INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2011; **85**:1687–1704
Published online 28 September 2010 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/nme.3041

A weighted least squares particle-in-cell method for solid mechanics

P. C. Wallstedt*, † and J. E. Guilkey

Department of Mechanical Engineering, University of Utah, Salt Lake City, UT 84112, U.S.A.

SUMMARY

A novel meshfree method is proposed that incorporates features of the material point (MPM) and generalized interpolation material point (GIMP) methods and can be used within an existing MPM/GIMP implementation. Weighted least squares kernel functions are centered at stationary grid nodes and used to approximate field values and gradients. Integration is performed over cells of the background grid and material boundaries are approximated with an implicit surface. The proposed method avoids nearest-neighbor searches while significantly improving accuracy over MPM and GIMP. Implementation is discussed in detail and several example problems are solved, including one manufactured solution which allows measurement of dynamic, non-linear, large deformation performance. Advantages and disadvantages of the method are discussed. Copyright © 2010 John Wiley & Sons, Ltd.

Received 2 March 2009; Revised 16 May 2010; Accepted 7 August 2010

KEY WORDS: generalized interpolation material point method; meshfree; marching cubes; weighted least squares; implicit surface; MPM; GIMP; PIC

1. INTRODUCTION

For many decades the finite element method (FEM) has been trusted with predictive computations for a wide range of structures and systems; it is known to be robust and accurate. However, there are classes of problems that remain difficult to solve with FEM. For example, the simulation of extrusion and molding operations produces extremely large deformations of the mesh, whereas simulation of failure processes involves tracking of arbitrary and complex cracks and material interfaces. A common way of dealing with moving discontinuities in FEM has been to re-mesh the domain frequently, perhaps at every time step. This leads to greater computational effort and relies critically on automated meshing algorithms that may only be reliable for simple domains or for linear triangles and tetrahedra.

During the last two decades a plethora of methods has been developed to circumvent the limitations of FEM. The Smooth Particle Hydrodynamics (SPH) method was developed by Lucy [1], Monaghan [2], and coworkers to model astrophysics problems and later extended to solid mechanics by Libersky, Petschek, and Randles [3, 4]. Although it originally suffered from instability and lack of convergence (see the work of Swegle and Hicks [5] for example) SPH has since been refined and corrected by Johnson and Beissel [6], Dilts [7], and others. While these modifications improved the accuracy and stability of SPH, they also introduced some inconvenience, such as keeping track of additional stress points.

*Correspondence to: P. C. Wallstedt, Department of Mechanical Engineering, University of Utah, Salt Lake City, UT 84112, U.S.A.
†E-mail: philip.wallstedt@utah.edu

By reforming the diffuse element method (DEM) of Nayroles *et al.* [8] in terms of moving least squares (MLS), Belytschko, Lu, and Gu [9–11] introduced the element free Galerkin (EFG) method that achieved substantial improvements in accuracy. Several other families of MLS-based methods appeared soon after including the Reproducing Kernel Particle Method (RKPM) of Liu and Jun and coworkers [12–15] and the Meshless Local Petrov Galerkin (MLPG) method of Atluri and Zhu [16]. Excellent overviews of meshfree methods include the papers of Belytschko *et al.* [17] and Fries and Matthies [18], and the text of Liu [19]. MLS-based methods were seen to be instances of a more general partition of unity framework [20, 21] and convergence properties were proven for broad classes of MLS-based methods [22].

The MLS-based methods were shown to be useful for new classes of problems for which FEM was ill-suited; development and application of these methods are on-going. Some limitations exist, however, such as the application of essential boundary conditions [23] and the substantially larger computational cost relative to FEM. The cell-based integration schemes used by some MLS-based methods are particularly troublesome for dynamic problems [24].

Substantial progress in crack propagation modeling has been made through MLS enrichment of finite element methods, such as the XFEM of Moës *et al.* [25]. XFEM builds on the generality and reliability of FEM while freeing crack propagation models from having to re-mesh along crack boundaries. However, XFEM still appears to share the mesh entanglement limitation of FEM for modeling of extreme deformations.

The material point method (MPM) was developed by Sulsky *et al.* [26–28] as an extension to the FLIP (Fluid-Implicit Particle) method of Brackbill and Ruppel [29], which itself is an extension of the particle-in-cell (PIC) method of Harlow [30]. MPM is extraordinarily easy to implement and use for complicated domains, such as foams, geologic formations, or biological structures, and can be initialized in seconds from imaging data such as a CT scan.

