Simulation of shock-body interactions using the Runge–Kutta discontinuous Galerkin method with adaptive mesh refinement implemented on graphic accelerators

B A Korneev¹,², A V Zakirov² and V D Levchenko¹,²
¹ Keldysh Institute of Applied Mathematics of the Russian Academy of Sciences, Miusskaya Square 4, Moscow 125047, Russia
² Kintech Lab Limited, 3rd Khoroshevskaya Street 12, Moscow 123298, Russia
E-mail: boris.korneev@phystech.edu

Abstract. In this paper, a numerical scheme for solving the equations of the dynamics of a compressible fluid, based on the Runge–Kutta discontinuous Galerkin (RKDG) method and an adaptively refined cubic mesh (AMR), is applied to the problems of shock interaction with solid and liquid obstacles. Features of effective implementation algorithm for the graphic processors (GPU) are described. Several well-known validation test problems are considered. Results of full detailed three-dimensional simulations of the shock with the gas or fluid bubble using the developed GPU–RKDG–AMR solver are presented.

1. Introduction
Fluid dynamics models are used to describe the processes in a gas or liquid at high pressures and temperatures, i.e., shock waves propagation, shock interaction with the matter [1], detonation and combustion phenomena. Fluid dynamics models are parts of multiphysical models for laser or electron beam interaction with metals, astrophysical models [2], plasma physics models.

Numerical modeling is the main tool for studying non-stationary hydrodynamic processes in four-dimensional space-time. Speaking about numerical methods, the following classes can be distinguished: finite-difference, finite-volume and finite-element schemes by the type of discretization in space, explicit and implicit by the type of discretization in time. Runge–Kutta discontinuous Galerkin (RKDG) method [3] for numerical solution of unsteady equations of gas dynamics, considered in the paper is a synthesis of the finite-volume method and the finite element Galerkin method. When approximating fluxes, a method based on solving the Riemann problem is used like in the Godunov-type methods [4, 5]. The method is explicit in time. Runge–Kutta discontinuous Galerkin method was originally formulated for the non-linear advection equation, has a generalization for hyperbolic problems, in particular for fluid dynamics problems in the framework of the system of Euler equations considered in the paper, and the method extensions have been developed to solve problems of other types: convection-diffusion problems, in particular, the system of Navier–Stokes equations, and the pure elliptic tasks [6].

Since the late twenties, high-performance computer technologies are commonly used for solving computational problems of fluid dynamics. One of the main challenges for the correct
and accurate simulation of problems with high Mach flows with shocks or spreading vortices is a very fine grid resolution required near the flow features. Among the variety of ways to obtain such a resolution, one is to refine the grid near such zones and, if necessary, rebuild the mesh followed by the movement of the shock or the vortex. Such a procedure is called the adaptive mesh refinement or shortly the AMR \cite{7} or $h$-AMR, if we want to stress the adaptivity in space, not the so-called $p$-adaptivity, when the order of approximation of the numerical scheme is locally increased \cite{8}. The $p$-adaptivity is not considered in the paper.

Among modern computers, graphic accelerators have established themselves as powerful computing units \cite{9}. In this paper we focus the implementation of the selected RKDG numerical scheme and the AMR technique using the graphic processors (GPU). It is known that the special data structure for the AMR is to be used. Commonly used data structure is based on the abstract structure of the quadtree for two-dimensional space or the octree for three-dimensional space with different optimizations \cite{7, 10}. Another aspect is to use block-structured or tiled AMR, which seems to be helpful for the optimization as well \cite{11, 12}. Finally, the approach should be well-compatible with the features of graphic processors.

In the next section the mathematical model and the numerical method is formulated. Then, the implementation of the AMR, including the data structure used in the work, the algorithm of calculating the mesh values and its remeshing if needed is described. The fourth section is dedicated to the results of the validation and simulation results of the implementation. Finally, the conclusion of the work is given.

