone mapped images at 100 FPS for 1 MPix images.

4 Proposed pipeline implementation

In this section, we describe the implementation of color HDR pipeline. Block scheme of the pipeline is shown in Fig. 3. The pipeline starts with application of inverse CRF function on input data to obtain data with linear response. Then, the HDR merging with ghost detection is applied. Finally, the TMO is applied, producing the video compatible with standard low dynamic range devices. The whole pipeline is designed with inspiration by work of Tamburrino et al. [31] where the HDR processing is applied directly on CFA pixels and the demosaicing is applied only after HDR processing.

4.1 Camera response function

The pixel values generated by CMOS sensors are not necessarily linear to incident light, even when they are captured in RAW format. The authors [4, 24, 29] proposed a CRF estimation algorithm, where they recover CRF from general images without additional information. In the proposed solution, we exploit a CRF function that is based on knowledge of the non-linearities and also knowledge of exposure times. We also experimentally discovered that expected linear dependency between two exposures was not fulfilled if

the expositions were extremely short, e.g., under ~ 150 µs. Both non-linearities were incorporated into a correction table, which implements the inverse CRF function for our CMOS sensor.

4.2 Proposed ghost-free HDR merging

We propose to use a ghost detection algorithm based on prediction of pixel value. It is based on similar principles as the work of Grosch [12]. Since the exposure time of each image is known, individual pixel values in image i can be predicted using values from j.

$$L_i \approx L_j \cdot \frac{t_i}{t_j}, \tag{2}$$

where t_x and t_y are exposition times of images. This relation holds only for non-saturated patches, so over/underexposed patches must be handled differently. The ghost detection is performed before the HDR merging phase, resulting in ghost pixel mask (further called as ghostmap), where the marked positions are treated differently from the non-marked ones during the HDR merging.

The function Ω tests the two images whether their pixels follows the prediction:

$$\Omega(L_i, L_j) = \begin{cases} 0 & L_i \cdot \frac{t_j}{t_i} / \alpha > L_j \\ 0 & L_i \cdot \frac{t_j}{t_i} \cdot \alpha < L_i \\ 1 & \text{else,} \end{cases} \tag{3}$$

where α represents the tolerance, which must be taken into account, since the sensor noise, quantization errors and CRF precision may influence the predicted value and thus cause the false ghost detections. According to our experiments, we use the tolerance $\alpha = 1.2$ as default, but this value can be adjusted based on the sensor features. In general, decreasing of the tolerance leads to more strict ghost detection, where more pixels are marked as ghosts, which eventually leads to worse dynamic range recovery. Increasing of ratio, on the other hand, decreases the chance of successful ghost detection. The ghostmap is defined as follows:

$$\overline{G} = \prod_{i=1}^{N-1} \Omega(L_i, L_{i+1}), \tag{4}$$

where non-zero value of G marks ghost pixels. The algorithm description is simplified by pixel range control—all of the under/overexposed pixels are omitted from the value prediction; Still, they are tested for extreme luminance changes (dark to bright and vice versa). The algorithm works per pixel and uses simple arithmetic operations and thus it is suitable for implementation on FPGA. The follow-up HDR merging algorithm is modified and in the areas, where ghostmap indicates motion, incorporates the pixels from only one image, called reference image, which is generally the best exposed or middle exposed image in the sequence.

4.3 HDR acquisition with ghost removal

We implemented the HDR merging algorithm from Debevec [4] with modification for ghost removal. Every pixel is assigned with the weight w, calculated using the triangle weight function by Debevec [4]. The function was modified for shortest and longest exposure, since in the original algorithm, the saturated pixels are assigned a value darker than pixels not completely saturated.

The HDR image H is calculated as a weighted sum of corresponding pixels in sequence of L:

$$H = \frac{\frac{1}{t_1} \sum_{i=1}^{n} \theta(i) \cdot w(L_i) \cdot L_i \cdot t_i}{\sum_{i=1}^{n} \theta(i) \cdot w(L_i)}, \tag{5}$$

where n is number of images in the sequence, sorted by ascending time. The time t_1 is used to shift pixel values to the common time base. The function θ incorporates the results of ghost detection, where $\theta = 1$ for reference image and $\theta = 1 - G$ otherwise.

4.4 Tone mapping

Tone mapping is the most complex stage in our pipeline. The proposed implementation uses, based on our experiments, smaller kernel than the one by [8]. To get the base layer, bilateral filter is the good choice in case of the small kernel. In our case, we selected implementation of BF with kernel size of 9×9 pixels. The size of kernel was selected to provide as high quality as possible, keeping reasonable computational complexity and also reasonable FPGA resource demands. Further resources are saved by implementation of TMO with fixed point arithmetics with the range adjusted exactly to the task. The input of TMO is CFA mosaiced HDR pixel and it produces tone mapped RGB image in range $< 0, 1 >$. TMO starts with computation of (R, G, B) value for every pixel; we use simple averaging for CFA demosaicing. The consequent step computes luminance from (R, G, B) pixel using the following equation:

