TECHNICAL TRANSACTIONS | **CZASOPISMO TECHNICZNE**
MECHANICS | MECHANIKA
1-M/2013

TADEUSZ CZYŻEWSKI*

PARALLEL COMPUTING IN KINEMATIC ANALYSIS OF HEAVY MACHINERY EQUIPMENT SYSTEM

OBLICZENIA RÓWNOLEGŁE W ANALIZIE KINEMATYKI OSPRZĘTU MASZYN ROBOCZYCH

Abstract

This paper presents a methodology to increase performance of the kinematics simulation process of heavy machinery equipment system. The proposed method is based on parallelization of calculations performed in a single simulation step. Parallelization is achieved by building application based on the OpenCL framework that allows to perform the necessary mathematical calculations using a graphics card processor. In the final part of the article were presented results of performance tests of the proposed method.

Keywords: *parallel computing*, *GPU unit*, *OpenCL*, *kinematic analysis*, *heavy machinery*

Streszczenie

W artykule przedstawiono metodykę zwiększania wydajności procesu symulacji kinematyki osprzętu maszyn roboczych. Zaproponowana metoda polega na zrównolegleniu obliczeń wykonywanych w ramach jednego kroku symulacyjnego. Zrównoleglanie realizowane jest poprzez zbudowanie oprogramowania opartego o szablon aplikacji OpenCL co pozwala na wykonanie niezbędnych obliczeń matematycznych za pomocą procesora karty graficznej. W końcowej części artykułu przedstawione zostały wyniki testów wydajności zaproponowanej metody.

Słowa kluczowe: *obliczenia równoległe*, *układ GPU*, *OpenCL, analiza kinematyczna*, *maszyny robocze*

* MSc. Eng. Tadeusz Czyżewski, Institute of Applied Informatics, Faculty of Mechanical Engineering, Cracow University of Technology.

1. Introduction

Designing of heavy machinery requires consideration of many aspects of the trajectory planning of particular links necessary to perform work cycle. The typical design process of heavy machinery is carried out using the 3D CAD systems. In these systems the assembly models of individual machine links are represented by solids with adequate material properties, mass, volume, etc. Between the machine links constraints are imposed. These constraints depending on the type reduce degrees of freedom to a specified number. Types of constraints are defined by the designer at the design stage. Information stored in a three-dimensional model of a modern CAD system can and should be used in the model simulations. Therefore, many CAD systems are equipped with modules enabling to conduct simulation investigations such as calculation of: kinematics, dynamics, strength, etc.

At the turn of the twentieth and twenty-first century on the hardware market appeared first graphics card equipped with a so-called GPU (*Graphics Processing Unit*). The primary purpose of these processors was to support the CPU (*Central Processing Unit*) in calculations related to the processing of 3D graphics. Over time, these systems has been used for calculations unrelated to graphics. The turning point on the way to develop a technique that allows performing general-purpose computation on GPUs (*GPGPU – General-Purpose Computing on Graphics Processor Units*) was the appearance of graphics cards equipped with programmable shader (NVIDIA GeForce 3 card series) [1,2]. The main task of shaders is to execute a short computer program written in specialized language (*shading language*), which is used to describe the properties of pixels and vertices. The syntax of this specialized language is very similar to the syntax of general-purpose programming languages such as C. Therefore, many scientific communities around the world very quickly began to experiment with writing code for a shader that implements the calculations not connected with graphics processing. As a consequence graphics card manufacturers have developed specialized platforms for the design and analysis of programs that perform calculations on GPUs. Each of these platforms is designed only for a specific manufacturer's devices. So this means that a program written for Nvidia graphics cards do not fully utilize the computing power of a computer CPU, and vice versa. This problem has been solved by the introducing in 2009 by Apple Inc. an OpenCL framework that allows to write programs implementing parallel computing on any hardware (CPU, GPU, DSP, etc.) [3].

Despite the significant growth of computing power of a hardware and capabilities of its use, CAD modules intended for simulations still do not provide parallel processing. Therefore, this article attempts to make full use of hardware computing power for parallel processing. For this purpose, an OpenCL-based application was built, which was used to conduct a series of performance tests of mathematical operations that are used in the analysis of the kinematics.

