# Improving Nanoobject Detection in Optical Biosensor Data

**Constantin TIMM**
**Department of Computer Science XII, Technical University of Dortmund, Otto-Hahn-Str. 16**
**Dortmund, 44227, Germany**

**Pascal LIBUSCHEWSKI**
**Department of Computer Science VII, Technical University of Dortmund, Otto-Hahn-Str. 16**
**Dortmund, 44227, Germany**

**Dominic SIEDHOFF**
**Department of Computer Science VII, Technical University of Dortmund, Otto-Hahn-Str. 16**
**Dortmund, 44227, Germany**

**Frank WEICHERT**
**Department of Computer Science VII, Technical University of Dortmund, Otto-Hahn-Str. 16**
**Dortmund, 44227, Germany**

**Heinrich MÜLLER**
**Department of Computer Science VII, Technical University of Dortmund, Otto-Hahn-Str. 16**
**Dortmund, 44227, Germany**

**and**

**Peter MARWEDEL**
**Department of Computer Science XII, Technical University of Dortmund, Otto-Hahn-Str. 16**
**Dortmund, 44227, Germany**

## ABSTRACT

The importance of real-time capable mobile biosensors increases in face of rising numbers of global virus epidemics. Such biosensors can be used for on-site diagnosis, e.g. at airports, to prevent further spread of virus-transmitted diseases, by answering the question whether or not a sample contains a certain virus. In-depth laboratory analysis might furthermore demand for measurements of the concentration of virus particles in a sample. The novel PAMONO sensor technique allows for accomplishing both tasks. One of its basic prerequisites is an efficient analysis of the biosensor image data by means of digital image processing and classification. In this study, we present a high performance approach to this analysis: The diagnosis whether a virus occurs in the sample can be carried out in real-time with high accuracy. An estimate of the concentration can be obtained in real-time as well, if that concentration is not too high.

The contribution of this work is an optimization of our processing pipeline used for PAMONO sensor data analysis. The following objectives are optimized: detection-quality, speed and consumption of resources (e.g. energy, memory). Thus our approach respects the constraints imposed by medical applicability, as well as the constraints on resource consumption arising in embedded systems. The parameters to be optimized are descriptive (virus appearance parameters) and hardware-related (design space exploration).

## Keywords

Biosensor, GPGPU, PAMONO, Design Space Exploration, Energy Awareness

## 1 INTRODUCTION

Advances in optical microscopy enable detection of viruses which can then be used for a rapid and distributed epidemic infection control. A novel technique which can achieve the latter is called PAMONO (Plasmon assisted Microscopy of Nano-Size Objects) [8, 10]. It provides the possibility to selectively detect different type of nanoobjects, especially viruses, in case the particles could be immobilized on the sensor surface. Virus types could be distinguished by applying different antigenes or antibodies to the gold layer. PAMONO not only enables an on-site detection of a small amount of viruses but also a detection in real-time which means that the result of a detection in progress can be visualized online while inserting the specimen. The PAMONO sensor unit produces a videostream, which is 1000 × 500 pixels in size and has a framerate of 30 frames per second. On-site application of mobile biosensors demands to take resource constraints into account, as sensor data analysis has to be carried out in embedded systems. On the other hand real-time analysis demands for high processing power. Thus a GPGPU (General Purpose computing on Graphics Processing Unit) approach was taken.

Despite an enormous amount of GPGPU application papers, in particular, related work in the context of design

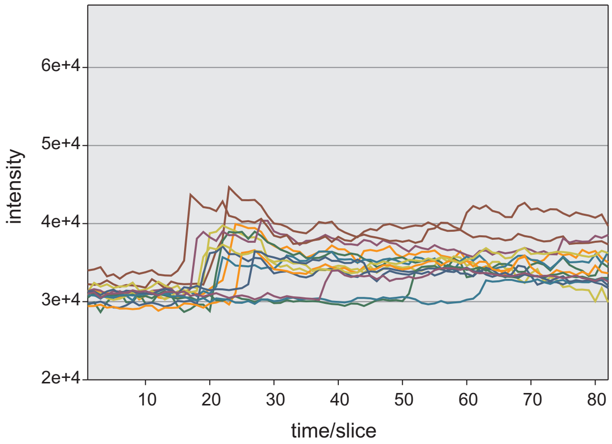

Figure 1: Intensity over time for different pixel positions

space exploration is important. Especially in the design process in the field of embedded systems a design space exploration can be done at design time to get an optimal system configuration. In this domain, many different approaches [1, 3, 6, 9] exist which provide the capability to automatically explore a design space with objectives such as energy, performance and heat dissipation. The design space exploration in this work takes the detection quality, performance and energy into account.

