Towards Mobile HDR Video

Tassio Castro
tknop@impa.br

Alexandre Chapiro
achapiro@impa.br

Marcelo Cicconet
cicconet@impa.br

Luiz Velho
lvelho@impa.br

Abstract

We present a method for High Dynamic Range video where the critical phases of the pipeline are based on histograms. It is possible to achieve high frame rates, since the algorithm generates one HDR frame per captured frame. Also, the method is of low computational cost, making it particularly suited for devices with less powerful processors. An implementation of the technique for the Nokia N900 smartphone, using the recent FCam API, is detailed.

1. Introduction

Since the development of the technique in the latter 90's, High Dynamic Range Imaging from pictures with different exposures has never ceased to be a hot topic in Computational Photography. In the beginning, softwares developed by pioneer researchers (like HDRShop and Photosphere) were released, and soon the feature became available in commercial imaging products such as Photoshop.

Through the past decade, several improvements were made in the technique, but the user still had to take several photographs and process them with computer software. Very recently, however, cameras with built-in HDR began to appear. The Pentax K-7, released in 2009, is said to be the first camera featuring HDRI. A more notable example is the iPhone 4 platform, which offers an HDR mode in the camera app since the release of the iOS 4.1 this year.

In the realm of HDR video, cameras with HDR sensors, like the RED Epic and the ARRI Alexa, are being widely used these days in film production. However, such devices are not available for the general consumer. An alternative is to use cameras that can capture sequences of frames with different exposures and apply a technique similar to the one used for still images. Fortunately, these cameras are more affordable, but methods for HDR video are still scarce.

In this work we present an HDR video reconstruction method for hand-held cameras, including those installed in some mobile phones. Being based on histograms, the method is more adequate for devices with less computational power. To the best of our knowledge, previous methods make use of (computationally expensive) optical flow techniques to find correspondences between frames in the photometric calibration phase of the algorithm. These methods assume that the exposure is constant for different frames, therefore they consider the interleaved pattern of exposures in the HDR video sequence. In this case the temporal resolution suffers considerably. Our method, in turn, does not reduce the video frame rate, which is another advantage.

In the next section some basic works on HDR Imaging are mentioned, especially those that provide more details and background to our approach, which is described in Section 3. In Section 4 we describe the mobile device implementation. Results are discussed in Section 5. Final comments and future directions are shown in Section 6.

2. Previous Work

The problem of High Dynamic Range image reconstruction for still images is virtually solved [13]. Furthermore, many different solutions appeared, since the classical work by Debevec and Malik [2]. HDR video reconstruction is a natural extension of the image related problem. Therefore, approaches tend to be built upon methods for still images. As it is out of the scope of this text to review all existing approaches, we refer the reader to [12], which is a nice review of the subject.

High Dynamic Range video reconstruction is a more challenging task because, from the hardware side, it requires a programmable camera; and, from the software side, the data is dynamic. The earlier reference in this case is [5], where classical vision methods for motion estimation (namely, optical flow) are used to deal with the motion between frames. For a review of methods we refer to [10], where components of the HDR pipeline are presented and discussed with the main focus on video.

Our approach for HDR video is based on histograms. It is efficient, simple and robust to noise. We will discuss the method in Section 3. The recovery of the camera response function from images is described with details in [4] and [9]. The HDR reconstruction algorithm is a modified version of [11]. The histogram-based image registration technique is brought from [14], and the Radiance Map reconstruction with ghost removal is made in a similar way to what is described in [8].

Regarding handheld devices, the problem of HDR reconstruction from misaligned and (possibly) blurred long-exposed photographs is treated in [7]. The authors of [1] provide a camera application with HDR mode. In fact they made available a full API for experiments with the low level aspects of the camera hardware. We have used their platform in this work, the video capture being made with a Nokia N900 smartphone.

3. HDR Video

The pipeline for HDR Photography from LDR images has two phases: first, photometric calibration must be performed, and the radiance map reconstructed, to generate the HDR picture; then, the dynamic range of the result should be reduced to allow the image to be shown in LDR displays.

In order to recover the camera response function and properly map radiance values, algorithms for HDR rely on pixel correspondences between frames, to relate differently exposed values of the same point in the scene. When it comes to video, this is obviously a difficult task.

The solution is to apply some sort of motion estimation. Classical vision methods for motion estimation (such as optical flow), assume that the exposure is constant. However, to obtain HDR video, a sequence where consecutive frames have different exposures should be captured.