2. Mathematical and numerical model
2.1. Governing equations

The system of equations describing the dynamics of a compressible fluid in three-dimensional space is presented as follows

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}_i}{\partial x_i} = 0,$$

where $\mathbf{U} = (\rho, \rho u_i, E)$ are conservative variables; $E = \rho(\varepsilon + \frac{1}{2} u^2)$ is full energy; $\mathbf{F}(\mathbf{U})_i = [\rho u_i, \rho u_i u_j + P_{ij}, u_i E + P_{ij} u^j]$ are fluxes; tensor $P_{ij} = p\delta_{ij}$ as we use the Eulerian model with neither viscous stresses not thermal conductivity. The system is completed by the equation of state $p = p(\rho, \varepsilon)$.

2.2. Runge–Kutta discontinuous Galerkin numerical method

The system 1 is in divergence form and is hyperbolic. The numerical solution is built in the domain $G \in \mathbb{R}^3$, $t > 0$ with the boundary $\partial G$ with given initial conditions at $t = 0$ and boundary conditions at $x \in \partial G$.

Domain $G$ is discretized using the finite volume grid. At every cell $L$ the number of basis functions $\{\varphi^n\}$ is introduced. The numerical solution is built for every component $\mathbf{U}$, as a basis expansion, while the basis coefficients are time-dependent. Thus, the numerical solution can be presented as an expression

$$\mathbf{U}(x, t) = u_n(t)\varphi^n(x).$$

Numerical approximation (2) is put into (1), then we require the residual to be orthogonal to the basis functions. Integrating over the volume $L$ is reformulated using the divergence theorem, so we obtain the surface integral term over $\partial L$. Finally we get

$$\frac{du_n(t)}{dt} \int_L \varphi^n \varphi_m dV + \int_{\partial L} \mathbf{F}_i \varphi_m n^i d\Sigma - \int_L \mathbf{F}_i \frac{\partial \varphi_m}{\partial x_i} dV = 0.$$
In case of orthonormal basis \( \int_L \varphi^n \varphi_m dV = \delta^m_n \) and
\[
\frac{du^m(t)}{dt} + \int_{\partial L} F^E_{iu_i} n^i d\Sigma - \int_L F^E_{iu_i} \frac{\partial \varphi_m}{\partial x^i} dV = 0.
\]
Here \( n^i \) is a normal vector to \( \partial L \).

In the terms of expression (4) containing integrals over the surface of the cell boundary, it is necessary to approximate the integral using the values in the neighboring cells forming the face of the surface. For the flux \( F^E_{iu_i} \), a special numerical flux is usually used based on the solution of the Riemann problem. In this work, the flux suggested by Toro [5] is used, called the HLLC flux. Integrals are replaced by quadrature formulas of the desired order, finishing the discretization in space.

The system of ordinary differential equations (4) is solved using the explicit total variation diminishing (TVD) Runge–Kutta method [13]. Special procedure of limiting must be used to remove spurious oscillations near the big gradients and discontinuous solutions, i.e., shocks and contacts. Limiter is a local function decreasing the possibly unphysical slopes using the neighboring cell data, not decreasing the order of accuracy in the smooth regions. Several types of limiters are developed [14–16].

3. Adaptive mesh refinement implementation

3.1. Data structure

The basic data structure is the octal tree forest, i.e., the array of structures of the following form:

```c
struct Node{
    int S, isLeaf, adviseS;
    Node* parent;
    Node* children;
    Node* nears[3];
    Tile data;
};
```

Such flags as being a leaf, rank of the node \( S \) and the preferable or advised rank are saved. Also we keep five pointers: one to the parent, one to the first child (access to other seven children can be obtained using simple pointer algebra as all eight children are put successively in the memory) and half of six neighbors. One can show that it is enough for \( O(1) \) access to all neighbors required in the stencil computations. One can considered the proposed structure to be modified version of the fully threaded tree [10].

Note that the data are kept in tiles. One tile is a cubic block of numerical cells. We define \( N_v \) as the width of the tile in cells. We use \( N_v = 4 \) for most simulations, therefore each tile contains \( 4 \times 4 \times 4 \) cells. Tiling is essential for the highly-parallel implementation [12]. Data storage can be illustrated using the two-dimensional cross-section as shown in figure 1.