$$I = 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B \tag{6}$$

Then, we convert I into logarithm base and perform the bilateral filtering to obtain the base layer. The detail layer is obtained as:

$$\text{Detail}(I) = \ln(I) - \text{Base}(I) \tag{7}$$

The minimum and maximum values from the base layer are used for calculation of compression factor c:

$$c = \frac{\alpha}{\max - \min} \tag{8}$$

Where the value of α controls the resulting contrast of tone mapped image, we typically use values within $< 2.5, 3.5 >$, while higher value of α results in brighter tone mapped image. The new intensity I_{new} for channels (R, G, B) is computed as follows:

$$\ln(I_{\text{new}}) = \text{Base}(I) \cdot \alpha + \text{Detail}(I) - \beta \tag{9}$$

where the value of β ensures that maximum value of the compressed base layer is equal to one.

We use Durand mapping function (10) with modified color correction presented by Mantiuk et al. [19], where parameter s is used to control the color saturation (we use value 0.8 as a default). The output of mapping equation is a number in range $< 0;1 >$. Finally, the result is linearly remapped into the range $< 0;255 >$ and enhanced by gamma function.

$$(R, G, B) = \left(\left(\frac{(R, G, B)}{I} - 1 \right) \cdot s + 1 \right) \cdot I_{\text{new}} \tag{10}$$

Our pipeline is intended for video processing and not for individual images; therefore, we use a smooth adaptation of dynamically changing average of the extreme values from a series of previous frames similarly to Kiser et al. [14]. This approach avoids adverse video effects, such as flickering, which may arise with quick changes of luminance. Moreover, we do not use the minimum and maximum value of the base layer, we use 5, 10 (by default) or 15% percentiles instead of minimum and maximum value to suppress the random noise (which can also produce flickering).

4.4.1 Bilateral filtering

Computation of the base layer by bilateral filter (BF) is the most resource-consuming part of the TMO. We chose to implement 9×9 pixel window which we found to be the minimum size suitable for the purpose, where the number of operations, especially multiplications, is still high and demanding for FPGA resources. The proposed implementation is optimized using the reduced spatial range and using spatial coefficients that are quantized to reduce the number of arithmetic operations to minimum. This is achieved using the coefficients containing only limited number of 1s in their binary representation while they stay as close as possible to their precise values. Multiplication by such coefficients is then reduced only to few shift operations in combination with addition (shift-and-add algorithm [1]). Spatial coefficient calculation can be precalculated and "hardwired", unlike the range coefficients, which are stored in a quantization table. In our implementation, we quantized the coefficients to keep two most significant "1" in binary representation. The proposed optimization does not have any fundamental adverse influence on filtering results, as summarized in Table 3.

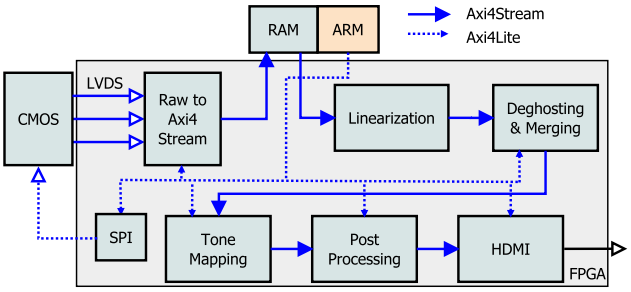

Fig. 4 Overview of HDR architecture and interconnections of individual blocks inside the FPGA of Xilinx Zynq SoC

4.5 HDR pipeline

Figure 3 presents the top level overview of data throughput requirements—the amount of data processed by individual top level blocks. The detailed block diagram of the complete HDR pipeline is illustrated in Fig. 4. The CMOS output consists from a raw CFA image data with a Bayer filter mosaic. The raw image is stored to DDR memory using DMA and double-buffering to avoid overwriting of the data. The HDR merge block reads three image streams simultaneously through the DMAs. It applies inverse CRF to obtain image with linear response, calculates Ghostmap and merge HDR image by algorithm described in Sect. 4.3. The essential part of the pipeline is the tone mapping, which compresses HDR by Durand [8] TMO.

4.5.1 Merging with deghosting

The first step, right before pixel merging is a ghost detection according to Eq. (3). Ghost flag θ is used in pixel merging (see Eq. (5)), where ghosted pixels ($\theta = 0$) are omitted using multiplexer with zero pixel value. The HDR merging algorithm was optimized to adapt to FPGA-based platforms. The input pixel is stored on 10.4 bits (meaning fixed point number with 10 integer and 4 fractional bits). The pixel weights are tabulated by 1024 entries and they are addressed only by integer bits. Because of numerical precision, we represented weights by 18-bit fixed point number which fits to port size of BRAM memory (memory block on FPGA). Multiplication of pixel by weight is performed in digital signal processing (DSP) blocks. The DSP is used also to multiply the weighted pixel by exposition ratio, which scales pixel values to the common range.

