Notes about the Generalized Interpolation Material Point Method

Miguel Molinos Pérez

Dpto. de Matemáticas Aplicadas
Universidad Politécnica de Madrid (UPM)
m.molinos@alumnos.upm.es

July 18, 2019

Contents

1 Introduction

The past several decades have brought tremendous advances in computing power and provided fertile ground for the development of the computational sciences. In computational solid mechanics the Finite Element Method (FEM) (METER CITA MOLONA) has been very successfully applied to a wide range of problems with good results. However, body fixed FEM meshes can be difficult and time consuming to generate for complex three-dimensional objects. Further, mesh distorsion associated with large deformations compromises solution accuracy, ultimately requiring re-meshing. These difficulties have spurred the development of alternative discretization strategies which avoid mesh distorsion by dicretizing at points and never maintaining a body-fixed mesh.

Quite a number of ”meshless methods” have been developed. Some of the ways in which the methods differ include whether or not a temporary mesh is used in the solution procedure, whether the discretization procedure begins with the

[**?**]

2 Governing Equations

In the following derivation of the discrete equations, bold face quantities indicate tensor, ∇ id the gradient operator, and $\cdot$, and $:$ are first order (vector) and second order tensor contraction, respectively. The subscript p is used to index material point variables and i grid vertex variables. The notation $\sum_p^{Np}$ and $\sum_i^{Np}$ is used to denote summation over all material points, and over all grid vertex, respectively.

Of interest in solid mechanics is the deformation and material response to a continuous solid body under prescribed loads and initial conditions, as governed by conservation of mass and momentum. Conservation of mass is satisfied implicitly by leaving discrete particle masses unchanged throughout a computation. Here we develop the discrete version of conservation of momentum, which permits evolution of particle momenta in time. We consider a deformable body acted upon by forces and subjected to either kinematic or traction boundary conditions every where on its surface. The continuum mechanics theory

- Balance of momentum
- Compatibility
-

We works over the balance of momentum in the strong formulation, as we can see in 1

$$\rho a = \nabla \cdot (\sigma) + \rho b \tag{1}$$

Multiplying (1) by ψ we get (2)

$$\psi^T \cdot \rho a = \psi^T \cdot \nabla \cdot (\sigma) + \psi^T \cdot \rho b \tag{2}$$

The function ψ belongs to a space $\mathscr{H}$ (vectorial space with scalar product defined) wich verifies $a(\ddot{u}, \dot{u}, u, \psi) = l(\psi) \quad \forall \ \psi \ \in \ \mathscr{H}$. In other hand, in every point of Ω where we have imposed the Dirichlet bounday condition,

$$\mathscr{H} = \{\psi(x) / \psi \mid_{\Gamma_D} = 0 \quad , \quad \psi \in H^1(\Omega)\} \tag{3}$$

In other words, $\mathscr{H}$ is the space defined by all the functions g, that ψ and $\nabla\psi$ are FUNCIONES DE CUADRADO INTEGRABLE in Ω

$$\int_\Omega \psi \ d\Omega < \infty \quad , \quad \int_\Omega \nabla\psi \ d\Omega < \infty \tag{4}$$

Integrating (2) over Ω we get (5)

$$\int_\Omega \psi^T \cdot \rho a \ d\Omega = \int_\Omega \psi^T \cdot \nabla \cdot (\sigma) \ d\Omega + \int_\Omega \psi^T \cdot \rho b \ d\Omega \tag{5}$$

integrating by parts the therm of the internal tension in (5), we get (6)

$$\int_\Omega \psi^T \cdot \rho a \ d\Omega = \int_\Omega \nabla \cdot \left(\psi^T \cdot \sigma\right) \ d\Omega - \int_\Omega \nabla\psi^T : \sigma \ d\Omega + \int_\Omega \psi^T \cdot b \ d\Omega \tag{6}$$

applying the Gauss theorem over (6), and following the definition of the stress vector $t = \sigma \cdot n$ where n is the vector orthogonal to a surface in the solid, finally we get the variational form for conservation of momentum may be written as 7

$$\int_\Omega \psi^T \cdot \rho a \ d\Omega = \int_\Gamma \psi^T \cdot \overbrace{\sigma \cdot n}^{t} \ d\Gamma - \int_\Omega \nabla\psi^T : \sigma \ d\Omega + \int_\Omega \psi^T b \ d\Omega \tag{7}$$