2. Simulation object

2.1. Kinematic structure

Fig. 2 presents the kinematic structure of the backhoe excavator equipment system, which was the object of research. This structure consists of a boom, arm and bucket connected to

each other and the base with revolute joints. Links and the basis connected in series form an open kinematic chain.

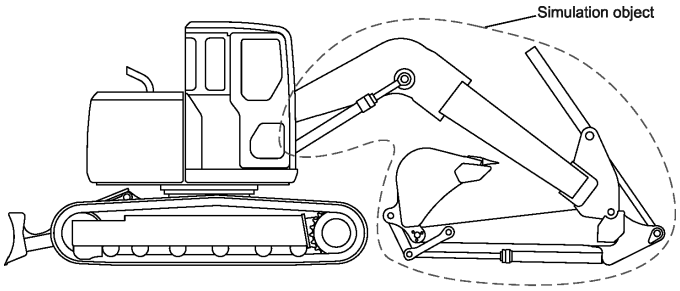

Fig. 1. Backhoe excavator with an isolated equipment system, which was the object of the simulation
Rys. 1. Koparka podsiębierna z wydzielonym osprzętem roboczym, który był obiektem symulacji

Motion of excavator links is forced by double-acting hydraulic cylinders, which together with the main kinematic chain create local kinematic loops, that was presented in Fig. 2. In order to increase bucket rotation angle, there was used a four-bar linkage mechanism, which is also a local kinematic loop. The links can be considered as a non-deformable.

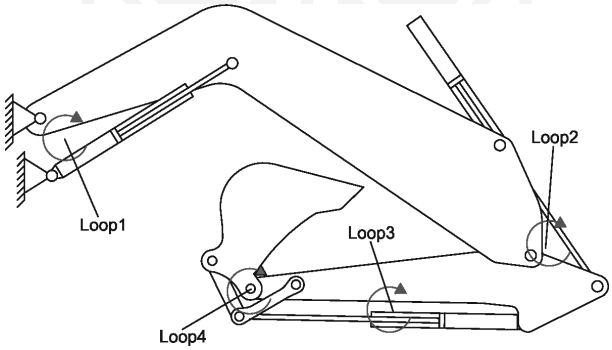

Fig. 2. Kinematic scheme of the equipment system
Rys. 2. Schemat kinematyczny osprzętu roboczego

2.2. Kinematic analysis

Kinematic analysis of the excavator equipment system can be performed by using of multibody formalism and the absolute coordinates, which describes the position and orientation of all links in every step of the simulation. In Fig. 3 are presented two links selected from the considered kinematic chain of the excavator. To each link a local reference system π is bounded. The position and orientation of any link of the excavator kinematic chain can be unequivocally defined by giving:

– vector $\boldsymbol{r}$ which specifies the position of the origin of the local reference system π of the link in global reference system π_0.

– three angles α, β, γ what describe orientation of the local reference system π of the link relative to global reference system π_0.

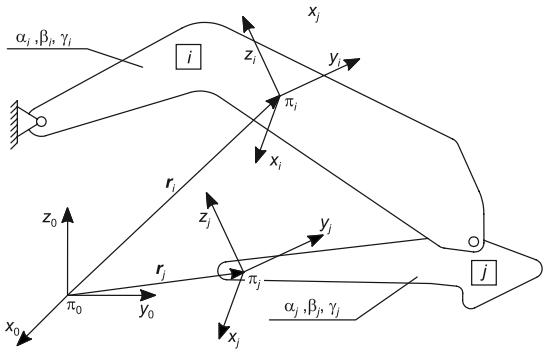

Fig. 3. Absolute coordinates

Rys. 3. Współrzędne absolutne

Thus, the absolute coordinates vector $\boldsymbol{q}_i$ of unconstrained link *i* in three dimensional space can be written as

$$\boldsymbol{q}_i = [\boldsymbol{r}_i^T, \alpha_i, \beta_i, \gamma_i]^T \tag{1}$$

where α, β, γ are the system of three Euler angles. After assuming the notation

$$\varphi_i = [\alpha_i, \beta_i, \gamma_i]^T \tag{2}$$

absolute coordinates vector $\boldsymbol{q}_i$ of unconstrained link *i* will have the following form

