Towards a Methodology for Creating Time-critical, Cloud-based CUDA Applications

Louise Knight*, Polona Štefanič*, Matej Cigale*, Andrew C. Jones* and Ian Taylor*
*School of Computer Science and Informatics, Cardiff University,
Queen's Buildings, 5 The Parade, Roath, Cardiff CF24 3AA, UK
Email: [KnightL2 | StefanicP | CigaleM | JonesAC | TaylorIJ1]@cardiff.ac.uk

***Abstract*—CUDA has been used in many different application domains, not all of which are specifically image processing-related. There is the opportunity to use multiple and/or distributed CUDA resources in cloud facilities such as Amazon Web Services (AWS), in order to obtain enhanced processing power and to satisfy time-critical requirements which cannot be satisfied using a single CUDA resource. In particular, this would provide enhanced ability for processing Big Data, especially in conjunction with distributed file systems (for example). In this paper, we present a survey of time-critical CUDA applications, identifying requirements and concepts that they tend to have in common. In particular, we investigate the terminology used for Quality of Service metrics, and present a taxonomy which summarises the underlying concepts and maps these terms to the diverse terminology used. We also survey typical requirements for developing, deploying and managing such applications. Given these requirements, we consider how the SWITCH platform can in principle support the entire life-cycle of time-critical CUDA application development and cloud deployment, and identify specific extensions which would be needed in order fully to support this particular class of time-critical cloud applications.**

***Index Terms*—Time-critical applications, CUDA, distributed cloud computing**

I. Introduction

CUDA™[1] is a parallel computing platform and programming model that works with NVIDIA GPUs, and it presents the opportunity to perform certain kinds of parallel computation rapidly, on relatively low-cost hardware [1]. In this paper we present a survey of a specific class of CUDA applications – time-critical applications – and the factors that should be considered when developing and deploying them in a distributed cloud architecture.

There are many time-critical applications with different purposes, and there are various approaches and tools that support their design, development and operation. In this survey, we have concentrated only on those time-critical applications that use CUDA. Therefore, the purpose of this study is to:

(i) present various time-critical applications that are implemented using CUDA,
(ii) divide them into groups, such as Environment-related, People/face detection, Medical applications ...,
(iii) analyze the time-critical requirements which are important for the specific groups so that applications can achieve their Quality of Service (QoS) and, based on this analysis, distinguish between the QoS attributes according to their importance (i.e. how critical each attribute is) for each specific group of time-critical CUDA applications,
(iv) discuss how using CUDA can have a positive influence on QoS and consequently on QoE for each specific group of time-critical applications,
(v) discuss the key implications for the SWITCH (Software Workbench for Interactive, Time Critical and Highly self-adaptive Cloud applications)[2] architecture and environment, if such applications are to be fully supported, and
(vi) consider whether the SWITCH architecture could be further refined in order to support CUDA to satisfy the QoS of more critical parts of applications, such as provisioning, verification of various constraints, image processing and so on.

In our survey, we have organized time-critical CUDA applications into four main themes or groups: Environment-related, People/face detection, Medical applications and Materials-related. We also found some studies which did not exactly fit into these four themes, and these are presented under the "Miscellaneous" heading.

It should be noted that many of the applications found during this survey, across a number of the above themes, have an image processing aspect to them, so these four themes are also distinguished according to whether they are image processing applications or not. Throughout this survey, we have found recurring ideas for which differing terminology were used, depending on the application domain in question (for example, there are several different ways of referring to runtime).

The rest of the paper is structured as follows. In Section II we introduce applications relating to each of the themes we have identified, detailing the terminology used within each theme. In Section III we present which attributes have the most influence on an application's general QoS. Section IV briefly describes the SWITCH project and its subsystems and finally, Section V concludes the paper by summarizing our findings and proposing further research directions towards time-critical applications using CUDA.

[1] http://www.nvidia.co.uk/object/cuda-parallel-computing-uk.html

[2] http://www.switchproject.eu/

TABLE I
QoS Parameters, Units Used, and Alternative Terms for Environment-related Applications

Parameter	Units	Other names
Runtime	Time units	Convergence time, execution time, processing speed, processing time
Processing rate	ms/pixel	
Accuracy	Percentage	Classification accuracy
Correlation coefficient (accuracy)		
Average absolute difference (accuracy)		

TABLE II
QoS Parameters, Units Used, and Alternative Terms for People/Face Detection Applications

Parameter	Units	Other names
Runtime	Time units	Computation time, detection time
Runtime	Frames per second (fps)	
Processing speed	Frames per second (fps), pixels per second	Detection speed, tracking rate
Accuracy	Percentage	Detection accuracy
Sensitivity	Percentage	Detection rate
False positives count		

II. Time-critical CUDA applications

In this section we discuss those applications found in our survey of time-critical applications implemented using CUDA. We have placed these applications into four categories: Environment-related, People/face detection, Medical applications, and Materials-related applications. We have a fifth, Miscellaneous, category for those applications that do not fit cleanly into any of the previously-listed four.

