EPJ Web of Conferences **45**, 01068 (2013)
DOI: 10.1051/epjconf/20134501068

Meshfree methods for computational fluid dynamics

P. Niedoba[1,a], L. Čermák[1], and M. Jícha[1]

[1]Faculty of Mechanical Engineering, Brno University of Technology, Technická 2896/2, 616 69 Brno, Czech Republic

Abstract. The paper deals with the convergence problem of the SPH (Smoothed Particle Hydrodynamics) meshfree method for the solution of fluid dynamics tasks. In the introductory part, fundamental aspects of meshfree methods, their definition, computational approaches and classification are discussed. In the following part, the methods of local integral representation, where SPH belongs are analyzed and specifically the method RKPM (Reproducing Kernel Particle Method) is described. In the contribution, also the influence of boundary conditions on the SPH approximation consistence is analyzed, which has a direct impact on the convergence of the method. A classical boundary condition in the form of virtual particles does not ensure a sufficient order of consistence near the boundary of the definition domain of the task. This problem is solved by using ghost particles as a boundary condition, which was implemented into the SPH code as part of this work. Further, several numerical aspects linked with the SPH method are described. In the concluding part, results are presented of the application of the SPH method with ghost particles to the 2D shock tube example. Also results of tests of several parameters and modifications of the SPH code are shown.

1 Introduction

Currently, fluid dynamics problems are solved mostly by traditional numerical methods, such as FDM, FVM or FEM. The common feature of these methods is the use of a lagrangian mesh or eulerian grid in the domain discretization process. However, for some types of problems these methods are suited poorly. Problems in question are especially those with an extreme deformation, a moving boundary or a free surface. The complications arising while solving these problems result from the use of a mesh or a grid.

And so the idea of meshfree methods evolves naturally, first being used in 1977 by Lucy L.B. [1] and Gingold R.A. & Monaghan J.J. [2]. Specifically, it was the SPH method applied to astrophysical problems of modeling the movement of stars and space objects.

The main idea of meshfree methods lies in modeling of the domain through field nodes without any information about relations between these nodes. Consequently, function approximation is performed with the help of field nodes in support domains.

2 Meshfree methods

A definition of meshfree methods according to [3] is following: a method used to create a system of algebraic equations for the entire domain without using a predefined mesh for the domain discretization is defined as a meshfree method.

2.1 Solution procedure

The procedure of meshfree methods consists of four basic steps:

- domain representation
- function approximation
- formation of system equations
- solving the global equations

2.1.1 Domain representation

First, the domain and its boundary is modeled (not discretized!) using sets of arbitrarily distributed nodes (see figure 1) in the domain and its boundary. The nodal distribution is usually not uniform. The density of nodes depends on the accuracy requirement of the analysis. Because the nodes carry the values of a field variable (e.g. density, velocity, etc), they are often called *field nodes*. Further in the text, a field variable will be refered to as a *field function*.

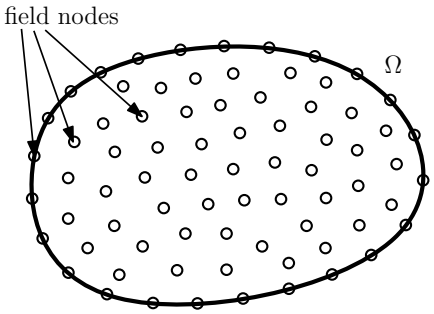

Fig. 1. Domain representation

[a] e-mail: P.Niedoba@seznam.cz

2.1.2 Function approximation

The field function u at any point at $\mathbf{x} = (x, y)$ within the domain is approximated using the values at its nodes within the "small" local domain of the point $\mathbf{x}$, i.e.

$$u(\mathbf{x}) = \sum_{i=1}^{n} \phi_i(\mathbf{x})\, u_i \tag{1}$$

where n is the number of nodes included in a local domain of the point at $\mathbf{x}$, u_i is the nodal field function at the $i-$th node in the local domain, and $\phi_i(\mathbf{x})$ is the *shape function* of the $i-$th node. The "small" local domain of $\mathbf{x}$ will be called the *support domain* of $\mathbf{x}$ and denoted $\Omega_{\mathbf{x}}$. The size of support domain defines the number of field nodes approximating $\mathbf{x}$. Some possible shapes of support domains are shown in figure 2.

