ATLANTIS PRESS

Advances in Engineering Research, volume 93

International Symposium on Mechanical Engineering and Material Science (ISMEMS 2016)

Method and Software for Modeling the Building Materials as Dispersions

SMIRNOV Vladimir Alexeevich[1, a *], KOROLEV Evgenij Valerjevich[1, b], EVSTIGNEEV Alexandr Viktorovich[1, c] and PODDAEVA Olga Igorevna[1, d]

[1]26 Yaroslavl hw., Moscow, Russian Federation

[a]smirnov@nocnt.ru, [b]korolev@nocnt.ru, [c]evstigneev@nocnt.ru, [d]info@nocnt.ru

Keywords: Building materials, Dispersion, Solvation shell, Particle system, Modeling.

Abstract. Particle system dynamics is the well known method that can be used in wide range of spatial scales – from molecular to astronomic. However, numerical experimentation with constructional composites requires adequate representation of peculiar forces that are due to solvation shells, steric conditions and technological actions. Thus, despite the fact that there is currently a lot of software suitable for particle system dynamics, the existing software is of limited use in constructional material science. In the present work we have developed the proper model of building material as dispersion with solvation shells. The software implementation of the developed model is also briefly described. The model and software were already successfully used in several basic and applied research works in material science.

Introduction

During the last decades, there was a considerable progress in numerical simulation methods that are based on particle system dynamics. Several distinct classes of such methods are evolved, including molecular dynamics, dynamics of complex fluids, smoothed particle hydrodynamics [1-6]. Efficient *N*-body algorithms, integration schemes, and parallel software implementations are already developed [7-9], including algorithms that are suitable for massively parallel hardware [10, 11].

However, during modeling the dispersions that are encountered in construction material science, there are some types forces (both pair forces between particles of fillers and aggregates and the forces that are due to steric conditions and processing technology) that can not to be taken into account by existing software packages. The example of complex pair force is the force caused by solvation shells. The particle with solvation shell can be modeled as two concentric spheres composed with rigid (impenetrable) core and viscous shell; such model is general enough to represent both micro- and macrostructure level of constructional composite. The viscous force can be estimated as a value proportional to cross section of the shells (outer spheres). Neither such estimation method nor the computational scheme that uses it is implemented in existing software.

Because of the absence of ready to use software package that is suitable for the modeling of building materials, the decision was made to implement such software.

Modeling Method

The building materials are typical dispersions that are made of dispersion medium (water, liquid polymer etc.) and fine fillers. Structure formation of such materials is mostly determined by the surface forces that are originating between closely located particles of the filler. The surface forces can lead to the forming of clusters, which have significant influence both on the rheological properties of mixtures and the performance properties of the building materials.

In many cases the motion law for particles of the fine filler can be written as:

$$m_i\ddot{\mathbf{r}}_i - k_i\left(\dot{\mathbf{r}}_i - \mathbf{u}_i\right) = -\nabla U_i,\ i = \overline{1,N}, \tag{1}$$

where, m_i is the mass of i-th particle, $\mathbf{r}_i = (x_i; y_i; z_i)$ is the position of particle, k_i is the parameter that depends on form of the particle and viscosity of the dispersion medium, $\mathbf{u}_i$ is the velocity of the dispersion medium at point $\mathbf{r}_i$, U_i is the scalar force potential at the point $\mathbf{r}_i$.

The potential in right side of Eq. 1 can be written as:

$$U_i = U_b + U_g + \sum_{\substack{j=1 \\ j \neq i}}^{N} U_p(r_{ij}), \tag{2}$$

where, U_b is the potential of boundary interaction, U_g is the gravitational potential, $U_p(r_{ij})$ is the pair potential (r_{ij} is the distance between surfaces of the particles), N is the number of particles.

In molecular dynamics, the pair potential is often taken in Mie form

$$U_p(r_{ij}) = \sum_{k=1}^{m} a_k r_{ij}^{b_k}, \tag{3}$$

in Morse form

$$U_p(r_{ij}) = \sum_{k=1}^{m} a_k \exp(b_k r_{ij} + c_k), \tag{4}$$

and also in form of combinations of Eq. 3, 4 with Gaussian (either additive or multiplicative).

The values of the ak, b_k, c_k in the expressions for the potentials can be determined on the basis of the preliminary information about number and locations of the minima that are corresponding to positions of equilibrium.

The particular case of Mie potential is the Lennard-Jones potential:

$$U_p(r_{ij}) = U_0 \left(\left(\frac{r_0}{r_{ij}} \right)^{12} - 2 \left(\frac{r_0}{r_{ij}} \right)^{6} \right), \tag{5}$$

where, U_0 is the specific energy of interaction, r_0 is the distance of equilibrium.