This work is structured as follows: Section 2 describes a parameterized processing pipeline for the detection of viruses in PAMONO sensor data, followed by a genetic optimization of the descriptive parameters for virus adhesions. Section 3 presents design-time optimization of the consumption of resources by performing a design-space exploration. Finally, sections 4 and 5 provide results and discussion.

## 2 GPGPU-based Analysis of PAMONO Sensor Data

Our approach is divided into two basic steps. First, the detection of the viruses and second, the optimization of the parameters for an effective detection.

### Detection of the Viruses

The detection of the viruses is accomplished by analyzing the video-stream produced by the PAMONO sensor using a graphics processing device. For maximizing the detection quality of the analysis we use a generic approach. The detection process is summarized as follows. First the input data is preprocessed by removing the constant background image and applying a Haar-wavelet noise reduction. The noise reduction process is done in a high parallel manner. For input images of size $M \times N$ pixels the filtering is done in $M \cdot N$ independent threads with $4M \cdot N$ global read/write accesses for each frame.

In the second step, each per pixel time series is matched to a variable pattern, resulting in a pre-classification of all pixels in space as virus candidates. Every virus adhesion to the PAMONO sensor produces a small ($< 6\%$) ascending slope of the intensity in the video-stream at the corresponding pixels. Before and after the adhesion the intensity remains constant. To identify these pixels we use pattern matching. Figure 1 shows some selected time series of virus adhesions.

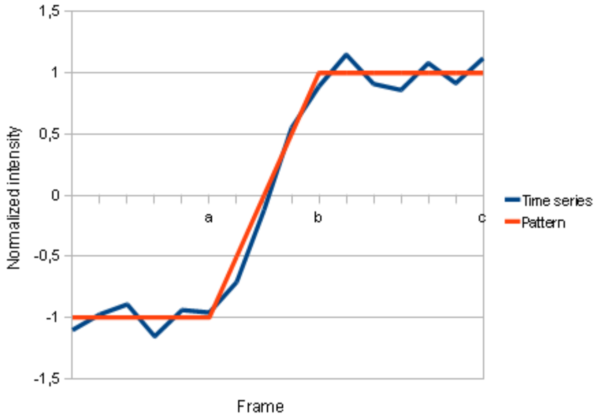

Figure 2: The scaled input time series (blue) is matched to a pattern (red). The pattern is represented by the descriptive parameters $a$, $b$ and $c$.

The pattern (figure 2) is represented by the descriptive parameters $a$, $b$ and $c$. The pattern is $-1$ for the first $a$ values, followed by a linear ascent for the next $b - a$ values and a value of 1 for the last $c - b$. The matching of the pattern to the time series is achieved by scaling the time series to the range of the pattern and then calculating the sum of squared differences (SSD) between these two. If the SSD is below a threshold, the pixel is classified as a virus candidate pixel. The scaled time series is for all types of viruses the same and could be encountered with one pattern. Different types affect the intensity change to some degree, but not the slope of the ascent. The slope can be influenced by how the sensor is functionalized, which could also change the absolute intensity of the ascent.

Each thread on the GPU matches one of the $M \times N$ time series (compare with figure 1).

In the last step the single pixels are combined to polygonal segments, by tracing the borders of the detected pixel areas. The polygons are tracked over time to combine associated polygons. The remaining polygons are then classified as viruses/non-viruses, based on their form factors [4].

All steps, except the tracing of the polygons, are done in time space. Each pixel position is independent from others. This allows the processing of exact one pixel position by each lightweight thread on the GPU without synchronization, which takes the full advantage of the GPU.

The scaling behavior of the algorithms in the steps can be evaluated by increasing the number of cores until the fur-

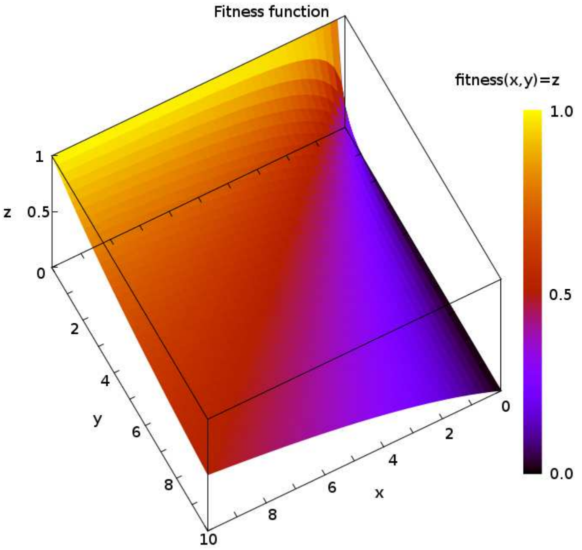

Figure 3: Fitness function with *x* as the true positive detections, *y* the sum of false positive and false negative detections and *z* the resulting fitness value.

ther increase does not lead to a speedup of the pipeline. Especially, in case that a GPGPU application is memory bounded, meaning that it needs a numerous number of main memory transfers, an increasing number of processing cores will not increase the performance when reaching a certain threshold. Furthermore, the performance will not scale with the number of cores if there are too many data dependencies inside the kernels making synchronization necessary. A detail analysis can be found in [7].

