UDC 004.75
DOI: 10.15587/2706-5448.2023.285543

Oleksandr Beznosyk,
Oleksandr Syrotiuk

USAGE OF A COMPUTER CLUSTER FOR PHYSICS SIMULATIONS USING BULLET ENGINE AND OPENCL

The study focuses on using a computer cluster for implementing real-time physical simulations, responding to a growing need for such use in various sectors, including medicine, video processing, automated transport management, robotics, and visualisation. The object of research is cluster and cloud technologies for conducting costly physical simulations for specific sectors, particularly high-budget and entertainment ones, such as cinematography and interactive entertainment.

Research methods include using a modified Bullet engine to carry out physical simulations, integrated with OpenCL to work with the cluster. The choice of these technologies was determined by their high performance and adaptability to cluster systems. The research was based on a typical Bullet framework's benchmark falling tower scene with the primary goal of measuring computational performance in frames per second.

Results showed that the use of clusters is not advisable in environments with a low network throughput and the use of non-uniform computers. Under those conditions, simulations using a cluster become unstable with many objects and contacts between them and show a degradation in performance by an average of 50–60 % (to values of 10–20 frames per second).

Despite the intermediate results of calculations on the cluster, the study met the expectations within the goals set and resources available. These results have significant implications for the further development of cluster and cloud technologies in physical simulations, providing valuable information about the limitations and capabilities of these systems.

Keywords: *3D space, continuous space, collision solution, collision detection, cloud technologies, distributed computing, high-performance computing.*

Received date: 12.06.2023
Accepted date: 02.08.2023
Published date: 04.08.2023

How to cite
Beznosyk, O., Syrotiuk, O. (2023). Usage of a computer cluster for physics simulations using Bullet engine and OpenCL. Technology Audit and Production Reserves, 4 (2 (72)), 6–9. doi: https://doi.org/10.15587/2706-5448.2023.285543

1. Introduction

Conducting physical simulations (and determining collisions between objects in particular) is an essential application of numerical methods in contemporary computer science. With the emergence of new technologies based on simulation results, the enhancement of simulation methods is becoming increasingly important [1–3]. Even with hardware improvements, the requirements for such computations should be simplified since new, more complex problems emerge.

This study explores the availability of considerable computational power for conducting physical simulations on computers (clients, thin clients) not adapted for this purpose by implementing a client-server connection where a computer cluster acting as a server will conduct the simulations. For this purpose, an experiment is being conducted to measure the performance of simulation calculations on a remote computer cluster, subsequently comparing this with the efficiency of performing the same simulation undertaken directly on the dedicated computers of different computational possibilities.

During the study, we researched the methods of calculating collisions in three-dimensional space to understand better at what level parallelization should occur in the cluster. This topic was chosen as it is relevant for many industries [1, 2]. Moreover, we found few publications exploring related themes of conducting such simulations on a cluster or computational clouds, and commercial analogues developing in this direction are just beginning to appear [1–3].

The research aims to describe the effect of computer cluster usage for the physical simulation (primarily for collision detection in 3D space) and discover the impact of cluster and its nodes characteristics on simulation efficiency. On the practical side of the research, the emergence of such systems will allow complex physical computations to be used on devices that cannot afford to perform such calculations in a reasonable time, like self-driving cars, medical equipment, mobile phones, or tablets. Moreover, those systems could lead to the creation of entirely new devices and save resources in existing processes.

2. Materials and Methods

The theoretical part of the study was conducted in two directions – investigating existing methods of collision detection and simulation in general, as well as examining

existing approaches and tools for their practical applicability [3–6]. Adjacent fields and technologies that will only be presented in the future at the time of the study were also considered. This is mainly dedicated to the commercial analogues of that research, but other open research is also considered [7].

Several existing frameworks for conducting physical simulations (physics engines or APIs) were investigated during the work, including Nvidia PhysX, BulletsPhysics, Havok, and Box2d [8, 9].

To build the cluster, several personal computers of different configurations were used. The hardware characteristics significant for this research are presented in Tables 1, 2.

As a result of API investigation, the BulletsPhysics API was selected as it meets the main requirements: open access to the code, ease of use, vast and complete documentation, and free licensing. OpenCL framework was selected as the tool for organizing computations on the cluster, as it is freely available and compatible with a wide range of hardware [10]. For the space partitioning structure in the scene, Dynamic Bounding Volume Tree was chosen [11].