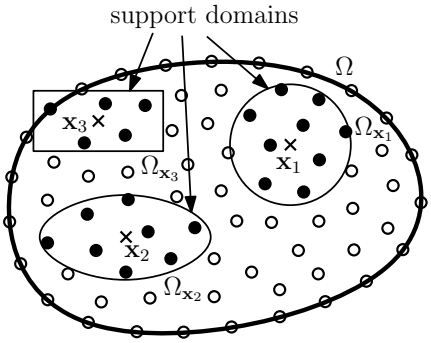

Fig. 2. Support domains (spherical is the most common one)

2.1.3 Formation of system equations

System equations can be formulated using the shape functions and strong or weak formulation. These equations are assembled into the global system matrices for the entire problem domain. For static problems, the global system equations are a set of algebraic equations. For general dynamics problems, it is a set of differential equations.

2.1.4 Solving the global equations

The last step depends on the type of equations (algebraic, differential, etc). Note that the global equations for computational fluid dynamics problems are basically nonlinear.

2.2 Classification

Meshfree methods will be classified according to the function approximation schemes [3]. Finite integral representation methods include SPH - Smoothed Particle Hydrodynamics method and RKPM - Reproducing Kernel Particle Method. Finite series representation methods include MLS - Moving Least Square, PIM - Point Interpolation Method, and FPM - Finite Point Method. The last category is called the finite differential representation methods and includes GFDM - General Finite Difference Method.

This paper details two of these methods - SPH and RKPM - that fall into the first category listed above.

3 SPH method

The smoothed particle hydrodynamics method belongs to basic meshfree methods. It is used for solving partial differential equations. A system of ordinary differential equations is produced after approximation of unknown functions (field function) and their spatial derivatives. This system is most often solved by explicit numerical methods.

3.1 Formulation

Function approximation of the field function $u(\mathbf{x})$ is based on an integral representation of the function and is given by the equation

$$< u(\mathbf{x}) > = \int_{\Omega_{\mathbf{x}}} u(\boldsymbol{\xi}) W(\mathbf{x} - \boldsymbol{\xi}, h)\, d\boldsymbol{\xi} \tag{2}$$

where $W\left(\mathbf{x} - \boldsymbol{\xi}_j, h\right)$ is the *weight function* (*smoothing function, kernel function*), h being the *smoothing length*, which defines the size of the support domain $\Omega_{\mathbf{x}}$, i.e. the smoothing length determines the number of particles approximating the function at $\mathbf{x}$. The weight function is usually chosen to be an even function and it satisfies number of conditions, e.g. the *normality condition*

$$\int_{\Omega_{\mathbf{x}}} W(\mathbf{x} - \boldsymbol{\xi}, h)\, d\boldsymbol{\xi} = 1. \tag{3}$$

Equation (2) is usually referred to as *kernel approximation*, or *SPH approximation* of function $u(\mathbf{x})$.

For practical calculation, the equation (2) must be discretized as follows

$$< u(\mathbf{x}) > = \sum_{j=1}^{n} u\left(\boldsymbol{\xi}_j\right) W\left(\mathbf{x} - \boldsymbol{\xi}_j, h\right) \frac{m_j}{\rho_j} \tag{4}$$

where m_j and ρ_j are mass and density of the $j-$th particle in $\Omega_{\mathbf{x}}$. Equation (4) is called a *particle approximation* of field function $u(\mathbf{x})$.

Note that the approximation (4) corresponds to the approximation (1) introduced for a general meshfree method. The shape function in this case has the form of

$$\phi_j(\mathbf{x}) = W\left(\mathbf{x} - \boldsymbol{\xi}_j, h\right) \frac{m_j}{\rho_j}. \tag{5}$$

Approximation of the spatial derivatives of the field function can be obtained by replacing the function $u(\mathbf{x})$ in equation (2) with its spatial derivative $\nabla \cdot u(\mathbf{x})$. Using the per-partes, the Green theorem and a discretization we obtain a particle approximation of the spatial derivative of the field function in the form of

$$< \nabla \cdot u(\mathbf{x}) > = \sum_{j=1}^{n} u(\boldsymbol{\xi}_j) \nabla_{\mathbf{x}} W(\mathbf{x} - \boldsymbol{\xi}_j, h) \frac{m_j}{\rho_j} \tag{6}$$

where $\nabla_{\mathbf{x}} W(\mathbf{x} - \boldsymbol{\xi}_j, h)$ is the spatial derivative of the weight function with respect to the variable $\mathbf{x}$.