2.1 Observations

3 State of art of the material point method

The material point method (MPM) was developed by Sulsky and coworkers, see [?], as an extension for solids mechanics of the fluid-implicit particle-method (FLIP) developed by (Brackbill). Further, the FLIP originally was proposed as an extension of the particle-in-cell method developed by (Harlow). More recent developments are centered in trying to solve the well-know grid-crossing error of the MPM, described deeply in 5.2, the first of this early attempts was the presented by [?] called the generalized interpolation material point method (GIMP), other approach to solve this problem are the implicit Generalized Interpolation Material Point (GIMP) presented by [?], the convected Particle Domain Interpolation (CPDI) and the second-order convected particle domain interpolation (CPDI 2) by [?], [?], the dual Domain Material Point Method (DDMPM) by [?], and the B-Spline MPM by [?].

All of this methods has a starting point the variation form for conservation momentum (7) presented in 2. In order to arrive to a finite set of equations, the continuum domain Ω is discreticed as as a finite sum of material points. Here, the index p is used for the material point values, I for the nodes of the grid nodes and the index i, j, k are used for the dimensions of the variables. Each material point has a position $x_{p,i}$ defined by the centroid of the volume Ω_p.

Each material point is assigned initial values of position, velocity, mass, volume and stress, denoted by $x_{p,i}$, $v_{p,i}$, m_p, $V_{p,i}$, $\sigma_{p,ij}$, and others. Given a material point property, f_p, a representation consistent with the initial discretization procedure is the sum over the material point as we can see in (8)

$$f(x,t) = \sum_{p}^{N_p} f_p(t)\chi_p(x) \tag{8}$$

As an example, the density field can be discreticed as

$$\rho(x,t) = \sum_{p}^{N_p} m_p \delta(x_i - x_{p,i}) \tag{9}$$

3.1 The original material point method

The particle characteristic function are required to be a partition of unity in the initial configuration as we can see in (9).

$$\sum_{p=1}^{Np} \chi_p^i(x) \; = \; 1 \tag{10}$$

Where χ_p^i denotes the particle characteristic functions restricted to their initial positions and undeformed state. In the simplest cases, particle characteristic function are taken to be initially non-overlapping. However, nothing precludes overlapping, or "fuzzy" particles, as discussed later. Initial particle volumes V_p^i are defined by (10).

$$V_p^i = \int_{\Omega^i} \chi_p^i(x)\; d\Omega \tag{11}$$

where Ω^i is the initial volume of the continuum body to be discredited. In addition to initial particle volumes, the material point initial masses, m_p^i, momenta, p_p^i ans stresses, σ_p^i, must be defined. These properties of the continuum against the particle characteristic functions, as we can see in the equations (11) and (12)

$$m_p^i = \int_{\Omega_i} \rho^i(x)\chi_p^i(x)\; d\Omega \tag{12}$$

$$p_p^i = \int_{\Omega_i} \rho^i(x) v^i(x)\chi_p^i(x)\; d\Omega \tag{13}$$

where ρ^i

3.2 Discrete Solution Procedure

Given a material point property, f_p, a representation consistent with the initial discretization procedure is the sum over the material points.

$$f(x) = \sum_{p}^{Np} f_p \chi_p(x) \tag{14}$$

The particle characteristic functions are used as a basis for representing particle data throughout the computational domain and determine the degree of smoothness of the spatial variation.