A major correction and improvement to MPM was offered by Bardenhagen and Kober [31] and called the Generalized Interpolation Material Point (GIMP) method. GIMP retains the same generality as MPM, though at additional computational cost. For realistic problems GIMP approaches first-order accuracy [32] but does not enjoy the reproducibility of MLS-based methods.

MPM and GIMP have been studied and used by many investigators; a subset of these contributions includes: analysis and improvement of time integration properties by Bardenhagen [33], Sulsky *et al.* [34], Wallstedt and Guilkey [32], and Steffen *et al.* [35]; membranes and fluid–structure interaction by York *et al.* [36, 37]; implicit time integration by Guilkey and Weiss [38] and Sulsky and Kaul [39]; conservation properties and plasticity by Love and Sulsky [40, 41]; contact by Bardenhagen *et al.* [42]; cracks and fracture by Nairn [43]; tracking of particle extents by Ma *et al.* [44]; and enhanced velocity projection and verification via the method of manufactured solutions by Wallstedt and Guilkey [32, 45].

In this work a new method is proposed that fits within the GIMP framework: it is based on a cartesian grid and it does not search for nearest neighbors or require a finite element mesh. The new method is based on a Weighted Least Squares approximation of data surrounding each grid node and is referred to in the remainder of this paper as 'WLS'. The method gives up some of the generality and convenience of GIMP while making substantial gains in accuracy.

Some of the volume decomposition features of WLS are analogous to a method that was presented by Belytschko *et al.* [46] for creating structured finite element models from solid models by means of implicit surface definitions. While their techniques focused on finite element discretizations, the cell decomposition and implicit surface estimation also apply to the Cartesian background grid used in the present method.

This paper is organized as follows: Several components of the algorithm are first described such as the PIC framework, weight functions, least squares scheme, implicit surface definition, marching cubes polygonization, and off-object node correction. Then these individual components are combined in a detailed description of the discrete governing equations of the main algorithm. Finally, several example problems are defined and solved, followed by a discussion of the utility of the method.

DOI: 10.1002/nme

2. GOVERNING EQUATIONS

An Updated Lagrangian formulation allows straightforward modeling of history-dependant materials with complicated constitutive models. The equation of motion is

$$\sigma\nabla+\rho\mathbf{b}=\rho\mathbf{a} \tag{1}$$

where σ is the Cauchy Stress, ρ is density, $\mathbf{b}$ is acceleration due to body forces, and $\mathbf{a}$ is acceleration. Essential boundary conditions may be set on velocity and acceleration, resulting in prescribed values of displacement $\mathbf{u}=\mathbf{x}-\mathbf{X}$, where $\mathbf{X}$ is position in the reference configuration and $\mathbf{x}$ is position in the current configuration. The deformation gradient is defined as $\mathbf{F}=\partial\mathbf{x}/\partial\mathbf{X}$. Surface tractions are $\sigma\mathbf{n}=\mathbf{t}$, where $\mathbf{n}$ is normal to the object surface and $\mathbf{t}$ is the traction vector on the surface. The conservation of mass is

$$\rho_0=\rho J \tag{2}$$

where ρ_0 is density in the reference configuration and $J=\det(\mathbf{F})$ is the Jacobian.

A FEM implementation might update the position and velocity of nodes and compute $\mathbf{F}$ from $\mathbf{u}$: $\mathbf{F}=(\mathbf{I}-(\partial\mathbf{u}/\partial\mathbf{x}))^{-1}$. However, the current method (as well as GIMP) updates position, velocity, and deformation gradient. Only linear reproducibility is required, so the WLS basis defined later in the manuscript can be planar.

The stress can be a function of virtually any variable used in the simulation, but in this work it is only a function of the deformation gradient $\mathbf{F}$. The rates of position, velocity, and deformation gradient are $\dot{\mathbf{x}}=\mathbf{v}$, $\dot{\mathbf{v}}=\mathbf{a}$, and $\dot{\mathbf{F}}=(\nabla\mathbf{v})\mathbf{F}$, respectively.

The weak form of the governing equations is obtained by multiplying Equation (1) by an admissible test function $\mathbf{w}$ and integrating over the current configuration Ω with boundary Γ. Integrating by parts over the highest order term, and assuming that $\mathbf{w}$ vanishes wherever essential boundary conditions are applied results in

$$\int_\Omega \rho\mathbf{w}\cdot\mathbf{a}\,\mathrm{d}\Omega=-\int_\Omega \sigma:\nabla\mathbf{w}\,\mathrm{d}\Omega+\int_\Gamma \mathbf{w}\cdot\tau\,\mathrm{d}\Gamma+\int_\Omega \rho\mathbf{w}\cdot\mathbf{b}\,\mathrm{d}\Omega \tag{3}$$

The discretization of the governing equations is developed in subsequent sections, culminating in Equation (25).