We can observe that an approximation of the spatial derivative of a field function is determined using only field function values and derivatives of the weight function.

3.2 Consistency

To ensure the convergence of meshfree method, it is necessary that the method satisfies certain order of consistency. Consistency is closely related to the reproduction of polynomials.

If an approximation can exactly reproduce polynomials of degree k, i.e.

$$< f(\mathbf{x}) > = f(\mathbf{x}) \tag{7}$$

where $f(\mathbf{x})$ is a polynomial of degree k, than we say that the approximation has k-th order of consistency, i.e. C^k consistency.

Specifically, consistency of the SPH approximation (4) is given by the following definition. The SPH approximation has C^k consistency if and only if weight functions satisfy the condition

$$\int_{\Omega_\mathbf{x}} \boldsymbol{\xi}^l W(\mathbf{x}-\boldsymbol{\xi},h)\, d\boldsymbol{\xi} = \mathbf{x}^l \quad \text{for} \quad l = 0, 1, \ldots, k. \tag{8}$$

It can be shown that C^0 consistency follows directly from the normality condition (3), which is a necessary condition for the weight functions. Thus the SPH approximation has always C^0 consistency.

Furthermore, we can prove the theorem, which says that the SPH approximation has C^1 consistency if and only if has C^0 consistency and also the appropriate weight function is an even function.

We see that the order of consistency depends only on the properties of weight functions.

3.3 Boundary treatment

The issue of boundary conditions is generally very difficult in the SPH method. We answer the question of properly defining the boundary condition that prevented particles from escaping out of the domain. Furthermore, we discuss consistency near the boundary of the domain (*near boundary area*).

3.3.1 Virtual particles

The first approach is the use of virtual particles. These particles are situated on the boundary and by repulsive force acting on the particles in the near boundary area (*near boundary particles*). Hence, virtual particles prevent an unphysical penetration through the boundary.

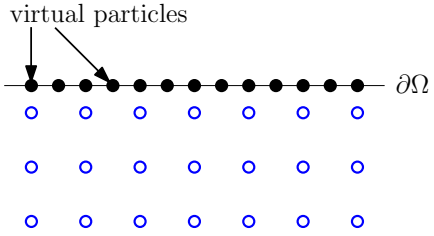

Fig. 3. Virtual particles

Unfortunately, this approach violates the condition for C^1 consistency of the SPH approximation in the near boundary area. This fact is due to the undesirable "cutting off" of the weight function support, see figure 4. Thus, the appropriate weight function is not an even function.

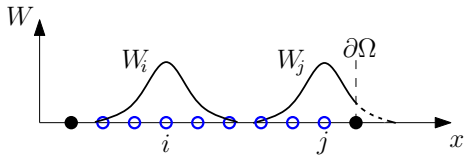

Fig. 4. Example of a 1D task, particle j is situated in the near boundary area

3.3.2 Ghost particles

A much better way is to use ghost particles as a boundary condition. In contrast to virtual particles, this approach creates a dynamic wall that is constructed at each time step. Ghost particles are formed symmetrically (according to the boundary) to the near boundary particles as "twin" particles, see figure 5.

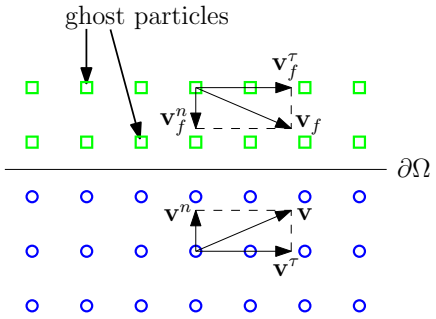

Fig. 5. Ghost particles, velocities are formed symmetrically (slip wall)

Using ghost particles ensures C^1 consistency of the SPH approximation, because the shape functions of the near boundary particles can be even functions.

