Advances in Engineering Software 86 (2015) 29–38

Contents lists available at ScienceDirect

Advances in Engineering Software

journal homepage: www.elsevier.com/locate/advengsoft

# A high performance crashworthiness simulation system based on GPU

Yong Cai [a], Guoping Wang [a,*], Guangyao Li [b,*], Hu Wang [b]

[a] School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China
[b] State Key Laboratory of Advanced Design and Manufacturing for Vehicle Body, Hunan University, Changsha 410082, China

ARTICLE INFO

*Article history:*
Received 17 November 2014
Received in revised form 24 March 2015
Accepted 6 April 2015
Available online 22 April 2015

*Keywords:*
Crashworthiness
Explicit finite element
Graphics processing units
CUDA
Parallel programming
CAE

ABSTRACT

Crashworthiness simulation system is one of the key computer-aided engineering (CAE) tools for the automobile industry and implies two potential conflicting requirements: accuracy and efficiency. A parallel crashworthiness simulation system based on graphics processing unit (GPU) architecture and the explicit finite element (FE) method is developed in this work. Implementation details with compute unified device architecture (CUDA) are considered. The entire parallel simulation system involves a parallel hierarchy-territory contact-searching algorithm (HITA) and a parallel penalty contact force calculation algorithm. Three basic GPU-based parallel strategies are suggested to meet the natural parallelism of the explicit FE algorithm. Two free GPU-based numerical calculation libraries, cuBLAS and Thrust, are introduced to decrease the difficulty of programming. Furthermore, a mixed array and a thread map to element strategy are proposed to improve the performance of the test pairs searching. The outer loop of the nested loop through the mixed array is unrolled to realize parallel searching. An efficient storage strategy based on data sorting is presented to realize data transfer between different hierarchies with coalesced access during the contact pairs searching. A thread map to element pattern is implemented to calculate the penetrations and the penetration forces; a double float atomic operation is used to scatter contact forces. The simulation results of the three different models based on the Intel Core i7-930 and the NVIDIA GeForce GTX 580 demonstrate the precision and efficiency of this developed parallel crashworthiness simulation system.



## 1. Introduction

Crashworthiness simulation system is one of the key computer-aided engineering (CAE) tools for the automobile industry [1]. It is widely used in crashworthiness design [2–4]. With the usage of simulation system, an enormous amount of expensive and time-consuming physical tests are greatly reduced. However, due to the high nonlinearity of contact–impact problems, such simulations contain several computationally intensive parts such as Gaussian integration and contact searching. For example, a crashworthiness simulation of an energy-absorbing structure with a fine-meshed FE model usually consumes dozens of hours. Therefore, computational expense is a major bottleneck of the crashworthiness simulation in real engineering problems. The purpose of this work is to improve the efficiency of crashworthiness simulation under high accurate solution significantly.

Previously, the most of contributions for improving the performance of solver are based on the FE theories and the contact algorithms. Hughes [5–7] and Belytschko [8–10] proposed several advanced shell theories to improve the accuracy and stability of the nonlinear explicit FE method. A large number of researchers suggested several contact search algorithms for contact detection [11,12]. Zhong [13] published the first book on Finite Element Method (FEM) modeling of contact–impact events and deals with FE procedures for solutions to both static and dynamic contact–impact problems. On the other hand, supercomputing involving parallel computing has become research hotspots. Parallel computing is a direct way to improve the computational efficiency. The parallel implementations of shell element formulation have been reported in both literatures and commercial codes. The results of these studies show that the nonlinear explicit FE analysis can be accelerated by parallel computation significantly [14–16]. Therefore, many researchers make their efforts to realize the parallel implementation of contact searching. In the early stages, Belytschko et al. [17] presented a parallel explicit FE method for the contact–impact problem on a SIMD computer. Namburu and Turner [18] presented a contact–impact algorithm on a data-parallel computer. Besides, great efforts have been made to develop decomposed impact simulation algorithms executing on network-based parallel computing architecture, such as the multi-processors based architectures using distributed memory

* Corresponding authors.
*E-mail addresses:* wgp@pku.edu.cn (G. Wang), gyli@hnu.edu.cn (G. Li).

http://dx.doi.org/10.1016/j.advengsoft.2015.04.003
0965-9978/

processing (DMP) and the multi-cores based architectures using shared memory processing (SMP) [19–21]. These efficient parallel implementations enlarged the scale of CAE model, thus more complex structures could be modeled with fine meshes [21]. However, the disadvantages of these traditional parallel computations are obvious. Firstly, the hardware cost of CPU-based parallel architecture is too expensive, and it is more difficult to use special programming language to code parallel program based on supercomputers with hundreds or thousands of processors. Secondly, the administration and maintenance costs are super-linear rising along with growing demand for computation power. Therefore, an alternative way for CPU-based parallel program is needed urgently to reduce the cost of parallel computing. In the past couple of years, we turned our attention to general purpose computation on graphics hardware (GPGPU).