3. DESCRIPTION OF THE METHOD

FEM relies on a connected mesh of nodes whereas meshfree methods typically rely on a cloud of disconnected nodes (along with a background integration grid and octree search structure). However, the WLS method of this work is embedded within a PIC framework which is also used for the MPM and GIMP methods. In this way WLS represents a hybrid of PIC and meshfree schemes.

Values of position, velocity, deformation gradient, and other variables are assigned to disconnected particles that are suitably spaced throughout an object such that sufficient information is present. A stationary cartesian grid facilitates a bucket-sorting scheme so that particle values are collected, via weighted least squares, to the regularly spaced nodes of the grid to create fields of acceleration and velocity throughout the domain. Gradients of stress and velocity on the grid are used to update particle position, velocity, and deformation gradient.

The stationarity of the local least squares equations makes the 'weighted' designation in WLS appropriate, rather than 'moving' least squares in which values are collected to mobile, disconnected nodes based on nearest neighbors.

The subdomains over which numerical integration is performed differ in this method from FEM and from GIMP, as illustrated in Figure 1.

FEM always forms a contiguous non-overlapping partition of an object into subdomains, which provides accurate and reliable integration, albeit at the cost of complicated mesh generation. But

DOI: 10.1002/nme

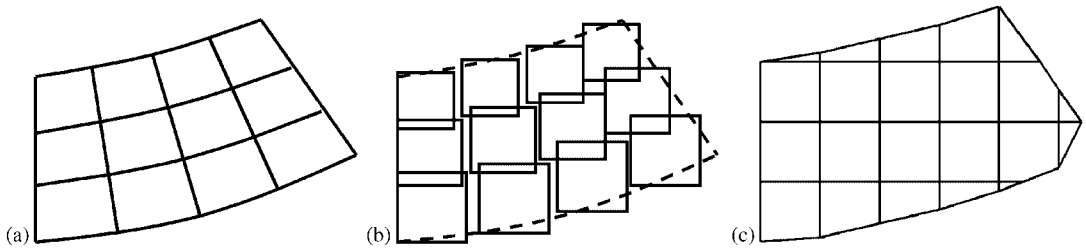

Figure 1. Comparison of volume partition strategies: (a) finite elements; (b) GIMP particles; and (c) WLS cell volumes.

the GIMP method only forms a contiguous non-overlapping partition of the material for the initial position; during problem evolution, gaps or overlaps may develop in the partition. Despite this shortcoming, GIMP's great advantage is the easy initialization of complicated domains.

The WLS method of this paper forms integration domains that remain contiguous and non-overlapping by using the cell structure of a PIC method. When a material boundary cuts through the middle of a cell, the cell is split into two and only a portion of the cell is allowed to contribute to the integration. The new cell integration method represents an improvement of accuracy compared to GIMP, while retaining the existing PIC framework, data structures, and initialization procedures. However, it involves the extra complication of marching cubes polygonization and it tends to ignore sharp corners, in a manner similar to GIMP.

The WLS method makes use of several mathematical components from the computer graphics and meshfree communities; detailed explanations of these are given in the following sections.

3.1. Weight functions

Two different shape or weight functions are used within WLS. The shape functions are centered on nodes of the background grid and are defined in terms of the cell size h. A wide second degree spline is defined in Equation (4) with 27 node support (in 3D) that allows nodes to 'see' particles that are farther away. A narrower piecewise linear function and its gradient are defined in Equations (5) and (6) with 8 node support that are used to interpolate grid data back to particles. This 'see wide, apply narrow' concept ensures that only nodes with sufficient data are allowed to influence particle updates. All weights and shapes are expressed in 1D as functions of $r = x - x_i$ and are implemented such that the first available choice in the list is always used; note that $\text{sign}(r) = (-1 \text{ if } r < 0,\ 1 \text{ otherwise})$.

$$W_i(r) = \begin{cases} 0.75 - \dfrac{r^2}{h^2} & |r| < \dfrac{h}{2} \\ \dfrac{(1.5h - |r|)^2}{2h^2} & |r| < \dfrac{3h}{2} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

$$S_i(r) = \begin{cases} 1 - \dfrac{|r|}{h} & |r| < h \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

$$G_i(r) = \begin{cases} -\dfrac{\text{sign}(r)}{h} & |r| < h \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Tensor products of these are used as shapes and weights in higher dimensions and the shapes and weights become functions of vectors.

