58th ILMENAU SCIENTIFIC COLLOQUIUM
Technische Universität Ilmenau, 08 – 12 September 2014
URN: urn:nbn:de:gbv:ilm1-2014iwk:3

ACCELERATION OF IMAGE RESTORATION ALGORITHMS FOR DYNAMIC MEASUREMENTS IN COORDINATE METROLOGY BY USING OPENCV GPU FRAMEWORK

Silvio Holder[1], Gerhard Linß[1]

[1]Faculty of Mechanical Engineering
Department of Quality Assurance and Industrial Image Processing
Technische Universität Ilmenau

ABSTRACT

This paper presents an approach for decreasing the processing time of dynamic image restoration for metrological applications. An extension of the Richardson-Lucy deconvolution algorithm for an optimal breaking of the restoration with respect to the change of the sub pixel-precise edge position is proposed. Therefore a powerful parallelization of the restoration algorithms was tested on graphics processing unit (GPU) by using OpenCV C++ framework. A comparison to a processing on multi-core central processing unit (CPU) with a variation of input data size and restoration calculation amount is performed. The experimental results reveal the optimal break-even point for an time efficient processing of the algorithms on GPU.

Index Terms - dynamic measurement, coordinate metrology, image restoration, graphics processing unit, deconvolution

1. INTRODUCTION

Dynamic measurements in mean of dynamic imaging at coordinate measuring machine (CMM) are degraded by the relative motion effects at image exposure time, called motion blur. Caused by motion blur the measurement uncertainties for estimation of sub pixel-precise edge positions (SPEP) are also degraded [1]. Our development deals with the restoration of this motion caused influences by deconvolution algorithms, which are primarily based on the famous Richardson-Lucy-algorithm [5], which is in our approach extended by image measures to break the iterative restoration process on an adequate point, see figure 1. Two main issues in consideration of dynamic image restoration for metrological applications are relevant:

a) accuracy of the restoration regarding to SPEP
b) restoration processing time

For decreasing the restoration calculation time amount a powerful parallelization on graphics processing unit (GPU) using Nvidia CUDA (Compute Unified Device Architecture) technology by the utilization of the C++ high-level programming of the OpenCV framework [6] is proposed in this approach. The calculation on GPU requires the transfer of the image memory data from main RAM (random access memory) to graphic cards RAM and back after finishing the restoration. These both transfers increase the total processing time and are the bottleneck in the processing chain. Regarding to this memory transfers there is a break-even point for an efficient working on GPU devices depending on the ratio of image data size and the amount of parallelizable calculations. This point is determined in the experimental results within this paper by the comparison to the processing on central processing unit (CPU).

2. DYNAMIC IMAGE FORMATION AND RESTORATION BASICS

The restoration of motion blurred images bases on modeling the imaging process with elements from linear system theory. The ideal image, described as $f(x, y)$, is transformed into the degraded image $g(x, y)$ by the convolution $(*)$ with the systems transfer function $h(x, y)$ and superposition of additional noise $n(x, y)$, see formula 1 and [3], [4]. The systems transfer function $h(x, y)$ is a superposition of different single influences, e.g. the optical transfer function $h_{optic}(x, y)$ of the optical system and in case of dynamic image acquisition also a motion blur kernel $h_{motion}(x, y)$, see formula 2.

$$g(x, y) = h(x, y) * f(x, y) + n(x, y) \qquad \textit{(formula 1)}$$

$$h(x, y) = h_{optic}(x, y) + h_{motion}(x, y) \qquad \textit{(formula 2)}$$

The motion blur kernel $h_{motion}(x, y)$ is estimated by analyzing the intensity edge transitions of the measuring scenes region-of-interest (ROI), which consists of search lines in motion direction, using an edge width estimation procedure as described in [2].

3. IMAGE RESTORATION USING AN EXTENDED RICHARDSON-LUCY ALGORITHM

The restoration of motion-blurred images in our approach is based on a famous non-blind deconvolution algorithm called Richardson-Lucy-algorithm (RLA), which computes ideal image approximation $\hat{f}(x, y)$ in an iterative process. Inputs for the RLA are the degraded image $g(x, y)$ and the pre-estimated transfer function $h(x, y)$. The original RLA is aborted on a predefined number of loops. Our algorithm is additionally extended by image analyses to break the RLA process on an adequate time. Therefore the image edges are analyzed in each iteration in regard to the SPEPs and the edge widths, which are important for the final geometric measurement accuracy. Figure 1 depicts the restoration process of the extended RLA. The RLA is stopped by a-priori breaking condition thresholds for the edge width of the intensity transition $x_{w,threshold}$ and the SPEP distance d_i between the current (i) and last iteration loop (i-1). The same extension is also applicable on a state-of-the-art acceleration of the RLA, which is published in [7]. The RLA calculation is mainly processed by convolutions, which are done in frequency domain to simplify the convolution operation to a simple multiplication. Therefore it is necessary to convert the data from spatial domain to frequency domain by fast fourier transformation (FFT) and convert the results back to spatial domain by the inverse fast fourier transformation (IFFT).