A. Environment-related

Applications relating to the environment mostly relate to disaster management. As such, the time-critical nature of these applications relates to the fact that disasters such as wildfires, floods, and earthquakes (for example) require immediate response. Wu et al. [2] used the Logistic Regression via variable Splitting and Augmented Lagrangian (LORSAL) algorithm to perform hyperspectral image classification [3]; alongside environmental applications they also suggested the method could be used for military reconnaissance. They measured the execution time (time units) and classification accuracy (percentage) of the method. Goodman et al. [4] used a method derived from that of Lee et al. [5], [6], [7] to analyze marine imaging spectroscopy data. The QoS-related parameters are processing time (time units), 'processing speed' (which appears to be the same as runtime) (time units), processing rate (ms/pixel), and accuracy (correlation coefficient; average absolute difference).

Kurte and Durbha [8] integrated two clustering algorithms, Partition Around Medoids (PAM) [9] and Clustering for Large Application (CLARA) [9] in order to process high-resolution satellite images. The convergence time (time units) was measured. Related to this, Bhangale and Durbha [10] used the Scale Invariant Feature Transform (SIFT) algorithm [11], [12] to extract features from high-resolution satellite images. The runtime (time units) was measured. Here we see an obvious instance where differing terminology is used. The convergence time and runtime are essentially the same; they are both the elapsed time until we find our final solution.

Apart from the previously-discussed papers which involve image processing, there were two we found which did not. Christgau et al. [13] implemented the tsunami simulation algorithm EasyWave [14] in parallel, and simply recorded the runtime (time units) of the implementation. Huang et al. [15] used the Kalman filter method [16] to estimate hidden states in linear dynamic systems. Apart from allowing weather forecasting, this method can also be used to track other systems, for example, military and financial, and to aid in robotic navigation. They just measured the runtime (time units).

Table I shows the terms used in environment applications. There is a strong focus on the runtime, which is unsurprising considering these are time-critical applications (this is a theme we see through the rest of the research areas), but also accuracy. In fact, it is the image analysis applications that are measuring the accuracy of the results; neither of the non-image analysis applications measured accuracy or results quality-related metrics. It could be argued that this is due to only selecting two papers that satisfy the criteria of being about the environment and non-image analysis, but this is actually a trend we see throughout this survey. The root of this is that measuring the accuracy of images is relatively straightforward compared to measuring the accuracy of other applications.

B. People/face detection

Examples of applications using methods for people/face detection include surveillance systems, virtual reality, video conferencing, pedestrian detection, etc. Sharma et al. [17] used two algorithms: a method based on Adaboost for face detection [18], and the method of Thota et al. [19] for face tracking. They recorded the detection speed (fps, pixels/s), tracking rate (fps), detection accuracy (percentage), and number of false positives (count). Herout et al. [20] used WaldBoost [21], derived from Adaboost for face detection. They just tracked the detection time (time units). Fauske et al. [22] compared three background subtraction methods for use with video streams: Codebook [23], Self-Organizing Background Subtraction (SOBS) [24], [25], and Horprasert et al.'s statistical approach [26]. They recorded the runtime in fps and time units. Weimer et al. [27] proposed their own method for pedestrian detection and tracking with applications in road safety. They measured the computation time (time units), processing speed (fps), and detection rate (percentage).

Table II shows the terms used in people/face detection applications. This category includes image analysis applications only, and, again, focuses on runtime and various measures of accuracy (the normal statistical definition of 'accuracy',

TABLE III
QoS Parameters, Units Used, and Alternative Terms for Medical Applications

Parameter	Units	Other names
Runtime	Time units	Execution time, processing time, reconstruction time, registration time
Processing speed	Projections per second (pps)	Reconstruction speed
Peak signal-to-noise ratio	dB	
Mean-square-error		
Throughput	GiB/s	
Throughput	Gflops/s	

sensitivity, and count of false positives) predominantly, but also looks at processing speed in terms of how much work we can get done per unit time (frames per second/pixels per second).

C. Medical applications

The field of medical applications is very broad; here we have focused on a small number of specific applications that we could find. These are: image registration, image reconstruction, and brain-computer interface feature extraction.

Muyan-Özçelik et al. [28] used the Demons algorithm [29] to perform image registration, which involves aligning two images and has many applications, such as image-guided surgery. They measured the runtime (time units), the instruction throughput (GFLOP/s), and the data throughput (GiB/s). Modat et al. [30] also performed image registration, but instead using the Free-Form Deformation (FFD) method; they just record the registration time (time units), which seems essentially analogous to the runtime.

