An Introduction to the Material Point Method

J.E. Guilkey
Department of Mechanical Engineering
University of Utah
Salt Lake City, Utah 84112

1 Introduction

The Material Point Method (MPM) as described by Sulsky, et al. [1, 2] is a particle method for structural mechanics simulations. Solid objects are represented by a collection of particles, or "material points." Each of these particles carries with it information for that part of the solid object that it represents. This includes the mass, volume, position, velocity and stress of that material. MPM differs from other so called "mesh-free" particle methods in that, while each object is primarily represented by a collection of particles, a computational mesh is also an important part of the calculation. Particles do not interact with each other directly, rather the particle information is interpolated to the grid, where the equations of motion are integrated forward in time. This time advanced solution is then used to update the particle state. An example of two disks initially approaching each other represented by material points on an overlying mesh is show in Figure 1.

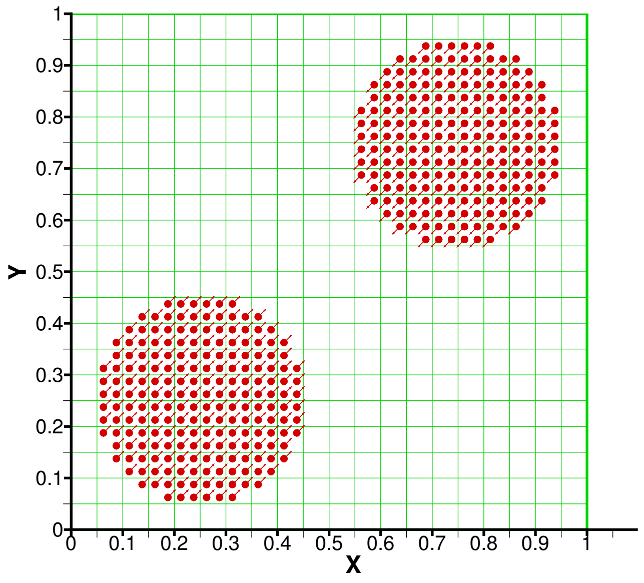


Figure 1: Initial particle representation of two colliding disks on an overlying mesh.

The method usually uses a regular structured grid as a computational mesh. While this grid, in principle, deforms as the material that it is representing deforms, at the end of each timestep, it is reset to it's original undeformed position, in effect providing a new computational grid for each timestep. The use of a regular structured grid for each time step has a number of computational advantages. Computation of spatial gradients is simplified. Mesh entanglement, which can plague fully Lagrangian techniques, such as the Finite Element Method (FEM), is avoided. MPM has also been successful in solving problems involving contact between colliding objects, having an advantage over FEM in that the use of the regular grid eliminates the need for doing costly searches for contact surfaces[3].

2 Algorithm

While a more detailed description of MPM can be found in [2], the algorithm is laid out here. The equations of motion are cast in the form:

$$\mathbf{M}_g \cdot \mathbf{a}_g = \mathbf{Fext}_g - \mathbf{Fint}_g \tag{1}$$

where $\mathbf{M}_g$ is the mass matrix, $\mathbf{a}_g$ is the acceleration vector, $\mathbf{Fext}_g$ is the external force vector (sum of the body forces and tractions), and $\mathbf{Fint}_g$ is the internal force vector resulting from the divergence of the material stresses. In general, $\mathbf{M}_g$ is a large, sparse matrix. In practice, and in what follows here, a "lumped" mass matrix is used, which only has entries on the diagonal, and is thus represented as a column matrix.

The solution procedure begins by interpolating the particle state to the grid, to form $\mathbf{M}_g$, $\mathbf{Fext}_g$, and to get a velocity on the grid $\mathbf{v}_g$. These quantities are calculated at each grid node by the following equations:

$$\mathbf{M}_i = \sum_p S_{ip} m_p \tag{2}$$

$$\mathbf{v}_i = \frac{\sum_p S_{ip} m_p \mathbf{v}_p}{\mathbf{M}_i} \tag{3}$$

$$\mathbf{Fext}_i = \sum_p S_{ip} \mathbf{Fext}_p. \tag{4}$$

m_p is the particle mass, $\mathbf{v}_p$ is the particle velocity, and $\mathbf{Fext}_p$ is the external force on the particle. The external force on the particle is generally an applied load of some type. In Equation 3, the numerator is the nodal momentum, which is then divided by the nodal mass to get a velocity. S_{ip} is a "shape function" for the *ith* node evaluated at $\mathbf{x}_p$. Traditionally, the shape functions are multiplicative combinations of one dimensional tent functions, as shown in Figure 2. The shape functions serve to distance weight the contribution of each particle to the grid nodes.