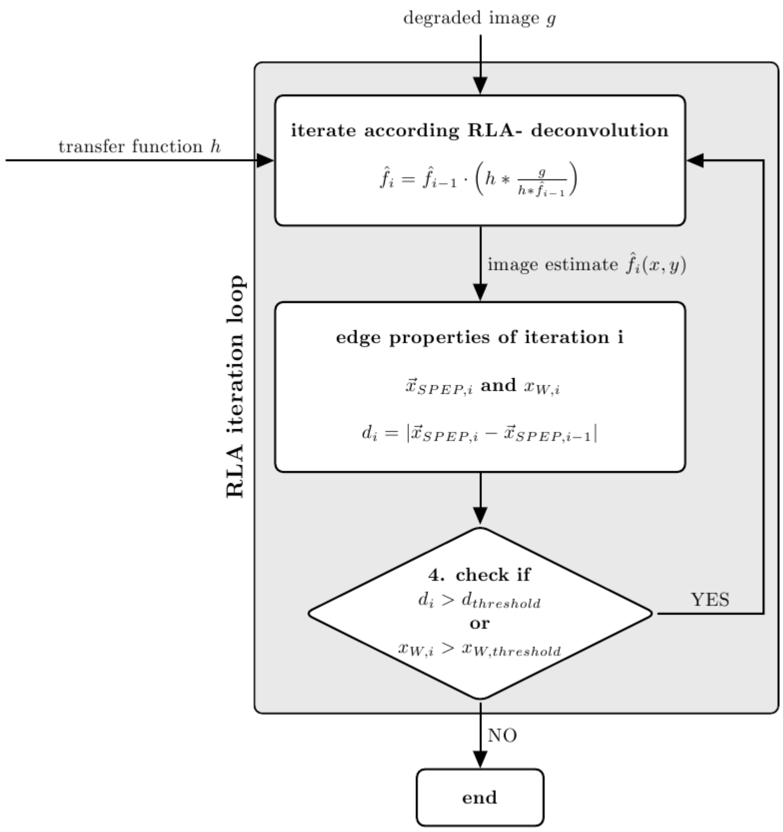

figure 1: extended Richardson-Lucy algorithm

3. PARALLELIZATION OF DYNAMIC IMAGE RESTORATION

The OpenCV framework delivers possibilities for working on GPU by high-level C++ programming. Therefore OpenCV has own matrix data types, pixel-wise and matrix-wise calculation methods on multi-core CPU and highly parallelized on GPU devices. It also delivers FFT and IFFT processing functions, which are main operations for restoration with RLA. For the calculation on GPU, additional steps before the main processing are necessary. The image data has to be transferred to an allocated memory on the GPU device over the common bus interface (GPU upload). The processed data is transferred back to the main memory after processing on GPU (GPU download), see figure 2.

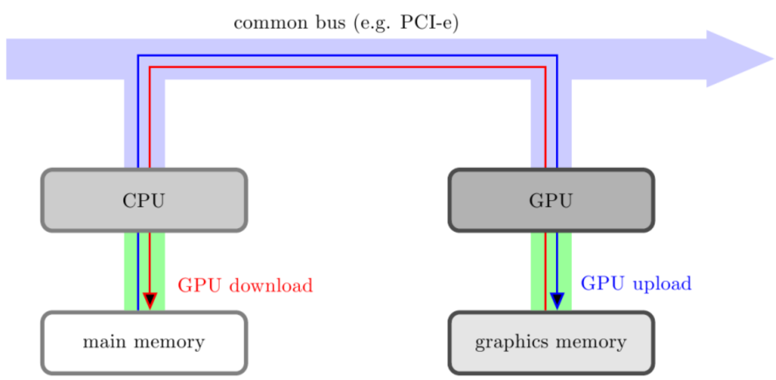

figure 2: schematic structure of device-memory interfaces

The convolution operations (figure 1) are processed by the FFT and IFFT conversions of the data and pixel-pixel-multiplications in frequency domain, this are strong parallelizable tasks. The GPU device is predestined for this independent operations. The bottleneck on restoration using GPU is the data transfer from main to graphics memory and back.