Since division is a time-and resource-demanding operation, we converted it into multiplication with a tabulated fractional value. The sum of weights, according to bit widths of intermediate results, needs to be represented by 11 bits (sum of three 9 bit values), thus the fraction value is tabulated on 2048 entries. The resulting HDR pixel is in 10.8 fixed point representation.

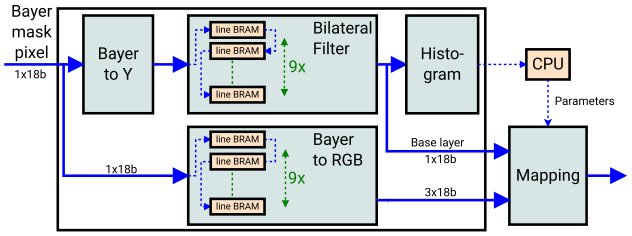

Fig. 5 Block scheme of tone mapping pipeline with detailed insight into parameters calculation block

Fig. 6 Hardware prototype of HDR camera

The processing of grayscale and color image are identical. We process individual pixels of color filter array (CFA), in our case a Bayer mask, in the same manner as the grayscale pixels [31].

The merging and deghosting algorithms are applicable on arbitrary number of images, but for practical reasons, our pipeline is designed for a sequences of three images. We choose the middle image in the sequence as the reference for the deghosting.

4.5.2 Tone mapping

We propose a new computing scheme for Durand TMO (see Fig. 5). This approach can be easily adopted also for various local TMOs. Input of TMO block is CFA mosaiced HDR pixel (e.g., in Bayer mask) and this pixel is passed into parallel blocks—base layer computation block and CFA demosaicing. Base layer block computes the luminance I from CFA pixels and performs the bilateral filtering. Demosaicing block contains FIFO and demosaicing is performed concurrently with corresponding output of base layer block. The serial connection of the FIFO line buffers represents the vertical displacement of the filtering masks. As a result of this architecture, only $\frac{1}{3}$ values are stored in FIFO (because CFA Bayer pixel is not stored as a RGB), which is advantageous especially when bilateral filter with large kernel is used.

Lapray [17] applies demosaicing after tone mapping, which requires additional delay lines for demosaicing process. We decided to overlap demosaicing and bilateral filtering delay lines (see Fig. 5). The benefit is that demosaicing algorithm with kernel size less or equal to kernel size of BF can be implemented without influence on FPGA resources.

4.5.3 Data representation

The whole FPGA design is implemented using only fixed point data representation and arithmetic which is natural and also very efficient for FPGA hardware. We know exactly the range of numbers in every part of pipeline, thus we are able to adapt the bit width of numbers to achieve sufficient precision without using the floating point representation. The resource requirements for floating point calculations are higher in general, which could increase total requirements considerably, since for every pixel, we perform a large number of arithmetic operations, e.g., during bilateral filtering.

The input of linearization block is 8 bit integer mosaiced pixel and output is pixel in fixed point representation of 10.8. The pixels (in 10.8 format) are propagated through ghost detection and HDR merging block to the tone mapping block. Two data types occurs in TMO—10.8 for input CFA mosaiced pixels, which are stored in FIFOs and processed by demosaicing algorithm, and 5.13 for logarithmic space calculations in bilateral filter. The intermediate results are often stored using even higher bit width to prevent precision losses. The division, logarithm, and exponential function are implemented based on tabulated values, bit shifts, additions, and multiplications. For example, the exponential function is implemented using two tables—one is indexed by higher half part of fixed point number and second by the lower half part. The 18-bit width is used through whole design due to the convenient size of BRAM (organization 1024×18 bits).

4.5.4 Camera hardware

We designed a custom-made camera platform based on SoC Xilinx Zynq XC7Z020 (see Fig. 6). The Zynq SoC contains dual-core ARM Cortex-A9 and FPGA on the same chip. Overall power consumption of camera is 8 W.

The platform is equipped by low noise global shutter CMOS sensor Python2000 from ON semiconductor, connected directly to FPGA through high-speed LVDS (Low Voltage Differential Lines) interface. The CMOS has a resolution of 1920×1280 pixels and is capable of capturing up to 255 FPS—this is the maximum speed of LVDS interface.

The described method itself, is not constrained to this camera or any particular features, such as frame rate and/or number of LDR exposures merged to HDR frame. However, it must be noted that multiple exposures must fit within the particular time frame and thus exposure times of individual LDR frames are limited by the frame rate. In our application,

we use 3 LDR frames and limit maximal output frame rate to 30 FPS which is constrained by the used hardware codec. This setting gives a sufficiently wide range of exposure times for many lighting conditions. When longer exposures are necessary, the frame rate can be reduced. The resulting image is streamed over HDMI from FPGA to hardware H.264 encoder Fujitsu MB86M25. To provide a constant camera frame rate, the whole pipeline is driven by HDMI timing circuit, which is generating interrupts with each output video frame. These interrupts are used to start capturing the image brackets. In the FPGA design, we follow the AXI4-Stream Video standard (subset of ARM AMBA standard[1]) for image data transfers between individual processing blocks. It ensures compatibility with third-party IP cores and easy extendability.