The practical part of the research consisted of using the components selected during the theoretical stage, forming the cluster, and conducting physical simulations in different configurations: using the computer cluster and, alternatively, the client machine.

Table 1

Characteristics of the first computer, A

Parameter Name	Value
CPU	Intel Celeron J4125
GPU	Integrate into the CPU
CPU core count	4
Hyper-Threading	Presented
GPU core count	12
CPU tick rate	2.0–2.7 GHz
GPU tick rate	250 MHz
RAM	4 Gb
VRAM	8 Gb

Table 2

Characteristics of the second computer, B

Parameter Name	Value
CPU	Intel i7-10700K
GPU	Nvidia 2060 Super
CPU core count	8
Hyper-Threading	Presented
GPU core count	2682
CPU tick rate	3.8 GHz
GPU tick rate	1450 GHz
RAM	32 Gb
VRAM	8 Gb

The final part of the research involves analysing the obtained results, explaining the reasons for these results, and establishing directions for further investigation and improvements of the received result.

3. Results and Discussion

In the first stage of the study, computers from Tables 1, 2 were used to conduct simulations, using these computers as client machines. The primary simulation selected for this study was the fall of a tower composed of 3000 cubes. The obtained results will serve as a benchmark for comparing the results of other experiments. Fig. 1 presents the fluctuations in frames per second (FPS) depending on the simulation time. The results for the computer from Table 1 are indicated in red, and those from Table 2 are in blue. Let's observe that machine B, being more powerful, supports the simulation more stably (the deviation of the average number of frames per second is smaller) and overall processes more frames per unit of time.

Next, it is necessary to investigate the impact of the OpenCL API on the execution of the simulations. The results of the simulations for computers A and B are presented below in Fig. 2.

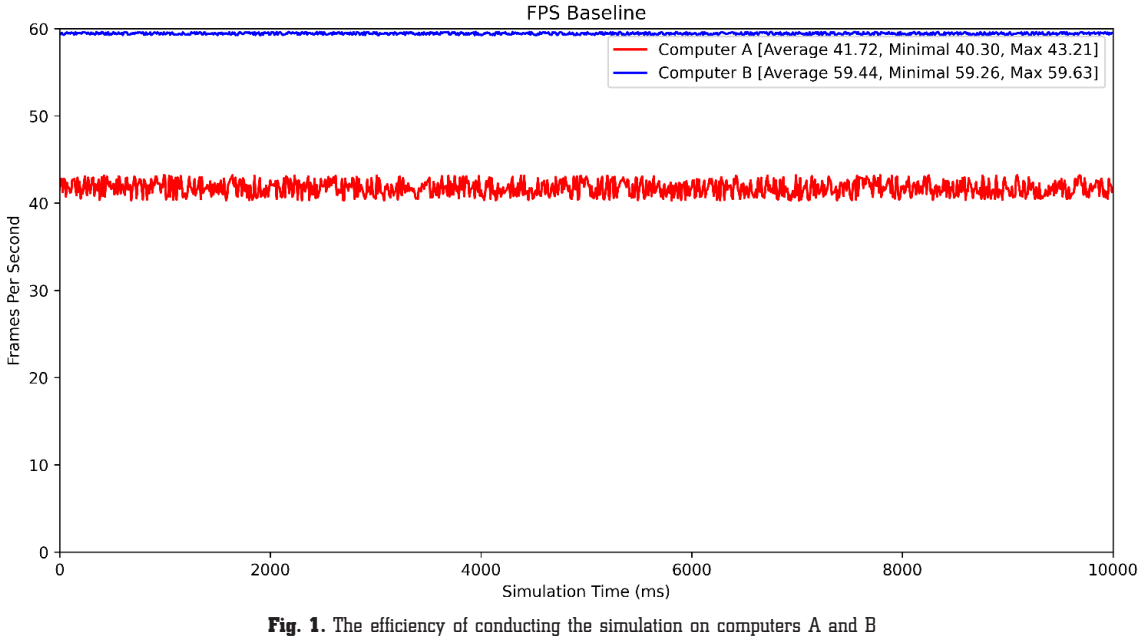

Fig. 1. The efficiency of conducting the simulation on computers A and B

ISSN 2664-9969

As a result of using OpenCL, the efficiency indicators of the calculations have decreased. For computer A, the average FPS rate dropped by 14.7 % (a decrease from 41.72 to 35.70), and the difference between the minimum and maximum value increased by 7 % (an increase from 2.91 to 3.12). For computer B, these indicators are 0.7 % (a decrease from 59.44 to 59.06) and 143 % (a rise from 0.37 to 0.9).