3.4 RKPM method

The reproducing kernel particle method belongs to the category of finite integral methods, and is a modification of the SPH method. This method adds the so-called correction function to the SPH formulation to ensure certain order of consistency. The particle approximation of the function $u(\mathbf{x})$ is defined as

$$< u(\mathbf{x}) > = \sum_{j=1}^{n} u(\boldsymbol{\xi}_j) C(\mathbf{x}, \boldsymbol{\xi}_j) W(\mathbf{x}-\boldsymbol{\xi}_j, h) \frac{m_j}{\rho_j} \tag{9}$$

where $C(\mathbf{x}, \boldsymbol{\xi})$ is the *correction function*.

Note that the approximation (9) corresponds to the approximation (1), where the shape function has the form of

$$\phi_j(\mathbf{x}) = C(\mathbf{x}, \boldsymbol{\xi}) W\left(\mathbf{x}-\boldsymbol{\xi}_j, h\right) \frac{m_j}{\rho_j}. \tag{10}$$

4 Shock tube 2D problem

In this section we show the results of the shock tube problem in 2-dimensions (2D) depending on the boundary conditions described in section 3.3.

The shock tube problem is a good numerical benchmark that shows the ability of numerical methods to deal with discontinuities. In addition, we know the exact solution of this problem in 1D and therefore we can compare it with the results of the 2D task. These results were obtained using a program based on the SPH method. The program was written in the *Fortran* programming language (Intel(R) Visual Fortran Compiler Professional 11.1.067) in the MS Visual Studio 2008 IDE for .NET Framework v. 3.5. The basis was the source code from the book [4]. Post-processing, i.e. graphs of particular variables was realized using the *matlab* software v. 7.11.0 (R2010b).

4.1 Description of the shock tube 2D problem

It is the examination of the phenomenon which occurs in the rectangular container after removal of an imaginary barrier. This barrier separates two different (density, pressure and energy) fluids. After removing the barrier, we will observe the spread of a shock wave, a rarefaction wave and a contact discontinuity.

4.2 Initial settings

The initial conditions were taken from (L. Hernquist, 1989, [5]) for 1D problem, which has been extended to 2D, in this way

$$\begin{aligned} &\rho_L = 1, && p_L = 1, && v_L = 0, \\ &\rho_R = 0.25, && p_R = 0.1795, && v_R = 0 \end{aligned}$$

where the subscripts L and R denote the fluid on the left and right side of the barrier. Assume that both fluids satisfy properties of an ideal gas, therefore the initial values of the internal energy e are calculated from the equation of state. So we have $e_L = 2.5$ and $e_R = 1.795$.

In the simulation 3200 particles were used. Specifically, 320 in x-direction and 10 in y-direction. The sizes of the domain (container) are $x_l = 0.8$ and $y_l = 0.025$. Spacing between particles is constant with $dx = dy = 2.5 \cdot 10^{-3}$. The center of the container is placed at the origin of the coordinate system ($x = y = 0$). The distribution of particles is shown in figure 6, which is only illustrative due to the large number of particles.

Boundary condition is set as the slip wall and the end time of the simulation was set to $t = 0.2$ s.

4.3 Parameter settings

The shock tube problem is generally a very fast phenomenon, therefore it can be assumed that it is very sensitive to the parameter settings. This assumption have been demonstrated by testing. To achieve the "correct" setting of these parameters, much effort is usually required.

Now we show the parameter settings for the shock tube 2D problem, which has been obtained by a number of tests.

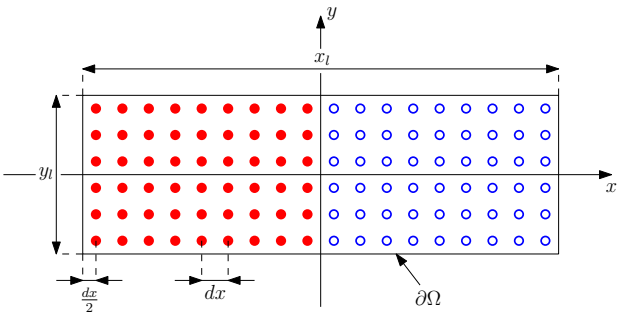

Fig. 6. Initial distribution of particles

Weight function: The exponential weight function has been used for the shock tube problem. The disadvantage of this approach is a global discontinuity at the boundaries of the support domains. The exponential function has derivatives of all orders, which has a positive effect on the stability of the solution. When testing other weight functions (cubic spline, quintic spline), there were significant oscillations in the solution.