The pipeline and its parameters are configurable by set of registers accessible over Axi-Lite interface. The configuration itself can be written by arbitrary state machine and/or processor, either synthesized on FPGA or physically present on the same board (see Fig. 4). In our case, we benefit from presence of the ARM processor on Zynq SoC, which is not nearly as powerful as to run the HDR processing itself but which is suitable for calculating, e.g., temporal parameters for TMO, setting up the optimal exposition times, etc.

5 Evaluation

This section is focused on evaluation of the HDR pipeline from the point of FPGA resource utilization, precision, and visual output quality including the HDR deghosting and TMO results.

5.1 Resource utilization

Table 1 summarizes resource utilization of the complete design, including HDR pipeline and also DMA controllers, memory interfaces, and configuration registers. This supporting circuitry covers approximately 40% of the occupied resources, where a significant part is used for configuration of deghosting, merging, and TMO parameters. The TMO demands are high due to bilateral filtering, which consumes majority of resources. The overall FPGA utilization is relatively high; however, we are using small and inexpensive FPGA, which could be upgraded. Still, there is enough resources for additional design updates.

Table 1 Design resource utilization for Xilinx Zynq Z-7020

	LUT	Registers	BRAM	DSP
DMAs and Mem. ctrl	14,660	21,981	43	0
CMOS controller	952	2361	0	0
Linearization	4	10	2	0
Ghost detection	229	384	0	4
Merging	329	1184	3	16
Tone mapping	14,706	20,316	34	38
Codec interface	3641	1949	9	0
Overall	34,521	48,185	91	58
Available	53,200	106,400	140	220
Utilization (%)	64.8	45.2	65.0	26.3

Table 2 Average precision of our HDR pipeline components

	PSNR	MSSIM
Merging	107.43	1.0
Intensity computation	103.05	1.0
Logarithm	76.10	0.99
Bilateral filtering	43.72	0.98
Tone mapping	48.54	0.99
Overall precision	46.37	0.94

Precision was evaluated against SW implementation and on a set of well-known HDR images from related publications

5.1.1 Image quality comparison

Many authors modified the original algorithms to be more suitable or even applicable at all for hardware acceleration. In fact, no "golden standard" among the TMOs exists, thus the common approach of the authors is to compare their SW reference implementation (typically in floating point) to the VHDL simulation or hardware results. The mutual comparison of TMOs using standard metrics is very rare.

We evaluated an output image quality according to the mean structural similarity index (MSSIM) and the peak signal-to-noise ratio (PSNR) metrics.

We created, as mentioned above, the design based on the original implementation of Durand TMO with Mantiuk et al. [20] color enhancement. The overall accuracy of the FPGA implementation (PSNR) was 46.37 dB on average compared to the SW implementation. The precision was evaluated on a set of 20 various well-known HDR images. As it can be observed in Table 2, the highest precision error arises in bilateral filter, whose Gaussian coefficients are approximated (see Sect. 4.4). However, the overall precision is still very high and the output of our FPGA implementation provides very good visual results.

[1] www.arm.com/products/system-ip/amba-specifications.

Table 3 Average precision of our BF used for the base layer separation

Method	PSNR	MSSIM
BF FX FPGA	48.38	0.99
BF FX1 FPGA	32.21	0.96
BF FX2 FPGA	43.72	0.98
BF Durand [8]	41.84	0.98

Precision of BF was evaluated comparing to OpenCV on a set of well-known HDR images from related publications. BF FX is fixed point implementation of BF, BF FX1 uses shift-and-add (based on [1]) for coefficients approximation and BF FX2 uses two shift-and-add steps. BF Durand is piece-wise linear approximation proposed by Durand and Dorsey for their TMO

5.2 Bilateral filter precision

Table 3 contains an accuracy overview of BF application in tone mapping. We compared our implementation to the state-of-the-art method presented by Durand and Dorsey [8] (BF Durand in Table 3). The original implementation replaces bilateral filter by usage of piece-wise linear approximation in spatial kernel. BF FX FPGA is our implementation of bilateral filter using fixed point arithmetic only (multiplication on DSP blocks). The FX1 FPGA variant of BF has the spatial and range coefficients represented by one bit-shift operation and the BF FX2 FPGA has spatial coefficients represented as a sum of two shifts and the range ones by one shift. We chose the variant BF FX2 for implementation since it has comparable precision to the pure fixed point implementation BF FX while occupying only fraction of FPGA resources. It also outperforms the precision of Durand's approximation [8] and thus it can be concluded that our BF FX2 is sufficient for HDR tone mapping.

5.3 Comparison to state-of-the-art

The results of the proposed method are mainly affected by the three functional blocks—HDR deghosting, merging, and tone mapping. Only a small number of related research works addressing deghosting and merging exists. Majority of FPGA implementations of HDR processing focus only TMO and disregard any ghosting effects.