DOI: 10.1002/nme

A shorthand is frequently used for these functions involving indices of particles or quadrature points. For example, if particles are indexed with p, then $W_{ip}=W_i(\mathbf{x}_p-\mathbf{x}_i)$, $S_{ip}=S_i(\mathbf{x}_p-\mathbf{x}_i)$, and $G_{ip}=G_i(\mathbf{x}_p-\mathbf{x}_i)$.

3.2. Weighted least squares framework

Although many basis functions can be chosen within a weighted least squares framework, the hyperplane is used throughout this method. The particles that fall within the non-zero region of a weight function centered at each node contribute to the equation at the node:

$$f_i^{\mathrm{LS}}(\mathbf{r})=c_0+c_1x+c_2y+c_3z \tag{7}$$

where x, y, and z are components of $\mathbf{r}=\mathbf{x}_p-\mathbf{x}_i$. The error norm that defines the WLS formulation is

$$L_2=\sum_p W_{ip}(f_i^{\mathrm{LS}}(\mathbf{r}_p)-f_p)^2 \tag{8}$$

Differentiating L_2 with respect to the coefficients $\boldsymbol{c}$ and setting each equation equal to zero results in the following system:

$$\begin{bmatrix} \sum_p W & \sum_p Wx & \sum_p Wy & \sum_p Wz \\ \sum_p Wx & \sum_p Wx^2 & \sum_p Wxy & \sum_p Wxz \\ \sum_p Wy & \sum_p Wxy & \sum_p Wy^2 & \sum_p Wyz \\ \sum_p Wz & \sum_p Wxz & \sum_p Wyz & \sum_p Wz^2 \end{bmatrix} \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} \sum_p Wf_p \\ \sum_p Wf_px \\ \sum_p Wf_py \\ \sum_p Wf_pz \end{pmatrix} \tag{9}$$

The system is rewritten as $\boldsymbol{Mc}=\boldsymbol{l}$ and $\boldsymbol{M}$, $\boldsymbol{c}$, and $\boldsymbol{l}$ are termed as the moment, coefficient, and load vectors/matrices, respectively. Each local function is found by the following, which is simply a different form of Equation (7).

$$f_i^{\mathrm{LS}}(\mathbf{r})=\boldsymbol{M}^{-1}\boldsymbol{l}\cdot(1 \;\; x \;\; y \;\; z)^{\mathrm{T}} \tag{10}$$

A shorthand is defined for the value and gradient of $f_i^{\mathrm{LS}}(\mathbf{r})$, at the location of the node:

$$\underset{o}{f_i} = f_i^{\mathrm{LS}}(\mathbf{r}=0)=c_0 \tag{11}$$

$$\nabla \underset{o}{f_i} = \nabla f_i^{\mathrm{LS}}(\mathbf{r}=0)=(c_1 \;\; c_2 \;\; c_3)^{\mathrm{T}} \tag{12}$$

3.3. Implicit surface definition

In FEM, surfaces are modeled with the edges or faces of elements, and are accurate insofar as the element can match the curve of the object. In GIMP, surfaces are implied but never defined. This means that GIMP does not resolve sharp convex or concave features; however, it also avoids many of the complications of FEM, such as meshing and special contact algorithms.

The treatment of material surfaces in WLS is one of its most critical components. The method uses flagged points that are placed on material surfaces and given special treatment in the algorithm. The surface particles add an extra component of difficulty to the initialization of the method, as compared to GIMP, but the difficulty is not severe. For geometrically simple objects, such as pressure vessels and beams, the surface of the object is clearly defined. And for complicated domains arising from three-dimensional scan data, surface particles can be located by placing them halfway between data samples that are 'in' and 'out'.

A cell subdivision method is developed based on the marching cubes polygonization algorithm of Lorensen and Cline [47]. For a 3D domain, the marching cubes algorithm requires a cartesian

DOI: 10.1002/nme

grid of points as input. The value stored at each point may be a simple binary flag indicating whether the point is inside or outside the object. However, a more sophisticated version uses the value on each point to represent a signed distance from the surface of the object, indicating how far the point is located away from the object's surface.