To describe the evolution of lyophilic dispersions with only repulsive forces it is enough to keep only the second term in Eq. 5. If only spatial configuration is of interest (but not the time of cluster formation), than we can take the potential in form:

$$U_p(r_{ij}) = \frac{k}{|r_{ij}|}, \tag{6}$$

where, k is the arbitrary constant; to select this constant only the computational stability have to be taken into account.

The boundary potential ensures finite nature of motion:

$$U_b(r_{i,b}) = U_0 \frac{r_0}{|r_{i,b}|}, \tag{7}$$

where, $r_{i,b}$ is the distance between boundary and surface of i-th particle, U_0 and r_0 are values of order of magnitude similar to ones in Eq. 3-5.

To avoid the use of spatial derivatives we can rewrite the Eq. 1 in form:

$$\begin{cases} \dot{\mathbf{r}}_i = \mathbf{v}_i \\ \dot{\mathbf{v}}_i = \mathbf{g} + \dfrac{1}{m_i} \left(\displaystyle\sum_{\substack{j=1 \\ j \neq i}}^{N} \mathbf{F}_{ij} + \mathbf{F}_{i,b} + \mathbf{F}_{i,e} \right) \end{cases}, \tag{8}$$

where, $\mathbf{F}_{ij} = -\mathbf{F}_{ji} = \frac{\mathbf{r}_{ij}}{r_{ij}} F_{ij}\left(r_{ij} - R_i - R_j\right)$ is the pair force, (F_{ij} is the modulus of this force, $F_{ij} = \frac{\partial}{\partial r} U_p\left(r_{ij}\right)$, R_i is the radius of particle), $\mathbf{F}_{i,b} = \frac{\mathbf{n}_{i,b}}{n_{i,b}} F_{i,b}\left(n_{i,b} - R_i\right)$ is the boundary force ($F_{i,b}$ is the modulus of this force, $F_{i,b} = \frac{\partial}{\partial n_{i,b}} U_b\left(n_{i,b}\right)$, $\mathbf{n}_{i,b}$ is the vector drawn from the center of the particle to the nearest point of the boundary), m_i is the mass of the particle, $\mathbf{g}$ is the gravitational acceleration, $\mathbf{F}_{i,e} = 6\pi\eta R_i\left(\mathbf{u}_i - \mathbf{v}_i\right)$ is the force of viscous friction.

The finite particle size is taken into account by adjusting the arguments in the expressions for the forces: the arguments above are $r_{ij} - R_i - R_j$ and $n_{i,b} - R_i$.

In case of modeling the dispersions with solvation shells the pair potential can be taken in form.

$$U_p\left(r_{ij}\right) = U_0\left(\left(\frac{r_0}{r_{ij}}\right)^{12} - 2\left(\frac{r_0}{r_{ij}}\right)^{6} + e^{-\left(r_{ij} - r_0\right)^2}\right), \tag{9}$$

and also the viscous force that is caused by crossed shells have to be taken into consideration. The latter can be done as follows.

We can assume that the force $\mathbf{F}_{ij,f}$ that is caused by crossed shells is proportional to the area S_{ij} of the cross section of the shells and to projection $\mathbf{v}_{i,t}$ of the velocity of particle to the plane of the cross section (Fig. 1):

$$S_{ij} = \pi h^2, \tag{10}$$

$$\begin{cases} d_i + d_j = d,\ d = \left|\mathbf{r}_i - \mathbf{r}_j\right| \\ d_i^2 + h^2 = r_i^2,\ d_j^2 + h^2 = r_j^2, \\ r_i = R_i + d_g,\ r_j = R_j + d_g \end{cases} \tag{11}$$

$$h^2 = r_i^2 - d_i^2 = r_i^2 - \frac{1}{4d^2}\left(r_i^2 - r_j^2 + d^2\right)^2, \tag{12}$$

$$S_{ij} = \pi\left(\left(R_i + d_g\right)^2 - \frac{1}{4d^2}\left(\left(R_i + d_g\right)^2 - \left(R_j + d_g\right)^2 + \left(\mathbf{r}_i - \mathbf{r}_j\right)^2\right)^2\right), \tag{13}$$

where, d, d_i, d_j, and h values are illustrated on Fig. 1.