Keck et al. [31] used the Simultaneous Algebraic Reconstruction Technique (SART) [32] to perform image reconstruction for CT scanners. They recorded the reconstruction time (time units). Pang et al. [33] also used SART for the same problem, recording the execution time (time units) (essentially the same as reconstruction time), reconstruction speed (projections per second, pps), and some measures of reconstruction quality: peak signal-to-noise ratio (dB), and mean-square-error. Riabkov et al. [34] used a modified FDK algorithm [35] to also perform image reconstruction; they simply recorded the execution time (time units), and briefly mentioned performance-per-watt and performance-per-dollar requirements.

A non-image analysis application is that of Wilson and Williams [36], who implemented a parallelized version of one of the BCI2000 algorithms [37] for brain-computer interface feature extraction; this has applications in implantable technology. They recorded the processing time (time units).

Table III shows the terms used in medical applications. Alongside the runtime, processing speed and results quality metrics (the latter two categories only being considered by image analysis applications), the throughput in GiB/s and Gflops/s is also considered. Although both described here as 'throughput', the measure with its units as GiB/s is probably the most traditionally thought-of as throughput, while the measure with units of Gflops/s is more a measure of how many instructions we can do per unit time.

TABLE IV
QoS Parameters, Units Used, and Alternative Terms for Materials-related Applications

Parameter	Units	Other names
Runtime	Time units	
Runtime	Frames per second (fps)	

TABLE V
QoS Parameters, Units Used, and Alternative Terms for Miscellaneous Applications

Parameter	Units	Other names
Runtime	Time units	Computation speed, computational time, execution time
Time per unit of computation	Time units	Processing time per frame, time per AO time-step
Processing speed	Frames per second (fps)	Processing rate
Communication latency	Time units	
Latency per frame	Time units	
Average absolute difference (accuracy)	Percentage	Percentage of pixels which differed over a given threshold between the ground truth and the results generated
Mean-square-error		Root mean square (RMS) between the ground truth and the results generated
Throughput	GB/s	Bandwidth, data transmission rate
Throughput	Gflops	
Memory access	MB	
Floating point operations count		

D. Materials-related

All of the applications that we found in this survey within this field are not related to image processing. The applications discovered during the survey cover ultrasonic array imaging and a Finite Element method for the analysis of soft tissues. Sutcliffe et al. [38] used a novel method utilizing Fermat's principle to perform ultrasonic array imaging for the identification and classification of defects within solid structures. They recorded the runtime (time units, fps), and briefly mentioned implementation and development costs. Strbac et al. [39] used a Finite Element method [40], [41] to analyse soft tissues; this also has applications in the medical domain. They measured runtime (time units), and mentioned accuracy.

Table IV shows the terms used in materials applications. Only runtime is actually measured in these applications; these applications are both non-image processing.

E. Miscellaneous

This category of methods covers any applications that would not fit into the previous sections.

Havel et al. [42] used PClines [43] to detect lines in raster images; this has applications in circuitry design, and detection of chessboard patterns. They measured the runtime (time units). Wang and Ellerbroek [44] used Multi-threaded Adaptive Optics Simulator (MAOS) [45] for real-time control of Adaptive Optics (AO) systems of large telescopes. They recorded the runtime (time units), time per AO time-step (time units), communication latency (time units), number of floating point operations, throughput (Gflops), memory access (MB), and bandwidth (GB/s). Pagès and Wilbertz [46] used a quantization tree algorithm [47], [48], [49], [50] for a computational finance application to do with the pricing of American style and multiple exercise options. They simply recorded the computational time (time units). Lattanzi et al. [51] used wave field synthesis [52] to artificially produce sound. They recorded the execution time (time units). Mehrabi et al. [53] used Visual Light Communication (VLC) to synchronise multiple video streams; the example use case given was that of having multiple cameras from e.g. smart phones, each filming the same event from a different angle. Only the runtime (time units) was measured. Maillard et al. [54] used several different components to produce an enhanced audience experience during sports events, including: 2D and 3D video streams; overlaying of metadata over the stream concerning, e.g. runner information; data about event location; etc. Each of the architectural components was already available; the project worked to bring all the components together. CUDA was used specifically for depth computation (relating to 3D video); the metrics relating to this specifically were computation speed (time units), the percentage of pixels which differed over a given threshold between the ground truth and the results generated, and the root mean square (RMS) between the ground truth and the results generated. Feldmann et al. [55] developed a system for immersive 3D videoconferencing, implementing in CUDA methods such as lens un-distortion and rectification, for example. They recorded the runtime (time units), processing time per frame (time units), latency per frame (time units), processing rate (frames per second, fps), and data transmission rate (MB/s).