For WLS, a cartesian grid is already used in parts of the algorithm. A means of imposing a signed distance on each grid node is developed that makes use of existing least squares machinery. A flag β_p is defined on all particles which is unity for surface particles and is larger for interior particles; a value of 2 is used. The flag values are summed to nearby grid nodes

$$\beta_i = \sum_p S_{ip} \beta_p \tag{13}$$

Based on the values of β_i a signed distance d_i for every node in the grid is estimated as

$$d_i = \begin{cases} h & \beta_i > 2 \\ -h & \text{otherwise} \end{cases} \tag{14}$$

This first estimate is rough but it serves to initialize all nodes as in or out.

Using the existing shape functions, a local surface location can be defined as the weighted average of nearby surface particles. The formula shown below forms sums over *surface particles only* to find an average distance to the surface for each node that has one or more surface particles within the non-zero region of its weight function.

$$\mathbf{x}_i^{\text{surf}} = \frac{\sum_p S_{ip} \mathbf{x}_p^{\text{surf}}}{\sum_p S_{ip}} \tag{15}$$

Note that $\mathbf{x}_i^{\text{surf}}$ is only defined on nodes near the surface, and for each such node the following steps are performed:

1. Define surface normal $\mathbf{n}$ at surface position $\mathbf{x}_i^{\text{surf}}$ as the gradient of the particle flags:

$$\mathbf{n} = \sum_i G_i(\mathbf{x}_i^{\text{surf}}) \beta_i \tag{16}$$

2. Signed distance is the projection of distance vector onto surface normal:

$$d_i = -\text{sign}((\mathbf{x}_i^{\text{surf}} - \mathbf{x}_i) \cdot \mathbf{n}) ||\mathbf{x}_i^{\text{surf}} - \mathbf{x}_i|| \tag{17}$$

Thus the creation of signed distances for all nodes takes place in two stages. In the first stage all signed distances are estimated with Equation (14), and in the second stage those nodes that fall near the surface are given more precise signed distances via Equations (15)–(17).

3.4. Subdivision of boundary cells via marching cubes

The integration domain for an object in WLS consists of the sum of the volumes of each full or partially full cell that is occupied by the object. Cells within the object are completely filled, cells on the boundary are partially filled, and cells away from the object are empty.

In order to subdivide each cell's volume in an efficient manner the Marching Cubes algorithm of Lorensen and Cline [47] is used. The 2D variation of this algorithm is called Marching Squares. For a 2D cell each of its four nodes may be flagged as 'in' or 'out' based on whether the node is located within the object or outside of it. Thus there exist $2^4 = 16$ combinations of node flags, which may be reduced, with symmetric reflections and rotations, to four unique cases involving triangles, quadrilaterals, or pentagons; see Figure 2. In 3D there exist $2^8 = 256$ combinations of node flags which reduce to 15 unique cases. This exhaustive listing of possible cases of cell subdivision makes the Marching Cubes algorithm speedy and robust.

While it is easy to perform integration over cells that are empty or full, the complex polyhedra generated by Marching Cubes may have variations of topology that, in general, cannot be expressed in terms of a single hexahedral or tetrahedral finite element. Although it may be theoretically

DOI: 10.1002/nme

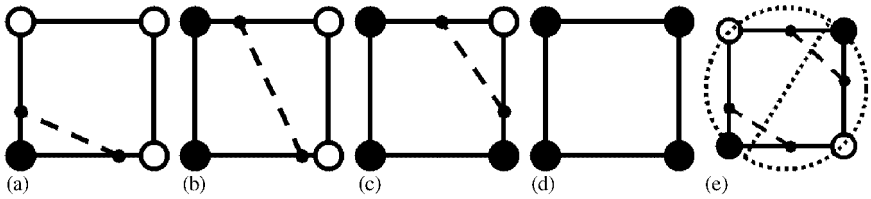

Figure 2. Subdivided cell regions: (a) triangle; (b) quad; (c) pentagon; (d) square; and (e) not allowed.

possible to compute the correct weights and locations of Gauss points for a 2D pentagon, for example, such is not generally done. And the task becomes unmanageable if Gauss points are to be computed for 15 different polyhedra in 3D.

In lieu of high-order Gauss integration, two methods of approximate integration are discussed, and one of these is ultimately chosen for implementation in WLS.

3.4.1. Multi-point integration over partially filled cells. The cells of the background grid may be integrated with increasing precision by filling them with a requisite number of Gauss points. For an interior cell with sufficient number of Gauss points the integration of a polynomial may be exact. But if such a cell is subdivided into two regions, only one of which is occupied by the object, then the integration cannot be exact. However, such a scheme may be considered an approximation.

