A Convected Particle Least Square Interpolation (CPLS) Material Point Method

Article *in* International Journal for Numerical Methods in Engineering · October 2019

DOI: 10.1002/nme.6257

4 authors, including:

Quoc-Anh Tran
Aalto University
16 PUBLICATIONS **37** CITATIONS

SEE PROFILE

A Convected Particle Least Square Interpolation (CPLS) Material Point Method

Quoc-Anh Tran, PhD student
Department of Civil Engineering, Aalto University, Espoo, Finland

Wojciech Sołowski, Assistant Professor
Department of Civil Engineering, Aalto University, Espoo, Finland

Martin Berzins, Professor
Scientific Computing and Imaging Institute, University of Utah, Salt Lake City, Utah, USA

James Guilkey, Research Associate Professor
Department of Mechanical Engineering, University of Utah, Salt Lake City, Utah, USA

Corresponding author:
Quoc Anh Tran
PhD student
Department of Civil Engineering, Aalto University
Rakentajanaukio 4 A, 02150 Espoo, FINLAND
Email: tran.quocanh@aalto.fi

ABSTRACT

Applying the Convected Particle Domain Interpolation (CPDI) to the Material Point Method has many advantages over the original Material Point Method, including significantly improved accuracy. However, in the large deformation regime, the CPDI still may not retain the expected convergence rate. The paper proposes an enhanced CPDI formulation based on least square reconstruction technique. The Convected Particle Least Square Interpolation (CPLS) Material Point Method assumes the velocity field inside the material point domain as non-constant. This velocity field in the material point domain is mapped to the background grid nodes with a Moving Least Squares reconstruction. In this paper, we apply the Improved Moving Least Squares method to avoid the instability of the conventional Moving Least Squares method due to a singular matrix. The proposed algorithm can improve convergence rate, as illustrated by numerical examples using the Method of Manufactured Solutions.

1. INTRODUCTION

The Material Point Method (MPM), proposed by Sulsky *et al.* [1], has been successfully used to solve large deformation problems. However, when the material points cross the cell boundaries, numerical noises are introduced into the numerical solution. The Generalized Interpolation Material Point Method (GIMP) [2] can mitigate that cell-crossing instability. GIMP introduced material point domains which are supported by charateristic functions. The GIMP shape functions are the convolution of a charateristic function and a grid shape function. To calculate the integral of the shape functions, it is essential to track the evolution of the material point domain during deformations. GIMP assumes that the material point domains do not change (referred to as uGIMP), or track only the axial deformation of the material point domains (referred to as cpGIMP). In uGIMP and cpGIMP, the material point domains remain rectangular in 2D or cuboid in 3D, which leads to some inaccuracies during large shear deformation. On the other hand, Convected Particle Domain Interpolation Method [3, 4] (CPDI) allows the material point domains to become parallelograms in 2D or parallelepiped in 3D. In such calculations, the CPDI approximates shape functions by an interpolation of the grid shape function at the parallelograms' corners using the 4-node quadrilateral finite element. The CPDI improves the numerical accuracy of the MPM and prevents the occurrence of the non-physical numerical fractures in the MPM, which is possible under the large extension condition. Given these advantages, the CPDI has been applied in the MPM simulations of, among others: nanoindentation [5], chemical/mechanical coupling of a silicon anode [6], the impact of thin-walled structures [7] as well as being enhanced to model fracture [8] or fluid-driven fracture [9]. The CPDI has also been used in simulations of geotechnical problems, such as modelling pile penetration [10], pile installation [11], soil liquefaction [12] and the coupled analysis of saturated porous media [13].

Although the CPDI can improve the numerical accuracy of the MPM and may lead to a 2nd

order of convergence when the deformation is relatively small compared with the grid cell size (i.e. when no material points cross cell boundaries), this paper demonstrates that the CPDI cannot maintain the same convergence rate in large deformation problems. We propose to enhance the CPDI method, so it retains the same convergence rate in large deformation problems. Our proposed method constructs the velocity field in the entire particle domain, in a similar way to the MPM gradient projection [14]. Then, the nodal velocity is calculated on the background grid with the Moving Least Squares reconstruction technique. We refer to this method as the Convected Particle Least Square Interpolation Material Point Method (CPLS).

The Least Square reconstruction technique has been used to improve the accuracy of the MPM beforehand [15, 16, 17, 18, 19, 20, 21]. However, these formulations require an inversion of the least squares moment matrix and a solution of a system of equations. This is numerically expensive, while requiring additional numerical treatments in case the moment matrix is singular. To avoid the instability of the original Moving Least Squares technique, this paper adopts an Improved Moving Least Squares (IMLS) technique, which has been applied elsewhere, to the element-free Galerkin method framework [22, 23, 24]. The difference is that IMLS applies orthogonal polynomials instead of monomials to overcome the instability of the conventional Moving Least Squares reconstruction.

The paper is organized as follows. The formulation and algorithm of the new improved moving least square shape function are presented in Section 2. Section 3 describes the combination of the IMLS shape function combined with the CPDI. Though, this combination is referred to as the CPLS. Section 4 demonstrates the capability of the proposed formulation, using the Method of Manufactured Solutions and examples in which the displacement magnitude is controlled by a single parameter to switch between a problem characterized by small or large deformations. In this paper, the subscript p denotes the particle index, the subscript j denotes the index of polynomial order and the subscript i denotes the nodal index.

2. IMPROVED MOVING LEAST SQUARES RECONSTRUCTION

2.1. Moving Least Squares (MLS) reconstruction

The MLS approximation of an unknown function $u(\mathbf{x})$ is:

$$u_h(\mathbf{x}) = \sum_{j=1}^{n_b} p_j(\mathbf{x})\, a_j(\mathbf{x}) = \mathbf{p}^T(\mathbf{x})\mathbf{a}(\mathbf{x}) \tag{1}$$

in which $\mathbf{p}(\mathbf{x})$ is the polynomial basis vector, $\mathbf{a}(\mathbf{x})$ is the unknown coefficient vector and n_b is number of basis functions. Given a set of N data points with position $\{x\}_{p=1}^{N}$ and values of these points $\{u\}_{p=1}^{N}$, the coefficient vector can be computed by minimizing the weighted least-square error J given by:

$$J = \sum_{p=1}^{N} w(\mathbf{x} - \mathbf{x}_p)\left[\mathbf{p}^T(\mathbf{x})\mathbf{a}(\mathbf{x}) - u_p\right]^2 \tag{2}$$

where $w(\mathbf{x} - \mathbf{x}_p) \geq 0$ is a positive weighting function. Then, after differentiating the weighted least-square error J with respect to $\mathbf{a}(\mathbf{x})$ and setting it to zero to we obtain:

$$\frac{\delta J}{\delta a} = 2\sum_{p=1}^{N} w(\mathbf{x} - \mathbf{x}_p)\mathbf{p}(\mathbf{x})\left[\mathbf{p}^T(\mathbf{x})\mathbf{a}(\mathbf{x}) - u_p\right] = 0 \tag{3}$$

Equation (3) can be rewritten in matrix form as:

$$\mathbf{A}(\mathbf{x})\mathbf{a}(\mathbf{x}) = \mathbf{B}(\mathbf{x})\mathbf{U}_p \tag{4}$$

where **A** is the moment matrix and **B** can be written as:

$$\mathbf{A}(\mathbf{x}) = \sum_{p=1}^{N} w(\mathbf{x} - \mathbf{x}_p)\mathbf{p}(\mathbf{x}_p)\mathbf{p}^T(\mathbf{x}_p) \tag{5}$$

$$\mathbf{B}(\mathbf{x}) = [w(\mathbf{x} - \mathbf{x}_1)\mathbf{p}(\mathbf{x}_1) \quad w(\mathbf{x} - \mathbf{x}_2)\mathbf{p}(\mathbf{x}_2) \quad \ldots \quad w(\mathbf{x} - \mathbf{x}_N)\mathbf{p}(\mathbf{x}_N)] \tag{6}$$

$$\mathbf{U}_p = [u_1\, u_2 \ldots u_N]^T \tag{7}$$

Solving equation (4) is expensive as it requires the inversion of matrix **A**, while it may also be numerically difficult when matrix **A** is singular. Therefore, instead of solving the linear

93 equation, we adopt the Improved Moving Least Squares technique, which uses orthogonal
94 polynomials as basis.

95 ## 2.2. Improved Moving Least Squares (IMLS) shape function for the MPM

96 For 2 arbitrary polynomials $f(\mathbf{x})$ in a domain Ω_f and $g(\mathbf{x})$ in a domain Ω_g, given a non-negative
97 weighting function $w(\mathbf{x}) \geq 0$, a weighted inner product of two polynomials f and g is defined
98 as:

$$\forall\, f(\mathbf{x}), g(\mathbf{x}) \quad \langle f, g\rangle = \int_{\Omega_f \cap \Omega_g} w(\mathbf{x}) f(\mathbf{x}) g(\mathbf{x}) dx \tag{8}$$

99 For the set of N material points with positions $\{\mathbf{x}_p\}_{p=1}^{N}$, the value of the weighted inner product
100 for a given position **x** could be approximated as:

$$\langle f, g\rangle \approx \sum_{p=1}^{N} w(\mathbf{x} - \mathbf{x}_p) f(\mathbf{x}_p) g(\mathbf{x}_p) \tag{9}$$

101 In this paper, a structured grid with the size of l_e is used for the numerical simulations. Then,
102 we select the weighting function as a cubic spline, suggested by Steffen *et al.* [25], for the
103 function reconstruction as:

$$w(x) = \begin{cases} \frac{2}{3} - \frac{x^2}{l_e^2} + \frac{|x|^3}{2l_e^3} & |x| \leq l_e \\ \frac{4}{3} - \frac{2x}{l_e} + \frac{x^2}{l_e^2} - \frac{|x|^3}{6l_e^3} & l_e \leq |x| \leq 2l_e \\ 0 & |x| \geq 2l_e \end{cases} \tag{10}$$

104 The cubic spline gives a smoother reconstructed function compared to the linear basis function.
105 Furthermore, the wider domain captures more material points to reconstruct the higher order
106 function and because the domain of the cubic spline is divisible to the size of the grid cell, it is
107 numerically more convenient to find the material points in the support of the nodal weighting
108 function when a structured grid is used. After taking into account equation (9), equation (4)
109 becomes:

$$\begin{bmatrix} \langle p_1, p_1 \rangle & \langle p_1, p_2 \rangle & \cdots & \langle p_1, p_{n_b} \rangle \\ \langle p_2, p_1 \rangle & \langle p_2, p_2 \rangle & & \langle p_2, p_{n_b} \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle p_{n_b}, p_1 \rangle & \langle p_{n_b}, p_2 \rangle & \cdots & \langle p_{n_b}, p_{n_b} \rangle \end{bmatrix} \begin{bmatrix} a_1(\mathbf{x}) \\ a_2(\mathbf{x}) \\ \vdots \\ a_{n_b}(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \langle p_1, u_p \rangle \\ \langle p_2, u_p \rangle \\ \vdots \\ \langle p_{n_b}, u_p \rangle \end{bmatrix} \tag{11}$$

If the basis function set $p_j(\mathbf{x})$ consists of orthogonal functions, then:

$$\langle p_j, p_k \rangle = 0, \quad \text{for } j \neq k \tag{12}$$

That means equation (11) becomes a diagonal matrix:

$$\begin{bmatrix} \langle p_1, p_1 \rangle & 0 & \cdots & 0 \\ 0 & \langle p_2, p_2 \rangle & & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \langle p_{n_b}, p_{n_b} \rangle \end{bmatrix} \begin{bmatrix} a_1(\mathbf{x}) \\ a_2(\mathbf{x}) \\ \vdots \\ a_{n_b}(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \langle p_1, u_p \rangle \\ \langle p_2, u_p \rangle \\ \vdots \\ \langle p_{n_b}, u_p \rangle \end{bmatrix} \tag{13}$$

Then, the coefficients $a_j(\mathbf{x})$ are given by:

$$a_j(\mathbf{x}) = \frac{\langle p_j, u_p \rangle}{\langle p_j, p_j \rangle}; \quad j = 1, \dots, n_b \tag{14}$$

Substituting equation (14) into equation (1), gives

$$u_h(\mathbf{x}) = \sum_{j=1}^{n_b} p_j(\mathbf{x}) \frac{\langle p_j, u_p \rangle}{\langle p_j, p_j \rangle} = \sum_{j=1}^{n_b} p_j(\mathbf{x}) \frac{\sum_{p=1}^{N} w(\mathbf{x} - \mathbf{x}_p) p_j(\mathbf{x}_p) u_p}{\langle p_j, p_j \rangle}, \tag{15}$$

and

$$u_h(\mathbf{x}) = \sum_{p=1}^{N} \sum_{j=1}^{n_b} \frac{w(\mathbf{x} - \mathbf{x}_p) p_j(\mathbf{x}) p_j(\mathbf{x}_p) u_p}{\langle p_j, p_j \rangle} = \Phi^{\mathbf{T}}(\mathbf{x}) \mathbf{U}_p = \sum_{p=1}^{N} \phi(\mathbf{x})\, u_p \tag{16}$$

where $\Phi(\mathbf{x})$ is the vector of IMLS shape function, with the IMLS shape function $\phi(\mathbf{x})$ being:

$$\phi(\mathbf{x}) = \sum_{j=1}^{n_b} \frac{w(\mathbf{x} - \mathbf{x}_p) p_j(\mathbf{x}) p_j(\mathbf{x}_p)}{\langle p_j, p_j \rangle} \tag{17}$$

2.3. Orthogonal basis function

In the Moving Least Squares method, the monomials, denoted by **q**, are used to construct the polynomial basis vector. For example, the 2D monomial basis functions are:

$$\mathbf{q} = (q_j) = \{1, x, y, x^2, xy, y^2\} \tag{18}$$

In the Improved Moving Least Squares method, orthogonal polynomials are used instead of monomials. However, in the MPM the positions of the material points are arbitrary and may not coincide with quadrature points, leading to inaccuracies. Therefore, the adoption of certain structured orthogonal polynomials such as Legendre polynomials may not be preferred, as their use does not ensure that the integration in equation (8) always satisfies the orthogonality condition given by equation (12). Instead, in order to use a set of polynomials which always satisfy the orthogonality conditions, the set of polynomials is updated in every time step and depends on the material point coordinates using the Gram-Schmidt orthogonalization method. For example, the 2D polynomials **p** satisfying the orthogonality condition, referred to as the orthogonal basis function, can be computed from the **q** in equation (18) as follows:

$$p_j = q_j - \sum_{k=1}^{j-1} \frac{\langle q_j, p_k \rangle}{\langle p_k, p_k \rangle} p_k \tag{19}$$