5.3.1 Deghosting

The only attempt to implement FPGA deghosting was presented by Bouderbane et al. [2]. We cannot compare to their result as it is working only in limited conditions (see Sect. 3). Moreover, they do not provide any FPGA resource utilization.

The results of our deghosting method are presented in Figs. 1 and 7. Our method is applicable for any surveillance purposes with static cameras and other similar applications; in this paper, the deghosting results are presented on traffic monitoring task, where the main goal is to preserve as many details as possible for the evidence purposes. Especially the license plate of approaching car has to be readable. Figure 7 contains standard motion of car approaching camera by 50 km/h. For Fig. 1, six frames of LDR images were intentionally omitted between every source LDR image to show capability of deghosting, e.g., faster moving objects.

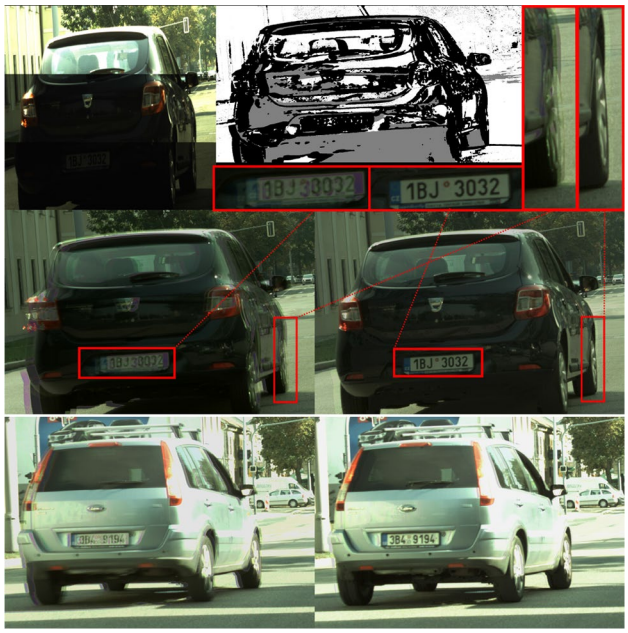

Fig. 7 Deghosting method results shown on moving cars. Please mind the marked areas, where the ghosting is significant. Further description is provided in Fig. 1

Table 4 Comparison of main parameters of our solution with Lapray et al. [16]

	This work	Lapray
Platform	Zynq 7020	Virtex 5
Resolution	1920 × 1080	1280 × 1024
TMO	Durand (local)	Reinhard (global)
Arithmetic	Fixed point	Floating point
Maximum speed	200 Mhz	126.733 Mhz
Throughput	200 Mpix/s	78.6 Mpix/s
Frame rate	96 FPS	60 FPS
Colors	Yes	Yes
Deghosting	Yes	No

5.3.2 FPGA HDR camera solutions

When we compare our work to Lapray [16] (see Table 4) in terms of resource utilization, our solution is more

demanding. It is mainly because we implemented local TMO with 9×9 pixel kernel and Lapray implemented global TMO only. As opposed to Lapray, we implemented the whole pipeline in fixed point, which is advantageous for resource utilization and also performance. Our solution differs from Lapray [16] also in capturing of the image sequence, where we take advantage of fast CMOS sensor and we capture three images for each output HDR image. The images are captured immediately in a sequence to reduce the ghosting effect. Moreover, we implemented a deghosting method to suppress any artifacts that may still arise. Lapray [16] captures images with constant time spacing and creates new HDR with every incoming LDR image.

Mann [18] developed a wearable HDR seeing aid, but they did not present any clear summary of resource consumption or evaluation of HDR pipeline precision. The precision of HDR imaging strongly depends on consumption of BRAMs, as explained in Sect. 3.1.

5.3.3 FPGA tone mapping solutions

Several works about accelerated tone mapping implementation on FPGA were proposed (see Sect. 3). We implemented a local TMO based on bilateral filter with quite sufficient kernel size 9×9 pixels, which helps to preserve local details. Still, according to Table 5, we do not have the highest FPGA resource consumption, even if it could be assumed.

As it can be observed in Table 5, Ofili et al. [25] and Vytla et al. [34] have the highest value of PSNR. This could be expected, since their tone mapping uses only small local neighborhood (3×3). In result, significantly less arithmetic operations are used and so the PSNR could be higher. Hassan and Carletta [13] achieved lower PSNR even in spite of the fact that they used four scale Gaussian pyramid.

Lapray et al. [16] have the lowest resource consumption because they implemented simple global TMO based on Duan [6] algorithm, which uses image histogram retrieved from their smart CMOS sensor. Also, their whole tone mapping uses only a small number of arithmetic operations and so the resource consumption is low. Ofili et al. [25] and Vytla et al. [34] also have lower resource consumption, but they implemented small local TMO with kernel of 3×3 pixels. Hassan and Carletta [13] have the highest resource utilization, mainly due to implementation of local TMO with multi-scale Gaussian filtering and they are processing HDR images with 28 bits per pixel, but HDR image is merged offline. Urena et al. [33] have high resource utilization because they implement tone mapping based on retina-like processing. High memory consumption (96%) is caused by HSV conversion implemented by lookup in BRAM memory and 7×7 convolution window.