For example, consider a 2D cell which is populated with 16 Gauss points, and assume that 11 points are within the object and 5 are outside of it. The approximation of the integral over the cell is then found as the sum of all the weight-value products of the Gauss points, where the value of the 'outside' Gauss points is taken to be zero. Another closely related approximation is to space 16 sample points evenly throughout the cell, rather than at the Gauss locations, and set the 'outside' points to zero as above. The weight of each 'inner' point is simply 1/16th of the cell volume. Such approximation schemes obviously lose their high order.

The multi-point approximate integration scheme is used by Belytschko *et al.* for the EFG method. In their overview of meshfree methods Belytschko *et al.* state that the background grid 'strikes many researchers as unacceptably crude, for within a cell, quadrature is performed over any discontinities and boundaries which do not coincide with the boundaries of the cells. Our experience so far suggests that the effects are quite minimal [17]'.

However, in a broad review of meshfree methods Fries and Matthies [18] are more critical, claiming that 'the integration error which arises from the misalignment of the supports and the integration domains is often higher than the one which arises from the rational character of the shape functions'.

3.4.2. Low-order integration over cell subdomains. In this approximate integration scheme the volume of integration is considered to be more important than the order of integration. The volume of a cell is subdivided into inner and outer portions and a single sample point is located at the centroid of the inner region.

In a FEM implementation the deformation gradient and stress are computed at Gauss points based on information interpolated from the nodes of the element. However, in WLS the stress at Gauss points is found in three stages. First, stress is computed on each particle; second, functions of average stress in the neighborhood of each node are created at the node (see Sections 4.1 and 4.3); and third, the stress at Gauss points is interpolated from the stress functions at the cell nodes (see Equations (21) and (24)). This smoothing and averaging process causes information from several nearby cells to be included at each Gauss point, rather than information from only the enclosing cell.

One advantage of this approximation is that the centroid is unique for the region so that ambiguous configurations are disallowed. The rectangularity and stationarity of the background cells are assured by construction. The uniqueness of the integration point, the stationarity of cell boundaries, and the broad and smooth information used at Gauss points, together suggest that zero

DOI: 10.1002/nme

energy modes, such as those occurring with single Gauss point integration of quadrilateral finite elements, are unlikely to occur. Generally speaking, the mathematical analogies between FEM, GIMP, and WLS remain tentative due to the major differences between the methods. Each family of methods must be independently analyzed for limitations and troublesome modes.

Another advantage of the centroidal approximation is that the value of the integral varies smoothly as the object boundary passes through the cell. In the multi-point approximation, Gauss points may abruptly 'turn on' or 'turn off' as the object surface passes through them. The single point integration method of this section is used throughout WLS.

3.4.3. Details of cell subdivision. The signed distance information developed in Section 3.3 is used to subdivide cell edge segments and enclose a shape within them. If any two nodes are on the same edge of a cell they are termed 'adjacent'. And if two nodes are adjacent, yet one of them is in, and the other is out, then it is clear that the surface of the object passes somewhere between them. Furthermore, the signed distance information is used to approximate the location on the segment at which the surface crosses. In the following equation two special node indices are denoted with capital letters: I is the index of the 'in' node and O is the index of the 'out' node. For any two adjacent nodes with one 'in' and one 'out', the following expression, involving the signed distances of Equation (17), gives global coordinates for the location at which the material boundary intersects the cell segment:

$$\mathbf{x}_{\text{cut}} = \mathbf{x}_I - d_I \frac{\mathbf{x}_I - \mathbf{x}_O}{d_I - d_O} \tag{18}$$

The cell boundary cut positions $\mathbf{x}_{\text{cut}}$ are used, together with the 'in' node or nodes of the cell to form the shape indicated by the Marching Cubes scheme. For example, if one node of a cell is in and the other three nodes are out, then two cut points are created between the 'in' node and its two adjacent nodes. The two cut points plus the 'in' node form a triangle, whose volume and centroid are used for integration.

This integration scheme eliminates the gaps and overlaps of GIMP integration but incurs the extra inconvenience of initializing surface particles and building a surface. Each WLS cell that is near a surface is divided into filled and empty portions. This results in a contiguous, non-overlapping description of the object, although corners tend to be rounded; see Figure 1.

3.5. Surface conditioning

While the weighted least squares scheme provides excellent results for values and gradients at nodes with sufficient numbers of particles, there are nodes near object boundaries that may end up with spurious values due to ill conditioning of the moment matrix. Any node whose values cause inaccuracy or instability due to ill-conditioning of its moment matrix is considered 'spurious'. A reliable way of detecting spurious nodes is described in this section, followed by two remedial procedures.