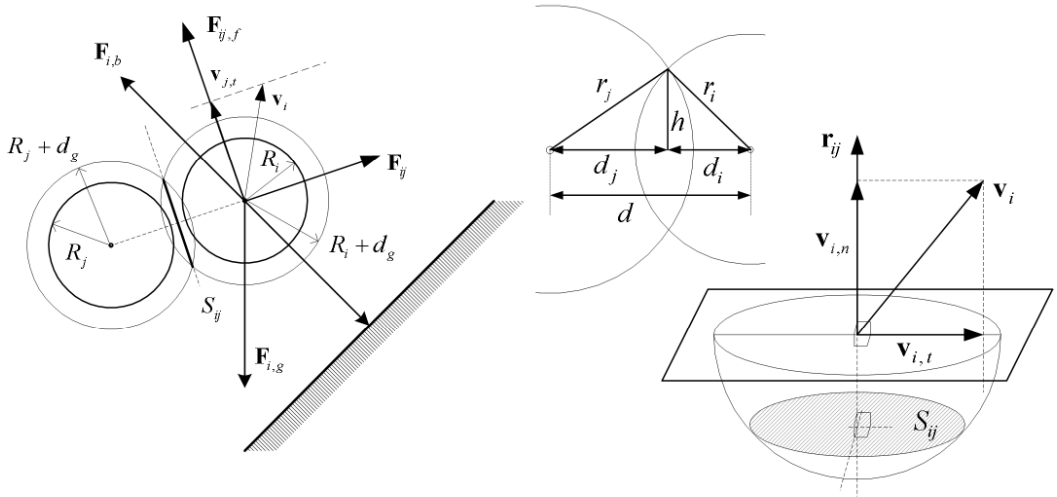

Fig. 1. Particles, distances, forces and velocities in dispersion with solvation shells

Thus, force $\mathbf{F}_{ij,f}$ will be equal to:

$$\mathbf{F}_{ij,f} = -\frac{S_{ij}\eta\mathbf{v}_{i,t}}{r_{ij} - R_i - R_j}. \quad (14)$$

The projection $\mathbf{v}_{i,t}$ can be found as:

$$\mathbf{v}_{i,t} = \mathbf{v}_i - \mathbf{v}_{i,n}, \quad (15)$$

where,

$$\mathbf{v}_{i,n} = \left(\mathbf{v}_i \cdot \mathbf{r}_{ij,0}\right)\mathbf{r}_{ij,0},\ \mathbf{r}_{ij,0} = \frac{\mathbf{r}_i - \mathbf{r}_j}{r_{ij}}. \quad (16)$$

Modeling Software

The Eq. 5-16 are implemented in software that correctly represents both tangent and normal pair forces, along with extra forces caused by gravity, technology and steric conditions.

The software is entirely written in ANSI C. The portability is one of the primary aims; there are several abstraction layers. Lower layer serves as isolation from the Windows and POSIX application programming interface (API) calls; this layer is very thin.

Middle layer is introduced as a support measure for object oriented programming in object agnostic ANSI C language. We believe that proper object oriented software design is far more important than using modern fashionable “object oriented” programming dialects; good object oriented design can easily be implemented in C language. Total amount of code is almost the same, while porting and maintaining are simplified significantly. Middle layer (middleware) incorporates basic algorithms, auxiliary functionality, lexical analysis and ODE solving algorithms. Latter are exposed through small number of interfaces (odeCreateSolver(), odeSet() / odeGet() and odeIntegrate()) which take opaque handle and pointer to the ODE right side; such implementation allows to exchange the solver without any alteration of the “applied” functionality.

Currently we do not use any complex scheme for pair forces computation. The primary argument for such a choice is following: the complex schemes (that are beneficial for CPU) sometimes do not perform adequately if used with massively parallel hardware (e.g. GPU, [11]).

Solution of ODE for dynamical systems requires appropriate initial and boundary conditions; this is especially important for many strongly nonequilibrium problems arising in constructional material science. For “perfect” solids (crystals) simple algorithms can be used to generate fragments of common lattices. To model the initial distribution of the dispersed phase in constructional mix it is necessary to employ several statistical densities which correspond to mixing conditions. Together with statistical distributions we have used constructive solid geometry (CSG) to model the boundary. The representation of the initial state in constrained dynamical problem has to be written in textual form. The example of such representation is shown below:

```
union
{
      cylinder [-101,0,0;1,0,0] [30;202]
      union {
            cylinder [0,-101,0;0,1,0] [30;202]
            union {
                  cylinder [0,0,-101;0,0,1] [30;202]
                  sphere [0,0,0] 60
            }
      }
}
```

The fragments similar to the above one are embedded in the task file (the solver is command line driven tool). During the parsing of the task file the CSG trees are constructed as necessary and particle generation performed for the selected types of spatial distributions. Both pre- and postprocessors of the dynamics package are able to export representation of the system in form of the MaxScript (built-in language of Autodesk 3DS MAX package) programs that can be used for visualization. The examples of rendered images are shown of the Fig. 2 and 3.