To get around this problem, we devised a method where motion estimation is based on histograms. Another advantage of using histograms is that it is less computationally expensive than methods based on optical flow techniques. This is particularly interesting for devices with less capable processors.

Our method has three steps: first, the camera response function is estimated using an histogram-based technique; second, multiresolution alignment of threshold images based on histogram cuts is performed; third, the radiance map is reconstructed observing the variances of radiance values for each pixel. The algorithm is detailed in the following subsections.

3.1. Photometric Calibration

The input of our algorithm is a sequence of triples[1] $\{F^i\}$, where $F^i = \{F^i_1, F^i_2, F^i_3\}$. The exposure of F^i_1 is constant, for all i, and is determined before the video capture, using the auto-exposure feature of the camera. F^i_2 and F^i_3 have exposures that are, respectively, twice and a half of the exposure of F^i_1, for all i.

Algorithms for photometric calibration require a correspondence between pixels of different frames in F^i to be known. We are assuming that exposure changes preserve monotonicity of pixel values. Intuitively, the n brightest pixels in a frame with exposure e_1 correspond approximately to the n brightest pixels in a subsequent frame with exposure e_2, even though their actual values are not the same. Let $\{p_i\}$ and $\{q_i\}$ be the sets of pixels from two consecutive frames (say, P and Q), of the same size, sorted according to the luminance value of the pixel. The radiance mapping $M_{P,Q}$, between P and Q, is defined simply by $M_{P,Q}(p_i) = q_i, \forall i$.

Finally, the actual *pixel value* to *radiance value* mapping can be recovered by applying any of the algorithms available in the literature. For this particular implementation we have used the parametric approach described in [9].

3.2. Histogram-Based Registration

Aiming not to reduce the HDR-reconstructed video frame rate when compared to the captured video, we generate an HDR frame for each frame in F^i. Therefore, once i is fixed, for each $j = 1, 2, 3$, the remaining F^i_k should be aligned with F^i_j. We perform a multi-resolution alignment that is described with details in [14].

Roughly speaking, the method is as follows: first, an image pyramid is constructed for each grayscale image exposure. Then, for each level of the pyramid a corresponding median threshold bitmap (MTB) image is constructed. An MTB image has 0's where the input pixel values are less than or equal to the median value and 1's where these values are greater. The overall offset for alignment is computed starting with the lowest resolution MTB pair, testing for an offset in the range $\{-1, 0, 1\}$ in the horizontal and vertical directions. At the next resolution level this offset is multiplied by 2 and the result is tested with its pair shifted in the range $\{-1, 0, 1\}$ in the mentioned directions. This continues up to the highest resolution, leading to the final alignment offset.

3.3. Radiance Map Reconstruction

The image alignment phase is necessary to deal with camera motion. However, there can also be movement in the scene, and such movement causes ghosting effects during the reconstruction of an HDR frame. We deal with this issue by analyzing the variance of radiance values over the corresponding (aligned) pixels of the images in F^i.

More precisely, for each F^i we reconstruct four HDR images, one for each F^i_j and one considering all the images in the triple. We call them $\hat{F}^i_j$, for $j = 1, 2, 3, 4$, in the mentioned order. Consider we are reconstructing the HDR frame for $\hat{F}^i_1$. Let us pick some pixel p in $\hat{F}^i_1$ and call r_1 its radiance value. Let r_j be the radiance values of the three corresponding pixels across images $\hat{F}^i_j$, $j = 2, 3, 4$. We define σ as the variance of the set $\{r_j\}_{j=1,2,3}$. For the final

[1]We chose 3, but any reasonably small number larger than 1 could be used.

$\hat{F}_1^i$, the radiance value for the pixel p will be set to a convex combination between r_1 and r_4 depending on the magnitude of σ: if the variance is high, some movement must be happening in this pixel across frames, so we weight the value of the radiance towards r_1; and vice versa if the variance is low. This procedure is repeated for $\hat{F}_2^i$ and $\hat{F}_3^i$.

4. Mobile Device Implementation

As discussed in [6], the lack of a fully programmable, portable camera is a problem for Computational Photography researchers. In this sense, the development of the FCam API [1] turns out to be a great step towards a new generation of mobile devices. With the FCam API, it is possible to have full control of the camera parameters, such as shutter speed, gain and focus. We may even change its algorithms, such as autoexposure, demosaicking and autofocus. In this work, we used a Nokia N900 running Maemo 5 (Open Source Linux distribution). We developed a HDR video application which allows the motion of both objects on the scene and the camera. Figure 1 illustrates an outline of the application.