Nowadays, GPU offers a tremendous amount of computing resources not only for graphics processes but also for general-purpose parallel computations. These general parallel computing resources include massive processing cores, high memory bandwidth, and general-purpose instruction sets. In the GPU-based parallel program, GPU is commonly used as a coprocessor to execute easy parallel sections. For now, a large number of high performance implementations of FE applications based on GPU have been reported. Göddeke et al. [22] have successfully implemented their FE algorithm on a GPU enhanced cluster to solve implicit FE problem with multi-grid algorithm. For nonlinear FE analysis, Mafi and Sirouspour [23] proposed a GPU-based implementation of FE method using implicit time integration for dynamic nonlinear deformation analysis. Ikushima and Shibahara [24] presented an idealized explicit FEM accelerated by GPU to predicted the residual stresses in multi-pass welded joint. Furthermore, FE analysis involves fluid–solid coupling [25], structural analysis [26], higher order numerical integration [27] and others are parallelized by GPU successfully. Our research team also developed a GPU-based parallel sheet metal forming simulation system, which achieved up to 27X speedup using GTX285 GPU [28].

In this paper, GPU was used to implement the explicit FE method with a hierarchy-territory contact-searching algorithm (HITA) and a penalty contact force calculation algorithm to simulate car crash. The remainder of this paper is organized as follows. In Section 2, the mathematical model and numerical scheme of contact–impact problems are briefly introduced. In Section 3, the typical CUDA programming model and the details of the GPU implementation for the whole simulation algorithm are presented. Numerical experiments are used to evaluate the performance of the developed parallel simulation system in Section 4. Finally, conclusion remarks are presented in Section 5.

## 2. Mathematical model and numerical scheme

To simulate contact–impact process with FE method, contact boundaries are usually approximated by a collection of segments, and contacts are considered at the discrete contacting nodes. The most widely used segments of shell structure are triangular shell element and quadrilateral shell element, as shown in Fig. 1. The discretized contact solution divides the solution domain into

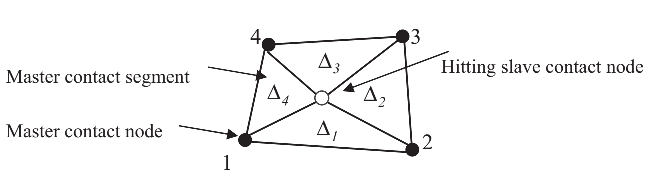


**Fig. 1.** A quadrilateral segment.

discrete elements, and then, numerical interpolations are performed within these elements through shape functions. Furthermore, to facilitate the contact searching and contact force calculation, these segments usually correspond to a low order shell element, such as the Belytschko–Tsay (BT) element [8].

For convenience, contact segments are usually described as a multiple hierarchy system consists of segments, edges and nodes. When two body boundaries come into contact, one is specified as a master one, and the other is specified as a slave one. The segment, edge and node on master body called master segment, master contact edge and master contact node, respectively. Similarly, the segment, edge and node on slave body called slave segment, slave contact edge and slave contact node, respectively.

### 2.1. Contact searching algorithm

Master–slave algorithm is the most common algorithm for contact searching, which was first introduced by Hallquist [29]. However, this algorithm must specify master and slave segments respectively, and in some situations two boundaries may come into contact before the searching begins. These two drawbacks limit its ability to search contacts during the crashworthiness simulation, which usually has large displacements and rotations. Therefore, a more efficient and effective contact searching algorithm named HITA is used in this work [13].

The HITA method reduces the redundant contact searches by taking the advantage of a multiple-hierarchy contact system to improve its searching efficiency. In a hierarchical contact system, contact searching should be first performed between the higher level hierarchies and then between the lower level ones. If the contacts are rejected between two higher level contact segments, searching in the lower level hierarchies will not be performed. More importantly, the data independences of each segment in each hierarchy are well suited for GPU implementation. The following section provides a brief introduction of searching strategy based on the hierarchy territory techniques.

Firstly, the hierarchy territory of a segment is the smallest rectangular box which has its edges parallel to the global coordinate axes and contains the complete segment, as the solid line box shown in Fig. 2. Mathematically, it is a domain which can be defined as

$$T = \{(x_1, x_2) | x_i^a \le x_i \le x_i^b, i = 1, 2\} \tag{1}$$

where $x_i^a$ and $x_i^b$ denote the minimum and the maximum coordinates of node 1 and node 2, respectively.

Secondly, a contact territory is defined due to the rounding error of computer, as the shaded parts shown in Fig. 2. $C_d$ is the control distance, and, $C_p$ is the allowed maximum penetration.

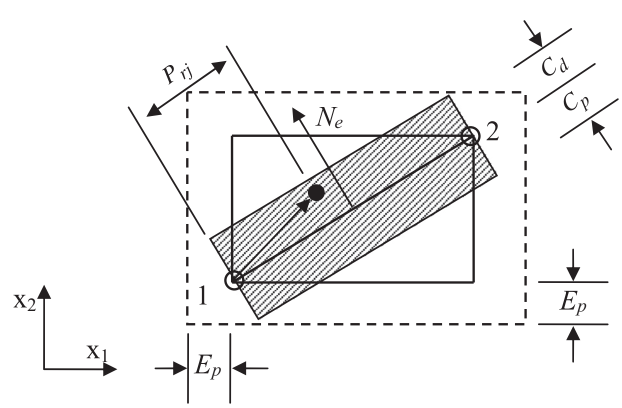


**Fig. 2.** Definition of different kind of territories for a segment.