A Case Study on Parallelization of Multibody Dynamic Equations for Graphic Processing Units (GPUs) in Real-Time Simulations - the Planar nR Serial Chain.

Louis Guigon, Paul Rosaz, Benjamin Boudon,
Chedli Bouzgarrou, Youcef Mezouar and Andres Kecskemethy

May 29, 2024

A Case Study on Parallelization of Multibody Dynamic Equations for Graphic Processing Units (GPUs) in Real-Time Simulations - The Planar nR Serial Chain.

Louis Guigon[*], **Paul Rosaz**[#], **Benjamin Boudon**[#], **Chedli Bouzgarrou**[#] **Youcef Mezouar**[#], **Andrés Kecskeméthy**[*]

[*] Chair of Mechanics and Robotics,
University Duisburg-Essen,
Lotharstr. 1, 47057 Duisburg, Germany
louis.guigon@uni-due.de
andres.kecskemethy@uni-due.de

[#] Université Clermont Auvergne,
Clermont Auvergne INP, CNRS,
Institut Pascal,
F-63000 Clermont-Ferrand, France
paul.rosaz@sigma-clermont.fr
chedli.bouzgarrou@sigma-clermont.fr
benjamin.boudon@sigma-clermont.fr
youcef.mezouar@sigma-clermont.fr

Abstract

Graphic Processing Units (GPUs) are the second heart of modern computers. Mainly used in displaying graphics, it became an additional tool for fast processing in the early 2000's. Thanks to the extension to General Purpose for Graphic Processing Units (GPGPU), many applications such as deep learning use this hardware to process tasks in parallel [1]. The present paper continues our investigations [2] on the usability of GPUs in the field of multibody dynamics, in this case for forward dynamics of planar serial chains using a single-step explicit Euler integration scheme, which is ubiquitous in real-time simulations. The GPU is a Single Instruction, Multiple Thread (SIMT) or also known as Multiple Data (SIMD) architecture. Thus, parallelizing on SIMT architecture requires to find independent and repetitive schemes, such as atomic operations $+, \times, /$ or $-$. However, kinematics of chains of bodies typically are computed sequentially resp. recursively, which create a data bottleneck difficult to parallelize [3, 4]. This work analyzes the possibility of forcing parallelization of kinematics of serial chains by using values of position and velocity from previous time steps of an explicit Euler integrator to break recursiveness and thus provide a means of fully parallelizing dynamical equations of minimal order:

$$\boldsymbol{M}(\underline{q}) \cdot \underline{\ddot{q}} + \underline{b}(\underline{\dot{q}}, \underline{q}) = \underline{Q}(\underline{\dot{q}}, \underline{q}) \tag{1}$$

The basic idea is that the system matrices $\boldsymbol{M}(\underline{q})$ and $\underline{b}(\underline{\dot{q}}, \underline{q})$ do not vary significantly from step to step and thus can be computed from intermediate expressions from previous time steps. In this abstract both the influence of using past time steps on accuracy and comparison of computation time for CPU vs. GPU is analyzed. The code is developed using NVIDIA CUDA as it is a well maintained and documented library [5]. The set of equations can be obtained in the minimal form using *kinematical differentials* resp. pseudo velocities and accelerations [6] corresponding to the Jacobian approach. Calculations can be divided into GPU blocks containing parallel calculations (depth) that are executed sequentially due to inter-dependency constraints (breadth) (see Fig. 1 for a 2R mechanism with five blocks).
By allowing the blocks to use previous values for position and velocity, the blocks can be computed in parallel, and thus the system becomes fully parallelizable. However, then, the equations will not be exact anymore, and thus errors might accumulate. An analysis of errors (Tab. 1) for a sample simulation of a 2R serial chain (Fig. 2a) using MATLAB shows however that the difference between the "exact" Euler step and the reference solution using Runge-Kutta scheme of ode45() is orders of magnitude larger than