Using computers A and B, let's build a cluster using OpenCL. The behaviour when calculating on a cluster, presented in Fig. 3, differs significantly from the results obtained in previous experiments. It can be seen that the system does not have a stable FPS indicator for most of the simulation. Let's observe a drop in performance indicators during the first 33 % of the simulation. The simulation's heaviest moment occurs during the tower's fall. The largest number of contacts is observed when the most significant number of intersections between the components occurs. In total, there are 3000 cubic bodies combined into a tower on the scene.

At the beginning of the simulation, only 650 contacts between bodies are active. At the moment of the 4000 milliseconds, 18500 contacts are already active, which leads to a sharp drop in performance. The system is at rest for some time, after which the performance recovery begins due to decreased contact between bodies.

At the same time, a separate question arises: why did the FPS indicators remain stable in previous experiments even at peak loads because all experiments' simulations are identical. It can be assumed that such a problem arises due to the need to synchronize data between cluster nodes (issue of parallelism organization by the OpenCL framework), as well as the heterogeneity of cluster elements (machine B is much more powerful than machine A) and the cluster's small size [2, 12].

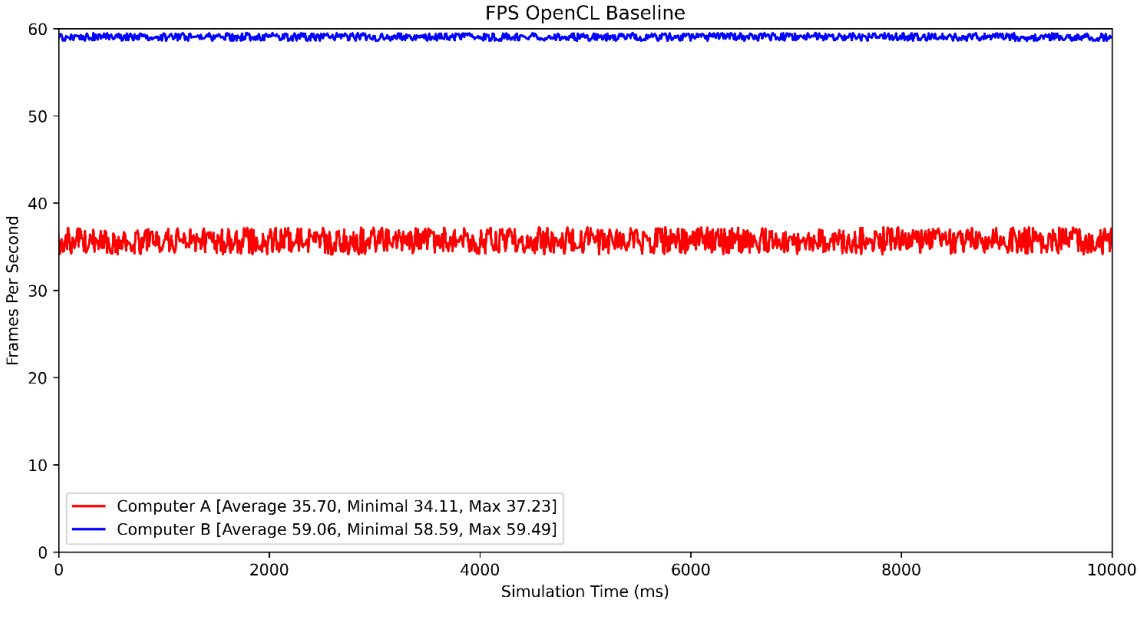

Fig. 2. The efficiency of conducting the simulation using OpenCL

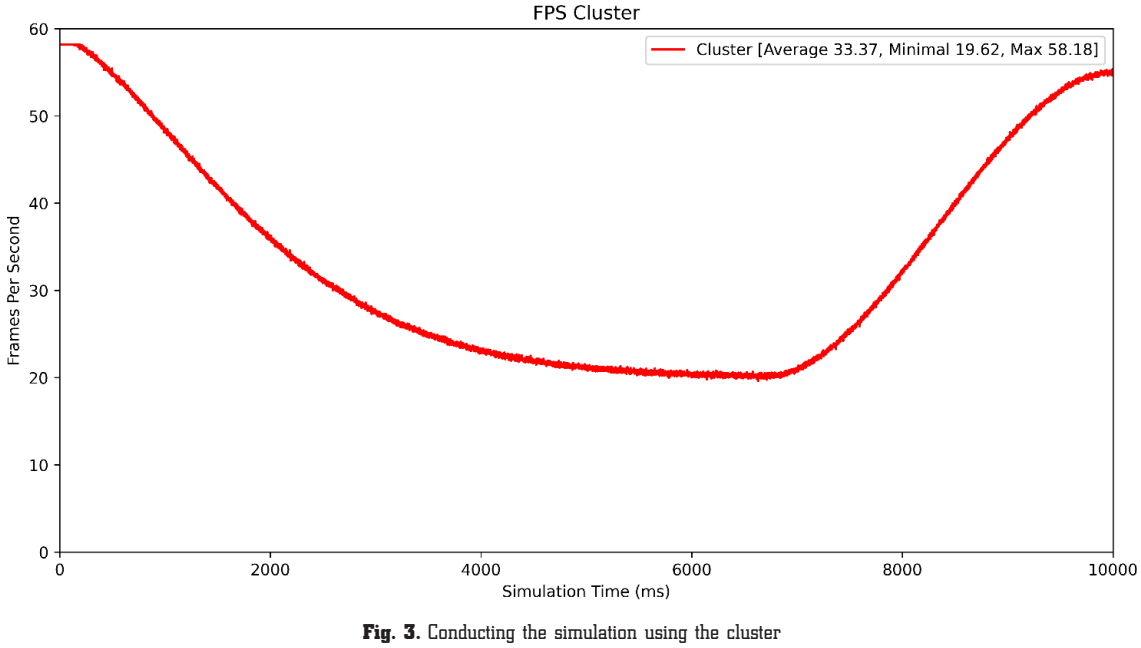

Fig. 3. Conducting the simulation using the cluster

The main limitations of this study are the small hardware base available at the time of this study. Limited access to powerful GPUs has forced to use urgent resources for personal access, limiting the number of computers in the cluster and the time allocated to conduct experiments. The limitations of this study can include the absence of investigation of the architecture of the software used to perform the experiments, as most of the available API is paid. Accordingly, no practical comparisons of the impact of different space partition structures and collision detection algorithms between various object forms were conducted [11, 13].