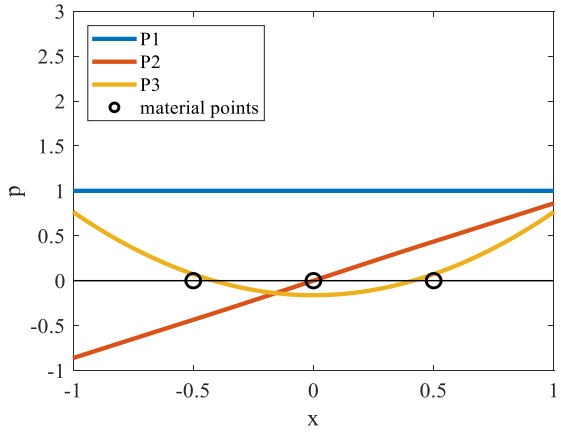


Figure 1 Polynomials in equal distribution of 3 material points with cubic Bspline

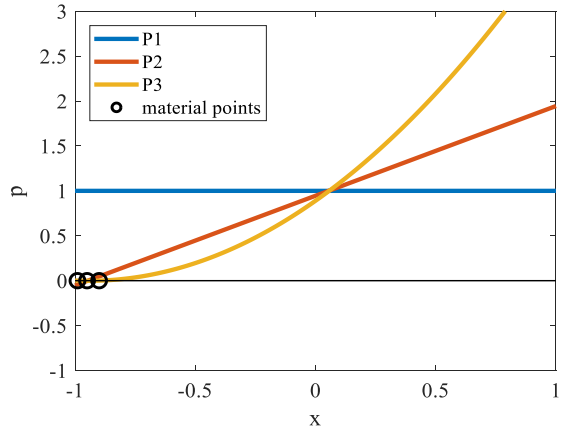


Figure 2 Polynomials in near-corner distribution of 3 material points with cubic Bspline

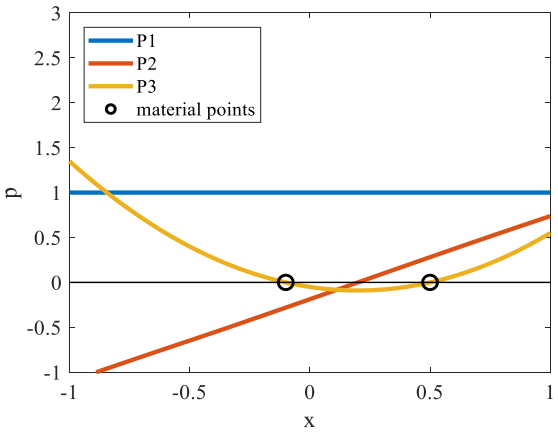


Figure 3 Polynomials for 2 material points, rank(A) = 2 with $\langle \mathbf{p_3}, \mathbf{p_3} \rangle = \mathbf{0}$

Figure 1 and Figure 2 show the change of polynomials in 1D corresponding to 2 different material points coordinates in the interval [-1,1] using a cubic Bspline function. The description of these calculations is presented in the Appendix. In case there is not enough material point data to reconstruct a high order function, the least squares moment matrix A may be singular. Then, given rank(**A**) = r < n_b, the weighted inner products of orthogonal basis functions, which lie in the null space of **A**, become zeros such that:

$$\langle p_j, p_j \rangle = 0 \tag{20}$$

In this case, the IMLS shape function in equation (19) will automatically remove the polynomials which lie in the null space component of **A**. Figure 3 shows the polynomials for 2 material points, using equation (19). In this case, because $p_3(x_p) = 0$ for all the x_p, using equation (9) leading to $\langle p_3, p_3 \rangle = 0$. The algorithm automatically applies to only the 2 first orthogonal basis functions p_1 and p_2 for the function reconstruction.

2.4. Algorithm to construct IMLS shape function

Consider a node x_i and the set of polynomials $\mathbf{q} = [1, x, y]$ for the linear function reconstruction ($n_b = 3$) or $\mathbf{q} = [1, x, y, x^2, xy, y^2]$ for the quadratic function reconstruction ($n_b = 6$), the IMLS nodal value of the shape function for the material point $\mathbf{x}_p$, $\phi_{ip} = \phi_i(\mathbf{x}_p^t)$, can be computed using the algorithm as follows:

Algorithm. IMLS shape function for node $\mathbf{x}_i$

Input:

$\mathbf{x}_p$, N – positions $\mathbf{x}_p$ and number of material points N interacting with $\mathbf{x}_i$

n_b – desired order of polynomials

$\mathbf{q}$ – set of monomials polynomials ($\mathbf{q} = [1, x, y]$ if $n_b = 3$/ $\mathbf{q} = [1, x, y, x^2, xy, y^2]$ if $n_b = 6$)

Output: ϕ_{ip} – IMLS nodal value of the shape function for material point $\mathbf{x}_p$

$\hat{p}_1 = 1$

for j = 1 **to** n_b

for p = 1 **to** N

$w(\mathbf{x}_i - \mathbf{x}_p)$ // Compute the weighting function **(10)**

$\langle \hat{p}_j, \hat{p}_j \rangle = \langle \hat{p}_j, \hat{p}_j \rangle + w(\mathbf{x}_i - \mathbf{x}_p)\hat{p}_j(\mathbf{x}_p)\hat{p}_j(\mathbf{x}_p)$ // Compute the inner product **(9)**

end

if $\langle \hat{p}_j, \hat{p}_j \rangle \neq 0$

for p = 1 **to** N

$\phi_{ip} = \phi_{ip} + \frac{w(\mathbf{x}_i - \mathbf{x}_p)\hat{p}_j(\mathbf{x}_i)\hat{p}_j(\mathbf{x}_p)}{\langle \hat{p}_j, \hat{p}_j \rangle}$ // Compute the IMLS shape function **(17)**

for k = 1 **to** j

for p = 1 **to** N

$\langle q_{j+1}, \hat{p}_k \rangle = \langle q_{j+1}, \hat{p}_k \rangle + w(\mathbf{x}_i - \mathbf{x}_p)q_{j+1}(\mathbf{x}_p)p_k(\mathbf{x}_p)$

end

// Compute the material points value of polynomials $\hat{p}_{j+1}(\mathbf{x}_p)$ **(19)**

$\hat{p}_{j+1}(\mathbf{x}_p) = q_{j+1}(\mathbf{x}_p) - \frac{\langle q_{j+1}, \hat{p}_k \rangle}{\langle \hat{p}_k, \hat{p}_k \rangle}\hat{p}_k(\mathbf{x}_p)$

// Compute the nodal value of polynomials $\hat{p}_{j+1}(\mathbf{x}_i)$ **(19)**

$\hat{p}_{j+1}(\mathbf{x}_i) = q_{j+1}(\mathbf{x}_i) - \frac{\langle q_{j+1}, \hat{p}_k \rangle}{\langle \hat{p}_k, \hat{p}_k \rangle}\hat{p}_k(\mathbf{x}_i)$

end

end

end

end

147 In this paper the linear function reconstruction, given above, is used for all numerical examples.

2.5. Accuracy and efficiency of improved moving least squares interpolation (IMLS)

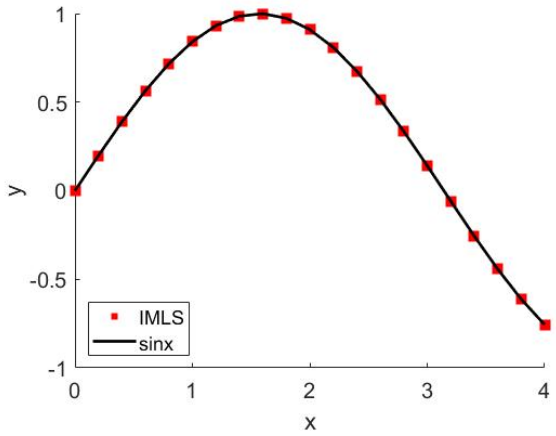


Figure 4 Reconstruction of sin(x) using improved least squares interpolation (IMLS)

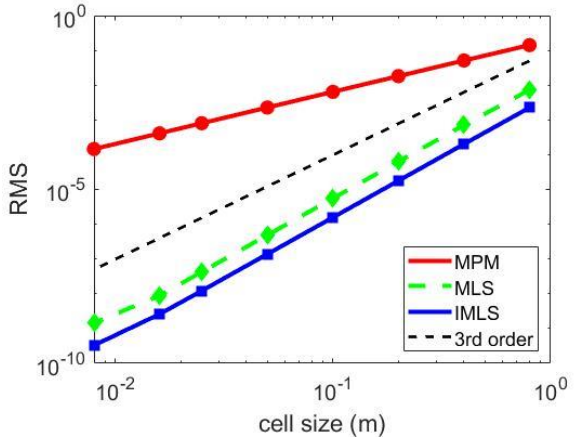


Figure 5 Convergence rate of Moving least squares interpolations

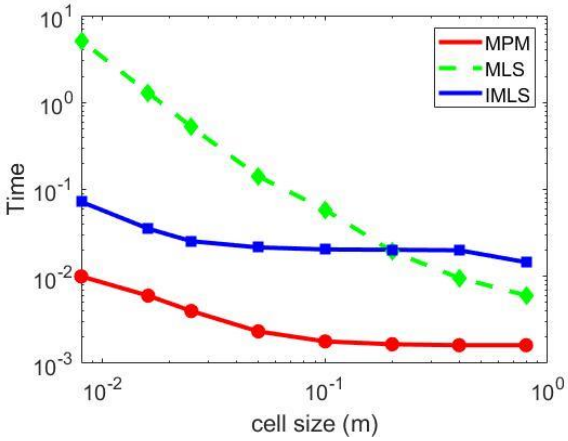


Figure 6 Computational efficiency of moving least squares interpolations

In this section, we examine the accuracy and efficiency of the improved moving least squares interpolation. Firstly, a function $f(x) = \sin(x)$ is considered on the interval $[1, 4]$. Then, we reconstruct the material point data by setting $f_p = f(x_p)$ on the domain with 20 cells and 2 material point equally distributed in each cell. Figure 4 shows the reconstructed nodal values using IMLS with 3 orthogonal polynomials $[p_1, p_2, p_3]$ determined in equation (19). The reconstruction is not exact but the root mean square (RMS) error (see equation (47)) reduces when refining grid. We also compare the IMLS reconstruction with the MPM interpolation and original MLS with the monomials $[1, x, x^2]$. Compared to original MLS, IMLS is demonstrated to maintain similarly high convergence rate (see Figure 5) while become more efficient in terms of computational cost (see Figure 6). The high computational cost of the original MLS stems

from solving the linear system in equation (4) by using the Cholesky decomposition for the symmetric positive-diagonal **A** (recall by the backslash operator in MATLAB).

In literature, different least squares reconstruction techniques have been used (see Table 1) and can be distinguished by the choice of the interpolator, the weighting function and the polynomial sets. The use of monomials in the original MLS requires an additional numerical treatment when the least square moment matrix **A** is singular. On the other hand, Taylor least squares method [18] is designed to enforce the mass and momentum conservation during the mapping from the material points to the nodes. This method requires the reconstruction of material point's data to integration points rather than nodes for the numerical integration on background grid. However, the integration errors will arise if the material points domain does not fully fill the cell [26] (partially-filled cells). This requires additional treatment to subdivide the partially-filled cell for the numerical integration, for example, using the multi-point approximate integration over the partially-filled cells [15] which becomes complex in 3D. In this paper, the improved moving least squares reconstruction is used only for the velocity projection from the material points to the nodes to avoid numerical inaccuracies arising from the partially-filled cells.

Table 1 Summary of least squares reconstruction for MPM

Method	Interpolator based on	Weighting function $w(\mathbf{x})$	Polynomial sets $\mathbf{p}(\mathbf{x})$
Weighted Least Squares [15]	GIMP [2]	Quadratic spline	Monomials
High-order PIC [16]	Particle-In-Cell (PIC) [27]	splines	Monomials
Improved MPM [17]	MPM [1]	B-splines	Monomials
PolyPIC [28]	Affine PIC [29]	B-splines	Monomials
MLS-MPM [21]	Affine PIC [29]/ PolyPIC [28]	B-splines	Monomials
Taylor least squares [18]	B-spline [25]	-	Taylor basis
MLS B-spline [20]	B-spline [25]	B-splines	Monomials

Improved MLS [19]	MPM [1]	B-splines	Orthogonal
This study	CPDI [3]	B-splines	Orthogonal

3. CONVECTED PARTICLE LEAST SQUARE INTERPOLATION (CPLS): COMBINATION OF THE CPDI AND THE IMLS FUNCTION RECONSTRUCTION

3.1. Material points variables field

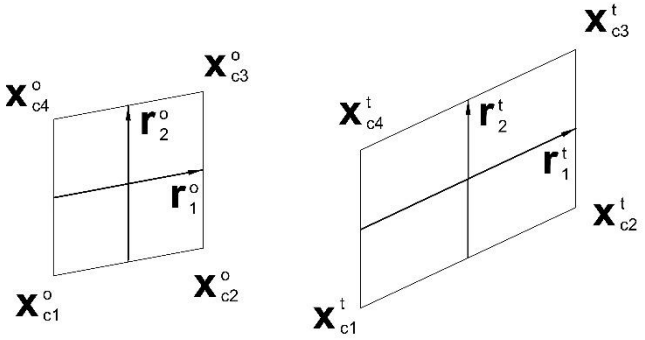


Figure 7 Material point domain in the CPDI

The CPDI method tracks the particle domain as a parallelogram described by two vectors $\mathbf{r}_1^o$ and $\mathbf{r}_2^o$ as illustrated in Figure 7. At a current time step, denoted by a superscript *n*, these two vectors are updated by a current deformation gradient $\mathbf{F}_p^n$ as:

$$\begin{aligned} \mathbf{r}_1^n &= \mathbf{F}_p^t \mathbf{r}_1^o \\ \mathbf{r}_2^n &= \mathbf{F}_p^t \mathbf{r}_2^o \end{aligned} \tag{21}$$