Approaching term by term of (7) using (13) we get the following :

- Acceleration term :

$$\int_\Omega \psi^T \rho \cdot a \cdot d\Omega = \int_\Omega \psi^T \tfrac{m}{V} \dot{v}\, d\Omega = \int_\Omega \psi^T \tfrac{1}{V} \overbrace{m\dot{v}}^{\dot{p}}\, d\Omega = \int_\Omega \psi^T \tfrac{\dot{p}}{V} d\Omega =$$
$$= \int_\Omega \psi^T \left[\textstyle\sum_{p=1}^{Np} \tfrac{\dot{p}_p}{V_p} \chi_p(x)\right] d\Omega = \textstyle\sum_{p=1}^{Np} \left[\int_{\Omega_p \bigcap \Omega} \psi^T \tfrac{\dot{p}_p}{V_p} \chi_p(x)\, d\Omega\right] \tag{15}$$

- Internal forces :

$$\int_\Omega \nabla\psi^T \sigma\, d\Omega = \int_\Omega \nabla\psi^T \left[\sum_{p=1}^{Np} \sigma_p \chi_p(x)\right] d\Omega = \sum_{p=1}^{Np} \left[\int_{\Omega_p \bigcap \Omega} \nabla\psi^T \sigma_p \chi_p(x)\, d\Omega\right] \tag{16}$$

- External forces :

$$\int_\Omega \psi^T \rho b\, d\Omega = \int_\Omega \psi^T \frac{m}{V} b\, d\Omega = \int_\Omega \psi^T \left[\sum_{p=1}^{Np} \frac{m_p}{V_p} \chi_p(x)\right] b\, d\Omega = \sum_{p=1}^{Np} \left[\int_{\Omega_p \bigcap \Omega} \psi^T \frac{m_p}{V_p} \chi_p(x) b\, d\Omega\right] \tag{17}$$

Finally we get the (1) with the GIPM discretization as (17):

$$\sum_{p=1}^{Np} \left[\int_{\Omega_p \bigcap \Omega} \psi^T \frac{\dot{p}_p}{V_p} \chi_p(x)\, d\Omega\right] + \sum_{p=1}^{Np} \left[\int_{\Omega_p \bigcap \Omega} \nabla\psi^T \sigma_p \chi_p(x)\, d\Omega\right] = \int_\Gamma \psi^T t\, d\Gamma + \sum_{p=1}^{Np} \left[\int_{\Omega_p \bigcap \Omega} \psi^T \frac{m_p}{V_p} \chi_p(x) b\, d\Omega\right] \tag{18}$$

where Ω_p denotes the current support of particle characteristic function p, and the current particle volumes are defined by (18)

$$V_p = \int_{\Omega_p \bigcap \Omega} \chi_p(x)\, d\Omega \tag{19}$$

Rewriting the balance of momentum, the equation

The other fundamental aspect of PIC methods is the use of a computational grid. In MPM the grid serves as a scratch pad for the solution of conservation of momentum, from which particle states are updated. To complete the discretization procedure, approximations to the admissible velocity fields, or test functions, are introduced in terms of grid vertex quantities and grid shape functions. This step is analogous to the development of FEM discrete equations. However, use of both grid and particle basis functions to represent test functions and trial functions, respectively, is a Petrov–Galerkin method, [Johnson (1987)], and therefore more akin to some of the meshless methods (in particular [Demkowicz and Oden (1986); Atluri and Zhu (2000)]) than the FEM. The continuous representation, $g(x)$, of grid data, g_i , then

$$g(x) = \sum_{i=1}^{Nn} = g_i N_i(x) \tag{20}$$

Here $N_i(x)$ is a computational grid shape function, which takes unit value at node i and zero value all the other nodes. Further, the shape function are required to be a partition of unity

$$\sum_{i=1}^{Nn} N_i(x) = 1 \tag{21}$$

$$\sum_{p=1}^{Np} \left[\int_{\Omega_p \bigcap \Omega} \psi^T \frac{\dot{p}_p}{V_p} \chi_p(x) d\Omega\right] = \textstyle\sum_{p=1}^{Np} \left[\tfrac{1}{V_p} \int_{\Omega_p \bigcap \Omega} N_i(x) \chi_p(x) d\Omega \dot{p}_p\right] =$$
$$= \textstyle\sum_{p=1}^{Np} \overline{S}_{ip} \cdot \dot{p}_p = \dot{p}_i \tag{22}$$