The martial law had a substantial impact on the conduct of this study because it was through it that there was a limitation of access to a large number of computing powers (part of the available computing resources was physically damaged or completely destroyed, another part of resource due to damage to communications during the study also was not available). One of the authors was in a non-controlled territory of the country when conducting the theoretical part of the study, which also led to further economic problems in performing the work.

Further research stems directly from existing constraints and results. This is, first of all, a study of the impact of the architecture of the simulation engine on the use of distributed computing, the impact of space partition structures and collision detection algorithms. It is also worth exploring the effect of data size transmitted between cluster nodes and the client itself. It is necessary to study the impact of the homogeneity of cluster nodes on the effectiveness of the calculation.

4. Conclusions

Considering the research and related studies, using a cluster to perform physical simulations is possible. However, at the moment, the efficiency and stability of these calculations could be more satisfactory. Only using OpenCL as a backend for calculations decreases average performance from 0.7 % to 14 % (according to results for machines A and B) and reduces the simulation's stability. The use of a cluster as a source of computational power reduced the efficiency of calculations by 45 % in the conducted experiment (compared to machine B) and 6 % (compared to machine A), which in general can be summarized as the conclusion that performance, in this case, is heading to the results of the weakest machine in the cluster (although this statement is subject to further research).

It should be noted that these results were obtained using technologies initially not intended for cluster computing, and the test scene is quite complex, considering the simple forms used in it. This study requires further development, as other factors, besides the cluster itself and the API to work with it, also influence the results of calculations.

Conflict of interest

The authors declare that they have no conflict of interest about this research, whether financial, personal, authorship or otherwise, that could affect the study and its results presented in this paper.