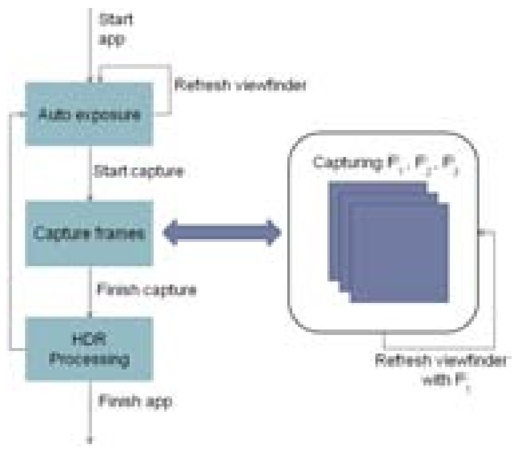

Figure 1. Application outline.

4.1. Capturing HDR Video

The major challenge concerning the capture and processing of HDR frames on a mobile device is to maintain a fine balance between frame rate, memory usage and processing power. On one hand, there is a goal to capture at least 25 frames per second to produce good quality video, and this rapidly consumes the device's memory. On the other hand, the application should process and save all the captured frames, and this stage is a little slower than the first, due to limited processing power and the fact that writing a file on disk requires more time than capturing a photo and saving it in the application memory. Although it is possible to use the method described in Section 3 with as many different exposures as desired, the current implementation only uses three different exposure settings. This provides good results without harming this balance. We limit our application to capture only short videos due to these memory limitations. It is a reasonable restriction: mobile devices usually are not designed to capture long duration videos.

The capture stage works as follows: while the user looks through the viewfinder (without recording), the application performs an autoexposure algorithm, corrects the white balance and displays a preview of the scene using the current settings. When the capture starts, the autoexposure algorithm is executed in the background, providing the optimal camera settings for a good exposure during the video capture process. The first shot uses the optimal computed exposure, the second and the third shots use twice and half of this value, respectively. This process is repeated until the video is fully recorded and the frames are sent to the processing stage (see Fig. 1). All shots use the same gain (which guarantees that they will have the same camera response function).

4.2. Processing HDR video

The processing of the captured frames is done after all the frames were captured. This is a way to guarantee that there is no slowdown during the capture process. Since this stage is done independently of the first, virtually any HDR method could be used here. We chose the method described in Section 3 because of its flexibility due to histogram analysis (instead of direct pixel-to-pixel correspondence). This stage could be performed on the mobile device, on another computer, or even on a cloud.

Our current implementation performs this step on a desktop computer for testing purposes; however, we are currently working on an application which will perform the full outline given in Fig. 1 on the camera itself (instead of only the capture process). This stage of implementation is straightforward, since the same code can run on both machines.

5. Results and Discussion

As was mentioned in Section 4, in the current implementation stage the algorithm being used on the Nokia N900 returns a sequence of images with varying exposure times. We then proceed by transferring the results to a desktop computer in order to process the data. This is done by applying the algorithm described in Section 3 to the images. After this step, we have a set of HDR images, which correspond to each frame of the captured video. In order to be able to visualize our results on regular LDR devices, a tone-mapping algorithm is also necessary. This step is done by using the `pfstmo` library, from Max Plank Institut[2]. After

[2]Available at http://www.mpi-inf.mpg.de/resources/tmo/

testing several tone-mapping methods, we have decided to use an implementation of [3] present in the library, due to both its speed and the quality of the results obtained. The final resulting tone-mapped output, along with a sequence of three differently exposed frames generated by our program can be seen on Figures 2 and 3.

Figure 2 shows the results obtained in an outdoor scene, with predominant camera movement. Notice that the background (resp., the building interior) is only well-exposed on the second (resp., third) captured frame. Both areas of the image are well show in the three tone-mapped results.

Figure 3 shows the results obtained capturing an indoor scene, with object and camera movement. Notice the movement between the captured frames.

6. Conclusions and Future Work

In its current form, the algorithm works as follows: firstly, the Nokia N900 with an FCam API is used to capture a sequence of frames with varying exposures. These frames are used on a desktop computer to generate a series of HDR images, which are then converted to regular LDR images through a tone-mapping algorithm.

One simple improvement we are currently working on is making the whole process run on the Nokia N900, without the use of any external hardware. This step is facilitated by the fact that the algorithm detailed in Section 3 is lighter than usual algorithms for HDR video, and thus should not require as much of the camera-phone's processor.