$$-\sum_{p=1}^{Np} \left[\int_{\Omega_p \bigcap \Omega} \nabla\psi^T \sigma_p \chi_p d\Omega\right] = -\textstyle\sum_{p=1}^{Np} \left[\int_{\Omega_p \bigcap \Omega} \nabla N_i(x) \sigma_p \chi_p d\Omega\right] =$$
$$= -\textstyle\sum_{p=1}^{Np} \left[\tfrac{V_p}{V_p} \int_{\Omega_p \bigcap \Omega} \nabla N_i(x) \chi_p d\Omega \sigma_p\right] =$$
$$= -\textstyle\sum_{p=1}^{Np} \left[V_p \overline{\nabla S}_{ip} \sigma_p\right] = f_i^{int} \tag{23}$$

$$\int_{\Gamma \equiv \partial\Omega} \psi^T t d\Gamma = \int_{\Gamma \equiv \partial\Omega} N_i(x) t d\Gamma = f_i^t \tag{24}$$

$$\sum_{p=1}^{Np}\left[\int_{\Omega_p\bigcap\Omega}\psi^T\frac{m_p}{V_p}\chi_p(x)bd\Omega\right]=\sum_{p=1}^{Np}\left[\int_{\Omega_p\bigcap\Omega}N_i(x)\frac{m_p}{V_p}\chi_p(x)bd\Omega\right]=\sum_{p=1}^{Np}\overline{S}_{ip}bm_p=f_i^b \tag{25}$$

$$\begin{gathered}\sum_{p=1}^{Np}\overline{S}_{ip}\cdot\dot{p}_p=-\sum_{p=1}^{Np}\left[V_p\overline{\nabla S}_{ip}\sigma_p\right]+\quad\int_{\Gamma\equiv\partial\Omega}N_i(x)td\Gamma+\quad\sum_{p=1}^{Np}\overline{S}_{ip}bm_p\\ \Downarrow\\ \dot{p}_p=f_i^{int}+f_i^t+f_i^b\end{gathered} \tag{26}$$

4 Explicit time integration

The equation of motion presented in

4.1 The standard time integration scheme

4.2 Generalized-α integration scheme for MPM

5 Stresses in the material-point-method

As we can see in [**?**]

5.1 Objective evaluation of stresses

5.2 Grid-crossing errors in the MPM

5.3 Gravitational loading in the MPM

5.4 Alternative approaches for updating stresses

The question of when to update stresses has been subjected of research by [**?**]. In it, Bardenhagen discusses two different ways to update the stresses, either before or after the calculation of internal forces.

5.5 Reduced integration of stress

6 Large strain formulation for the material-point-method

6.1 Tracking large deformation

The geometric area associated with a material point is the given the notion *voxel*. In the reference configuration, the voxel is defined such that V_p corresponds to the size of the *voxel*. The *voxel* notion, was introduced by [**?**]. The location of the corners associated with each voxel at a deformed state is specified through the deformation-gradient tensor which needs to be tracked throughout the simulation. Adopting this concept, a deformation gradient is associated with each material point in the current state as

$$F_p = \frac{x_p}{\partial x^0} = \frac{1}{V_p^0} \int_{\Omega_p^0} F(x)\; d\Omega \tag{27}$$

In the initial configuration, the deformation gradient tensor is know. Typically, an undeformed material state will be prescribed ($F^0 = I$, where I is the identity tensor). An explicit forward difference updating scheme for updating the deformation gradient is employed as we can see in

$$F^{k+1} = \Delta F^{k+1} F^k \tag{28}$$

This way of integrating the deformation gradient, using the grid velocities, in also employed by [**?**] and [**?**]