4. EXPERIMENTAL RESULTS

An efficient calculation on GPU is depending on the ratio of data transfer and repeated calculations to minimize to memory transfer bottleneck. Therefore the processing times on CPU vs. GPU of the restoration processing were analyzed and compared. The following setup was used for our tests, see table 1.

table 1: hardware and software setup for the tests

hardware setup	
CPU	Intel Core i5 - 3210M 2.5GHz, 4 Cores
Main RAM	8GB, DDR3
GPU	NVIDIA Geforce GT640M, Intel HD Graphics 4000 chipset (optional)
Graphics RAM	2GB, DDR3, 128bit interface
CUDA compute capability	3.0
software setup	
operating system	MS Windows 7 Professional x64
compiler	MS Visual Studio 2010 x64
CUDA version	Cuda 5.5.20
OpenCV version	2.4.6.0 x64, compiled with Intel TBB 4.1, Eigen-library

The RLA, an accelerated version of RLA [7] and the extended versions (figure 1) of them are implemented for CPU and GPU in our test software (see figure 3) using the software frameworks and compiler setup as depicted in
table 1. The accelerated RLA [7] decreases the necessary iteration loops compared to standard RLA to a ratio of 1:5 for reaching nearly the same restoration level.

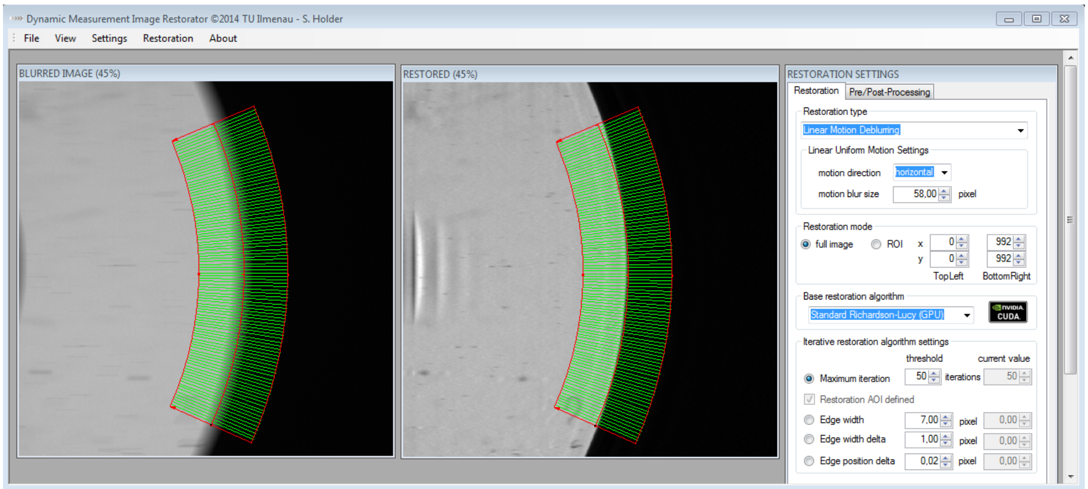

figure 3: test software "Dynamic Measurement Image Restorator" (DMIR) with degraded (left) and restored (right) sample image

In the tests a variation of the image size and a variation of the iteration loops of RLA was performed. Every processing for each configuration was repeated 20 times to evaluate statistical mean values.