Artificial viscosity: The artificial viscosity was used to remove the oscillations which originated in areas of discontinuities. It is a component that is added to the pressure terms of the Euler equations or Navier-Stokes equations.

Variable smoothing length: Due to the variable density of particles, the variable smoothing length is required to preserve an almost constant number of particles in particular support domains during the approximation.

Nearest Neighboring Particle Searching (NNPS): Because of the large number of particles and the variable smoothing length, the tree method was used as a search algorithm.

Time step: To ensure the stability of the shock tube problem, a sufficiently small time step is necessary. After the tests, the time step was set to $dt = 0.0005$ and for achieving the simulation end time ($t = 2$), 400 steps were required.

Boundary conditions: In section 3.3, we have shown that the SPH method in conjunction with virtual particles does not satisfy the C^1 consistency condition at the near boundary area. This problem was corrected by using ghost particles as a boundary condition.

Now we show the implementation problems and their solutions in the application of these boundary conditions.

- Virtual particles: We solved the problem how to set the repulsive force large enough to prevent particles escape out of the domain, and small enough to avoid degradation of the solution by acting of this force. It is therefore a compromise which could not be found for the classical location of these particles (on the domain boundary). We have to solve this problem by moving the virtual particles outside of the domain as shown in figure 7. This approach allowed us to set a repulsive force that prevents particles from escaping out of the domain and is small enough.
  We chose this approach because it preserves the number of particles and their spacing. However, there is a

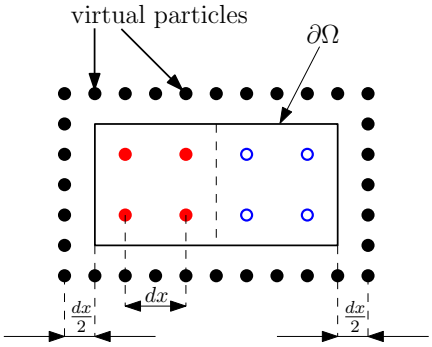

Fig. 7. Corrected position of virtual particles

domain extension (by dx) in each coordinate direction. If there was a strict requirement to preserve the domain size then it would be necessary to change the disposition of particles, i.e. the number of particles or their spacing.

- Ghost particles: In this case, we discussed how to set the properties and position of ghost particles that are created in adition to particles located at the domain corners. Using [6], the position of ghost particles has been resolved. Thus, three ghost particles are always created to the particles at each corner of the domain, as shown in figure 8.

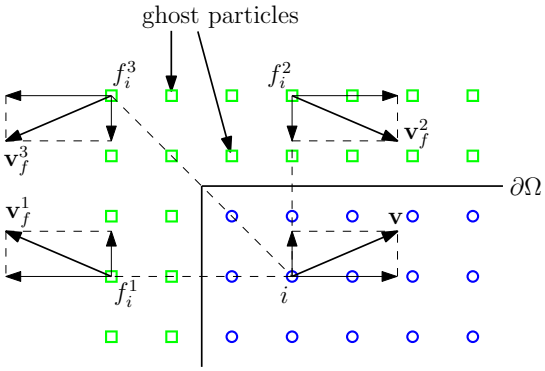

Fig. 8. Positions and velocities of ghost particles, which are created to particle i situated at the domain corner

We have also solved the problem how to properly set the mass of these ghost particles. The best variant was to choose the same mass as particle i, which also corresponds to the text [7].

4.4 Simulation results

The simulation was performed for both types of boundary conditions, i.e. first for virtual particles only, and than for ghost particles only. Ghost particles were implemented into the SPH code within this work. Figures 9, 10, 11 and 12 show the progressions of axial velocity, density, pressure and internal energy in the longitudinal section, i.e. $y = 0$. The solid line shows the exact solution of the Euler equations for a 1D task.