6.2 Large strain analysis by the material point method

6.2.1 Evaluationg the weighting functions using Gauss quadrature

6.2.2 Analytical approximation of the weighting functions

6.3 Refinement in the material-point-method

6.3.1 Material-point splitting

6.3.2 Material-point splitting in case of extreme deformations

6.3.3 Refining the computational grid

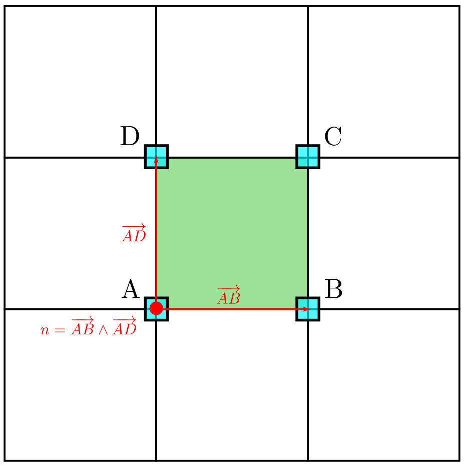

Figure 1: Normal vector of an element

7 Algorithms for the Material Point Method

In this chapter, we develop several algorithms that we need for a material point code.

7.1 Calculus of the natural coordinates of a Gauss point

7.2 Local search of Gauss points

Exists several algorithms proposed by other authors to do the search of a particle in the MPM scheme. In this case, we adopt one based on the Gauss-Point velocity field

1. Get the list of elements that share each vertex, for this,is interesting to use a pointer-based programming language, we suggest to use a table of pointers in order to store the data in a more compact and flexible way. This allows also to extend this algorithm to any kind of mesh (IE: Quadrilaterals, triangular and hybrid). This step will be done at the beginning of the calculus because it is an information that will not change during the simulation unless we use some kind of automatic re-meshing algorithm.

2. Check in first instance if the material point is in the same element. For this, get in first place the normal vector to the plane of the element with a scalar product (28)

$$\vec{n} = \overrightarrow{AB} \wedge \overrightarrow{AD} \tag{29}$$

   As we can see in the figure 1. Loop over the sides of the element and check if each point satisfies the condition (29).

$$\vec{n} \cdot (\overrightarrow{AB} \wedge \overrightarrow{AP_i}) > 0 \tag{30}$$

   where $\vec{n}$ is the normal vector of the element defined in (28), $\overrightarrow{AB}$ is the array of the side of the element, and $\overrightarrow{AP_i}$ is the array to the material point measured from the initial vertex of the side. If it is true, for all the sides, the point is inside of the element, else if this condition is false in some of the sides, the point it is outside and we can break the loop.

3. If the point is not in the same element, search in the neighbour elements of the initial one. In this step of the local algorithm, we have two chooses, the first one and the easy one to program is search in the elements around the initial one. The second one is which we have implemented in our code and consist in use the velocity field to predict in which element will be the particle. For this algorithm, this are the steps :

   (a) For each vertex of the element, get the direction of search of this vertex as the sum of the arrays of the sides that reach to this vertex.

$$\overrightarrow{n_B} = \overrightarrow{AB} + \overrightarrow{CB} \tag{31}$$

   (b) For each dimension of $\overrightarrow{n_B}$, multiply it component by component by the velocity vector of the material point, like some kind of projection, if all the components of the resultant vector are positive we should search the material point in this direction, else if try in the next node.

   (c)

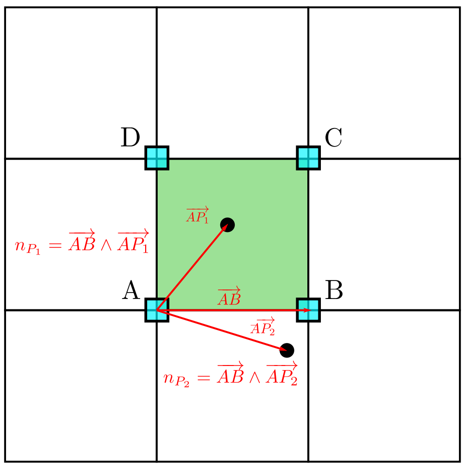
D
C
$\overrightarrow{AP_1}$
$n_{P_1} = \overrightarrow{AB} \wedge \overrightarrow{AP_1}$
A
B
$\overrightarrow{AB}$
$\overrightarrow{AP_2}$
$n_{P_2} = \overrightarrow{AB} \wedge \overrightarrow{AP_2}$

Figure 2: Look if the point is in our out of the element

Como vemos en [**?**]

References