Some cases of ill-conditioning are easily detected, such as an insufficient number of particles to form the planar basis (three particles are required in 2D; four in 3D) or a matrix determinant that is within machine precision of zero. But other spurious nodes frequently arise that cannot be detected by these means. A more robust and general method of predicting ill-conditioned moment matrices is as follows.

For weight function $W_i(r)$ of Equation (4) a particle has greater weight as it gets closer to the node; therefore the weight function for each contributing particle is a good approximation for a dimensionless radius from the particle to the node; see Figure 3(a) and note how the weight contours are nearly circular. If all sample particles fall outside a chosen weight, then remedial procedures are performed. Experience suggests that a trigger weight `TW` in the range $0.1 < \texttt{TW} < 0.25$ will indicate an appropriate amount of extra surface conditioning; the `TW` used throughout this paper is 0.12 and an example of predicted spurious nodes is shown in Figure 3(b).

For every spurious node the dimension of the moment matrix is reduced. This is accomplished by replacing Equation (7) with $f_i^{\text{LS}}(\mathbf{r}) = c_0$. Then the inverse of the moment matrix is formed by setting all values to zero, except for the first row and column: $M_{00}^{-1} = 1/M_{00}$.

DOI: 10.1002/nme

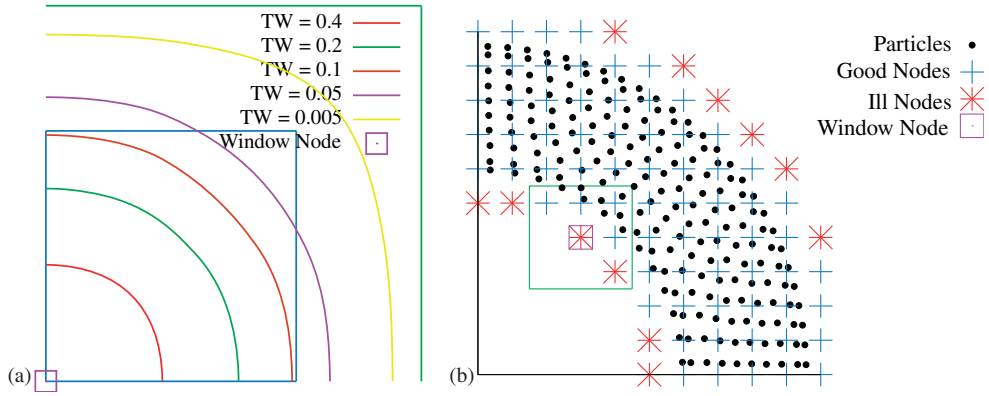

Figure 3. Trigger weight for spurious nodes: (a) trigger weight and (b) spurious nodes. (a) is a magnified view of the region near the window node of (b).

For the computation of internal forces, experience shows that moment matrix dimension reduction is insufficient to remedy the effects of spurious nodes, and a more drastic step must be taken. In these cases the weighted least squares coefficients for internal force at a node are discarded, and values at the node are found from an average of 26 nodes that surround it in 3D, or 8 nodes that surround it in 2D. This procedure eliminates the influence of spurious nodes on the gradient of stress.

4. ALGORITHM SEQUENCE AND DISCRETE EQUATIONS

In this section each stage of the WLS time step is described and the least squares, marching cubes, and matrix conditioning techniques of previous sections are combined to form the algorithm.

4.1. Current particle density, body force, and stress

From the current particle values of position $\mathbf{x}_p$, acceleration due to body forces $\mathbf{b}_p$, and deformation gradient $\mathbf{F}_p$, the Jacobian $J_p = \det(\mathbf{F}_p)$, density $\rho_p = \rho_0 / J_p$, and body force density $\mathbf{f}_p^{\text{ext}} = \rho_p \mathbf{b}_p$ are computed.

Stress is computed on each particle from a constitutive equation $\sigma_p = \sigma(\mathbf{F}_p)$, which may be very general due to the Lagrangian formulation.

4.2. Least squares moment matrix and object surface

A least squares moment matrix is formed at each node to be used for subsequent calculations. Particles that fall within the non-zero region of the weight function of each node contribute to its matrix; see Section 3.2.

Special flagged surface particles are used via the procedure of Section 3.3 to determine how far 'in' or 'out' of the object each node is located.