Table V shows the terms used in those applications which do not fit into the earlier categories. Alongside runtime and accuracy-related metrics, we also see measures relating to communication latency, throughput, memory access and the number of floating point operations performed.

III. QoS metrics/attributes that have the most influence for each theme

As should be apparent from the previous sections, many studies, even within the same field, use different terms to mean the same thing. All of the QoS-related parameters are given in Table VI, alongside the alternative terms certain papers used, and the units used to measure such parameters. It should be noted that certain terms in this table, particularly those relating to cost, are not present in the smaller tables already shown.

What we have also noticed is that each theme (environment-related applications, etc.) has a focus on different QoS metrics. We shall now identify the most-used metrics in each theme, and surmise why these are prevalent.

In the field of **environment-related applications**, there is a focus on runtime (as there is for all other types of application, unsurprisingly). Image processing applications within this field look at processing rate and especially the accuracy of the results obtained (for example, using measures such as the correlation coefficient). We believe that there is such a focus on accuracy because applications within this field usually relate to disaster management, such as working out where the most damage is so that we can direct resources to that area. This is such a serious field of applications that it is truly critical that we get the correct answer.

For the field of **people/face detection**, alongside runtime and processing speed, we see, again, a focus on accuracy and its related measures, sensitivity and a count of the false positives. Applications within this field can range from more 'serious' ones (relating to safety), such as surveillance systems in secure environments and pedestrian detection, to the less 'serious', for example, virtual reality. Taking first the example of surveillance systems, this can involve tracking a person across physical space. For example, if we wanted to investigate an incident involving a specific person, we would want to know that that is the same person, so a measure of the accuracy would make sense in this context. This also makes sense in the case of virtual reality applications for video games, for example.

We now consider **medical applications**, a field where we identified some image processing-related applications, and other applications not relating to image processing. Alongside runtime, which was recorded by all applications in this field, we have some accuracy-related measures, such as peak signal-to-noise ratio, and mean-square-error. If we are considering applications relating to intra-operative settings, for example, where the surgeon is responding to information they are seeing while performing an operation, then the need for accuracy is quite obvious. There are also a few throughput-related measures considered, which again, makes sense in that for such a critical application as image-guided surgery, we need to work within the constraints we have.

For **materials-related applications**, the only measurement considered is runtime.

For the **miscellaneous applications**, unsurprisingly there is a wide variety of different QoS measurements used; we have runtime-related measures, accuracy measures, and other parameters relating to throughput, memory access and the number of floating point operations.

IV. Software Workbench for Time-critical and Highly Self-Adaptive Cloud applications

A. SWITCH Subsystems

The SWITCH (Software Workbench for Time-critical and Highly Self-Adaptive Cloud applications) project aims to simplify the development of time-critical[56] applications by providing the functionality to develop, deploy and adapt the systems during their entire life-cycle. The SWITCH platform

TABLE VI
QoS Parameters, Units Used, and Alternative Terms

Parameter	Units	Other names
Time		
Runtime	Time units	Computation speed, computation(al) time, convergence time, detection time, execution time, processing speed, processing time, reconstruction time, registration time
Runtime	Frames per second (fps)	
Time per unit of computation	Time units	Processing time per frame, time per AO time-step
Processing speed	Frames per second (fps) Projections per second (pps)	Detection speed, processing rate, reconstruction speed, tracking rate
Processing rate	ms/pixel	
Communication latency	Time units	
Latency per frame	Time units	
Quality of results		
Accuracy	Percentage	Classification accuracy, detection accuracy
Correlation coefficient (accuracy)		
Average absolute difference (accuracy)	No units, percentage	Percentage of pixels which differed over a given threshold between the ground truth and the results generated
Sensitivity	Percentage	Detection rate
False positives count		
Peak signal-to-noise ratio	dB	
Mean-square-error		Root mean square (RMS) between the ground truth and the results generated
Data		
Throughput	GB/s, GiB/s	Bandwidth, data transmission rate
Throughput	Gflops/s, Gflops	
Memory access	MB	
Compute		
Floating point operations count		
Costs (mentioned only briefly)		
Performance-per-watt		
Performance-per-dollar		
Implementation cost	Dollars	Development cost

has three main parts. The SWITCH IDE (SIDE) deals with creating an application from microservices, and creating the UI for the rest of the system. It also enables the user to define what kind of infrastructure (s)he wants to have. The goal of SIDE is to provide the maximum flexibility to the user when creating applications. The Dynamic Real-Time Infrastructure Planner (DRIP) deals with the planning and provisioning of the virtual machines and deploying the composed application. DRIP functions as middleware for the system, that is internally composed of independent components that provide the necessary functionality. The Autonomous System Adaptation Platform (ASAP) focuses on monitoring, controlling and adapting an application so that it can maintain the required QoS. To facilitate this it includes a Monitoring System, that collects the metrics from the applications, an alarm trigger system, that deals with analyzing these metrics and starting events when certain conditions are met and a self adapter that decides what actions are to be taken and triggers adaptation of the system, taking the form of scaling VMs, horizontal scaling of containers, etc.