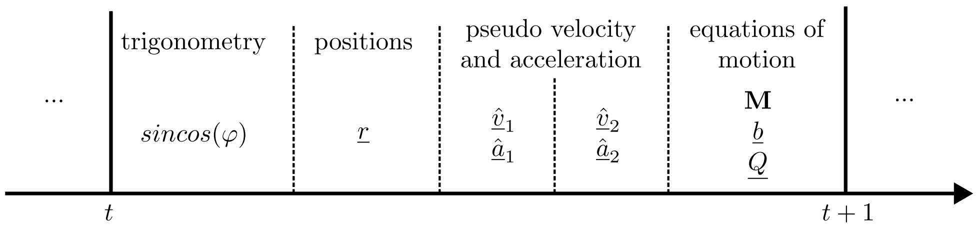
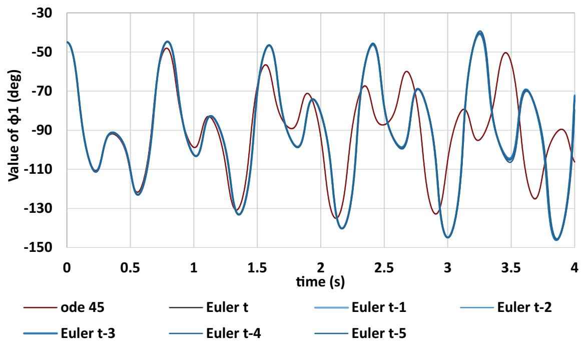


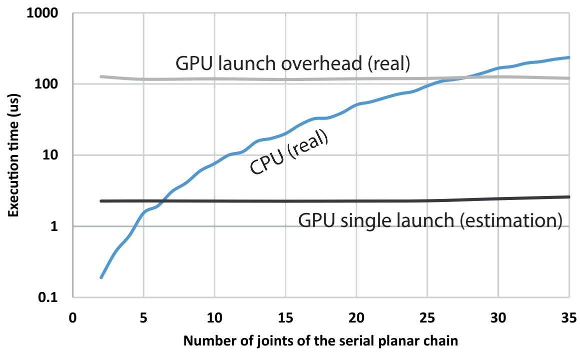
Figure 1: Execution timeline of parallel blocks.

Table 1: Errors caused by delayed values on Euler compared to ode45.

Timestep	Angle	Euler	t-1		t-3		t-5	
μs	deg	diff. ode45	diff. ode45	diff. Euler	-	-	-	-
1000	φ_1	55.2858	55.2947	0.0089	55.2645	-0.0213	55.6067	0.3209
	φ_2	128.3012	128.2902	-0.0110	128.3398	0.0386	128.2627	-0.0385
100	-	7.2605	7.2612	0.0007	7.2583	-0.0022	7.2509	-0.0096
	-	16.7775	16.7789	0.0014	16.7734	-0.0041	16.7598	-0.0177
50	-	2.6862	2.6863	0.0001	2.6856	-0.0006	2.6834	-0.0028
	-	6.2337	6.2340	0.0003	6.2325	-0.0012	6.2284	-0.0053



(a) Values for φ_1 with different solvers and delays for an integration timestep of 1ms.



(b) GPU execution time using delayed values.

Figure 2: Effect of delayed values on the calculation of dynamics.

the difference between the Euler scheme with increasing delay (t-1, t-3, t-5) and the one without, with no visible difference between the Euler schemes with different delays.
The result of parallelization on the GPU is shown on Fig. 2b where the performance of a GTX 1060 GPU is compared to an Intel i7-7700 CPU for the execution of a planar serial chain with a growing number of joints. The GPU timing remains closely to constant while the CPU shows a quadratic complexity. However, the GPU is slowed-down by constant hundred microseconds due to kernel launching and synchronizing overhead (upper curve). An estimation of GPU time in which the launching is executed only once using persistent kernels (lower curve) is planned to be implemented and running at the conference.
In conclusion, this work shows that there is a significant potential for computation-time optimization for forward-dynamic simulations of multibody dynamics using GPUs which has not yet been analyzed in the literature. Further research is currently in progress to analyse the errors due to time shifts, introduce divide-and-conquer method for chain slicing, remove the launching overhead, include closed loops and spatial systems and regard multi-step integrators.

References

[1] Wu, E. and Liu, Y.: Emerging technology about GPGPU. IEEE Asia Pacific Conference on Circuits and Systems, pages 618–622, 2008.

[2] Guigon, L., Boudon, B. and Kecskemethy, A.: Dynamics of a 3R spatial robot based on a GPU approach. ECCOMAS Thematic Conference on Multibody Dynamics, 2023.

[3] Postiau, T.: Génération et Parallélisation des Equations du Mouvement de Systèmes Multicorps par l'Approche Symbolique. PhD thesis, Université catholique de Louvain, 2004.

[4] Featherstone, R.: A Divide-and-Conquer Articulated-Body Algorithm for Parallel O(log(n)) calculation of Rigid-Body Dynamics. The International Journal of Robotics Research, 1999.

[5] NVIDIA.: CUDA Toolkit Documentation. NVIDIA Corporation, 2023.

[6] Angeles, J. and Kecskemethy, A.: Kinematics and Dynamics of Multi-Body Systems. CISM International Centre for Mechanical Sciences, Springer, 2014.