As the particle domain is prescribed as a parallelogram, the corner coordinates could be calculated from the particle coordinate $\mathbf{x}_p^n$ and two vectors $\mathbf{r}_1^n$ and $\mathbf{r}_2^n$ as:

$$\begin{aligned} \mathbf{x}_{c1}^n &= \mathbf{x}_p^n - \mathbf{r}_1^n - \mathbf{r}_2^n \\ \mathbf{x}_{c2}^n &= \mathbf{x}_p^n + \mathbf{r}_1^n - \mathbf{r}_2^n \\ \mathbf{x}_{c3}^n &= \mathbf{x}_p^n + \mathbf{r}_1^n + \mathbf{r}_2^n \\ \mathbf{x}_{c4}^n &= \mathbf{x}_p^n - \mathbf{r}_1^n + \mathbf{r}_2^n \end{aligned} \tag{22}$$

The CPDI shape function and gradient of the shape function can be calculated using the linear basis function φ_i [3] as follows:

$$N_{ip} = N_i\left(\mathbf{x}_p^n\right) = \frac{1}{4}\left(\varphi_i(\mathbf{x}_{c1}^n) + \varphi_i(\mathbf{x}_{c2}^n) + \varphi_i(\mathbf{x}_{c3}^n) + \varphi_i(\mathbf{x}_{c4}^n)\right) \tag{23}$$

$$\nabla N_{ip} = \nabla N_i\left(\mathbf{x}_p^n\right) = \frac{1}{2V_p}\left\{\begin{array}{l}\left(\varphi_i(\mathbf{x}_{c1}^n) - \varphi_i(\mathbf{x}_{c3}^n)\right)\begin{bmatrix}\mathbf{r}_{1y}^n - \mathbf{r}_{2y}^n \\ \mathbf{r}_{2x}^n - \mathbf{r}_{1x}^n\end{bmatrix} \\ + \left(\varphi_i(\mathbf{x}_{c2}^n) - \varphi_i(\mathbf{x}_{c4}^n)\right)\begin{bmatrix}\mathbf{r}_{1y}^n + \mathbf{r}_{2y}^n \\ -\mathbf{r}_{2x}^n - \mathbf{r}_{1x}^n\end{bmatrix}\end{array}\right\} \tag{24}$$

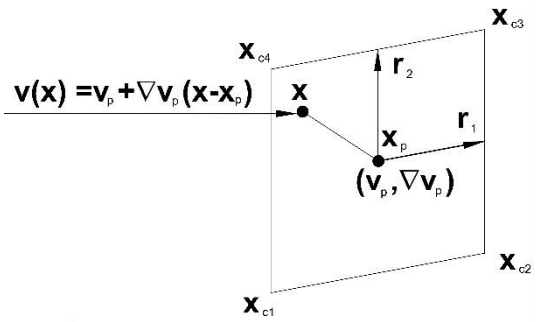


Figure 8 Particle velocity field

In the CPDI, the material points momenta $(m\mathbf{v})_p$ are interpolated to the grid nodes using the CPDI shape function given in equation (23). This implies that the velocity/momentum is locally constant in the material point domain, then distributed evenly to the material points corners (by the weight of 0.25 in the area of a quadrilateral particle domain) and interpolated from the material points corners to the grid nodes using the linear basis functions φ_i. This may reduce the spatial numerical accuracy as the velocity field in each material point domain are zero-order polynomials. Different approaches have been studies to improve the velocity projection for the MPM. Wallstedt and Guilkey [14] employed the velocity gradient to GIMP's velocity projection to reduce the velocity projection errors. Jiang *et al.* [29, 30] applied a velocity gradient to the momentum projection for the angular momentum conservation. While both approaches above consider velocity gradient in the velocity projection, the former considers the linear velocity gradient field while the latter considers both linear and rotation velocity gradient field to obtain the conservation of the angular momentum. In this study, we use a similar approach from Wallstedt and Guilkey [14] but for the CPDI. The velocity field in the material point's domain depends on the velocity gradient $\nabla\mathbf{v}_p$ as follows:

$$\mathbf{v}(\mathbf{x}) = \mathbf{v}_p + \nabla \mathbf{v}_p(\mathbf{x} - \mathbf{x}_p) \tag{25}$$

In other words, the velocity is approximated as a linear function in the material point domain. As such, this approximation does not require an extra material point variable because the velocity gradient $\nabla \mathbf{v}_p$ is already available within the MPM algorithm. Therefore, no extra step is required to compute the velocity gradient. However, the velocity gradients need to be stored as a state variable at material points. When the velocity is prescribed as a velocity field, the velocities at material point corners can be calculated using Equation (25). Then, each material point domain consists of 5 velocity data points including the velocity at material points and four corners, ($\mathbf{v}_p$, $\mathbf{v}_{c1}$, $\mathbf{v}_{c2}$, $\mathbf{v}_{c3}$, $\mathbf{v}_{c4}$). These data points can be used to reconstruct the velocity function in the background grid using the Improved Moving Least Squares method as described in the previous section.

3.2. Time discrete equations

The numerical solution is obtained at discrete time steps using the Updated Stress Last approach [31] and the semi-implicit Euler time integration where Δt is the time step and N_t is the total number of time steps. The solution in both nodes and material points in the current time step is denoted by a superscript *n* and in the next time step is:

$$t^{n+1} = t^n + \Delta t \tag{26}$$

Initially, state variables for solids are generated including mass m_p, position $\mathbf{x}_p$, velocity of material points and velocities of material point corners $\mathbf{v}_p, \mathbf{v}_{c=1:4}$, volume V_p, stress $\boldsymbol{\sigma}_p$, body force $\mathbf{b}_p$, deformation gradient $\mathbf{F}_p$. At the first of each time step, the nodal mass m_i^n, internal forces $\mathbf{f}_i^{b,n}$ and external forces $\mathbf{f}_i^{int,n}$ are computed as:

$$m_i^n = \sum_p N_{ip}\, m_p \tag{27}$$

$$\mathbf{f}_i^{b,n} = \sum_p N_{ip} \mathbf{b}_p^n m_p \tag{28}$$

$$\mathbf{f}_i^{int,n} = -\sum_p \nabla N_{ip} \boldsymbol{\sigma}_p^n V_p \tag{29}$$

In the original CPDI method, the nodal velocity is updated as using the CPDI shape function N_{ip}:

$$\mathbf{v}_i^n = \left(\sum_p N_{ip} \, m_p \mathbf{v}_p^n \right) \Big/ m_i^n \tag{30}$$

However, in this paper, the nodal velocity is updated using the IMLS shape function in equation (17) evaluated at the position of material points $\phi_{ip} = \phi_i(\mathbf{x}_p^n)$ and at the position of material point corners $\phi_{ic} = \phi_i(\mathbf{x}_c^n)$ as:

$$\mathbf{v}_i^n = \sum_p \left(\phi_{ip} \mathbf{v}_p^n + \sum_{c=1}^{4} \phi_{ic} \mathbf{v}_c^n \right) \tag{31}$$

where $\mathbf{v}_i^n$ is the nodal velocity, $\mathbf{v}_p^n$ is the material point's velocity, $\mathbf{v}_c^n$ is the velocity at the corners of the material point. ϕ_{ip} and ϕ_{ic} are the IMLS nodal shape functions. Because the least squares shape function does not guarantee the partition of the unity, the linear momentum may not be conserved. Therefore, the conservation errors of the proposed least squares interpolation are examined in the given numerical examples. Subsequently, the CPDI algorithm [3] is followed. The total force $\mathbf{f}_i^n$, nodal acceleration $\mathbf{a}_i^n$ and velocity at the next time step $\mathbf{v}_i^{n+1}$ are calculated as:

$$\mathbf{f}_i^n = \mathbf{f}_i^{int,n} + \mathbf{f}_i^{b,n} + \mathbf{f}_i^{ext,n} \tag{32}$$

$$\mathbf{a}_i^n = \frac{\mathbf{f}_i^n}{m_i^n} \tag{33}$$

$$\mathbf{v}_i^{n+1} = \mathbf{v}_i^n + \mathbf{a}_i^n dt \tag{34}$$

After solving the equations of motion, the velocities and positions of the material points are updated:

$$\mathbf{v}_p^{n+1} = \mathbf{v}_p^n + \sum_i N_{ip}\mathbf{a}_i^n dt \tag{35}$$

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \sum_i N_{ip}\mathbf{v}_i^{n+1} dt \tag{36}$$

The velocity gradients, using the gradient of the shape function, can be computed at particle position based on nodal velocities as:

$$\nabla\mathbf{v}_p^{n+1} = \sum_i \nabla N_{ip}\mathbf{v}_i^{n+1} \tag{37}$$

Subsequently, the quantities such as deformation gradient $\mathbf{F}_p^{n+1}$, volume of the material point V_p^{n+1} and density of the material points ρ_p^{n+1} are updated:

$$\mathbf{F}_p^{n+1} = (\mathbf{I} + \nabla\mathbf{v}_p^{n+1} dt)\mathbf{F}_p^n \tag{38}$$

$$V_p^{n+1} = \det\left(\mathbf{F}_p^{n+1}\right) V_p^o \tag{39}$$

$$\rho_p^{n+1} = \frac{m_p}{V_p^{n+1}} \tag{40}$$

After that, the topology parameters $\mathbf{r}_1^{n+1}$, $\mathbf{r}_2^{n+1}$ are updated using equation (21) and position of material point corners $\mathbf{x}_c^{n+1}$ in equation (22). Now, the extra step required beyond the algorithm [3] is to update the velocities at material point domain corners:

$$\mathbf{v}_c^{n+1} = \mathbf{v}_p^{n+1} + \nabla\mathbf{v}_p^{n+1}\left(\mathbf{x}_c^{n+1} - \mathbf{x}_p^{n+1}\right) \tag{41}$$

Then, a constitutive model is called to update the stress $\boldsymbol{\sigma}_p^{n+1}$ at the material points at the end of the time step.

4. NUMERICAL EXAMPLES

4.1. Method of Manufactured Solutions

Method of Manufactured Solutions (MMS) has been used for the verification and validation for the MPM solution. The advantage of MMS is to test codes with boundaries or nonlinearities in which the exact solution is difficult to obtain. Furthermore, in MMS, the magnitude of the displacement in the solution can be controlled by a single parameter. It is convenient to study the cell-crossing errors in the MPM because the displacement magnitude can be modified ranging from small to large deformation. Steffen *et al.* [32] applied MMS for a 2D axis-aligned vibration problem; Wallstedt and Guilkey [33] adopted MMS to study the convergence rate of GIMP by simulating a large deformation radial expansion of a ring. Later, Kamojjala *et al.* [34] presented a serial of verification tests for solid mechanics including a MMS generalized vortex problem. In this paper, the axis-aligned vibration [32], the radial expansion of a ring [33] and the MMS generalized vortex of a ring [34] were used to study the convergence rate of the CPDI and the CPLS, presenting in section 4.2, 0 and 4.4 respectively. This section summarizes the general derivation of the method of manufacture solution for hyper-elastic materials. Firstly, the linear momentum balance equation can be written in the total Lagrangian form and updated Lagrangian form as follows:

$$\rho_o \mathbf{a} = \nabla . \mathbf{P} + \rho_o \mathbf{b} \tag{42}$$

$$\rho \mathbf{a} = \nabla . \boldsymbol{\sigma} + \rho \mathbf{b} \tag{43}$$

where

- ρ_o Density in reference configuration,
- ρ Density in current configuration,
- $\mathbf{a}$ Acceleration vector,
- $\mathbf{P}$ 1st Piola-Kichhoff Stress,
- $\boldsymbol{\sigma}$ Cauchy Stress,
- $\mathbf{b}$ Body force vector.

To manufacture the solution, a non-linear time-dependent displacement solution **u** is defined. Then the deformation gradient is computed as:

$$\mathbf{F} = \mathbf{I} + \frac{\delta \mathbf{u}}{\delta \mathbf{X}} \tag{44}$$

After that, the 1st Piola-Kichhoff stress of the hyper-elastic neo-Hookean model to the reference configuration, denoted as **X,** can be written as:

$$\mathbf{P} = \lambda \ln J \mathbf{F}^{-1} + \mu \mathbf{F}^{-1}(\mathbf{F}\mathbf{F}^{\mathrm{T}} - \mathbf{I}) \tag{45}$$

where J is the determinant of deformation gradient **F**, μ and λ are the shear modulus and Lame constant. The Cauchy stress to the current configuration, denoted as **x,** is:

$$\boldsymbol{\sigma} = \frac{\mathbf{P}\mathbf{F}^{\mathrm{T}}}{J} = \frac{\lambda \ln J}{J}\mathbf{I} + \frac{\mu}{J}(\mathbf{F}\mathbf{F}^{\mathrm{T}} - \mathbf{I}) \tag{46}$$

The acceleration is computed by twice differentiating the displacement **u** with respect to time. Then, substituting the given stress and acceleration to the govern equation (42), the body force can be found. The body forces are used as the source term. The source term (body force) is manufactured as the input for the model. The Root Mean Square (RMS) errors for numerical simulations are defined as:

$$\mathrm{RMS} = \sqrt{\frac{\sum_{p=1}^{N_p} \left\| \mathbf{f}_{\mathrm{numerical}}(\mathbf{x}_p) - \mathbf{f}_{\mathrm{exact}}(\mathbf{x}_p) \right\|^2}{N_p}} \tag{47}$$

4.2. Axis-aligned vibration

The first numerical example is a 1D vibration problem which the displacement and the velocity are:

$$\mathbf{u}(X,t) = A \sin\left(2\frac{\pi X}{L}\right) \sin\left(\frac{c\pi t}{L}\right) \tag{48}$$

$$\mathbf{v}(X,t) = \frac{d\mathbf{u}}{dt} = Ac\pi \sin\left(2\frac{\pi X}{L}\right) \cos\left(\frac{c\pi t}{L}\right) \tag{49}$$

where A is the maximum amplitude of displacement, L is size of the unit square (L=1m), X is the position of material points in the reference configuration, c is the wave speed ($c = \sqrt{\frac{E}{\rho}}$). The deformation gradient computed using equation (44) is:

$$\mathbf{F}(X,t) = 1 + 2\frac{A\pi}{L}\cos\left(2\frac{\pi X}{L}\right)\sin\left(\frac{c\pi t}{L}\right) \tag{50}$$

Using the Method of Manufactured Solutions for the equation of motion, the body forces required to obtain the displacement and velocity field above are:

$$\mathbf{b}(X,Y,t) = \frac{\pi\,\mathbf{u}(X,t)}{L^2}\left(4\frac{\mu}{\rho_o} - c^2 - 4\frac{\lambda\left[\ln\left(\mathbf{F}(X,t)\right) - 1\right] - \mu}{\rho_o\mathbf{F}(X,t)^2}\right) \tag{51}$$