Financing

The research was performed without financial support.

Data availability

The manuscript has no associated data.

References

1. Lewis, N. S., Winter, A. O., Bonus, J., Motley, M. R., Eberhard, M. O., Arduino, P., Lehman, D. E. (2023). Open-source simulation of strongly-coupled fluid-structure interaction between non-conformal interfaces. *Frontiers in Built Environment, 9*. doi: https://doi.org/10.3389/fbuil.2023.1120518
2. Merchant, N., Sampson, A. T., Boiko, A., Falconer, R. E. (2023). Dense agent-based HPC simulation of cell physics and signaling with real-time user interactions. *Frontiers in Computer Science, 5*. doi: https://doi.org/10.3389/fcomp.2023.1085867
3. Jorissen, K., Vila, F. D., Rehr, J. J. (2012). A high performance scientific cloud computing environment for materials simulations. *Computer Physics Communications, 183 (9)*, 1911–1919. doi: https://doi.org/10.1016/j.cpc.2012.04.010
4. Serpa, Y. R., Rodrigues, M. A. F. (2020). Broad Phase Collision Detection: New Methodology and Solution for Standardized Analysis of Algorithms. *Anais Estendidos Da Conference on Graphics, Patterns and Images (SIBRAPI Estendido 2020)*. doi: https://doi.org/10.5753/sibgrapi.est.2020.12982
5. Cohen, J. D., Lin, M. C., Manocha, D., Ponamgi, M. (1995). I-COLLIDE. *Proceedings of the 1995 Symposium on Interactive 3D Graphics – SI3D'95*. doi: https://doi.org/10.1145/199404.199437
6. Kockara, S., Halic, T., Iqbal, K., Bayrak, C., Rowe, R. (2007). Collision detection: A survey. *2007 IEEE International Conference on Systems, Man and Cybernetics*. doi: https://doi.org/10.1109/icsmc.2007.4414258
7. Xiaoguang, A., Ling, L. (2016). A Study of Collision Detection Algorithm Based on Cloud Computing Model. *2016 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS)*. doi: https://doi.org/10.1109/icitbs.2016.9
8. Erez, T., Tassa, Y., Todorov, E. (2015). Simulation tools for model-based robotics: Comparison of Bullet, Havok, MuJoCo, ODE and PhysX. *2015 IEEE International Conference on Robotics and Automation (ICRA)*. doi: https://doi.org/10.1109/icra.2015.7139807
9. Larsson, T., Akenine-Möller, T. (2006). A dynamic bounding volume hierarchy for generalized collision detection. *Computers & Graphics, 30 (3)*, 450–459. doi: https://doi.org/10.1016/j.cag.2006.02.011
10. Barak, A., Ben-Nun, T., Levy, E., Shiloh, A. (2010). A package for OpenCL based heterogeneous computing on clusters with many GPU devices. *2010 IEEE International Conference On Cluster Computing Workshops and Posters (CLUSTER WORKSHOPS)*. doi: https://doi.org/10.1109/clusterwksp.2010.5613086
11. Yoon, J., Son, B., Lee, D. (2023). Comparative Study of Physics Engines for Robot Simulation with Mechanical Interaction. *Applied Sciences, 13 (2)*, 680. doi: https://doi.org/10.3390/app13020680
12. Chau, S.-C., Fu, A. W.-C. (2004). Load Balancing between Heterogeneous Computing Clusters. *Lecture Notes in Computer Science*, 75–82. doi: https://doi.org/10.1007/978-3-540-24679-4_19
13. Ströter, D., Mueller-Roemer, J. S., Stork, A., Fellner, D. W. (2020). OLBVH: octree linear bounding volume hierarchy for volumetric meshes. *The Visual Computer, 36 (10-12)*, 2327–2340. doi: https://doi.org/10.1007/s00371-020-01886-6

Oleksandr Beznosyk, *PhD, Associate Professor, Department of System Design, National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute», Kyiv, Ukraine, ORCID: https://orcid.org/0000-0003-2775-6070*

✉***Oleksandr Syrotiuk,*** *Postgraduate Student, Department of System Design, National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute», Kyiv, Ukraine, ORCID: https://orcid.org/0000-0002-4531-6290, e-mail: oleksandr.syrotiuk.dev@gmail.com*

✉*Corresponding author*