Table 5 shows that our pipeline is superior not only in highest resolution and frequency, but also in terms of

Table 5 Comparison of tone mapping operator with similar research works

	Tone mapping	Type	Platform	Image type	Image size	Freq. (Mhz)	Throughput (MPix/s)	Speed (FPS)	LUT	FF	Memory (Kbits)	PSNR (dB)
Ofili [25]	Glozman	Local	FPGA	Color	1024×768	115	99	126	8546	10442	68	54.2
Hassan [13]	Reinhard [28]	Local	FPGA	Gray	1024×768	77	100	60	34806	N/A	3079	33.7
Vytla [34]	Fattal [10]	Local	FPGA	Color	1 Mpix	114	100	100	9019	6589	300	53.8
Chiu [3]	Reinhard [28] and Fattal [10]	Both	SoC	N/A	1024×768	100	47.2	60	N/A	N/A	N/A	45 and 35
Urena [33]	Urena [33]	Both	FPGA	Color	640×480	40	18.5	60	30086	N/A	702	N/A
Lapray [15]	Duan [6]	Global	FPGA	Color	1280×1024	127	127	114	3433	4045	540	N/A
This work	Durand [8]	Local	FPGA	Color	1920×1080	200	200	96	14,706	20,316	1368	46.69

PSNR values are for Memorial image (VHDL vs. SW implementation). For our and Lapray solution we selected size of the TMO block only