At this point, a velocity gradient, $\nabla \mathbf{v}_p$ is computed at each particle using the grid velocities $\mathbf{v}_g$:

$$\nabla \mathbf{v}_p = \sum_i \mathbf{G}_{ip} \mathbf{v}_I \tag{5}$$

where $\mathbf{G}_{ip}$ is the gradient of the *ith* node's shape function, evaluated at $\mathbf{x}_p$. A one dimensional example of $\mathbf{G}_{ip}$ is shown in Figure 3. Note that in going to multiple dimensions, the $\mathbf{G}_{ip}$ are found by taking gradients of the multidimensional S_{ip} NOT by forming multiplicative combinations of the one-dimensional $\mathbf{G}_{ip}$.

This velocity gradient is used as input to a constitutive model (stress-strain relationship) which is evaluated at each particle. The specifics of this calculation are dependent on the constitutive model. The result of this calculation is the Cauchy stress at each particle, $\boldsymbol{\sigma}_p$. With this, the internal force due to the divergence of the stress is calculated:

$$\mathbf{Fint}_g = \sum_p \mathbf{G}_{ip} \boldsymbol{\sigma}_p v_p \tag{6}$$

where v_p is the particle volume. The internal force can be thought of as the force that holds a material together. For a given deformation, this force is larger for stiffer materials.

Everything is now available to solve Equation 1 for $\mathbf{a}_g$. With that, the backward Euler method is used for all time integrations. A convective grid velocity $\mathbf{v}_g^L$ is computed:

$$\mathbf{v}_g^L = \mathbf{v}_g + \mathbf{a}_g dt \tag{7}$$

While the following calculation is never carried out, in principal, the nodes of the grid also move with that convective velocity:

$$\mathbf{x}_g^L = \mathbf{x}_g + \mathbf{v}_g^L dt \tag{8}$$

During this part of the computation, the particles move with the deforming grid. Their position and velocity is explicitly updated by:

$$\mathbf{v}_p(t+dt) = \mathbf{v}_p(t) + \sum_i S_{ip} \mathbf{a}_i dt \tag{9}$$

$$\mathbf{x}_p(t+dt) = \mathbf{x}_p(t) + \sum_i S_{ip} \mathbf{v}_i^L dt \tag{10}$$

This completes one timestep. Note that not carrying out the calculation in 8 explicitly has the effect of resetting the deformed grid to it's undeformed position at the end of the timestep cycle.

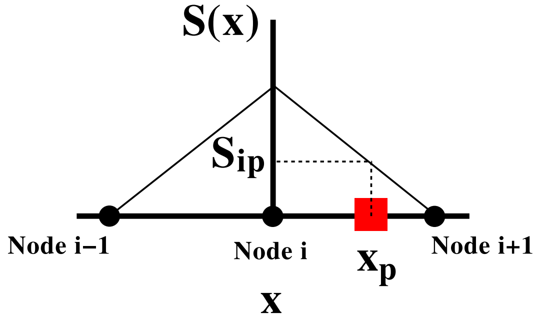
As with all explicit time integration methods, a timestep size limit must be enforced such that $dt < dx/(|\mathbf{v}_p| + c)$ for all particles, where dx is the computational grid spacing and c is the speed at which stress waves propagate through the material. Failure to adhere to this condition will cause the solution to become unstable and blow up. The material wavespeed depends on the material model used, as well as on the particular parameters chosen for that model. Specifics of calculating the wavespeed are given in the appendix.

References

[1] D. Sulsky, Z. Chen, and H.L. Schreyer. A particle method for history dependent materials. *Comput. Methods Appl. Mech. Engrg.*, 118:179–196, 1994.

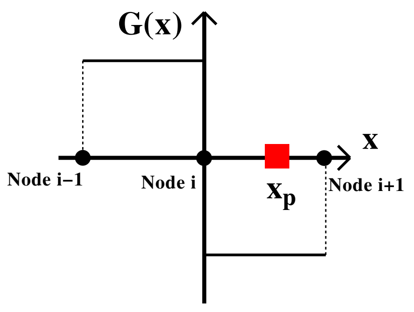
[2] D. Sulsky, S. Zhou, and H.L. Schreyer. Application of a particle-in-cell method to solid mechanics. *Computer Physics Communications*, 87:236–252, 1995.

[3] S.G. Bardenhagen, J.U. Brackbill, and D. Sulsky. The material-point method for granular materials. *Comput. Methods Appl. Mech. Engrg.*, 187:529–541, 2000.



$S(x) = (x - x_{i-1})/(x_i - x_{i-1})$	$x_{i-1} < x < x_i$	
$S(x) = (x_{i+1} - x)/(x_{i+1} - x_i)$	$x_i < x < x_{i+1}$	
$S(x) = 0$	$x < x_{i-1}$	$x > x_{i+1}$

Figure 2: One dimensional linear shape function, $S(x)$.



$G(x) = 1/(x_i - x_{i-1})$	$x_{i-1} < x < x_i$	
$G(x) = -1/(x_{i+1} - x_i)$	$x_i < x < x_{i+1}$	
$G(x) = 0$	$x < x_{i-1}$	$x > x_{i+1}$

Figure 3: One dimensional linear shape function derivative, $G(x)$.