Table 2 Numerical parameters for 1D vibration

Parameters	Symbol	Value
Young's modulus	E	$10e^7$ Pa
Poisson's ratio	ν	0
Shear modulus	μ	$3.85e^6$ Pa
Lamé constant	λ	$5.77e^6$ Pa
Initial density	ρ_o	1000 kg/m^3
Final time	T	0.02s
Time step	dt	$0.4\frac{h}{c}$
Displacement amplitude	A	0.0001m and 0.1m

A 1m-length bar is discretized into 8, 16, 32, 64,128, 256, 512 cells with 4 material points/particles per cell, labeled "PPC" in the figure's legend. This discretization corresponds to cell sizes of 0.125m, (0.125/2)m, (0.125/4)m, (0.125/8)m, (0.125/16)m, (0.125/32)m and (0.125/64)m respectively. Table 2 summarizes the numerical parameters. The time step dt is equal to $0.4\frac{h}{c}$. The final time of the simulation is 0.02s corresponding to 1 cycle motion of material points. We compared the CPLS, with the CPDI [3] and the MPM [1] in the small deformation A = 0.0001m and the large deformation A = 0.1m. All the algorithms show similar accuracy and approximate 2nd order convergence rate for displacements (Figure 9), velocities

(Figure 11) and stresses (Figure 13) when the deformation is small and cell-crossing does not occur. However, when the deformations are large leading to excessive cell-crossing, the MPM diverges and the CPDI does not retain the expected convergence rate. Furthermore, increasing the initial number of material points per cell (1 PPC, 4 PPC and 9PPC equally distributed in a square grid cell) increases slightly the convergence rate for the CPDI. In contrast, the CPLS retains the same order convergence rate for displacements (Figure 10), velocities (Figure 12) and stresses (Figure 14) in the large deformation. Similarly, the CPLS convergence rates shown are almost independent of the number of material points per cell. Figure 15 shows the computational cost of the CPLS-1PPC and the CPDI with different PPC. Visually, the CPLS with 1PPC has a similar computational cost with the CPDI with 9PPC in 2D but can lead to a higher accuracy. As the accuracy of the CPLS is independent from the number of the material points per cells, it is advised to avoid the excessive number of material points in cell. Furthermore, another reason for avoiding the excessive number of material points per cell is that they may contribute to the ringing instability/null-space instability. This instability was demonstrated for the MPM [35, 36] as well as for GIMP [37] and Dual Domain Material Point Method [38, 39].

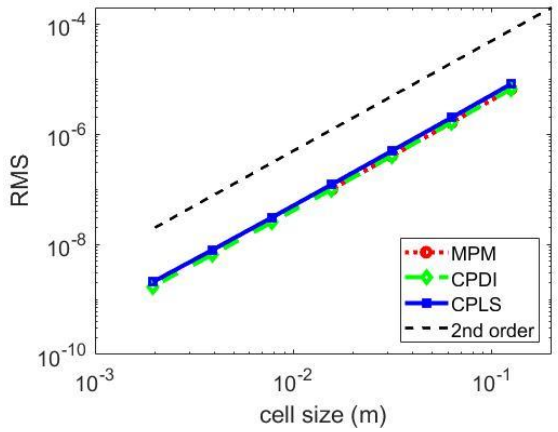


Figure 9 Displacement Convergence rate at small deformation (A=0.0001m)

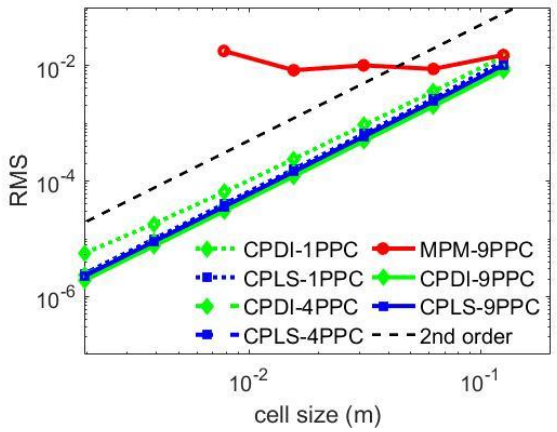


Figure 10 Displacement Convergence rate at large deformation (A=0.1m)

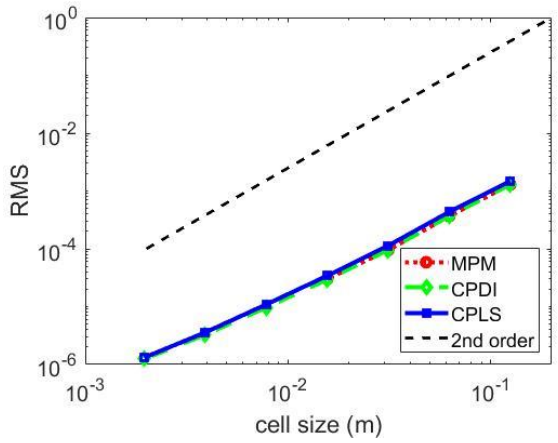


Figure 11 Velocity Convergence rate at small deformation (A=0.0001m)

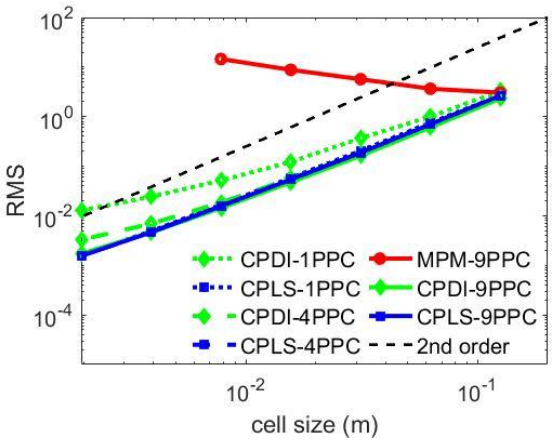


Figure 12 Velocity Convergence rate at large deformation (A=0.1m)

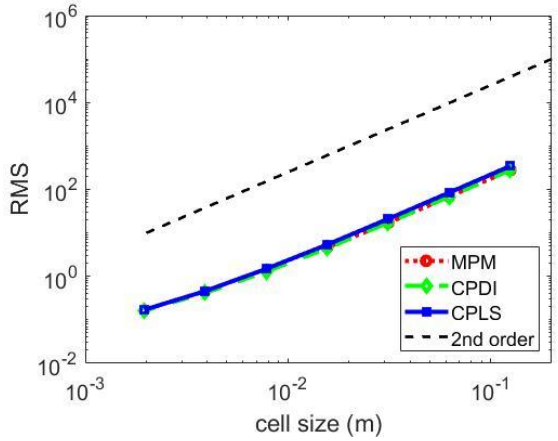


Figure 13 Stress Convergence rate at small deformation (A=0.0001m)

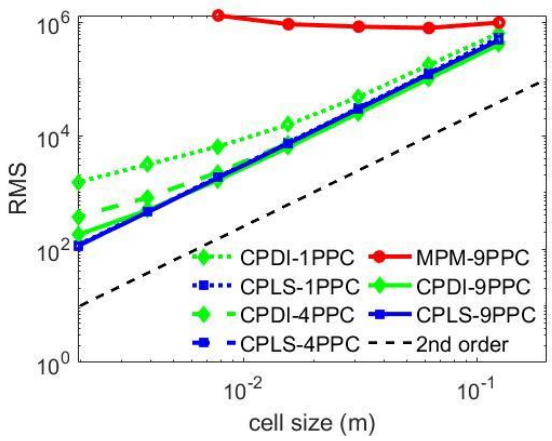


Figure 14 Stress Convergence rate at large deformation (A=0.1m)

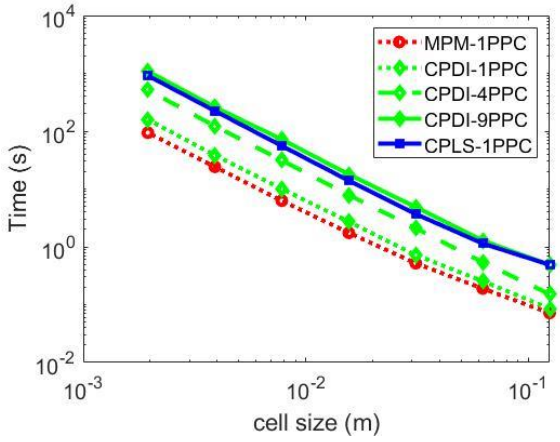


Figure 15 Computational cost of the CPLS

311 The conservation errors of the least-squares interpolation are examined by comparing the total
312 linear momentum of all material points and the total linear momentum of all nodes, which are
313 defined by:

$$P_p = \sum m_p |\mathbf{v}_p| \tag{52}$$

$$P_i = \sum m_i |\mathbf{v}_i| \tag{53}$$

314 Figure 16 and Figure 17 show the total linear momentum of both the CPDI and the CPLS for
315 the cell size of 0.125m. While the CPDI shape function enforces a momentum conservation by
316 the partition of unity, the CPLS shows a negligible conservative error (momentum difference
317 between material points and grid nodes) and this error will get smaller by refining the mesh.

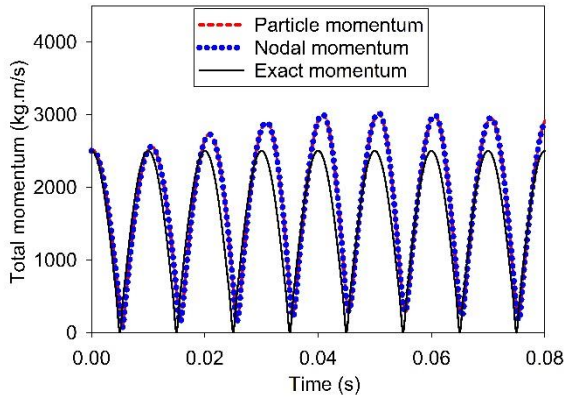


Figure 16 Total momentum of the CPDI for cell size 0.125m (A=0.1m)

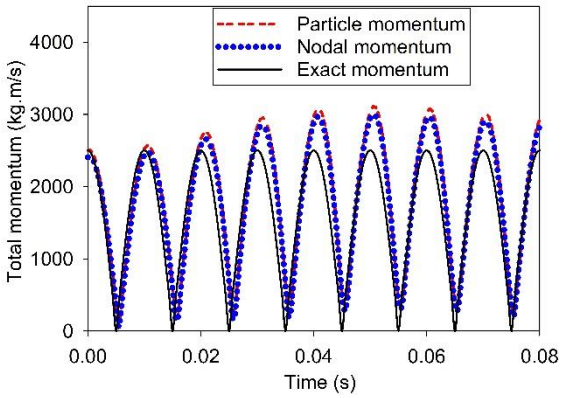


Figure 17 Total momentum of the CPLS for cell size 0.125m (A=0.1m)

318 To check the energy conservation, the total energy E is the sum of the strain U and the kinetic
319 energy K which are computed in each time step as:

$$K = \frac{1}{2}\sum_p m_p \|\mathbf{v}_p\|^2 \tag{54}$$

$$U = \frac{1}{2}\sum_p \boldsymbol{\sigma}_{p,ij}\boldsymbol{\varepsilon}_{p,ij}V_p \tag{55}$$

$$E = U + K \tag{56}$$

Figure 18, Figure 19 and Figure 20 present the evolutions of the kinetic energy, the strain energy and the total energy. Both the CPDI and the CPLS show similar energy evolutions with the maximum energy being slightly higher than the exact solutions, but this energy discrepancy decreases when refining the mesh.

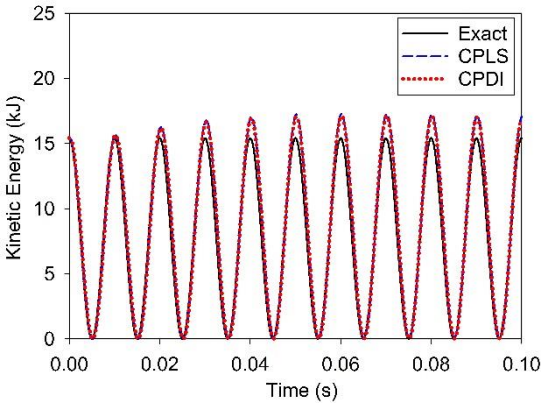


Figure 18 Kinetic energy evolution for cell size 0.125m (A=0.1m)

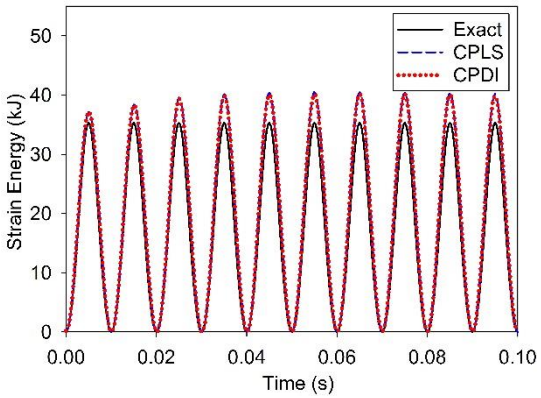


Figure 19 Strain energy evolution of the CPDI for cell size 0.125m (A=0.1m)

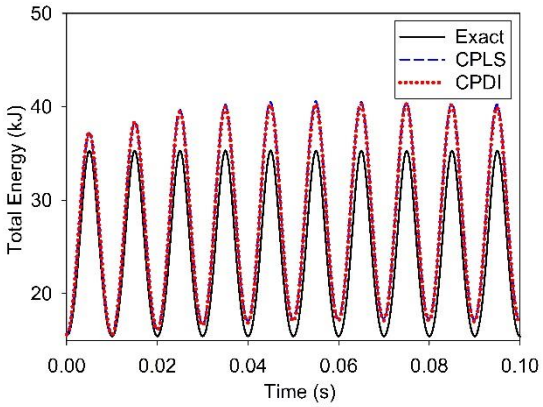


Figure 20 Total energy evolution of the CPDI for cell size 0.125m (A=0.1m)

4.3. 2D Ring expansion

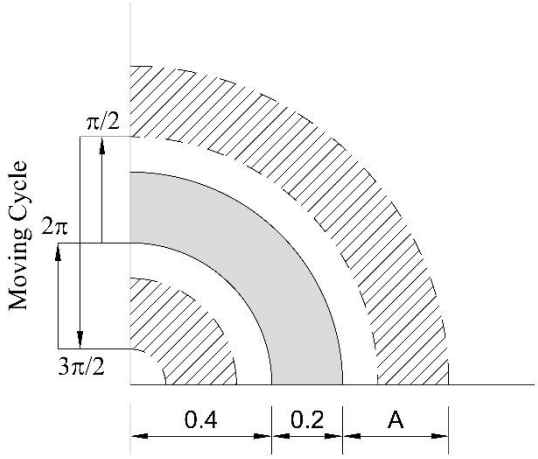


Figure 21 Schematic of 2D Ring expansion

The displacement and velocity field are prescribed with the polar coordinates for a ring as:

$$\mathbf{u}(r,t) = A\sin\left(\frac{c\pi t}{2\bar{r}}\right)(c_3 r^3 + c_2 r^2 + c_1 r_1) \tag{57}$$

$$\mathbf{v}(r,t) = \frac{d\mathbf{u}}{dt} = \frac{Ac\pi}{2\bar{r}}\cos\left(\frac{c\pi t}{2\bar{r}}\right)(c_3 r^3 + c_2 r^2 + c_1 r_1) \tag{58}$$

where A is the maximum amplitude of displacement, r and Θ are radial and angular coordinates in the reference configuration, t is the time, c is the wave speed ($c = \sqrt{\frac{E}{\rho}}$). The inner and outer radius of the ring are r_i and r_o respectively and $\bar{r}$ is the mean radius of the ring $\bar{r} = (r_i + r_o)/2$. The constants c_3, c_2, c_1 are:

$$c_1 = \frac{-6r_i}{r_o(r_o - 3r_i)}; c_2 = \frac{3(r_i + r_o)}{r_o^2(r_o - 3r_i)}; c_3 = \frac{-2}{r_o^2(r_o - 3r_i)} \tag{59}$$

The body forces and stresses are calculated based on the Method of Manufactured solutions.