$$q_i = [\boldsymbol{r}_i^T, \varphi_i^T] \tag{3}$$

By using eq. (1)–(3) it is possible to write the absolute coordinates vector, which describes position and orientation of all links of backhoe excavator from Fig. 1. According to [4] the absolute coordinates vector will be as follows

$$\boldsymbol{q} = [\boldsymbol{q}_1^T, \boldsymbol{q}_2^T, \boldsymbol{q}_3^T, \boldsymbol{q}_4^T, \boldsymbol{q}_5^T, \boldsymbol{q}_6^T, \boldsymbol{q}_7^T, \boldsymbol{q}_8^T, \boldsymbol{q}_9^T, \boldsymbol{q}_{10}^T, \boldsymbol{q}_{11}^T]^T \tag{4}$$

Kinematics links between particular components of excavator is expressed by imposing on vector (4) suitable holonomic constraints in form

$$\Phi\mathrm{K} = (\boldsymbol{q}) = \boldsymbol{0} \tag{5}$$

Similar restrictions as to positions have to be applied to velocities and accelerations of excavator links. Constraints which bound the velocity vector are obtained by differentiating with respect to time equation (5)

$$\dot{\boldsymbol{\Phi}}^K(\dot{\boldsymbol{q}}, \boldsymbol{q}) = \boldsymbol{\Phi}_{\boldsymbol{q}}^K \dot{\boldsymbol{q}} = \boldsymbol{0} \tag{6}$$

where $\boldsymbol{\Phi}_{\boldsymbol{q}}^K$ is a Jacobian matrix of absolute constraints vector (4). Constraints which bind the velocity vector can be obtained by double differentiating with respect to time equation (5)

$$\ddot{\boldsymbol{\Phi}}^K(\ddot{\boldsymbol{q}},\dot{\boldsymbol{q}},\boldsymbol{q}) = \boldsymbol{\Phi}_{\boldsymbol{q}}^K\ddot{\boldsymbol{q}} = -(\boldsymbol{\Phi}_{\boldsymbol{q}}^K\dot{\boldsymbol{q}})_{\boldsymbol{q}}\dot{\boldsymbol{q}} = \boldsymbol{\Gamma}^K \tag{7}$$

For conducting kinematic analysis it is necessary to determine specific motion of the object. For this purpose there are defined additional constraint equations, which are dependent on absolute coordinates vector (4) and time. These equations are called driving constraints and are written as follows

$$\boldsymbol{\Phi}^D = (\boldsymbol{q},\, t) = \mathbf{0} \tag{8}$$

Driving constraints (8), as in the case of kinematic constraints (5), impose additional restrictions on the generalized velocities and accelerations. By differentiating twice the equation (8) with respect to time there are obtained equations of driving constraints for generalized velocities and accelerations, which take the form

$$\dot{\boldsymbol{\Phi}}^D(\dot{\boldsymbol{q}},\boldsymbol{q},t) = \boldsymbol{\Phi}_{\boldsymbol{q}}^D\dot{\boldsymbol{q}} = -\boldsymbol{\Phi}_t^D \tag{9}$$

$$\ddot{\boldsymbol{\Phi}}^D(\ddot{\boldsymbol{q}},\dot{\boldsymbol{q}},\boldsymbol{q},t) = \boldsymbol{\Phi}_{\boldsymbol{q}}^D\ddot{\boldsymbol{q}} = -(\boldsymbol{\Phi}_{\boldsymbol{q}}^D\dot{\boldsymbol{q}})_{\boldsymbol{q}}\dot{\boldsymbol{q}} - 2\boldsymbol{\Phi}_{\boldsymbol{q}t}^D\dot{\boldsymbol{q}} - \boldsymbol{\Phi}_{tt}^D = \boldsymbol{\Gamma}^D \tag{10}$$

After collecting equations (5) and (8) a system of nonlinear algebraic equations is obtained. Solution of this system gives position vector of all excavator links

$$\boldsymbol{\Phi}(\boldsymbol{q},t) = \begin{bmatrix} \boldsymbol{\Phi}^K(\boldsymbol{q}) \\ \boldsymbol{\Phi}^D(\boldsymbol{q},t) \end{bmatrix} = \mathbf{0} \tag{11}$$