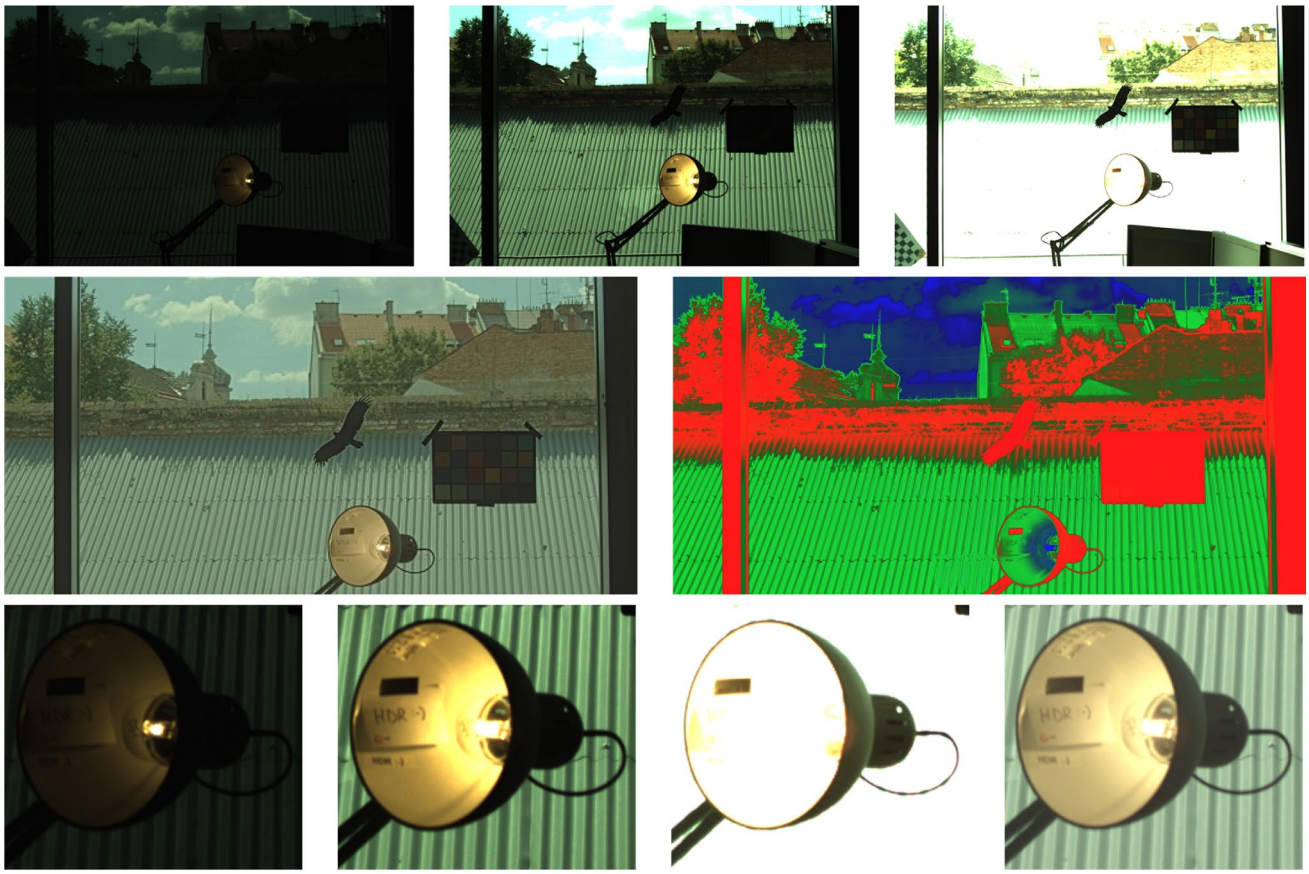

Fig. 8 The figure shows inputs and output of our HDR pipeline. The sequence of three images is shown in upper row, the resulting tone mapped HDR image is shown at left bottom. The color map at right bottom shows the contribution of individual images to the resulting HDR (blue—the shortest exposure, green—middle exp., red—the longest exp.). The detail of lamp is shown at the bottom. Three images from the left are LDR, TMO HDR is on the right

pipeline throughput. This is achieved by fully pipelined implementation, which is sufficiently powerful for real-time processing systems. Also, the resource consumption for the price of TMO with relatively large kernel is more than convenient. An example of input LDR sequence and tone mapped output are shown in Fig. 8.

6 Conclusion

In this paper, we presented a complete real-time HDR pipeline, which can take place in standalone FPGA-based devices and/or as an enhancement or preprocessing part/block in smart cameras, allowing the capture of more details in the scene, typically in difficult light conditions.

Our custom-designed Zynq-7020 based camera is capable of capturing 30 FPS of full HD HDR video.[2] The whole design is implemented using fixed point arithmetics, thanks to which it can fit into the small and inexpensive FPGAs on the market. Our pipeline is also highly customizable during the run time and so it can be extended by automatic parameter calibration to even further improve capturing quality. The main highlights of our design are local tone mapping operator with relatively large kernel and application of ghost removal algorithm which significantly improves perceived quality of HDR image. The main field of applications of our HDR camera in the industry is automatic quality control, automotive or surveillance cameras, etc.

Acknowledgements This work was supported by The Ministry of Education, Youth and Sports of the Czech Republic from the National Programme of Sustainability (NPU II); project IT4Innovations excellence in science—LQ1602.

[2] Short example videos captured by our camera prototype are available at http://www.fit.vutbr.cz/research/groups/graph/downloads/jrtip2018_hdr_video.zip.