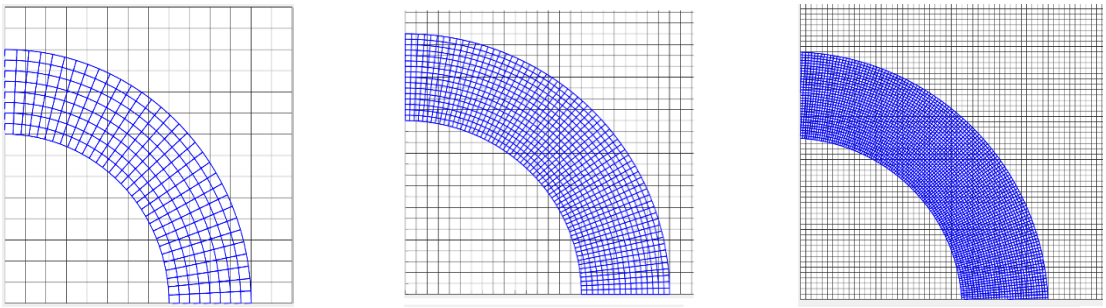
Because of the symmetric condition, only a quarter of a ring is considered with a symmetric boundary condition. Five different cell sizes (l_e) including 0.1m, (0.1/2)m, (0.1/4)m, (0.1/8)m and (0.1/16)m are used to calculate the spatial convergence rate. The number of material points increases proportionally to the grid size. There are $N_r = 0.4/l_e$ material points distributed equally in the radial direction (r=0.2m) and there are $N_\Theta = 1.6/l_e$ material points distributed

equally in the angular coordinate (see Figure 22 for the discretization of material points domain and background grid).

Table 3 Numerical parameters

Parameters	Symbol	Value
Young's modulus	E	$10e^7$ Pa
Poisson's ratio	ν	0
Shear modulus	μ	$3.85e^6$ Pa
Lamé constant	λ	$5.77e^6$ Pa
Initial density	ρ_o	1000 kg/m^3
Final time	T	0.02s
Time step	dt	$0.4\frac{h}{c}$
Displacement amplitude	A	0.0001m and 0.1m

Table 3 shows the numerical parameters for the 2D ring expanding examples. Two algorithms are compared (CPDI and CPLS) in small deformation A = 0.0001m and large deformation A = 0.1m. For small deformation, both the CPDI and the CPLS show a 2^{nd} order convergence rate for displacement (Figure 23), velocity (Figure 25) and 1^{st} order convergence rate for stress (Figure 27). However, in large deformation, the CPDI cannot retain the same order convergence rate for the displacement and the velocity and 1^{st} order convergence rate for the stress. In contrast, the CPLS retains a same order convergence rate for the displacement (Figure 24), the velocity (Figure 26) and 1^{st} order convergence rate for the stress (Figure 28). The conservation errors of the CPLS are also negligible in this example (see Figure 29 and Figure 30).



Cell size (0.1/2)m Cell size (0.1/4)m Cell size (0.1/8)m

351 **Figure 22 Grid and material point domain discretization for the 2D Ring Expanding**

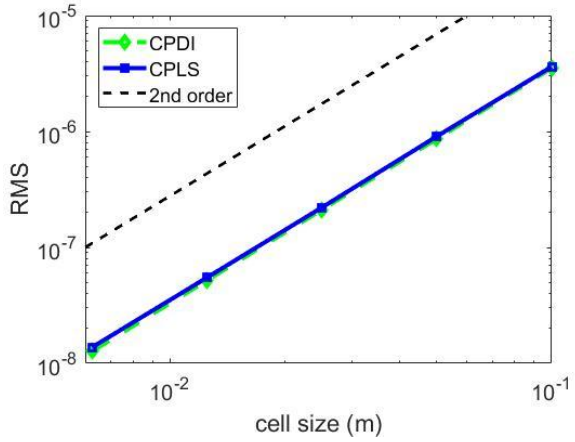


Figure 23 Displacement Convergence rate at small deformation (A=0.0001m)

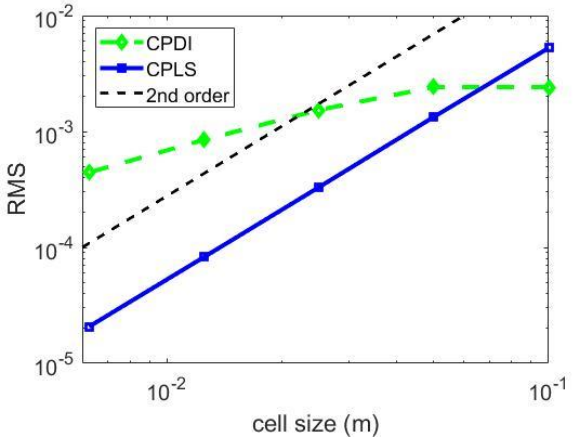


Figure 24 Displacement Convergence rate at large deformation (A=0.1m)

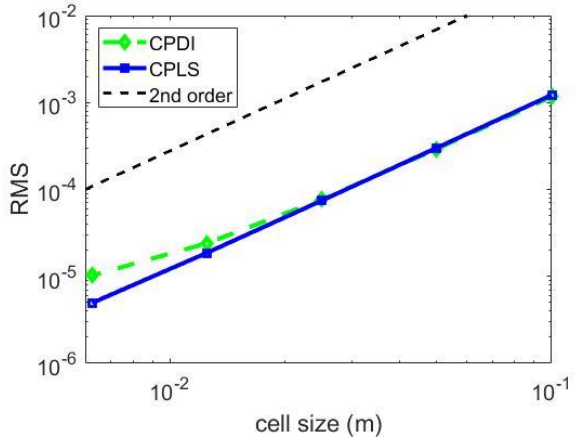


Figure 25 Velocity Convergence rate at small deformation (A=0.0001m)

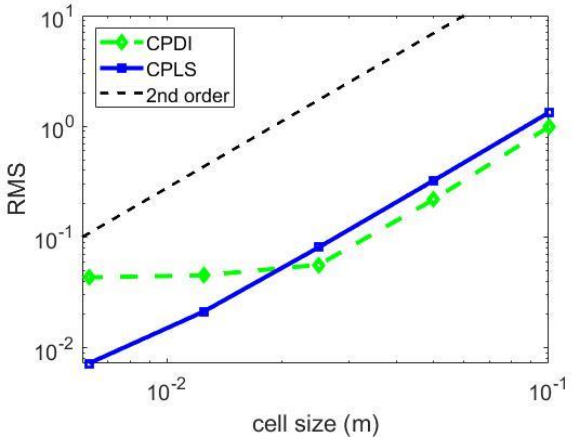


Figure 26 Velocity Convergence rate at large deformation (A=0.1m)

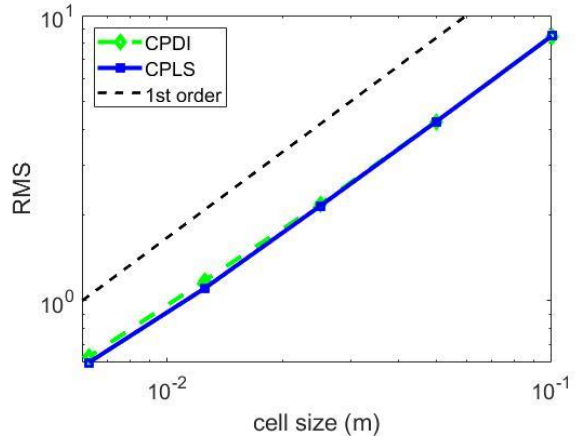


Figure 27 Stress Convergence rate at small deformation (A=0.0001m)

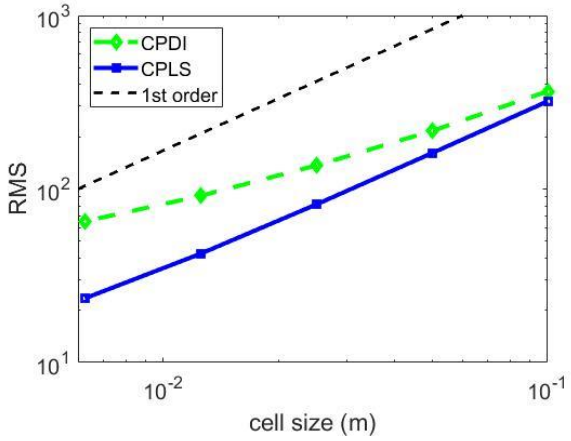


Figure 28 Stress Convergence rate at large deformation (A=0.1m)

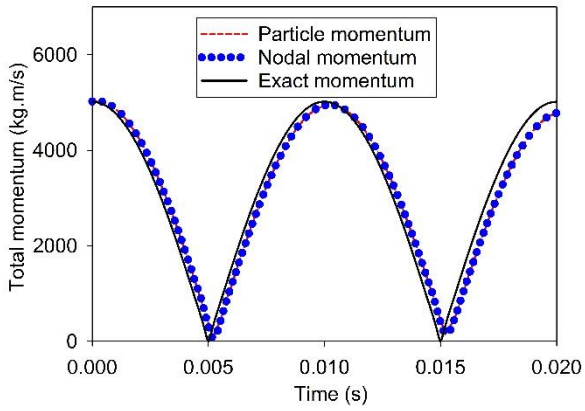


Figure 29 Total momentum of the CPDI for cell size 0.1m (A=0.1m)

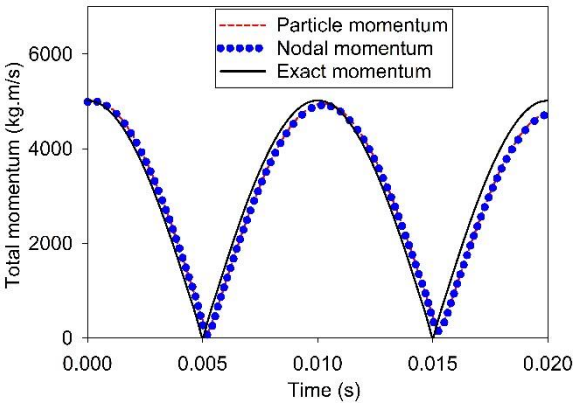


Figure 30 Total momentum of the CPLS for cell size 0.1m (A=0.1m)

4.4. Generalized vortex

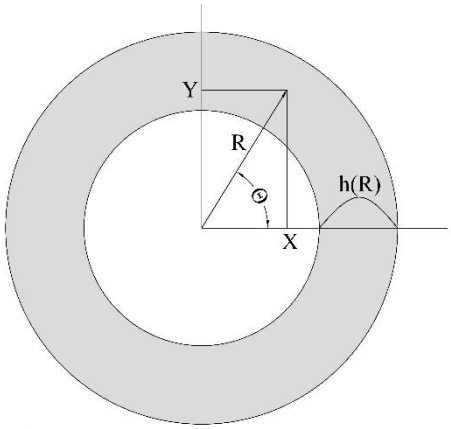


Figure 31 Polar coordinates for the ring

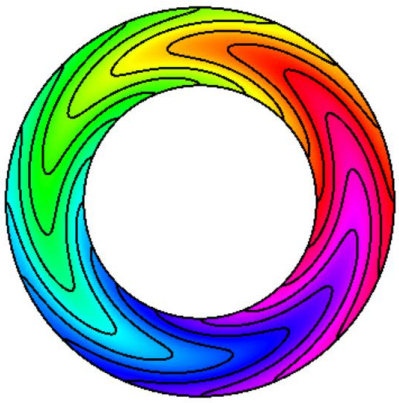


Figure 32 Illustration of the displacement field of the MMS vortex [34]

This example demonstrates the capability of the CPDI and the CPLS in extreme large distortion. Firstly, we consider a ring with an inner radius of $r_i = 0.75\text{m}$ and an outer radius of $r_o = 1.25\text{m}$. The ring is rotated with a maximum rotation angle A= 1(radian) at $r = \frac{r_i + r_o}{2} = 1\text{m}$ and zero rotation at the inner and outer boundary (see the illustration of the displacement field in Figure 32). The radius of each material points at the reference configuration is:

$$R = \sqrt{X^2 + Y^2} \tag{60}$$

The initial angle of each material point is:

$$\Theta = \arctan(X, Y) \tag{61}$$

We define a constant which is zero at the inner and outer boundary of the ring as follows:

$$h(R) = 1 - 32(R-1)^2 + 256(R-1)^4 \tag{62}$$

The induced rotation angle is:

$$\alpha = A \sin(c\pi t)\, h(R) \tag{63}$$

where A is the maximum rotation angle and c is the wave speed ($c = \sqrt{\frac{E}{\rho}}$). The angle of each material point at current configuration is:

$$\beta = \Theta + \alpha \tag{64}$$

The induced rotation angle corresponds to the displacement and velocity field as follows:

$$\mathbf{u}(X, Y, t) = \begin{bmatrix} A(X\cos\alpha - Y\sin\alpha) \\ A(X\sin\alpha + Y\cos\alpha) \end{bmatrix} \tag{65}$$

$$\mathbf{v}(X, Y, t) = \frac{d\mathbf{u}}{dt} = \begin{bmatrix} A^2\pi c h(R)\cos(\pi c t)(-X\sin\alpha - Y\cos\alpha) \\ A^2\pi c h(R)\cos(\pi c t)(X\cos\alpha - Y\sin\alpha) \end{bmatrix} \tag{66}$$