Collecting equations (6) and (9), (7) and (10) give a system of linear algebraic equations. Solution of this system allows to obtain the velocity and acceleration vector of all excavator links

$$\dot{\boldsymbol{\Phi}}(\dot{\boldsymbol{q}},\boldsymbol{q},t) = \boldsymbol{\Phi}_{\boldsymbol{q}}\dot{\boldsymbol{q}} = -\boldsymbol{\Phi}_t = \begin{bmatrix} \mathbf{0} \\ -\boldsymbol{\Phi}_t^D \end{bmatrix} \tag{12}$$

$$\ddot{\boldsymbol{\Phi}}(\ddot{\boldsymbol{q}},\dot{\boldsymbol{q}},\boldsymbol{q},t) = -(\boldsymbol{\Phi}_{\boldsymbol{q}}\dot{\boldsymbol{q}})_{\boldsymbol{q}}\dot{\boldsymbol{q}} - 2\boldsymbol{\Phi}_{\boldsymbol{q}t}\dot{\boldsymbol{q}} - \boldsymbol{\Phi}_{tt} = \begin{bmatrix} -(\boldsymbol{\Phi}_{\boldsymbol{q}}^K\boldsymbol{q})_{\boldsymbol{q}}\dot{\boldsymbol{q}} \\ -(\boldsymbol{\Phi}_{\boldsymbol{q}}^D\dot{\boldsymbol{q}})_{\boldsymbol{q}}\dot{\boldsymbol{q}} - 2\boldsymbol{\Phi}_{\boldsymbol{q}t}^D\dot{\boldsymbol{q}} - \boldsymbol{\Phi}_{tt}^D \end{bmatrix} \tag{13}$$

Conducting kinematic analysis of backhoe excavator will be consist in solving system of equations (11)–(13).

3. Simulation parallelization

The basic mathematical methods in kinematic analysis based on multibody system formalism is matrix analysis. One of the most common mathematical operations in kinematic analysis is the matrix multiplication. In a single simulation step it is necessary to make tens or even hundreds of such multiplications. Depending on the size of the time step and the time interval in which the simulation is performed, the number of simulation steps may vary from a few to several hundred, and even more. Therefore, one of the key aspects that determine the efficiency of the simulation is to use a very efficient algorithm for matrix

multiplication. A good approach in this situation is to use a multi-core architecture, which enables parallelization of the matrix multiplication process.

According to [5] the mathematical definition of matrix multiplication can be written as

$$\boldsymbol{C}_{i,j} = \boldsymbol{C}_{i,j} + \sum_{k=0}^{P} \boldsymbol{A}_{i,k} \cdot \boldsymbol{B}_{k,j} \tag{14}$$

where

$$0 \le i \le N, \qquad 0 \le j \le M$$

This is shown schematically in Fig. 4. It is well known that the classical approach to matrix multiplication allows in one calculation step of designating one of the elements of the resulting. Use of OpenCL framework enables to write a special program that will be executed

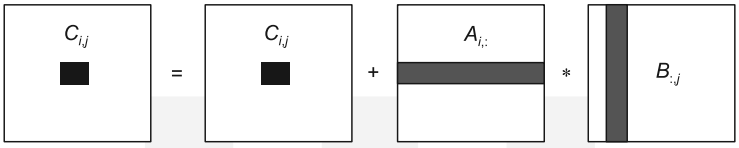

Fig. 4. Classical matrix multiplication

Rys. 4. Klasyczne mnożenie macierzy

on multiple threads of CPU or GPU. Unlike the classical approach parallel matrix multiplication allows to determine in one calculation step not only one element of the resulting matrix but the entire row or column. This was shown schematically in Fig. 5.

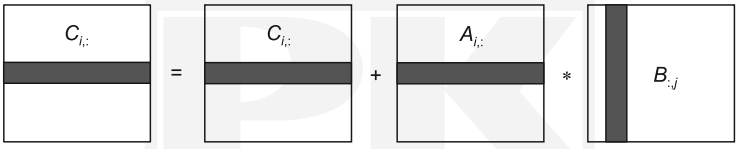

Fig. 5. Parallel matrix multiplication

Rys. 5. Równoległe mnożenie macierzy

4. Test results

Kinematic analysis of excavator equipment system shown in fig. 1 consists in solving three equations describing the position, velocity and acceleration of the entire system in time. The system of equations describing the system position is a system of nonlinear algebraic equations. To solve this system the Newton-Raphson method was used. This is an iterative method, in which one of the mathematical operations performed in a single step is to multiply the inverted Jacobian matrix of an absolute coordinates vector $\boldsymbol{q}$ with constraints imposed on these coordinates. Constraints vector is treated as a column matrix, and therefore these multiplication is simply a multiplication of two matrices. This matrix multiplication was the subject of performance test which was carried out in this article. Tests were conducted on various hardware configurations which are shown in table 1.