It can be seen that the solution obtained with the help of ghost particles captures the shape of the exact solution better than a solution that uses virtual particles. We

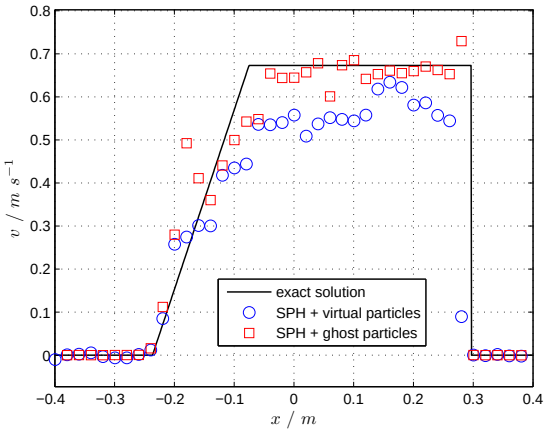

Fig. 9. Axial velocity, longitudinal section, $t = 0.2$ s

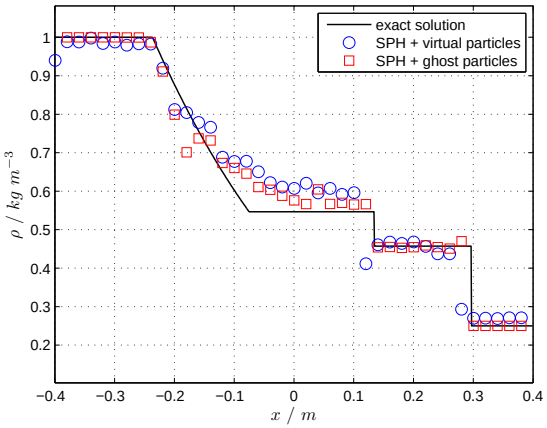

Fig. 10. Axial density, longitudinal section, $t = 0.2$ s

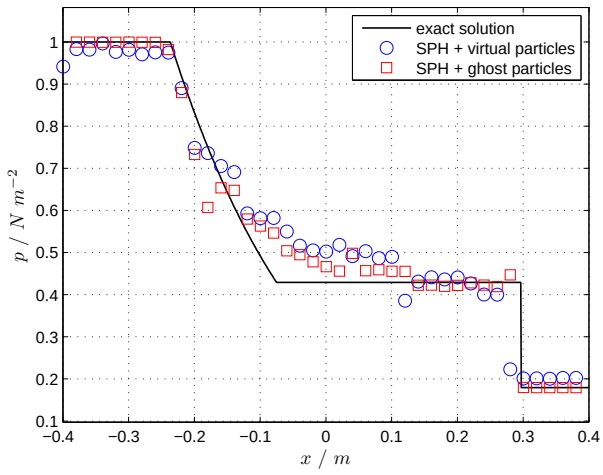

Fig. 11. Axial pressure, longitudinal section, $t = 0.2$ s

demonstrate the influence of C^1 consistency of the SPH approximation at the near boundary area. Let us remind that the use of ghost particles ensures C^1 consistency, while the use of virtual particles does not. The position of the shock wave front observed around $x = 0.3$ can be determined from figures. The rarefaction wave is situated between $x = -0.3$ and $x = 0$ and the contact discontinuity is