The Cauchy stress in the polar coordinate $\boldsymbol{\sigma}^*$ [40] is computed using the Method of Manufactured Solutions, leading to

$$\boldsymbol{\sigma}^* = \begin{bmatrix} \sigma_{rr} & \sigma_{r\beta} \\ \sigma_{\beta r} & \sigma_{\beta\beta} \end{bmatrix} \tag{67}$$

where.

$$\sigma_{rr} = 0$$

$$\sigma_{r\beta} = \sigma_{\beta r} = AR\mu\sin(\pi ct)(-64(-1-R) + 1024(-1+R)^3) \quad (68)$$

$$\sigma_{\beta\beta} = \mu(AR)^2\sin^2(\pi ct)(-64(-1-R) + 1024(-1+R)^3)^2$$

Then, the Cauchy stress in the Cartesian coordinate $\boldsymbol{\sigma}$ is converted from the Cauchy stress in the polar coordinate using a transformation tensor $\mathbf{q}$ to the current configuration by:

$$\boldsymbol{\sigma} = \mathbf{q}\boldsymbol{\sigma}^*\mathbf{q}^T \quad (69)$$

where the transformation tensor $\mathbf{q}$ to the current configuration is:

$$\mathbf{q} = \begin{bmatrix} \cos\beta & -\sin\beta \\ \sin\beta & \cos\beta \end{bmatrix} \quad (70)$$

Solving equation (69), the Cauchy stress components are:

$$\sigma_{xx} = \sin\beta\left(\sigma_{\beta\beta}\sin\beta - 2\sigma_{r\beta}\cos\beta\right)$$

$$\sigma_{xy} = \sigma_{yx} = \sigma_{r\beta}(\cos^2\beta - \sin^2\beta) - \sigma_{\beta\beta}\sin\beta\cos\beta \quad (71)$$

$$\sigma_{yy} = \cos\beta\left(2\sigma_{r\beta}\sin\beta + \sigma_{\beta\beta}\cos\beta\right)$$

The body forces required to obtain the displacement and velocity field above are:

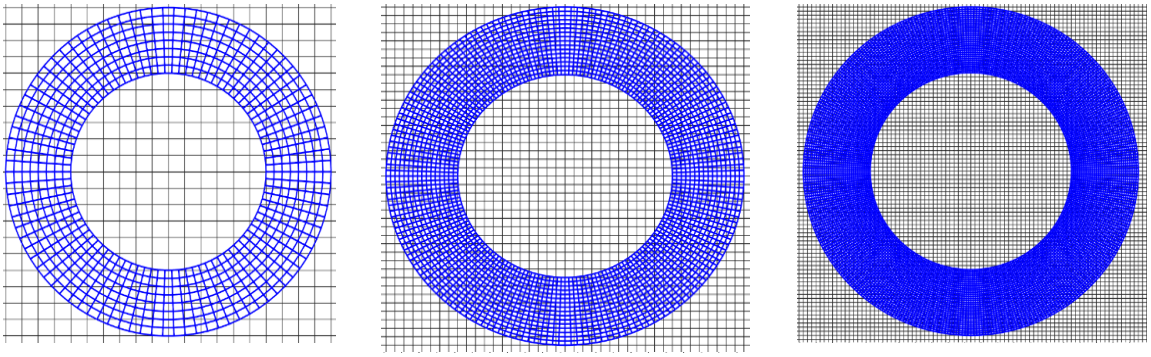
$$\mathbf{b}(X,Y,t) = \mathbf{q}\begin{bmatrix} b_r \\ b_\beta \end{bmatrix} = \begin{bmatrix} b_r\cos\beta - b_\beta\sin\beta \\ b_r\sin\beta + b_\beta\cos\beta \end{bmatrix} \quad (72)$$

where radial body forces and angle body forces in the polar coordinate are:

$$b_r = -RA^2(4R-5)^2(4R-3)^2$$

$$\left(-c^2\pi^2\cos^2(c\pi t)\,(4R-3)^2(4R-5)^2 + \frac{4096\mu}{\rho_o}(R-1)^2\sin^2(c\pi t)\right) \quad (73)$$

$$b_\beta = -RAc^2\pi^2\sin(c\pi t)\,(4R-3)^2(4R-5)^2 + \frac{64\mu(96R^3 - 240R^2 + 188R - 45)}{\rho_o} \quad (74)$$



Cell size 0.125m | Cell size (0.125/2)m | Cell size (0.125/4)m

Figure 33 Grid and material point domain discretization for the 2D generalized Vortex

Table 4 Numerical parameters for generalized vortex problem

Parameters	Symbol	Value
Young modulus	E	10e3 Pa
Poisson's ratio	ν	0.3
Shear modulus	μ	$3.85e^2$ Pa
Lame constant	λ	$5.77e^2$ Pa
Initial density	ρ_o	1000 kg/m^3
Final time	T	1s
Time step	dt	$0.1\frac{h}{c}$
Displacement amplitude	A	1 radian

In this example, the material point domains are distributed in the radial direction and the background grid is structured. Four different cell sizes including 0.125m, $(0.125/2)$m, $(0.125/4)$m and $(0.125/8)$m are used to calculate the spatial convergence rate. In a similar way to the previous example, the number of material points increase proportionally to the number of grid cells. There are $N_r = 1/l_e$ material points distributed equally in the radial direction (r = 0.5m) and $N_\Theta = 45/4l_e$ material points distributed equally in the angular coordinate (see Figure 33 for the discretization of material points domain and background grid). The numerical parameters for the generalized vortex problem are shown in Table 4. In this example, the MPM performs poorly as the ring is broken under excessive rotation (see Figure 34). Both the CPDI and the CPLS perform better as the ring keeps the similar shape with the

initial configuration at the final configuration (see Figure 35 and Figure 36). Figure 37 and Figure 38 show the norm of the displacement errors of material points for the cell size of 0.125m. Because of the symmetry. Only a quarter of the ring is plotted. While high displacement errors for the CPDI are obtained in the middle and the inner boundary of the ring, high displacement errors for the CPLS are obtained in the middle and the outer boundary of the ring. Similar velocity and stress error distribution are observed for both the CPDI and the CPLS with a lower magnitude for the CPLS (see Figure 39, Figure 40, Figure 41 and Figure 42). Figure 43, Figure 44 and Figure 45 show the spatial convergence rate of the displacement, the velocity and the stress respectively. Overall, the CPLS shows approximately 2^{nd} order convergence while the CPDI, at a very fine mesh, loses convergence. Unlike the CPDI, the CPLS interpolation does not ensure the linear momentum conservation with a small deviation between total momentum of material points and nodes (see Figure 46 and Figure 47), these errors also reduce significantly when refining mesh in a similar way to the previous examples.

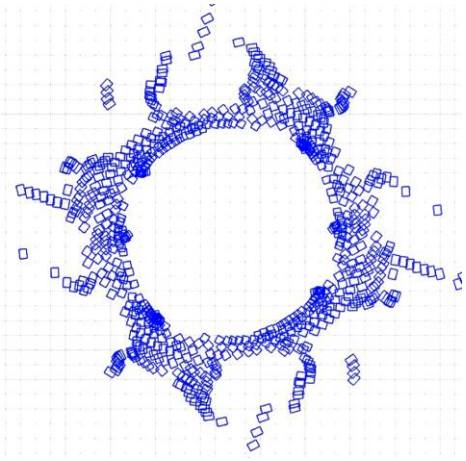


Figure 34 MPM solution for cell size 0.125m

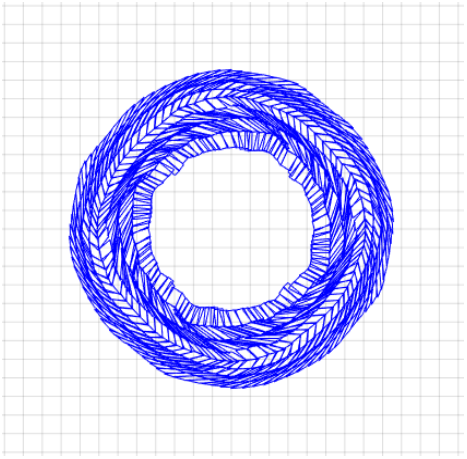


Figure 35 Final configuration of the CPDI material point domain for the cell size 0.125m

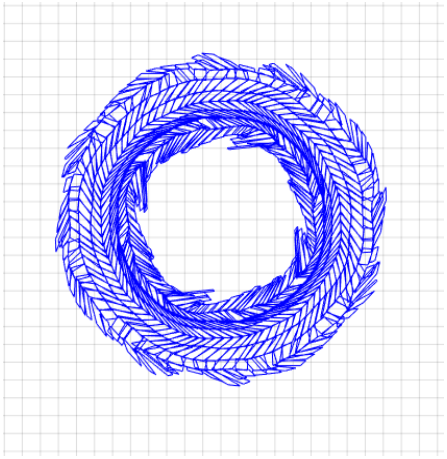


Figure 36 Final configuration of the CPLS material point domain for the cell size 0.125m

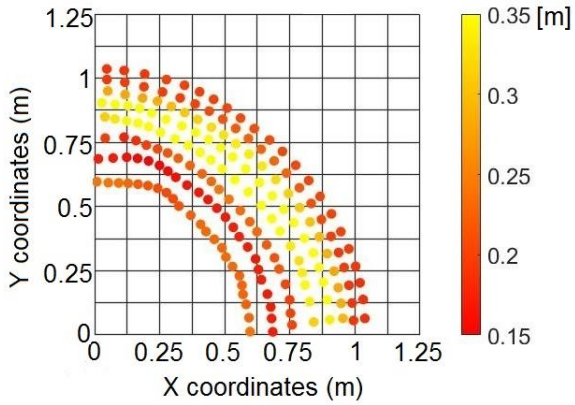


Figure 37 Displacement error of the CPDI material point for the cell size 0.125

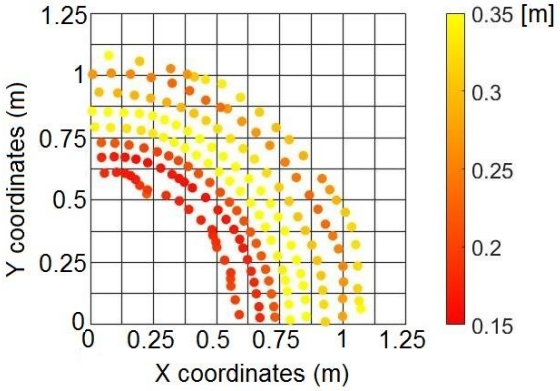


Figure 38 Displacement error of the CPLS material point for the cell size 0.125

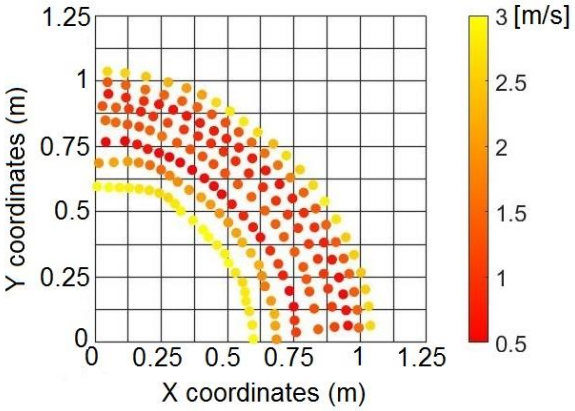


Figure 39 Velocity error of the CPDI material point for the cell size 0.125

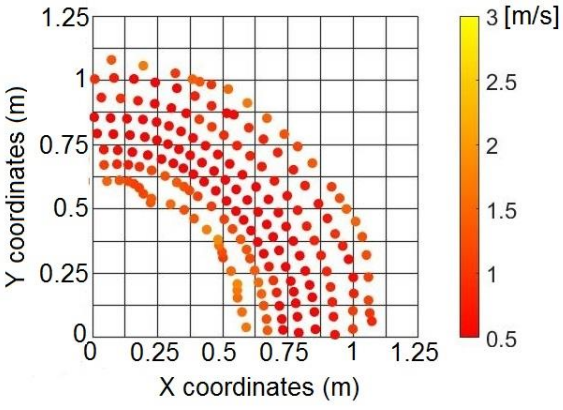


Figure 40 Velocity error of the CPLS material point domain for the cell size 0.125