SWITCH is motivated not only by Quality of Service concerns, but also by Quality of Experience — the actual performance of an application from users' perspective. The problem is that the user experience cannot be easily measured. To this end QoS is a metric, or a group of them, that approximate, and are to some extent proxies for, QoE. This enables the measurement of the experience of the application and enables adapting the application if necessary. From the perspective of CUDA applications this is mostly useful to determine how powerful an infrastructure is needed to facilitate the analysis of certain problems in a specific time, usually real-time.

B. Possible SWITCH extensions using CUDA

Two of the types of applications considered in Section II-E relate to applications which are already supported by SWITCH, but in a non-CUDA implementation. For example, the papers by Mehrabi et al. [53] and by Maillard et al. [54] are similar in some ways to the SWITCH use case of MOG[3]. Likewise, the paper by Feldmann et al. [55] is similar to the WT use case[4]. However, CUDA is not currently supported by SWITCH. That said, CUDA applications are an obvious candidate for development using SIDE. As the capabilities of cloud providers increase, making it possible to provision instances that include GPU systems, CUDA applications can

[3]http://www.mog-technologies.com/
[4]http://www.wtelecom.es/?lang=en

be migrated to the cloud. This marks another shift in the way applications are developed, as it allows the system to be viable even if it requires substantial CUDA capabilities for a short period of time. These kinds of applications rely on systems that enable quick creation of applications and efficient, automatic deployment and tear-down. SIDE and DRIP could be, with minimal development time, adapted to function with applications that require CUDA compatible instances. These systems could also provide significant cost savings, as they could be dynamically moved to a location where the instances are cheaper. It should be noted that "time critical" does not equate to "fast". In many cases the consistency of the time within which the result is provided is as valuable as the speed of such operations.

What does need to be taken into account, if SWITCH were to support CUDA, is that the CUDA architecture supports a very specific type of application; it is a SIMD (Single Instruction Multiple Data) architecture, which means that it excels at running the same instruction on large amounts of data at the same time, before moving on to running another instruction on the same large amount of data. This means that whatever the application being considered, it would need to be evaluated for how it would suit the CUDA architecture, and possibly adapted to perform best using CUDA.

This could open a way for a new class of application: applications which require fast response times but limited investments. For instance, algorithms that analyse the performance of the application, similar to the modelling system described by Štefanič et al. [57] could be redeveloped to use CUDA, thus enabling larger sets of examples to be analysed in the process of deriving the model. This would enable the system to produce better models faster and thus adaptation could become cheaper.

V. Conclusions

In this paper, we have analysed four research themes identified while surveying time-critical CUDA applications for their associated QoS metrics. These themes are: Environment-related, People/face detection, Medical applications, and Materials-related. (We have also identified a fifth, Miscellaneous, category.) We found that overall, all of the categories besides Materials-related applications consider the quality of results as well as the runtime of the method to obtain those results. In particular, the main attributes considered by each group are:

- Environment-related: time (runtime, processing rate), quality of results (percentage accuracy, correlation coefficient, average absolute difference)
- People/face detection: time (runtime, processing speed), quality of results (percentage accuracy, sensitivity, false positives count)
- Medical applications: time (runtime, processing speed), quality of results (peak signal-to-noise ratio, mean-square error), data (throughput)
- Materials-related: time (runtime)

We also (unsurprisingly) found a larger diversity of QoS measures in those applications that did not fit into the above categories (Miscellaneous applications), which also included measures on the number of floating point operations, for example.

The terminology used by the different papers can vary significantly. Generally, however, all of the time-critical CUDA applications surveyed are concerned with a limited number of QoS parameters: time-related measures (of which runtime is the largest, being referred to by around ten different names); measures relating to the quality of results, such as accuracy; data measures, such as throughput and memory access; and compute measures like the number of floating point operations performed.

We also found a pattern in that image processing-related applications tended to be the only applications to even consider the quality of results obtained, and we believe this may be due to the associated applications of these image processing methods. In particular, we surmised that the reasons why quality and accuracy are considered so much by certain applications is due to the 'seriousness' of those applications. For example, one type of medical application is that of image-guided surgery, which involves the surgeon receiving information during an operation and acting based on that information. This is obviously one which needs a great importance to be placed on results accuracy, as it is an application that directly affects a human life.