### Parameter Optimization

Since the optimal values for the parameters of the pattern matching – *a*, *b*, *c* and the threshold – are unknown, an optimization with a genetic algorithm is employed. For an introduction to genetic algorithms see [5]. The chromosome, which is used by the genetic algorithm to generate a population, is built of four genes. Each gene represents one parameter of the optimization. The genes one to three define the size and shape of the pattern and are given by *a*, *b* and *c* as used for defining the pattern. The fourth gene defines a threshold for the SSD between the observed time series and the pattern.

To determine the fitness of a given population, the detected particles are compared to manually annotated data. The true positives (tp), false positives (fp) and false negatives (fn) are computed by automatically matching the detected polygons with manual segmented data. With these values the accuracy, defined as $\frac{\text{tp+tn}}{\text{tp+fn+fp+tn}}$ [2], is given and selected for the fitness function. As illustrated in figure 3, the fitness function returns a larger value if the true positives outweigh the false classified. The true negatives (tn) are set to zero, because they are not properly defined in this context. A population size of 80 is chosen, the starting population is created with random chromosomes and the fittest individual of each population is propagated to the next generation. If the fitness function reaches the desired accuracy or a specified number of generations is attained, the genetic algorithm returns the best obtained parameters for the pattern matching.

| Datasets | Precision | Recall | Accuracy |
|---|---|---|---|
| 280 nm | 95 % | 90 % | 86 % |
| 200 nm | 87 % | 91 % | 80 % |
| VLPs | 80 % | 86 % | 71 % |

Table 1: Detection results of different datasets, based on true positives, false positives and false negatives.

## 3 DESIGN SPACE EXPLORATION

The process of designing a GPGPU-accelerated device like the data analysis system presented, involves a design space exploration (DSE). DSE optimizes the hardware-related parameters of the system with respect to resource restrictions. Especially the objectives *run time* and *energy consumption* are of concern here. The first objective is important in order to realize a system which is capable of visualizing results of the virus detection in real-time while the latter is crucial for mobile devices. The parameters which are evaluated are e.g. the number of required processing units, the amount of memory used and the mapping on the processing cores.

## 4 RESULTS

The results to be presented in this section comprise the optimization of the detection rate and the design space exploration for different numbers of parallel processing cores.

The experiment set-up was made with a QPhotonics superluminiscent diode QSDM-680-9 and a Kappa DX40-1020FW camera which recorded the images. Different sizes and types of particles are used, amounting to datasets with 280 nm- and 200 nm-sized synthetic particles and virus like particles (VLPs).

The genetic algorithm took about four hours for each dataset with about 22.000 single runs of the pipeline. Due to the parallel GPGPU approach, and caching of the input images, the resulting frame rate exceeds 220 frames per second on a GeForce GTX 480. If the images could be provided faster, the possible throughput of the GPU pipeline is about 2.000 frames per second (images 512 × 50 pixels, scaling to bigger images is linear).

The results for the improvement of the detection quality are shown in table 1, quantifying precision, recall and accuracy, where the latter serves as the fitness function. The detection quality depends on the particle size.

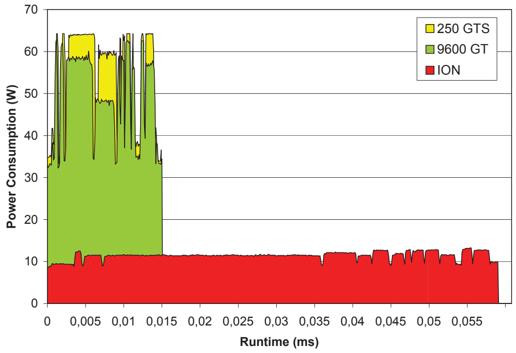

Figure 4: Power Consumption over Application Runtime with Different Processing Cores Number

Three different graphics cards where used for the design space exploration: ION (16 Cores), 9600GT (64 Cores) and 250GTS (128 Cores). As can been seen from figure 4, the runtimes of the image processing and analysis are improving with the number of cores but with a lower factor when scaling from 64 to 128 cores. This indicates that one bottleneck for the speedup is the memory and the second bottleneck is the data dependency of the application. On the other hand, the energy consumption – integral of the power consumption curve over time –, is not improving anymore when switching from 9600GT to 250GTS.