3.2. Order of calculations

The implementation is developed using the compute unified device architecture (CUDA). Inside the tile, the calculations are parallelized using the CUDA threads, while each tree is processed by the CUDA block from the root to the leaves. The synchronization is made for the grid after each stage, i.e., the Runge–Kutta step, limiter step or mesh reform step.

Note that all operations needed to perform for the implementation algorithm, such as the numerical stencil operations [17] or AMR operations, are local, so the synchronization is not necessary at each time step. For such cases we suggest to use the locally-recursive non-locally asynchronous algorithms for that [18]. However, in this paper, we used the simplest stage-wise synchronization algorithm.
4. Simulation results

4.1. Accuracy validation

We use RKDG numerical methods of second and third orders in the implementation. For the validation of the order we use a simple advection test with a smooth initial condition.

The following setting is used. The domain is a unit cube $[0,1]^3$, where pressure is constant, density field is defined as follows

$$\rho(x, y, z, 0) = \sin 2\pi x \sin 2\pi y \sin 2\pi z,$$

$(x, y, z) \in (0, 1)^3$, (5)

velocity $(u, v, w) = (1,1,1)$. Periodic boundary conditions are used. This test has the analytical solution, so we can compare it with the numerical one.

The error is defined by the following equation:

$$\varepsilon_1 = ||U_a - U_h||_1,$$

(6)

where $||\cdot||_1$ is the $L_1$ norm, $U_a$ is the analytical solution, $U_h$ means the numerical solution, index $h$ denotes the space discretization step.

The convergence test is carried out, where the dependency of the discretization step $h$ on the error $\varepsilon_1$. The Courant number is kept constant for all the simulations. The result of the test, presented in figure 2, shows, that, at least for smooth solutions, we expect the solver to have the proper order of convergence.

4.2. Riemann problem. Shock capturing

We use a number of tests known as the Riemann problem when the initial condition is determined by two constant states

$$U(x, 0) = U_L, \quad x \leq x_0; \quad U(x, 0) = U_R, \quad x > x_0$$

(7)

with the discontinuity at $x = x_0$.  

---

**Figure 1.** AMR tiles data structure in two-dimensional section. From left to right: one tile of coarse level (blue), two tiles of more fine level (green), and four tiles of the finest level (red). 4 × 4 numerical cells are contained in the tile of each level.
Figure 2. Advection test problem. The error of the numerical solution $\varepsilon_1$ (6) depending on the mesh size $h$, is shown using the logarithmic scale.

The first test is a one-dimensional Riemann problem with

$$\begin{align*}
(p_L, u_L, p_L) &= (1, 1, 1), \\
(p_R, u_R, p_R) &= (1, -1, 1).
\end{align*}$$

The solution is two shocks moving left and right. In figure 3 we see that the grid is refined near the shocks and is coarsened elsewhere.
Figure 4. Sedov blast wave propagation. Pressure ratio equals to $p_{\text{blast}}/p_0 = 10^5$. Mesh cells for the coarse levels of refinement are drawn.

Also we set the Sedov-type [19] problem of explosion of a spherical object. As our method is on the Cartesian grid, the problem has the three-dimensional setting. Numerical domain is a cube $[-1,1]^3$ filled with gas having unit density, pressure and zero velocity. In the center of the domain we set a small spherical region with the pressure increased $10^5$ times. The cross-section shown in figure 4 at a specific time moment represents the impact of the developed adaptive grid refinement technique to capture the shock. Despite of being naturally spherically asymmetric due to the Cartesian grid effect, the solution seems to keep high value of symmetry.

4.3. Bubble-shock interactions

The problem of shock interacting with an inhomogeneity in the fluid, in particular, the spherical region filled up with another gas, or shortly the bubble-shock interaction problem in the gas is considered here as another illustrative problem of well-applicability of the AMR approach [20].