References

1. Baba, N., Isobe, S., Norimoto, Y., Noguchi, M.: Stellar speckle image reconstruction by the shift-and-add method. Appl. Opt. **24**, 1403–1405 (1985)
2. Bouderbane, M., Dubois, J., Heyrman, B., Lapray, P.J., Ginhac, D.: Ghost removing for HDR real-time video stream generation. In: Real-time image and video processing, 98970F (2016)
3. Chiu, C.T., Wang, T.H., Ke, W.M., Chuang, C.Y., Huang, J.S., Wong, W.S., Tsay, R.S., Wu, C.J.: Real-time tone-mapping processor with integrated photographic and gradient compression using 0.13 um technology on an arm soc platform. J. Signal Process. Syst. **64**, 93–107 (2011)
4. Debevec, P.E., Malik, J.: Recovering high dynamic range radiance maps from photographs. In: ACM transactions on graphics (TOG), SIGGRAPH '97, pp. 369–378. ACM, New York (1997)
5. Drago, F., Myszkowski, K., Annen, T., Chiba, N.: Adaptive logarithmic mapping for displaying high contrast scenes. In: Computer graphics forum, pp. 419-426. Blackwell Publishing, Inc, Oxford, UK (2003)
6. Duan, J., Bressan, M., Dance, C., Qiu, G.: Tone-mapping high dynamic range images by novel histogram adjustment. Pattern Recogn. **43**, 1847–1862 (2010)
7. Duan, J., Qiu, G., Chen, M.: Comprehensive fast tone mapping for high dynamic range image visualization. In: Pacific Graphics (2005)
8. Durand, F., Dorsey, J.: Fast bilateral filtering for the display of high-dynamic-range images. In: ACM transactions on graphics (TOG), SIGGRAPH '02. pp. 257–266. ACM, New York (2002)
9. Farbman, Z., Fattal, R., Lischinski, D., Szeliski, R.: Edge-preserving decompositions for multi-scale tone and detail manipulation. In: ACM SIGGRAPH 2008 Papers, SIGGRAPH '08. ACM, New York, NY, USA (2008)
10. Fattal, R., Lischinski, D., Werman, M.: Gradient domain high dynamic range compression. In: ACM Transactions on Graphics (TOG), SIGGRAPH '02, pp. 249–256. ACM, New York (2002)
11. Froehlich, J., Grandinetti, S., Eberhardt, B., Walter, S., Schilling, A., Brendel, H.: Creating cinematic wide gamut HDR-video for the evaluation of tone mapping operators and HDR-displays. In Proc. SPIE 9023 (2014)
12. Grosch, T.: Fast and robust high dynamic range image generation with camera and object movement. Vision, Modeling, and Visualization 2006: Proceedings, pp. 277–283 (2006)
13. Hassan, F., Carletta, J.E.: A real-time FPGA-based architecture for a Reinhard-like tone mapping operator. In: SIGGRAPH/EUROGRAPHICS symposium on Graphics Hardware, pp. 65–71 (2007)
14. Kiser, C., Reinhard, E., Tocci, M., Tocci, N.: Real time automated tone mapping system for hdr video. In: IEEE Proceedings of the IEEE International Conference on Image Processing (2012)
15. Lapray, P.J., Heyrman, B., Ginhac, D.: HDR-ARtiSt: A 1280 x 1024-pixel adaptive real-time smart camera for high dynamic range video. In: SPIE Photonics Europe. Brussels, Belgium (2014)
16. Lapray, P.J., Heyrman, B., Ginhac, D.: Hdr-artist: an adaptive real-time smart camera for high dynamic range imaging. J. Real Time Image Process. **12**, 747–762 (2016)
17. Lapray, P.J., Heyrman, B., Ross, M., Ginhac, D.: HDR-artist: High dynamic range advanced real-time imaging system. In: 2012 IEEE International Symposium on Circuits and Systems (2012)
18. Mann, S., Lo, R.C.H., Ovtcharov, K., Gu, S., Dai, D., Ngan, C., Ai, T.: Realtime HDR (high dynamic range) video for eyetap wearable computers, FPGA-based seeing aids, and glasseyes (eyetaps). In: 2012 25th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE) (2012)
19. Mantiuk, R., Daly, S., Kerofsky, L.: Display adaptive tone mapping. ACM Trans. Graph (TOG), **27**, 68 (2008)
20. Mantiuk, R., Mantiuk, R., Tomaszewska, A., Heidrich, W.: Color correction for tone mapping. In: Computer graphics forum, pp. 193–202. Blackwell Publishing Ltd, Oxford, UK (2009)
21. Marsi, S., Impoco, G., Ukovich, A., Carrato, S., Ramponi, G.: Video enhancement and dynamic range control of HDR sequences for automotive applications. EURASIP J. Adv. Signal Process. **2007**, 080971 (2007)
22. Mertens, T., Kautz, J., Reeth, F.V.: Exposure fusion. In: Computer Graphics and Applications, 2007. PG '07. 15th Pacific Conference on (2007)
23. Min, T.H., Park, R.H., Chang, S.: Histogram based ghost removal in high dynamic range images. In: IEEE International Conference on Multimedia and Expo, 2009. ICME 2009 (2009)
24. Mitsunaga, T., Nayar, S.K.: Radiometric self calibration. In: Proceedings of 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149), vol. 1 (1999)
25. Ofili, C., Glozman, S.: Yadid-Pecht: hardware implementation of an automatic rendering tone mapping algorithm for a wide dynamic range display. J. Low Power Electron. Appl. **3**, 337–367 (2013)
26. Pece, F., Kautz, J.: Bitmap movement detection: Hdr for dynamic scenes. In: IEEE 2010 Conference on Visual Media Production (CVMP) (2010)
27. Popadić, I., Todorović, B.M., Reljin, I.: Method for HDR-like imaging using industrial digital cameras. Multimed. Tools Appl. **76**, 12801–12817 (2017)
28. Reinhard, E., Stark, M., Shirley, P., Ferwerda, J.: Photographic tone reproduction for digital images. ACM Trans. Graph. **21**, 267–276 (2002)
29. Robertson, M.A., Borman, S., Stevenson, R.L.: Estimation-theoretic approach to dynamic range enhancement using multiple exposures. J. Electron. Imaging **12**, 219–229 (2003)
30. Sidibe, D., Puech, W., Strauss, O.: Ghost detection and removal in high dynamic range images. In: 2009 17th European Signal Processing Conference (2009)
31. Tamburrino, D., Alleysson, D., Meylan, L., Süsstrunk, S.: Digital camera workflow for high dynamic range images using a model of retinal processing. In: IST/SPIE Electronic Imaging: Digital Photography IV, vol. 6817 (2008)
32. Tocci, M.D., Kiser, C., Tocci, N., Sen, P.: A versatile hdr video production system. In: ACM SIGGRAPH 2011 Papers, SIGGRAPH '11. ACM, New York, NY, USA (2011)
33. Ureña, R., Martínez-Cañada, P., Gómez-López, J.M., Morillas, C., Pelayo, F.: Real-time tone mapping on GPU and FPGA. EURASIP J. Image Video Process. 1 (2012)
34. Vytla, L., Hassan, F., Carletta, J.E.: A real-time implementation of gradient domain high dynamic range compression using a local poisson solver. J. Real Time Image Process. **8**, 153–167 (2013)
35. Zhao, H., Shi, B., Fernandez-Cull, C., Yeung, S.K., Raskar, R.: Unbounded high dynamic range photography using a modulo camera. In: ICCP (2015)