Moving towards developing CUDA applications on the SWITCH architecture, and adapting the SWITCH architecture so that it is suitable for CUDA, we have begun experimenting with the CUDA-enabled instance types already present in AWS. Briefly, AWS EC2 supports several GPU instances (e.g. P2 instances), and one can launch instances with a single GPU, or multiple GPUs present. One can also launch these instances across different Regions (geographical locations), and Availability Zones within a Region, and have the instances communicate through different mechanisms. In order to investigate how SWITCH might support CUDA in the future, we are in the early stages of experimenting with running CUDA applications on multiple, single-GPU instances in one geographical Region in AWS, and having them 'communicate' through the use of a shared volume (using Amazon Elastic File System (EFS)). In the future we wish to investigate how changing various parameters, such as the number of instances running, their geographical location, etc. may influence the QoS of an application. This information could give some useful insight into how SWITCH may support CUDA applications.

Acknowledgment

The research reported in this paper was funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 643963 (SWITCH project).

References

[1] nVidia, "Gpu-based deep learning inference: A performance and power analysis," November 2015. [Online]. Available: http://www.nvidia.com/content/tegra/embedded-systems/pdf/jetson_tx1_whitepaper.pdf

[2] Z. Wu, Q. Wang, A. Plaza, J. Li, L. Sun, and Z. Wei, "Real-Time Implementation of the Sparse Multinomial Logistic Regression for Hyperspectral Image Classification on GPUs," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 7, pp. 1456–1460, 2015.

[3] J. M. Bioucas-Dias and M. Figueiredo, "Logistic regression via variable splitting and augmented Lagrangian tools," Inst. Superior Técnico, TU Lisbon, Lisbon, Portugal, Tech. Rep., 2009.

[4] J. A. Goodman, D. Kaeli, and D. Schaa, "Accelerating an imaging spectroscopy algorithm for submerged marine environments using graphics processing units," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 4, no. 3, pp. 669–676, 2011.

[5] Z. P. Lee, K. L. Carder, T. G. Peacock, C. O. Davis, and J. L. Mueller, "Method to derive ocean absorption coefficients from remote-sensing reflectance," *Applied Optics*, vol. 35, no. 3, p. 453, 1996.

[6] Z. Lee, K. L. Carder, C. D. Mobley, R. G. Steward, and J. S. Patch, "Hyperspectral remote sensing for shallow waters. I. A semianalytical model." *Applied optics*, vol. 37, no. 27, pp. 6329–38, 1998.

[7] ——, "Hyperspectral remote sensing for shallow waters: 2 Deriving bottom depths and water properties by optimization," *Applied Optics*, vol. 38, no. 18, p. 3831, 1999.

[8] K. R. Kurte and S. S. Durbha, "High resolution disaster data clustering using Graphics Processing Units," *2013 IEEE International Geoscience and Remote Sensing Symposium - IGARSS*, pp. 1696–1699, 2013.

[9] J. Han, M. Kamber, and A. K. H. Tung, "Spatial Clustering Methods in Data Mining: A Survey," *Geographic Data mining and knowledge discovery*, vol. 2, pp. 188–217, 2001.

[10] U. M. Bhangale and S. S. Durbha, "High performance SIFT feature classification of VHR satellite imagery for disaster management," *ACM International Conference Proceeding Series*, pp. 324–329, 2015.

[11] D. G. Lowe, "Object recognition from local scale-invariant features," *Proceedings of the Seventh IEEE International Conference on Computer Vision*, pp. 1150–1157 vol.2, 1999.

[12] ——, "Distinctive image features from scale invariant keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–11 020 042, 2004.

[13] S. Christgau, J. Spazier, B. Schnor, M. Hammitzsch, A. Babeyko, and J. Wächter, "A comparison of CUDA and OpenACC: Accelerating the Tsunami Simulation EasyWave," *Workshop Proceedings of ARCS 2014*, 2014.

[14] J. Wächter, A. Babeyko, J. Fleischer, R. Häner, M. Hammitzsch, A. Kloth, and M. Lendholt, "Development of tsunami early warning systems and future challenges," *Natural Hazards and Earth System Science*, vol. 12, no. 6, pp. 1923–1935, 2012.

[15] M.-Y. Huang, S.-C. Wei, B. Huang, and Y.-L. Chang, "Accelerating the Kalman filter on a GPU," *Proceedings of the International Conference on Parallel and Distributed Systems - ICPADS*, pp. 1016–1020, 2011.

[16] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, vol. 82, no. 1, p. 35, 1960.

[17] B. Sharma, R. Thota, N. Vydyanathan, and A. Kale, "Towards a robust, real-time face processing system using CUDA-enabled GPUs," *2009 International Conference on High Performance Computing (HiPC)*, pp. 368–377, 2009.