## 5 DISCUSSION

A method to optimize the descriptive and hardware-related parameters of our processing pipeline for PAMONO sensor data was presented. An automated detection of viruses in such data can be carried out in real-time, using GPGPU computing. The detection can be used to distinguish different types of viruses by applying different coatings to the gold layer. Despite the early stage of development, the detection results are promising in terms of accuracy.

Future research in the analysis of PAMONO sensor data aims at an enhancement of the presented processing pipeline, followed by a multi-objective optimization using machine learning techniques. This enables decomposition of the objective of accuracy into precision and recall, as well as a simultaneous optimization of the objectives speed and consumption of resources. Suitable features beyond time series are to be identified that take into account not only the temporal but also the two spatial dimensions of PAMONO data.

Beyond the descriptive and hardware-related parameters covered in this work, an extended pipeline gives rise to a third class of parameters to be optimized: the computational parameters of the processing pipeline, like e.g. choice of algorithms for preprocessing and classification, thresholds, and the features to be computed for matching. These computational parameters affect all objectives: detection quality, speed and consumption of resources. The next step to be taken is optimizing these computational parameters on a given platform. This encompasses learning the descriptive parameters with respect to the chosen features, i.e. learning the characteristic values those features assume for viruses. The goal of this optimization is improving detection quality (precision and recall) while maintaining the real-time capabilities of the current pipeline and minimizing consumption of resources on the given mobile platform. Achieving these goals means developing the foundation for a mobile PAMONO sensor and analysis device.

## Acknowledgment

Part of the work on this paper has been supported by Deutsche Forschungsgemeinschaft (DFG) within the Collaborative Research Center SFB 876 “Providing Information by Resource-Constrained Analysis”.

## References

[1] L. Benini, M. Lombardi, M. Milano, and M. Ruggiero. A constraint programming approach for allocation and scheduling on the cell broadband engine. In P. J. Stuckey, editor, *CP*, volume 5202 of *Lecture Notes in Computer Science*, pages 21–35. Springer, 2008.

[2] T. Fawcett. ROC Graphs: Notes and Practical Considerations for Researchers. Technical report, HP Labs, 2004.

[3] J. Keinert, M. Streubühr, T. Schlichter, J. Falk, J. Gladigau, C. Haubelt, J. Teich, and M. Meredith. Systemcodesigner: an automatic esl synthesis approach by design space exploration and behavioral synthesis for streaming applications. *ACM Trans. Des. Autom. Electron. Syst.*, 14, 2009.

[4] G. Landini. Quantitative analysis of the epithelial lining architecture in radicular cysts and odontogenic keratocysts. *Head & Face Medicine*, 2, 2006.

[5] K. F. Man, K. S. Tang, and S. Kwong. *Genetic algorithms: concepts and designs*. Springer, 2001.

[6] H. Nikolov, M. Thompson, T. Stefanov, A. Pimentel, S. Polstra, R. Bose, C. Zissulescu, and E. Deprettere. Daedalus: Toward composable multimedia mp-soc design, 2008.

[7] C. Timm, F. Weichert, P. Marwedel, and H. Müller. Design space exploration towards a realtime and energy-aware gpgpu-based analysis of biosensor data. *Computer Science - Research and Development, Special Issue ”International Conference on Energy-Aware High Performance Computing (ENA-HPC)”, Springer (accepted for publication)*, 2011. Publikation.

[8] F. Weichert, M. Gaspar, C. Timm, A. Zybin, E. Gurevich, M. Engel, H. Müller, and P. Marwedel. Signal Analysis and Classification for Surface Plasmon Assisted Microscopy of Nanoobjects. *Sensors and Actuators B: Chemical*, 151:281–290, 2010.

[9] E. Zitzler, K. Giannakoglou, D. Tsahalis, J. Periaux, K. Papailiou, T. F. (eds, E. Z. Ler, M. Laumanns, and L. Thiele. Spea2: Improving the strength pareto evolutionary algorithm for multiobjective optimization, 2002.

[10] A. Zybin, Y. Kuritsyn, E. Gurevich, V. Temchura, K. berla, and K. Niemax. Real-time Detection of Single Immobilized Nanoparticles by Surface Plasmon Resonance Imaging. *Plasmonics*, 5:31–35, 2010.