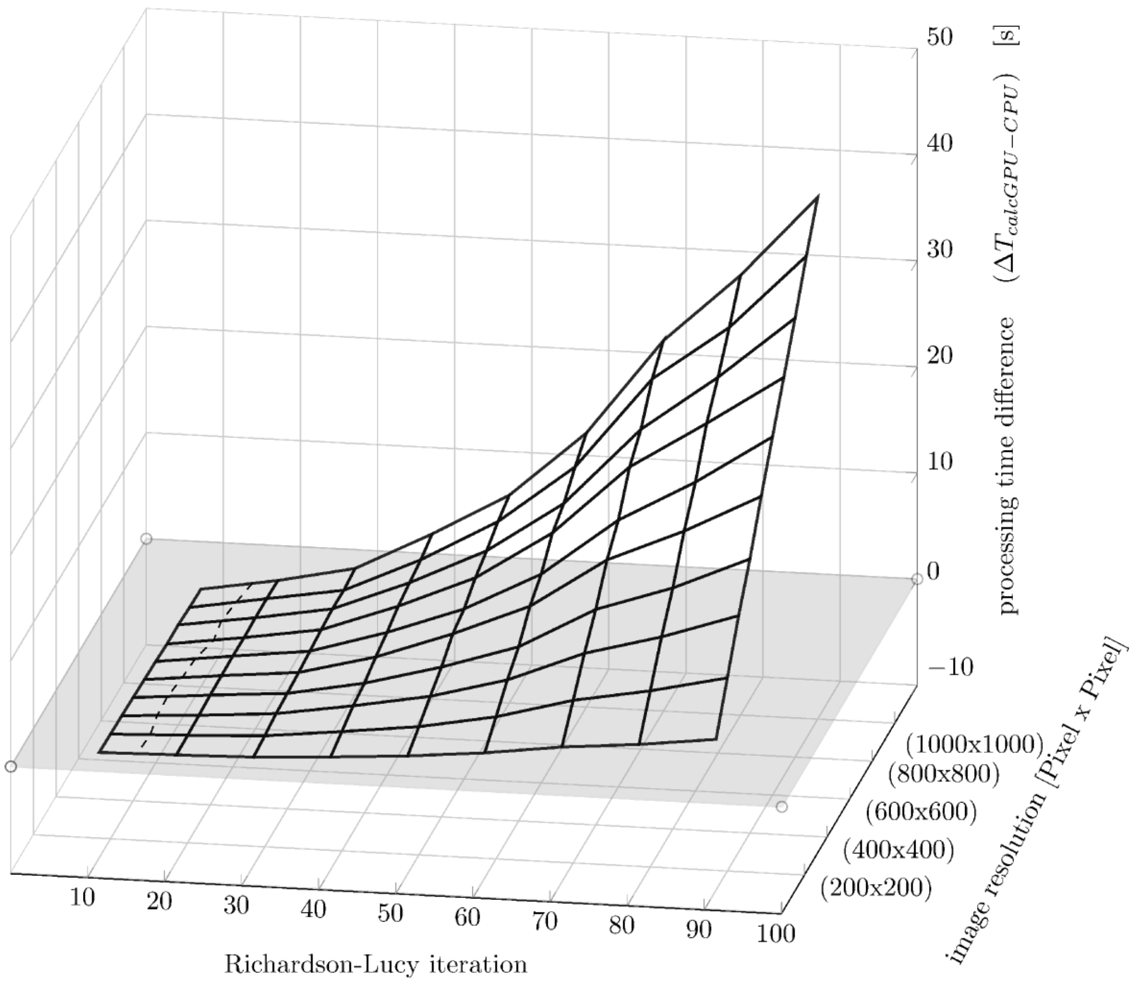

figure 4: illustration of processing time difference $\Delta T_{calcGPU-CPU}$ as a function of iteration loop and image resolution for standard RLA

The illustration of figure 4 depicts the difference time amount $\Delta T_{calcGPU-CPU}$ for the processing on GPU and CPU of our test setup for the standard RLA. The zero plane shows the state where both processings have the same time amount. The dashed curve in the chart characterizes the break-even point for the efficient working on GPU. It is nearly independent from the image size and the amount of iterations. The processing time saving is immense at increasing image sizes and rising number of iteration loops. Similar results appear on the tests for the accelerated RLA, see figure 5. The best time efficient calculation is reached by using of the accelerated RLA on graphics processing unit.