Another improvement might come from other means of pixel correspondence for object movement on the scene, such as optical flow, to further enhance the quality of the results. Also, other Tone Enhancement techniques could be used to improve the quality of the captured videos.

A more difficult challenge for future works involves finding a better way to deal with the device's small memory, in order to increase the amount of frames that can be captured. A possible partial solution lies in creating a low priority thread that would save images to the device's hard disk while the program is still running.

Finally, we believe that the use of a fully programmable camera brings many more possibilities besides the ones that have been explored here. Many ideas involving user-aided capture and processing, or the retrieval of geometric proprieties of objects through intelligent capture processes, can be achieved by having access to low level hardware parameters, as was done in this work.

References

[1] A. Adams, E.-V. Talvala, S. H. Park, D. E. Jacobs, B. Ajdin, N. Gelfand, J. Dolson, D. Vaquero, J. Baek, M. Tico, H. P. A. Lensch, W. Matusik, K. Pulli, M. Horowitz, and M. Levoy. The frankencamera: an experimental platform for computational photography. *ACM Trans. Graph.*, 29(4):1–12, 2010. 2, 3

[2] P. E. Debevec and J. Malik. Recovering high dynamic range radiance maps from photographs. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 369–378, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co. 1

[3] F. Drago, K. Myszkowski, T. Annen, and N. Chiba. Adaptive logarithmic mapping for displaying high contrast scenes. *Computer Graphics Forum*, 22:419–426, 2003. 4

[4] M. Grossberg and S. Nayar. Determining the Camera Response from Images: What is Knowable? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(11):1455–1467, Nov 2003. 1

[5] S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski. High dynamic range video. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, pages 319–325, New York, NY, USA, 2003. ACM. 1

[6] M. Levoy. Experimental platforms for computational photography. *IEEE Computer Graphics and Applications*, 30:81–87, 2010. 3

[7] P.-Y. Lu, T.-H. Huang, M.-S. Wu, Y.-T. Cheng, and Y.-Y. Chuang. High dynamic range image reconstruction from hand-held cameras. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 0:509–516, 2009. 2

[8] T.-H. Min, R.-H. Park, and S. Chang. Histogram based ghost removal in high dynamic range images. In *ICME'09: Proceedings of the 2009 IEEE international conference on Multimedia and Expo*, pages 530–533, Piscataway, NJ, USA, 2009. IEEE Press. 2

[9] T. Mitsunaga and S. Nayar. Radiometric Self Calibration. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 374–380, Jun 1999. 1, 2

[10] K. Myszkowski. *High Dynamic Range Video*. Morgan and Claypool Publishers, 2008. 1

[11] M. Robertson, S. Borman, and R. Stevenson. Dynamic range improvement through multiple exposures. In *Proceedings of the IEEE International Conference on Image Processing*, volume 3, pages 159–163, Kobe, Japan, Oct. 1999. IEEE. 1

[12] A. M. Sa. *High Dynamic Range Image Reconstruction*. Morgan & Claypool Publishers, 2008. 1

[13] L. Velho. Histogram-based hdr video. In *SIGGRAPH '07: ACM SIGGRAPH 2007 posters*, page 62, New York, NY, USA, 2007. ACM. 1

[14] G. Ward. Fast, robust image registration for compositing high dynamic range photographs from handheld exposures. *JOURNAL OF GRAPHICS TOOLS*, 8:17–30, 2003. 1, 2