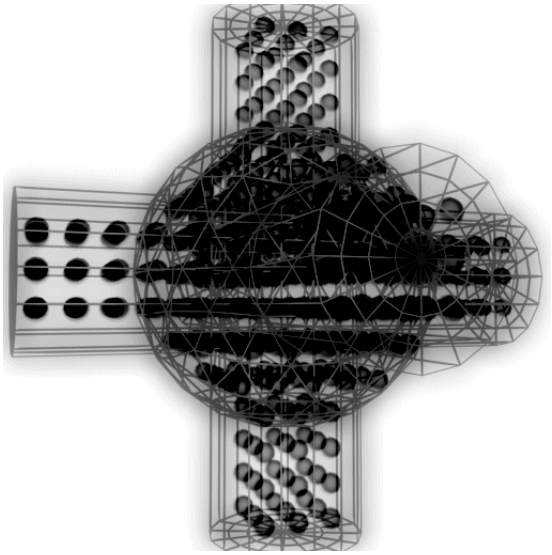

Fig. 2 Regular cubic lattice for the listed CSG code.

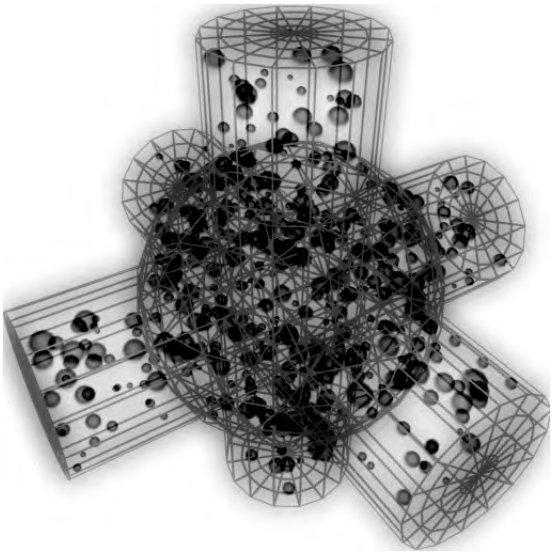

Fig. 3 Uniform distribution for the listed CSG code.

Real-time visualization functionality is also implemented in solver (portable OpenGL API is used for rendering), though quality of the produced images are still far from typographic. Such visualization is only suitable for monitoring of the solution process.

Summary

Due to the fact that particle system dynamics is widely used for numerical modeling in material science, many software packages are developed for such simulations. However, in constructional material science we constantly encounter cases when adoption of the existing software leads to unacceptable time consumption for exploration of features and extending the functionality; because of this, at some point the decision was made to implement the software that is specifically tailored for the tasks of construction. The developed software allows examination of the influence of mixture and technology to the properties of composition (technological mix) and constructional composite in great detail. The software is of layered object oriented design, is implemented entirely in ANSI C and uses only documented and portable programming interfaces. Currently the software can be compiled for Microsoft Windows (with Microsoft Visual C, Intel C, Open Watcom and GNU C compilers) as well as for many POSIX (in particular, Linux and FreeBSD) environments. This software was successfully used in several basic and applied research works in material science.

The goals of the further software development are parallelization of algorithms, code cleanup and implementation of the advanced visualization features.

Acknowledgements

This work is supported by the Ministry of Science and Education of Russian Federation, Project #7.11.2014/K “Theoretical and experimental study of the dynamics of constructions”.

References

[1] W. G. Hoover, Molecular Dynamics, Springer, Berlin, 1986.

[2] A. J. C. Ladd, Molecular Dynamics, in: Computer Modelling of Fluids Polymers and Solids, C. R. A. Catlow, S. C. Parker, M. P. Allen (Eds.), Kluwer Academic Publishers, Dordrecht, 1988, pp. 55-82.

[3] R. W. Hockney, J. W. Eastwood, Computer simulation using particles, IOP Publishing, Bristol 1988.

[4] K. Ohno, K. Esfarjani, Y. Kawazoe (Eds.), Computational Materials Science, Springer-Verlag, Berlin, 1999.

[5] J. G. Lee, Computational Materials Science: An Introduction, CRC Press, Boca Raton, 2012.

[6] R. LeSar, Introduction to Computational Materials Science, Cambridge University Press, Cambridge, 2013.

[7] H. Berendsen, D. van der Spoel, R. Vandrunen, GROMACS – a Message-passing Parallel Molecular-Dynamics Implementation, Comput. Phys. Commun. 91 (1995) 43-56.

[8] S. Plimpton, Fast Parallel Algorithms for Short-range Molecular Dynamics, J. Comput. Phys. 117 (1995) 1-19.

[9] S. Plimpton, B. Hendrickson, A New Parallel Method for Molecular Dynamics Simulation of Macromolecular Systems, J. Comput. Chem. 17 (1996) 326-337.

[10] R. Yokota, L. Barba, Hierarchical N-body Simulations with Autotuning for Heterogeneous Systems, Comput. Sci. Eng. 14 (2012) 30-39.

[11] E. Palmer, R. Taylor, GPU Accelerated Barnes-Hut N-Body Simulation, URL: http://www. andrew. cmu. edu/user/esp/