[18] P. Viola and M. J. Jones, "Robust real-time face detection," *International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.

[19] R. Thota, A. Kalyansundar, and A. Kale, "Modeling and Tracking of Faces in Real-life Illumination Conditions," *Proceedings of the International Conference of Acoustics, Speech and Signal Processing*, pp. 761–764, 2009.

[20] A. Herout, R. Jošth, R. Juránek, J. Havel, M. Hradiš, and P. Zemčík, "Real-time object detection on CUDA," *Journal of Real-Time Image Processing*, vol. 6, no. 3, pp. 159–170, 2011.

[21] J. Šochman and J. Matas, "WaldBoost – Learning for Time Constrained Sequential Detection," in *Proceedings of Computer Vision and Pattern Recognition, 2005*, 2005.

[22] E. Fauske, L. M. Eliassen, and R. H. Bakken, "A Comparison of Learning Based Background Subtraction Techniques Implemented in CUDA," *Proceedings of the First Norwegian Artificial Intelligence Symposium*, pp. 181–192, 2009.

[23] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, "Real-time foreground-background segmentation using codebook model," *Real-Time Imaging*, vol. 11, no. 3, pp. 172–185, 2005.

[24] L. Maddalena and A. Petrosino, "A Self-Organizing Approach to Background Subtracion for Visual Surveillance Applications," *IEEE Transactions on Image Processing*, vol. 17, no. 7, pp. 1168–1177, 2008.

[25] ——, "Multivalued background/foreground separation for moving object detection," in *Fuzzy Logic and Applications: 8th International Workshop*, 2009.

[26] T. Horprasert, D. Harwood, and L. S. Davis, "A Robust Background Subtraction and Shadow Detection," *Proceedings of Asian Conf. Computer Vision*, 2000.

[27] D. Weimer, S. Köhler, C. Hellert, K. Doll, U. Brunsmann, and R. Krzikalla, "GPU architecture for stationary multisensor pedestrian detection at smart intersections," *IEEE Intelligent Vehicles Symposium, Proceedings*, no. Iv, pp. 89–94, 2011.

[28] P. Muyan-Özçelik, J. D. Owens, J. Xia, and S. S. Samant, "Fast deformable registration on the GPU: A CUDA implementation of demons," *Proceedings - The International Conference on Computational Sciences and its Applications, ICCSA 2008*, pp. 223–233, 2008.

[29] J.-P. Thirion, "Image matching as a diffusion process: an analogy with Maxwell's demons," *Medical Image Analysis*, vol. 2, no. 3, pp. 243–260, 1998.

[30] M. Modat, G. R. Ridgway, Z. A. Taylor, M. Lehmann, J. Barnes, D. J. Hawkes, N. C. Fox, and S. Ourselin, "Fast free-form deformation using graphics processing units," *Computer Methods and Programs in Biomedicine*, vol. 98, no. 3, pp. 278–284, 2010.

[31] B. Keck, H. Hofmann, H. Scherl, M. Kowarschik, and J. Hornegger, "GPU-accelerated SART reconstruction using the CUDA programming environment," *Proc. SPIE Medical Imaging*, p. 72582B, 2009.

[32] A. H. Andersen and A. C. Kak, "Simultaneous Algebraic Reconstruction Technique (SART): a superior implementation of the ART algorithm," *Ultrasonic Imaging*, vol. 6, pp. 81–94, 1984.

[33] W.-M. Pang, J. Qin, Y. Lu, Y. Xie, C.-K. Chui, and P.-A. Heng, "Accelerating simultaneous algebraic reconstruction technique with motion compensation using CUDA-enabled GPU," *International Journal of Computer Assisted Radiology and Surgery*, vol. 6, no. 2, pp. 187–199, 2011.

[34] D. Riabkov, X. Xue, D. Tubbs, and A. Cheryauka, "Accelerated cone-beam backprojection using GPU-CPU hardware," *Proceedings of the 9th International Meeting on Fully Three-Dimensional Image Reconstruction in Radiology and Nuclear Medicine*, pp. 4–7, 2007.

[35] L. A. Feldkamp, L. C. Davis, and J. W. Kress, "Practical cone-beam algorithm," *Journal of the Optical Society of America A*, vol. 1, no. 6, p. 612, 1984.

[36] J. A. Wilson and J. C. Williams, "Massively Parallel Signal Processing Using the Graphics Processing Unit for Real-Time Brain-Computer Interface Feature Extraction," *Frontiers in Neuroengineering*, vol. 2, no. July, 2009.

[37] G. Schalk, D. J. McFarland, T. Hinterberger, N. Birbaumer, and J. R. Wolpaw, "BCI2000: A general-purpose brain-computer interface (BCI) system," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 6, pp. 1034–1043, 2004.