Table 1

Hardware configurations used in tests

Configuration symbol	Description of component	
	Processor	**Graphics card**
K1	Intel Core i7-3610QM 2.3 GHz	NVIDA GeForce GT 635M
K2	2 x Intel Xeon E5520 2.27 GHz	NVIDA GeForce GTX 285
K3	Intel Core i7-2630QM 2.0 GHz	NVIDA Quadro 3000M
K4	Intel Core i7-2670QM 2.2 GHz	NVIDIA GeForce GTX 570M

On each hardware configuration was run an OpenCL-based computer program, which performed mentioned matrix multiplication in three ways: classical approach, parallel on CPU, parallel on GPU. The test results were presented in Fig. 6. This chart shows that the largest increase of a performance compared to the classical approach has been obtained by performing parallel computations using the GPU. The resulting increase of efficiency is varying from about thirteen times in the case of K2 configuration to about fifteen times in the case of K4 configuration. The test results also show that the parallelization of matrix multiplication process using the main computer processor enables to obtain several times increase of performance. Obtained increase of efficiency is varying from about two times for configuration K3 and K4 to five times for configuration K2.

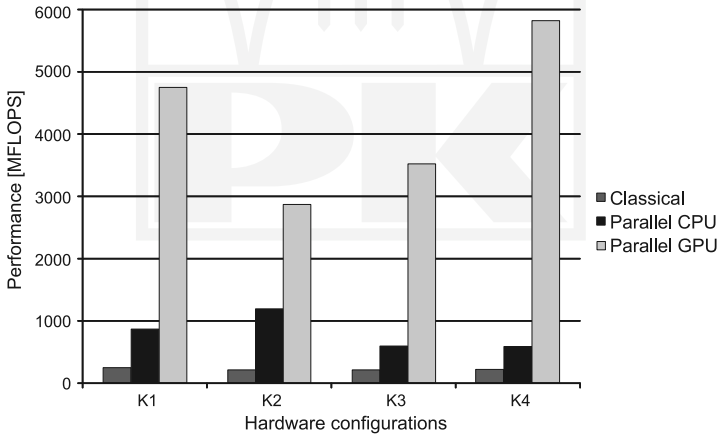

Fig. 6. Test results

Rys. 6. Wyniki testów

5. Conclusions

The article shows how the performance of mathematical operations, such as matrix multiplication, can affect the efficiency of the kinematic simulation of heavy machinery equipment system. Performed tests shown that the performance of matrix multiplication

can be repeatedly increased by parallelization of multiplication process. The best results were obtained with parallelization of the calculation process using the GPU and OpenCL framework. Obtained results leads to conclusion that the parallelization of other mathematical operations occurring in the kinematic calculations algorithm (such as matrix inversion, matrix norm determination) can further increase the efficiency of the simulation process.

References

[1] Kird B.D., Hwu W.W., *Programming massively parallel processors. A hands-on approach*, Morgan Kaufmann, Burlington 2010.

[2] Tasora A., Negrut D., Anitescu M., *GPU-Based Parallel Computing for the Simulation of Complex Multibody Systems with Unilateral and Bilateral Constraints: An Overview*, Computational Methods in Applied Sciences vol. 23, Springer, New York 2011.

[3] Munshi A., Gaster W.W., Mattson T. G., Fung J., Ginsburg D., *OpenCL programming guide*, Addision-Wesley, Boston 2012.

[4] Haug E.J., *Computer Aided Kinematics and dynamics of mechanical systems. Volume I: Basic methods*, Allyn and Bacon, Massachusetts 1989.

[5] Trefethen L.N., Bau. D., *Numerical linear algebra*, Society for Industrial and Applied Mathematics, Philadelphia 1997.