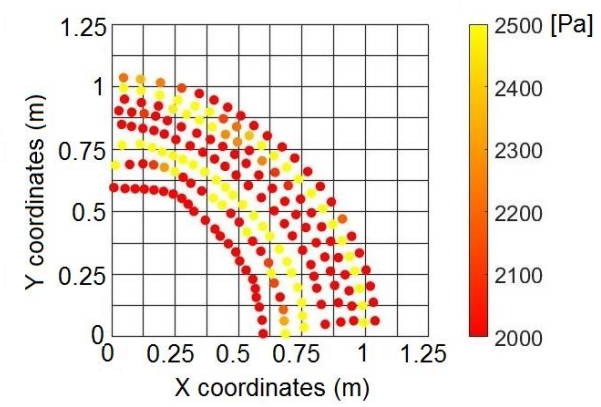


Figure 41 Stress error of the CPDI material point for the cell size 0.125

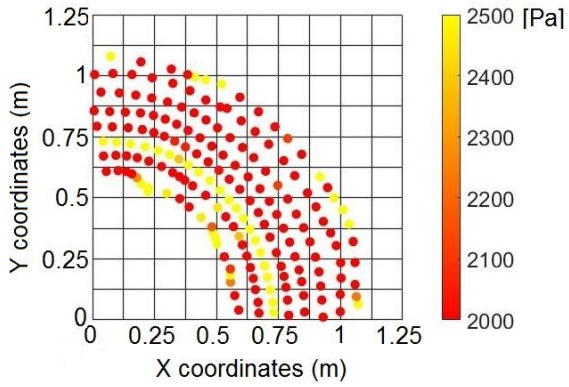


Figure 42 Stress error of the CPLS material point for the cell size 0.125

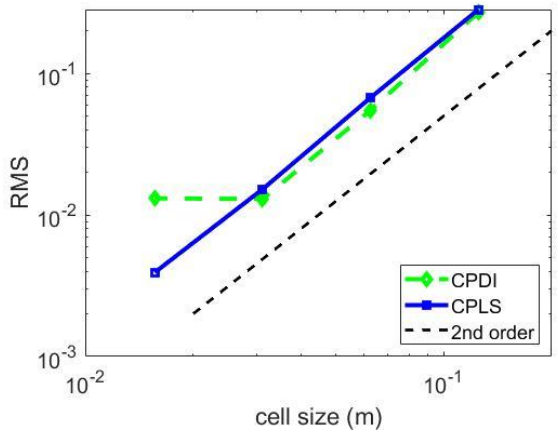


Figure 43 Convergence rate of the **displacement**

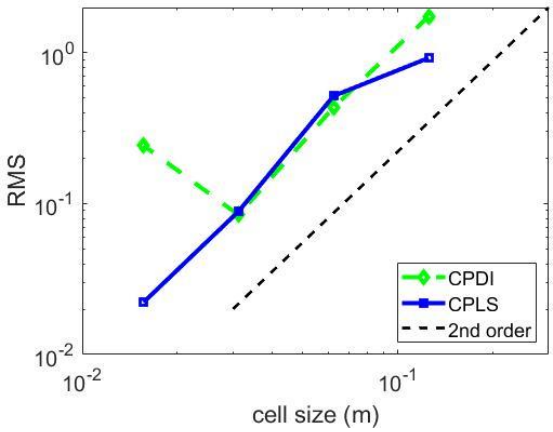


Figure 44 Convergence rate of the velocity

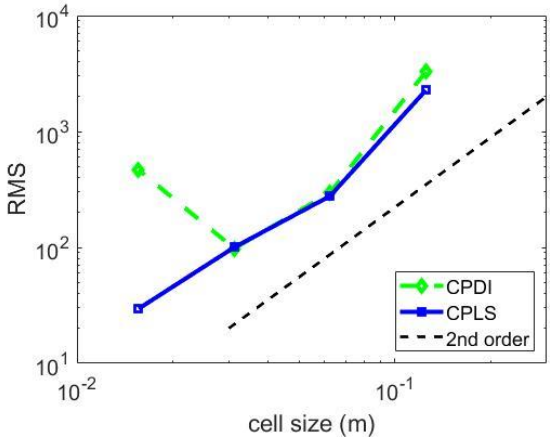


Figure 45 Convergence rate of the **stress**

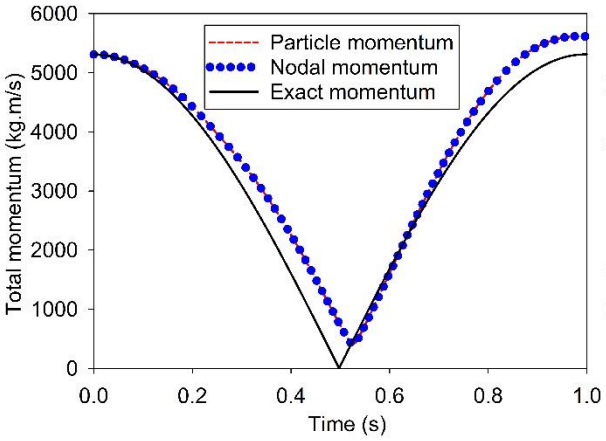


Figure 46 Total momentum of the CPDI for the cell size 0.0625m

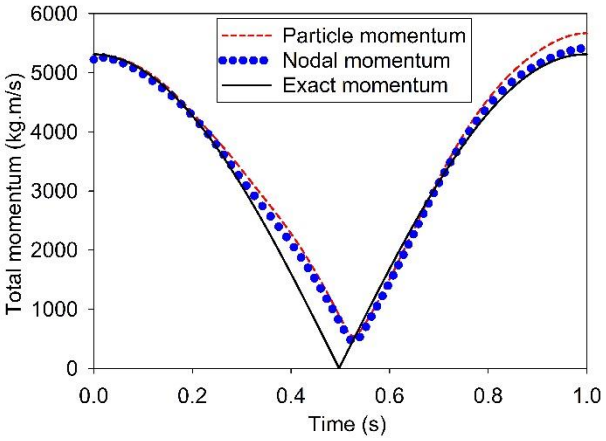


Figure 47 Total momentum of the CPLS for the cell size 0.0625m

4.5. Two elastic disks impact

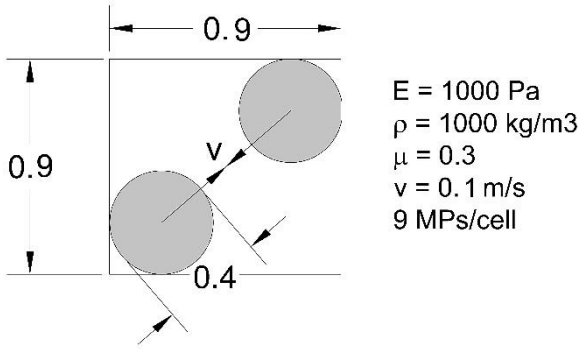


Figure 48 Colliding disks problem

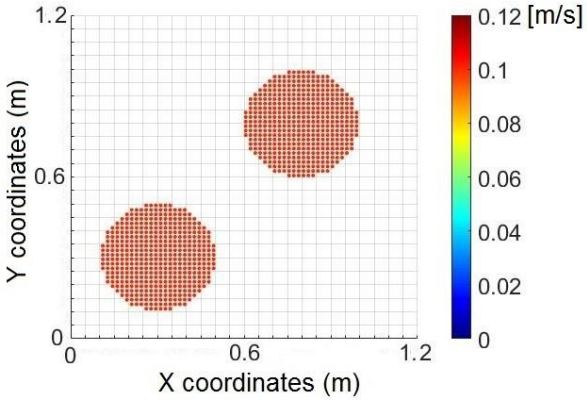


Figure 49 Initial condition (speed colorbar)

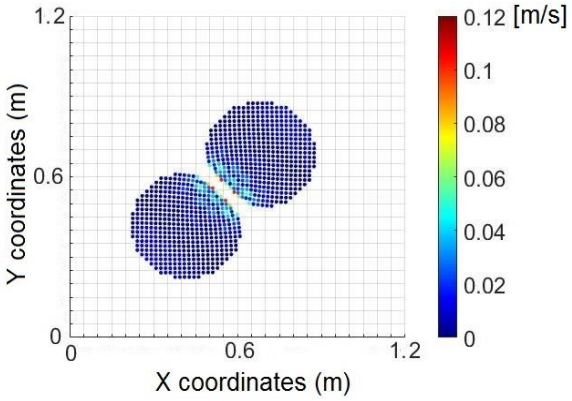


Figure 50 Colliding Impact (t=1.05s)

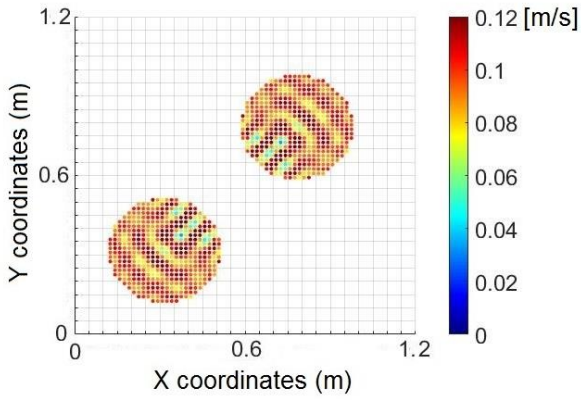


Figure 51 Releasing disks (t=2.5s)

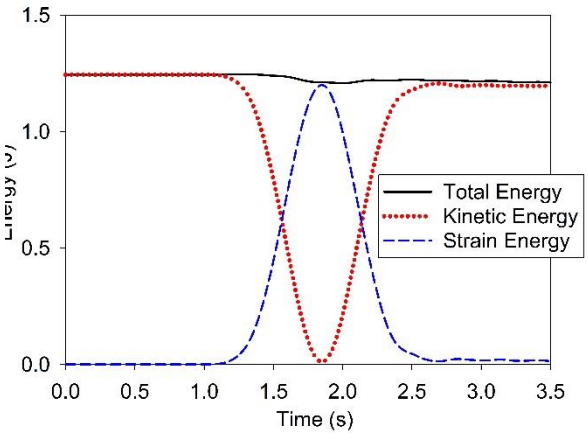


Figure 52 CPDI energy evolution

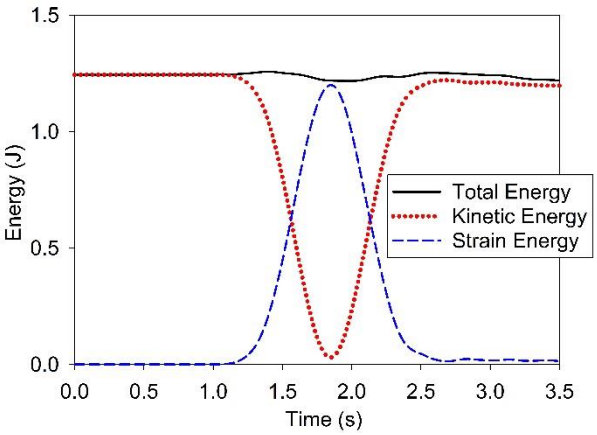


Figure 53 CPLS energy evolution

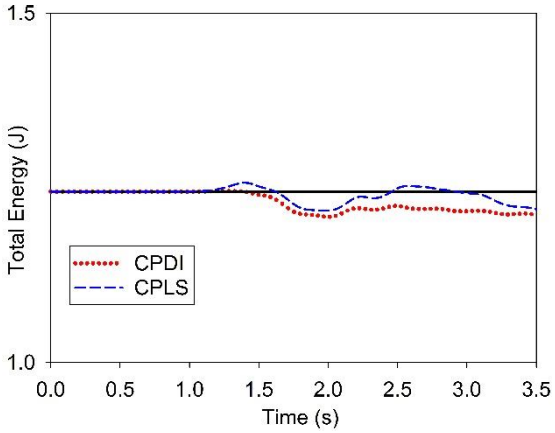


Figure 54 Total energy evolution

The colliding impact of two elastic disks is simulated to examine the no-slip contact with the new interpolation function, replicated the simulation by Sulsky et al. [1] with given geometry and parameters in Figure 48. The structured grid has a square cell of 0.05m and there is a total of 896 material points for the 2 disks. The impact velocity is 0.1 m/s. Figure 49, Figure 50 and Figure 51 show the evolution of the speed during the colliding. Energy plots are given for the CPDI in Figure 52 and for the CPLS in Figure 53. In both the CPDI and the CPLS, the kinetic energy decreases and recovers after the impact while the strain energy reaches a maximum during the impact and decreases close to zero after the impact. That results in a similar total energy evolution of both the CPDI and the CPLS (see Figure 54). This example demonstrates that there is no need for the additional treatment for the no-slip contact for the CPLS.

4.6. An elastic disk rebound with the wall

The next example is the impact of an elastic disk with the frictionless wall at the boundary of the background grid domain with given geometry and parameters in Figure 55. The numerical parameters are the same for the colliding impact of two disks, but the impact now is between the elastic disk and the wall which enforces the Dirichlet boundary of the zero velocity. Energy plots are given for the CPDI in Figure 56 and for the CPLS in Figure 57. Both the CPDI and the CPLS show a slight decrease of total energy when rebounding from the boundary of the background grid. Like previous example, the CPLS show a slightly less dissipation than the CPDI (see Figure 58). The energy loss indicates some dissipation due to the use of the lumped mass matrix [41]. In general, the least square interpolation does not require any additional boundary treatment in this example.

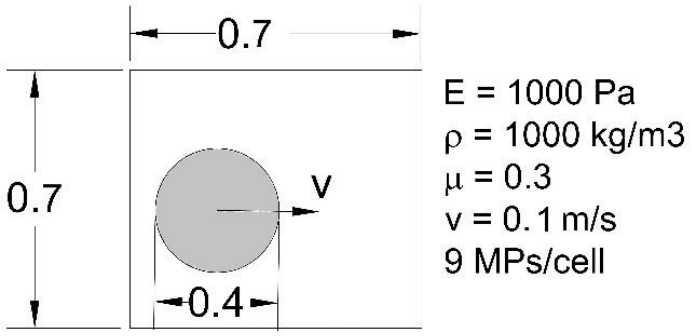


Figure 55 Wall rebound problem

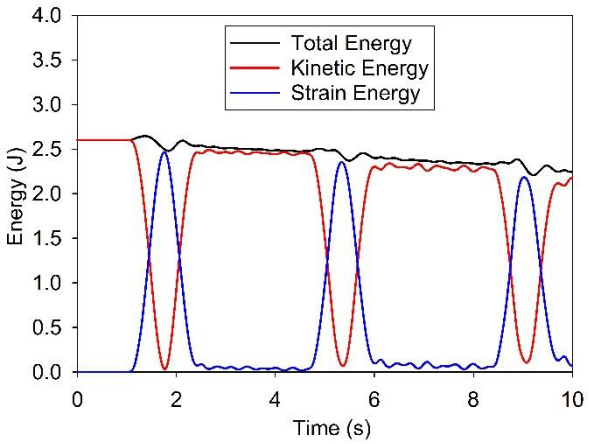


Figure 56 CPDI energy evolution

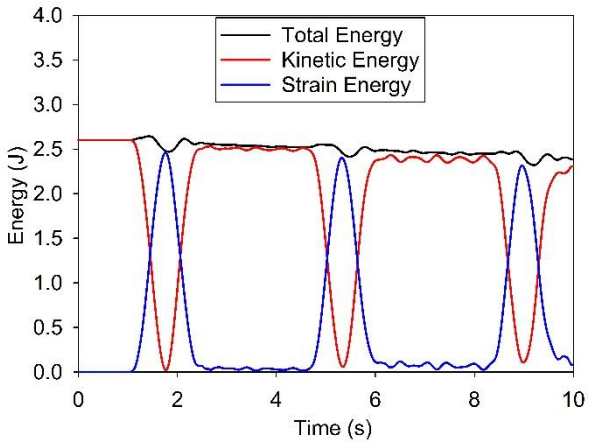


Figure 57 CPLS energy evolution

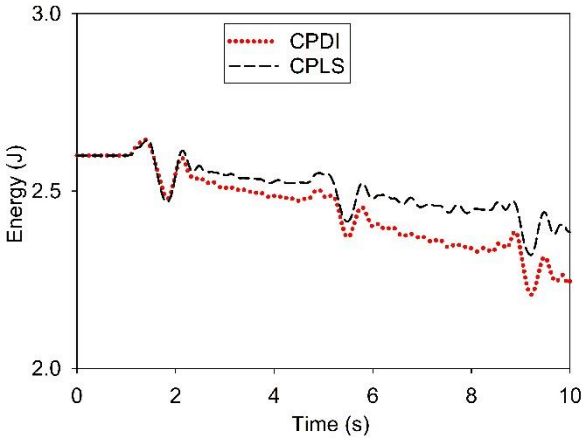


Figure 58 Total energy evolution

5. CONCLUSION

The CPLS, an improved version of the CPDI, has been proposed in this paper to obtain higher accuracy in the large deformation regime compared with the original CPDI and MPM. The accuracy enhancement is achieved by employing the Improved Moving Least Squares method, which improves the robustness compared with the conventional Moving Least Squares method. Furthermore, by using the velocity gradient, the velocity in the CPLS is considered as linear function rather than being locally constant within the material point domains. Three verification examples using the Method of Manufactured Solutions demonstrate the improved capabilities of the CPLS compared with the CPDI and the MPM. Furthermore, the CPLS improves the accuracy of the CPDI and maintains automatically the non-slip contact in a similar way to the CPDI. Further works are needed to examine whether the least square interpolation may affect the friction contact implementation and general boundary conditions. We also found that the Improved Moving Least Squares reconstruction does not always guarantee the linear momentum conservation in the mapping from the material points to the nodes although the conservative errors are minor in the given numerical examples. The work presented in this paper is not only limited to the CPDI with quadrilateral particle domain but also is able to be extended to other CPDI variants such as using the tetrahedron particle domain [42, 43].