[38] M. Sutcliffe, M. Weston, P. Charlton, K. Donne, B. Wright, and I. Cooper, "Full matrix capture with time-efficient auto-focusing of unknown geometry through dual-layered media," *Insight: Non-Destructive Testing and Condition Monitoring*, vol. 55, no. 6, pp. 297–301, 2013.

[39] V. Strbac, J. Vander Sloten, and N. Famaey, "Analyzing the potential of GPGPUs for real-time explicit finite element analysis of soft tissue deformation using CUDA," *Finite Elements in Analysis and Design*, vol. 105, pp. 79–89, 2015.

[40] H. Gründemann, *Computational Methods for Transient Analysis*, T. Belytschko and T. J. R. Hughes, Eds. Amsterdam: North-Holland, 1983.

[41] K. Miller, G. Joldes, D. Lance, and A. Wittek, "Total Lagrangian explicit dynamics finite element algorithm for computing soft tissue deformation," *Communications in Numerical Methods in Engineering*, vol. 23, pp. 121–134, 2007.

[42] J. Havel, M. Dubská, A. Herout, and R. Jošth, "Real-time detection of lines using parallel coordinates and CUDA," *Journal of Real-Time Image Processing*, vol. 9, no. 1, pp. 205–216, 2014.

[43] M. Dubská, A. Herout, and J. Havel, "PClines - Line detection using parallel coordinates," *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pp. 1489–1494, 2011.

[44] L. Wang and B. Ellerbroek, "Computer simulations and real-time control of ELT AO systems using graphical processing units," in *Proceedings Volume 8447, Adaptive Optics Systems III*, 2012.

[45] L. Wang, "MAOS," 2010. [Online]. Available: https://github.com/lianqiw/maos

[46] G. Pagès and B. Wilbertz, "GPGPUs in computational finance: massive parallel computing for American style options," *Concurrency and Computation: Practice and Experience*, vol. 24, pp. 837–848, 2012.
[47] V. Bally and G. Pagès, "A quantization algorithm for solving multidimensional discrete-time optimal stopping problems," *Bernoulli*, vol. 9, no. 6, pp. 1003–1049, 2003.
[48] V. Bally, G. Pagès, and J. Printems, "A quantization tree method for pricing and hedging multi-dimensional American options," *Mathematical Finance*, vol. 15, no. 1, pp. 119–168, 2005.
[49] O. Bardou, S. Bouthemy, and G. Pagès, "Optimal quantization for the pricing of swing options," *Applied Mathematical Finance*, vol. 16, no. 2, pp. 183–217, 2009.
[50] A. L. Bronstein, G. Pagès, and B. Wilbertz, "How to speed up the quantization tree algorithm with an application to swing options," *Quantitative Finance*, vol. 10, no. 9, pp. 995–1007, 2010.
[51] A. Lattanzi, E. Ciavattini, S. Cecchi, L. Romoli, and F. Ferrandi, "Real-Time implementation of Wave Field Synthesis on NU-Tech Framework using CUDA Technology," in *128th Convention of Audio Engineering Society*, London, UK, 2010.
[52] A. J. Berkhout, D. de Vries, and P. Vogel, "Acoustic control by wave field synthesis," *Journal of the Acoustic Society of America*, vol. 93, no. 5, pp. 2764–2778, 1993.
[53] M. Mehrabi, S. Lafond, and L. Wang, "Frame Synchronization of Live Video Streams Using Visible Light Communication," *2015 IEEE International Symposium on Multimedia (ISM)*, pp. 128–131, 2015.
[54] J. Maillard, M. Leny, and H. Diakhaté, "Enhancing the audience experience during sport events: Real-time processing of multiple stereoscopic cameras," *Annales des Telecommunications/Annals of Telecommunications*, vol. 68, no. 11-12, pp. 657–671, 2013.
[55] I. Feldmann, W. Waizenegger, N. Atzpadin, and O. Schreer, "Real-time depth estimation for immersive 3D videoconferencing," *3DTV-CON 2010: The True Vision - Capture, Transmission and Display of 3D Video*, 2010.
[56] Z. Zhao, A. Taal, A. Jones, I. Taylor, V. Stankovski, I. G. Vega, F. J. Hidalgo, G. Suciu, A. Ulisses, P. Ferreira, and C. d. Laat, "A software workbench for interactive, time critical and highly self-adaptive cloud applications (switch)," in *2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, May 2015, pp. 1181–1184.
[57] P. Štefanič, M. Cigale, A. Jones, and V. Stankovski, "Quality of service models for microservices and their integration into the switch ide," in *2017 IEEE 2nd International Workshops on Foundations and Applications of Self* Systems (FAS*W)*, Sept 2017, pp. 215–218.