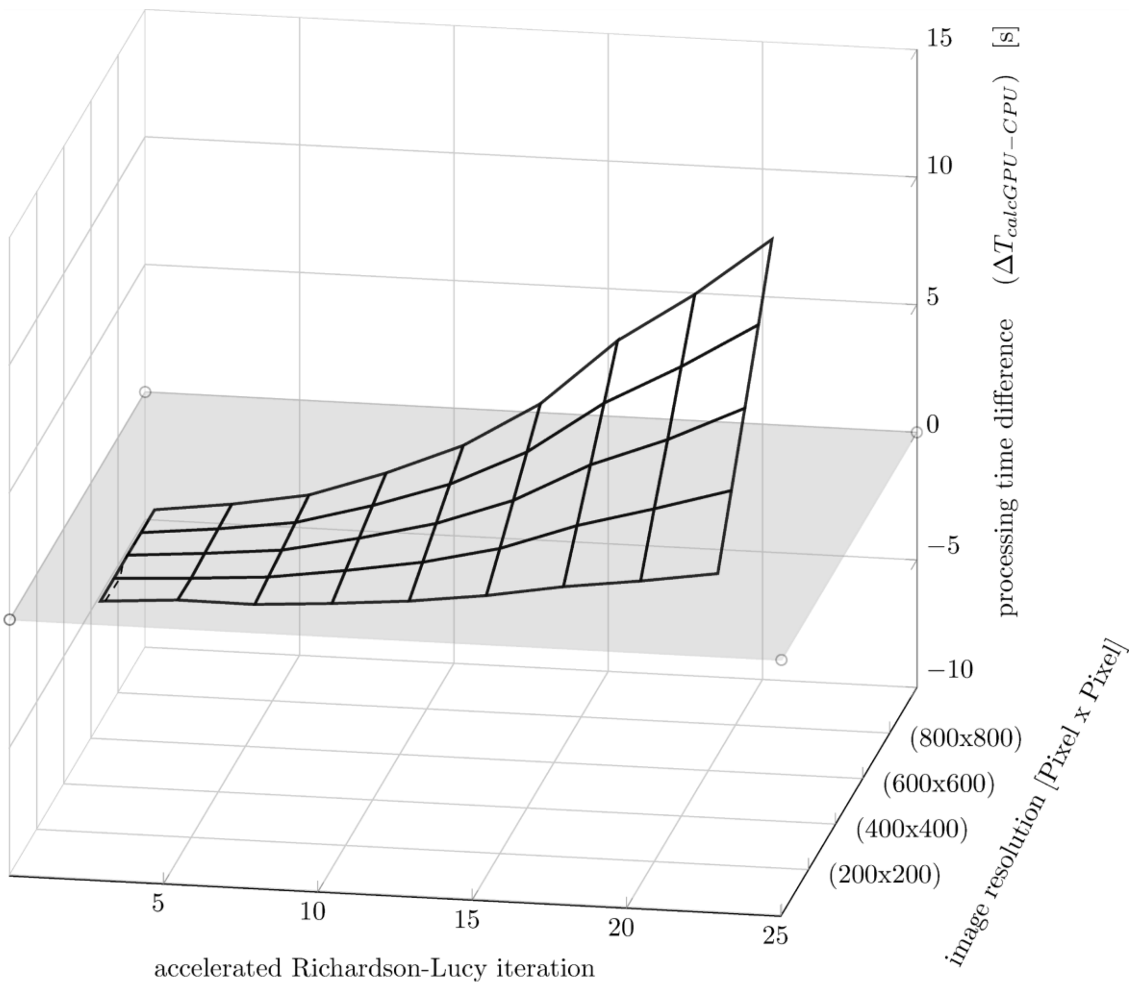

figure 5: illustration of processing time difference $\Delta T_{calcGPU-CPU}$ as a function of iteration loop and image resolution for accelerated RLA[7]

5. CONCLUSION

We presented within this paper a possibility to accelerate dynamic image restoration based on the famous Richardson-Lucy algorithm by utilization of graphics processing units via OpenCV C++ framework. Depending on the image size and the level of restoration, the break-even point for processing on GPU was estimated by experimental tests on a high-performance computer setup with multi-core CPU and CUDA GPU device. The experimental test revealed a general time advantage for processing dynamic image restoration on GPU. The memory transfer bottleneck is comparatively small to the amount of parallelizable operations. An acceleration by the parallelization on the graphics processing unit is recommended in case of image restoration by Richardson-Lucy methods.

REFERENCES

[1] Weißensee, K.; Holder, S.: Uncertainty estimation of image-based measurements affected by motion-blur, In: Innovation in mechanical engineering - shaping the future : programme; 56th IWK, International Scientific Colloquium, Ilmenau University of Technology, 2011

[2] Holder, S. et al: Method for a robust search line based estimation of intensity edge width in blurred gray scale images for quantification of motion- and out-of-focus-blur, In: Innovation in mechanical engineering - shaping the future : programme; 56th IWK, International Scientific Colloquium, Ilmenau University of Technology, 2011

[3] Gonzales, R.; Woods, R.: Digital image processing, Pearson Prentice Hall, New Jersey, 2008

[4] Bovik, A.: The essential guide to image processing - 2nd Ed., Elsevier Academic Press, New York, 2008

[5] Molina, R. et al: Signal Processing Magazine, IEEE 18 11–29, 2001

[6] OpenCV documentation: URL: http://opencv.org/documentation.html, 20.07.2014

[7] Biggs, D.S.C.; Andrews, M.: Acceleration of Iterative Image Restorations Algorithms, Applied Optics, Vol. 36 No. 8, 1997

CONTACTS

Silvio Holder silvio.holder@tu-ilmenau.de
Univ.-Prof. Dr.-Ing. habil. Gerhard Linß gerhard.linß@tu-ilmenau.de