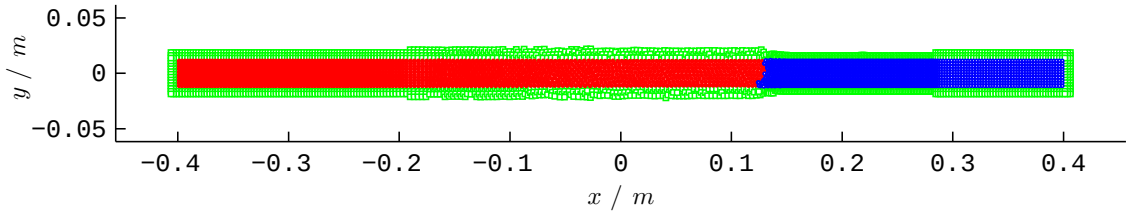

Fig. 13. Position of particles, $t = 0.2$ s

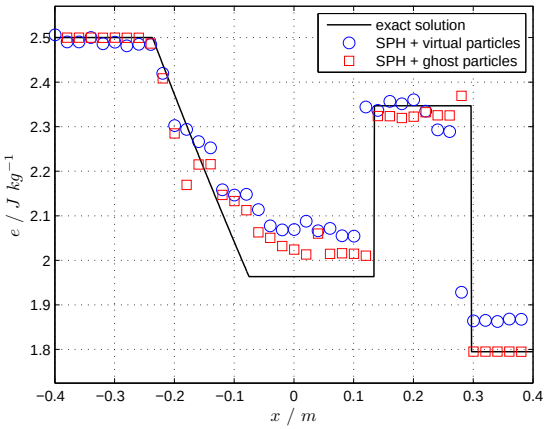

Fig. 12. Axial internal energy, longitudinal section, $t = 0.2$ s

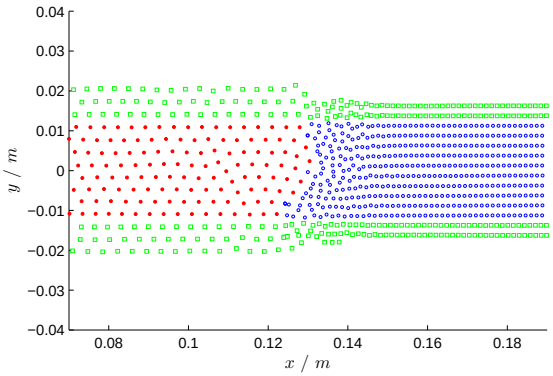

Fig. 14. Position of particles, the detail of the contact discontinuity area, $t = 0.2$ s

located around $x = 0.13$. Slight deviations from the exact solution of both simulations, especially in the rarefaction wave area, are probably caused by the continuity method used for the calculation of density. The continuity method does not preserve mass exactly, but is able to deal with the discontinuity.

Figure 13 shows the specific location of particles at the end of the simulation in the case of using ghost particles. We can also notice the relation between the number of ghost particles and the density of fluid particles distribution. The number of ghost particles is reduced in areas where the fluid particles are closer. This is due to decreasing smoothing length in areas with greater density of particle distribution. This is necessary in order to preserve an almost constant number of particles in each support domain. Similarly, we obtain that the number of ghost particles increases in the places where the fluid particles are closer. Figure 14 shows the detail of the contact discontinuity area around $x = 0.13$.

5 Conclusion

This paper covers our work on the convergence problem of the SPH method. The classical form of boundary condition, i.e. virtual particles, does not preserve C^1 consistency of the SPH approximation in the near boundary area. This negative feature has a direct impact on convergence and thus also on the accuracy of the method. This problem has been solved by using ghost particles as a boundary condition. This type of particles has been implemented to the SPH code. The previously described theory was demonstrated on the 2D shock tube example. The SPH method with ghost particles expressed the form of the exact solution better than the SPH method with virtual particles. We have confirme the influence of C^1 consistency of the SPH approximation on the accuracy of the method. Especially in the near boundary area.

6 Acknowledgements

This research was funded by Brno University of Technology, Faculty of Mechanical Engineering through the project number FSI-S-11-6.

References

1. L.B. Lucy, Astronomical Journal **82**, (1977) 1013-1024
2. R.A. Gingold, J.J. Monaghan, Monthly Notices of the Royal Astronomical Society **181**, (1977) 375-389
3. G.R. Liu, *Mesh Free Methods: Moving beyond the Finite Element Method* (CRC Press, Boca Raton, 2002)
4. G.R. Liu, M.B. Liu, *Smoothed Particle Hydrodynamics - a meshfree particle method* (World Scientific Publishing, New Jersey, 2003)
5. L. Hernquist, N. Katz, The Astrophysical Journal Supplement Series **70**, (1989) 419-446
6. A. Colagrossi, G. Colicchio, D. Le Touzé, Proc. 2nd SPHERIC Workshop **2**, (2007) 59-62
7. S. Børve, Proc. 6th SPHERIC Workshop **6**, (2011) 313-320
8. G.R. Liu, Y.T. Gu, *An Introduction to Meshfree Methods and Their Programming* (Springer Dodrecht, Berlin, 2005)