6. ACKNOWLEDGEMENT

The work has been funded by the Academy of Finland under the project 'Progressive failure and post-failure modelling of slopes with Generalized Interpolation Material Point Method (GIMP)' under decision number 286628. The authors would like to thank Prof. Lu Wen Zhang from Shanghai Jiao Tong University for her support on the implementation of IMLS shape function. The authors also thank to Alireza Sadeghirad and Krishna Kamojjala for sharing their CPDI code and numerical benchmarks using the Method of Manufactured Solutions for the paper.

7. APPENDIX

This Appendix describes in detail how to calculate the set of polynomials in Figure 1-3. The input requires the material point coordinates x_p and the total number of material points in the interval [-1, 1] as follows:

Figure 1: Material point's coordinates: x_p = [-0.5 0 0.5]; N = 3

Figure 2: Material point's coordinates: x_p = [-0.99 -0.95 -0.9]; N = 3

Figure 3: Material point's coordinates: x_p = [-0.1 -0.5]; N = 2

Firstly, we define the set of polynomials in 1D: $\mathbf{q} = [1, x, x^2]$, Then, the domain is discretized into $N_i = 1000$ points $\{x\}_{i=1}^{N_i}$, the value of the orthogonal polynomials of these points, denoted as $p_1(x_i), p_2(x_i), p_3(x_i)$, can be calculated using equation (19) as:

for i = 1 **to** N_i

$$p_1(x_i) = 1; \tag{75}$$

$$p_2(x_i) = q_2(x_i) - \frac{\langle q_2, p_1 \rangle}{\langle p_1, p_1 \rangle} p_1(x_i) = x_i - \frac{\sum_{p=1}^{N} w(x_i - x_p) x_p}{\sum_{p=1}^{N} w(x_i - x_p)} \tag{76}$$

$$p_3(x_i) = q_3(x_i) - \frac{\langle q_3, p_1 \rangle}{\langle p_1, p_1 \rangle} p_1(x_i) - \frac{\langle q_3, p_2 \rangle}{\langle p_2, p_2 \rangle} p_2(x_i) \tag{77}$$

And

$$p_3(x_i) = {x_i}^2 - \frac{\sum_{p=1}^{N} w(x_i - x_p) x_p^2}{\sum_{p=1}^{N} w(x_i - x_p)} - \frac{\sum_{p=1}^{N} w(x_i - x_p) x_p^2 p_2(x_p)}{\sum_{p=1}^{N} w(x_i - x_p) p_2(x_p) p_2(x_p)} p_2(x_i) \quad (78)$$

where $w(x_i - x_p)$ is the cubic spline function in equation (10).

Then, the set of the orthogonal polynomials $\mathbf{p} = [p_1, p_2, p_3]$ are presented as the scatter plots of all N_i connected with straight lines.

8. References

[1] D. Sulsky, Z. Chen and H. L. Schreyer, "A particle method for history-dependent materials," *Computer Methods in Applied Mechanics and Engineering,* vol. 118, no. 1-2, pp. 179-196, 1994.

[2] S. Bardenhagen and E. Kober, "The generalized interpolation material point method," *Computer Modeling in Engineering & Science,* vol. 5, pp. 477-495, 2004.

[3] A. Sadeghirad, R. Brannon and J. Burghardt, "A convected particle domain interpolation technique to extend applicability of the material point method for problems involving massive deformations," *International Journal for numerical methods in Engineering,* vol. 86, pp. 1435-1456, 2011.

[4] A. Sadeghirad, R. M. Brannon and J. E. Guilkey, "Second-order convected particle domain interpolation (CPDI2) with enrichment for weak discontinuities at material interfaces," *International Journal for Numerical Methods in Engineering,* vol. 95, no. 11, pp. 928-952, 2013.

[5] C. Hammerquist and J. A. Nairn, "Modeling nanoindentation using the Material Point Method," *Journal of Materials Research,* vol. 33, no. 10, pp. 1369-1381, 2018.

[6] C. Gritton, J. Guilkey, J. Hooper, D. Bedrov, R. Kirby and M. Berzins, "Uisng the material point method to model chemical/mechanical coupling in the deformation of a silicon anode," *Modelling and simuation in Materials Science and Engineering,* vol. 25, no. 4, p. 034005, 2017.

[7] S. Sinaie, D. T. Ngo, V. P. Nguyen and T. Rabczuk, "Validation of the material point method for the simulation of thin-walled tubes under lateral compression," *Thin-Wall Structures,* vol. 130, pp. 32-46, 2018.

[8] M. A. Homel and E. B. Herbold, "Field-Gradient Partitioning for fracture and frictional contact in the Material Point Method," *International Journal for Numerical Methods in Engineering,* vol. 109, no. 7, pp. 1013-1044, 2016.

[9] C. C. Hammerquist and Y. Aimene, "Nested manifold MPM simulations for modeling fluid-driven fracture propagation," Cambridge, 2019.

[10] F. Hamad, "Formulation of the axisymmetric CPDI with application to pile dring in sand," *Computers and Geotechnics,* vol. 74, pp. 141-150, 2016.

[11] C. Moormann, S. Gowda and S. Giridharan, "Numerical Simulation of Open Ended Pile Installation in Saturated Sand," in *Energy Geotechnics. SEG 2018. Springer Series in Geomechanics and Geoengineering,* L. L. (. Ferrari A., Ed., Springer, 2018.

[12] S. Giridharan, D. Stolle and C. Moormann, "Modeling liquefaction using the Material Point Method: an evaluation using two constitutive models," Cambridge, UK, 2019.

[13] Y. Zheng, F. Gao, H. Zhang and M. Lu, "Improved convected particle domain interpolation method for coupled dynamic analysis of fully saturated porous media

involving large deformation," *Computer Methods in Applied Mechanics and Engineering,* vol. 257, pp. 150-163, 2013.
[14] P. Wallstedt and J. Guilkey, "Improved velocity projection for the Material Point Method," *Computer Modeling in Engineering and Sciences* , vol. 19, no. 3, 2007.
[15] P. C. Wallstedt and J. E. Guilkey, "A weighted least squares particle-in-cell method for solid mechanics," *International journal for numerical methods in engineering,* vol. 85, no. 13, pp. 1687-1704, 2009.
[16] E. Edwards and R. Bridson, "A high-order accurate particle-in-cell method," *International Journal for Numerical Methods in Engineering,* vol. 90, no. 9, pp. 1073-1088, 2012.
[17] D. Sulsky and M. Gong, "Improving the Material Point Method," in *Innovative Numerical Approaches for Multi-Field and Multi-Scale Problems. Lecture notes in Applied and Computational Mechanics,* vol. 81, 2016, pp. 217-240.
[18] E. Wobbes, M. Moller, V. Galavi and C. Vuik, "Conservative Taylor least squares reconstruction for the Material Point Methods," *International Journal of numerical methods in engineering,* vol. 117, no. 3, pp. 271-290, 2018.
[19] Q. A. Tran, M. Berzins and W. Solowski, "An improved moving least square for the Material Point Method," in *Proceedings of the 2nd International Conference on the Material Point Method for Modelling Soil-Water-Structure Interaction (MPM 2019),* Cambridge, United Kingdom, 2019.
[20] Q. A. Tran, E. Wobbes, W. Sołowski, T. Möller and C. Vuik, "Moving least squares reconstruction for B-spline Material Point Method," in *Proceedings of the 2nd International Conference on the Material Point Method for Modelling Soil-Water-Structure Interaction (MPM 2019),* Cambridge, United Kingdom, 2019.
[21] Y. Hu, Y. Fang, Z. Ge, Z. Qu, Y. Zhu, A. Pradhana and C. Jiang, "A moving least squares material point method with displacement discontinuity and two-way rigid body coupling," *ACM Transactions on Graphics,* vol. 37, no. 4, 2018.
[22] K. M. Liew, Y. Cheng and S. Kitipornchai, "Boundary element-free method and its application to two dimensional elasticity problems," *International journal for numerical methods in engineering,* vol. 65, pp. 1310-1332, 2006.
[23] L. Zhang and K. Liew, "An improved moving least-square Ritz method for two-dimensional elasticity problems," *Applied Mathematics and Computation,* vol. 246, pp. 268-282, 2014.
[24] L. Zhang, Y. Deng and K. Liew, "An improved element-free Galerkin method for numerical modelling of the biological population problems," *Engineering analysis with boundary elements,* vol. 40, pp. 181-188, 2014.
[25] M. Steffen, R. M. Kirby and M. Berzins, "Analysis and reduction of quadrature errors in the material point method (MPM)," *International Journal for Numerical Methods in Engineering,* vol. 76, pp. 922-948, 2008.
[26] J. Dolbow and T. Belytschko, "Numerical integration of the Galerkin weak form in meshfree methods," *Computational Mechanics,* vol. 23, pp. 219-230, 1999.
[27] F. H. Harlow, "The particle-in-cell method for numerical solution of problems in fluid dynamics," *Proceedings ofSymposia in Applied Mathematics,* vol. 15, no. 269, 1963.
[28] C. Fu, Q. Guo, T. Gast, C. Jiang and J. Teran, "A Polynomial Particle-in-Cell Method," *ACM Transactions on Graphics,* vol. 36, no. 6, p. Article No. 222, 2017.
[29] C. Jiang, C. Schroeder and J. Teran, "An angular momentum conserving affine-particle-in-cell method," *JOurnal of Computational Physics,* vol. 338, pp. 137-164, 2017.
[30] C. Jiang, C. Schroeder, A. Selle, J. Teran and A. Stomakhin, "The affine Particle-in-cell method," *ACM Transactions on Graphics,* vol. 34, no. 4, p. Article 51, 2015.
[31] S. G. Bardenhagen, "Energy Conservation Error in the Material Point Method for Solid Mechanics," *Journal of Computational Physics,* vol. 180, pp. 383-403, 2002.

[32] M. Steffen, P. C. Wallstedt, J. E. Guilkey, R. M. Kirby and M. Berzins, "Examination and analysis of implementation Choices within the Material Point Method (MPM)," *Computer Modeling in Engineering and Sciences,* vol. 31, no. 2, pp. 107-127, 2008.
[33] P. C. Wallstedt and J. E. Guilkey, "An evaluation of explicit time integration schemes for use with the generalized interpolation material point method," *Journal of Computational Physics,* vol. 227, pp. 9628-9642, 2008.
[34] K. Kamojjala, R. Brannon, A. Sadeghirad and J. Guilkey, "Verification tests in solid mechanics," *Engineering with Computers,* vol. 31, no. 2, pp. 193-213, 2013.
[35] C. Gritton, M. Berzins and R. Kirby, "Improving accuracy in particle methods using null spaces and filters," in *In: Onate E, Bischoff M, Owen DRJ, Wriggers P, Zohdi T (eds) Proceedings of the IV International Conference on Particle-based Methods,* Barcelona, Spain, 2015.
[36] C. Gritton and M. Berzins, "Improving accuracy in the MPM method using a null space filter," *Computational Particle Mechanics,* vol. 3, no. 12, pp. 1-12, 2016.
[37] Q. A. Tran and W. Solowski, "Temporal and null-space filter for the material point method," *International Journal for Numerial methods in Engineering,* vol. 120, no. 3, pp. 328-360, 2019.
[38] Q. A. Tran, W. Sołowski and E. Cummings, "Hydro-mechanical coupled dual domain material point method stabilized with a null-space filter," in *V International Conference on Particle-based Methods. Fundamentals and Applications,* 2017.
[39] Q. A. Tran and W. Solowski, "Large strain consolidation modelling using Dual Domain Material Point Method," in *The 15th International Conference of the International Association for Computer Methods and Advances in Geomechanics,* 19-23 October, 2017 Wuhan, China, September 2017.
[40] R. Brannon, "https://csmbrannon.net/2018/11/09/matlab-and-mms-source-files/," [Online].
[41] E. Love and D. L. Sulsky, "An unconditionally stable, energy-momentum consistent implementation of the material point method," *Computer methods in applied mechanics and engineering,* vol. 195, pp. 3903-3925, 2006.
[42] V. P. Nguyen, C. T. Nguyen, T. Rabczub and S. Natarajan, "On a family of convected particle domain interpolations in the material point method," *Finite Elements in Analysis and Design,* vol. 126, pp. 50-64, 2017.
[43] R. B. Leavy, J. E. Guilkey, B. R. Phung, A. D. Spear and R. M. Brannon, "A convected-particle tetrahedron interpolation technique in the material point method for the mesoscale modeling of ceramics," *Computational Mechanics,* pp. 1-21, 2019.