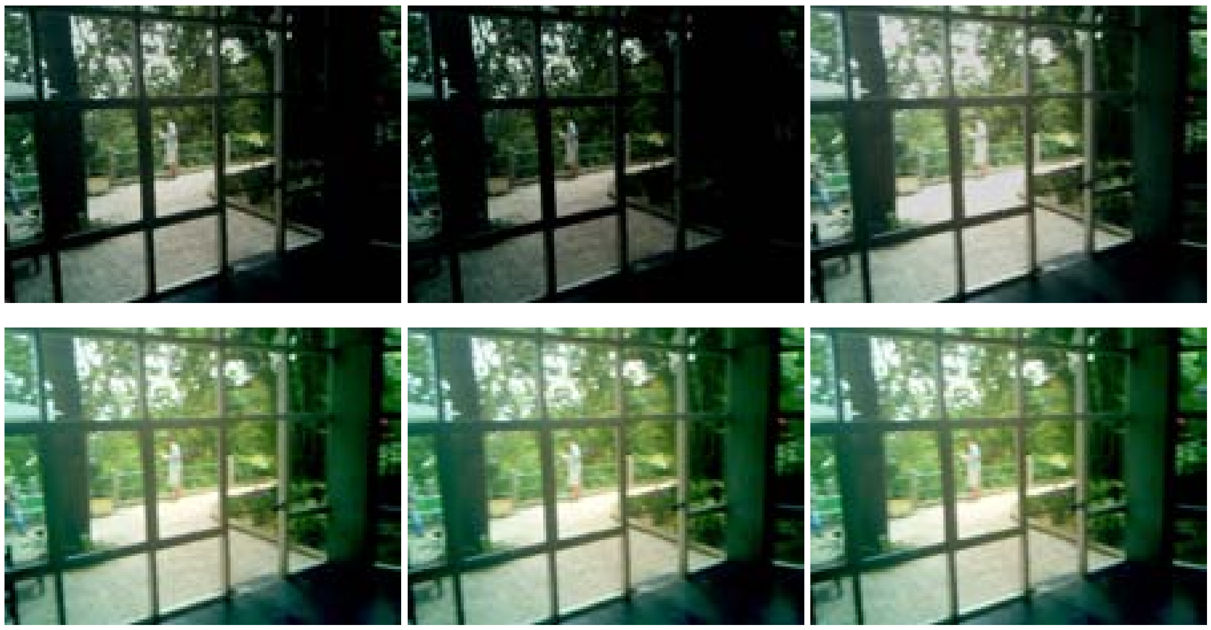

Figure 2. (Top) Captured frames. From left to right: optimally-, sub- and super-exposed shots. (Bottom) Corresponding results with tone-mapping.

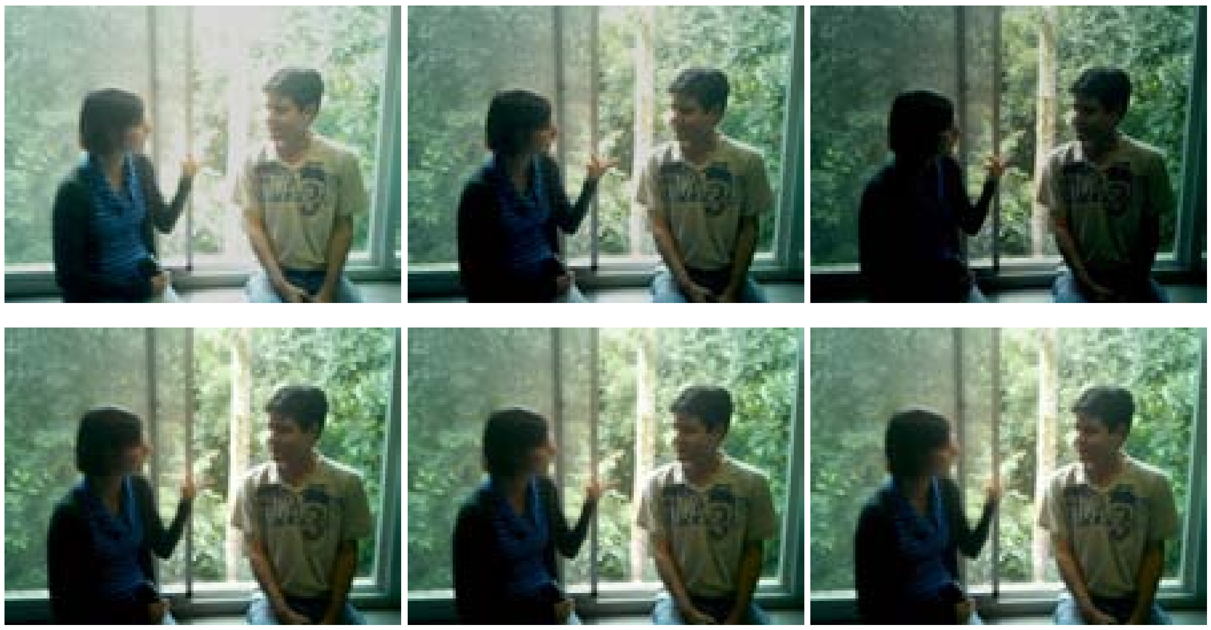

Figure 3. (Top) Captured frames. From left to right: sub-, normal- and super-exposed shots, according to initial calibration on a brighter scene. (Bottom) Corresponding results with tone-mapping.