Setting with the low-density bubble is simulated and the results are shown in figure 5. For the dense bubble the results of the simulations are shown in figure 6. For both settings we suppose the diameter of the bubble $D$ and the density outside the bubble $\rho_\infty$ to be unit. The shock condition is defined by the Rankine–Hugoniot conditions. Unit of measure for time in figures 5 and 6 is selected as the ratio of initial diameter of the bubble $D$ and the shock movement $M_{c\infty}$, so $[t] = [D][M_{c\infty}]^{-1}$.

We see how the features of the solution are resolved on the finest grid, while the grid near the non-perturbed gas is being coarsen. The solution is similar to the well-known patterns for both cases [20].
Figure 5. Helium bubble in air interacting with a shock-wave front moving with Mach number $M = 3$. Mesh cells for the coarse levels of refinement are drawn. Time evolution is shown from top left to bottom. $t = 0.1$ (a), 0.6 (b), 1.1 (c), 2.0 (d), 3.0 (e).
Figure 6. Heavy bubble in air (the ratio of the density inside the bubble to the density around the bubble is $\rho_B/\rho_\infty \approx 4.65$) interacting with a shock moving with speed $M = 3$. Mesh cells for the coarse levels of refinement are drawn. Time evolution is shown from top left to bottom. $t = 0.1$ (a), 0.6 (b), 1.1 (c), 2.0 (d), 3.0 (e), 4.0 (f).
5. Conclusion
The effective GPU implementation of the RKDG method on adaptively refined three-dimensional cubic mesh for the simulation of large-scale fluid dynamics problems with shocks is discussed. We suggest the data structure and its parallel usage algorithm. Solver validation tasks and results of simulation of the bubble-shock interaction problem are shown.

Acknowledgments
The work is supported by the Russian Science Foundation, grant No. 18-71-10004.

References
[1] Korneev B and Levchenko V 2016 Fluid Dyn. Res. 48 061412
[2] Rybakin B and Goryachev V 2018 Comput. Fluids 173 189–94
[3] Cockburn B and Shu C W 2001 J. Sci. Comput. 16 173–261
[4] Godunov S K 1959 Mat. Sb. 89 271–306
[5] Toro E F 2009 Riemann Solvers And Numerical Methods for Fluid Dynamics: A Practical Introduction (Springer)
[6] Cockburn B and Shu C W 1998 SIAM J. Numer. Anal. 35 2440–63
[7] Berger M J and Colella P 1989 J. Comput. Phys. 82 64–84
[8] DeCounguy H, Devine K, Flaherty J, Loy R, Ozturan C and Shephard M 1994 Appl. Numer. Math. 16 157
[9] Nickolls J and Dally W J 2010 IEEE Micro 30 56–69
[10] Kravtsov A V, Klypin A A and Khokhlov A M 1997 Astrophys. J., Suppl. Ser. 111 73
[11] Van der Holst B et al 2011 Astrophys. J., Suppl. Ser. 194 23
[12] Ivanov A, Perepelkina A, Levchenko V and Pershin I 2019 Memory-optimized tile based data structure for adaptive mesh refinement Russian Supercomputing Days pp 64–74
[13] Gottlieb S and Shu C W 1998 Math. Comput. 67 73–85
[14] Krivodonova L 2007 J. Comput. Phys. 226 879–96
[15] Zhu J, Qiu J, Liu T and Khoo B C 2011 Appl. Numer. Math. 61 554–80
[16] Liang C and Xu Z 2014 J. Sci. Comput. 58 41–60
[17] Korneev B and Levchenko V 2018 Simulating three-dimensional unsteady viscous compressible flow on GPU using the DiamondTorre algorithm Preprint 105 (Keldysh Institute of Applied Mathematics, Russian Academy of Sciences)
[18] Korneev B and Levchenko V 2015 Procedia Comput. Sci. 51 1292–302
[19] Sedov L I 1946 J. Appl. Math. Mech. 10 241–50
[20] Niederhaus J H, Greenough J, Oakley J, Ranjan D, Anderson M and Bonazza R 2008 J. Fluid Mech. 594 85–124