4.3. Node values via WLS

The least squares values $\overset{o}{\mathbf{v}}_i$, $\overset{o}{\rho}_i$, $\overset{o}{\mathbf{f}}{}_i^{\text{ext}}$, and $\overset{o}{\sigma}_i$ are computed on the nodes via the weighted least squares framework of Equation (10), keeping in mind the shorthand of Equations (11) and (12).

4.4. Integrate over full and partial cell volumes

Let q be an index over the quadrature points (one for each non-zero volume cell) and i an index over nodes. Let $\texttt{maxW} = max(W_i)$ be the maximum weight of any particle within the support of

DOI: 10.1002/nme

node i (see Equation (4)). Then the values of density, body force density, and stress are found at each quadrature point by

$$\rho_q = \frac{\sum_i \alpha_i S_{iq} \overset{o}{\rho}_i}{\sum_i \alpha_i S_{iq}} \tag{19}$$

$$\mathbf{f}_q^{\text{ext}} = \frac{\sum_i \alpha_i S_{iq} \overset{o}{\mathbf{f}}_i^{\text{ext}}}{\sum_i \alpha_i S_{iq}} \tag{20}$$

$$\sigma_q = \frac{\sum_i \alpha_i S_{iq} \overset{o}{\sigma}_i}{\sum_i \alpha_i S_{iq}} \tag{21}$$

where α is unity if a node's $\texttt{maxW} > \texttt{TW}$ and zero otherwise. By using α in this manner the values on quadrature points are only based on well-conditioned nodes. In practice the quadrature point values are not stored; they are computed on-the-fly during integration.

Letting V_q be the volume of each cell (see Section 3.4) the following integrations are performed:

$$m_i = \sum_q S_{iq} \rho_q V_q \tag{22}$$

$$\mathbf{f}_i^{\text{ext}} = \sum_q S_{iq} \mathbf{f}_q^{\text{ext}} V_q \tag{23}$$

$$\mathbf{f}_i^{\text{int}} = \sum_q \sigma_i G_{iq} V_q \tag{24}$$

4.5. Extrapolation

If a node's $\texttt{maxW} < \texttt{TW}$, then the averaging process of Section 3.5 is performed for $\mathbf{f}_i^{\text{int}}$ only.

4.6. Equations of motion

All necessary data are now collected on the grid and the equation of motion can be solved and particle variables updated. The grid velocity is found directly via weighted least squares: $v_i = \overset{o}{v}_i$.

The equation of momentum is solved on the grid by

$$\mathbf{a}_i = \frac{\mathbf{f}_i^{\text{int}} + \mathbf{f}_i^{\text{ext}}}{m_i} \tag{25}$$

4.7. Time update

The time update is a centered-difference scheme commonly used in FEM. Grid acceleration is used to update grid velocity

$$\mathbf{v}_i^{n+\frac{1}{2}} = \mathbf{v}_i^{n-\frac{1}{2}} + \mathbf{a}_i \Delta t \tag{26}$$

Acceleration is interpolated back to particles and used to update particle velocity

$$\mathbf{v}_p^{n+\frac{1}{2}} = \mathbf{v}_p^{n-\frac{1}{2}} + \sum_i S_{ip} \mathbf{a}_i \Delta t \tag{27}$$

Position is updated by interpolated grid velocity, not by current particle velocity. This at first seems unnecessary but is done to ensure that particles at the same point in space have the same velocity

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \sum_i S_{ip} \mathbf{v}_i^{n+\frac{1}{2}} \Delta t \tag{28}$$

DOI: 10.1002/nme

The gradient of velocity is calculated on each particle and used to update the deformation gradient

$$\nabla \mathbf{v}_p^{n+\frac{1}{2}} = \sum_i G_{ip} \mathbf{v}_i^{n+\frac{1}{2}} \tag{29}$$

$$\mathbf{F}_p^{n+1} = \mathbf{F}_p^n + \nabla \mathbf{v}_p^{n+\frac{1}{2}} \mathbf{F}_p^n \Delta t \tag{30}$$

The centered-difference update scheme requires that velocity be initialized to a negative half time step. Velocities at the $-\frac{1}{2}$ time step are sometimes available, but for typical simulations they may be impossible to find. Instead, the approach used in this paper is to multiply the grid acceleration values of Equation (25) by $\frac{1}{2}$ for the first time step only. This propagates the $\frac{1}{2}$ through the algorithm and corrects the first-order error that would otherwise be incurred.

5. EXAMPLE PROBLEMS

A neo-Hookean constitutive model [48] is used for the example problems of this section. The strain energy function is

$$\Psi(\mathbf{F}) = \tfrac{1}{2}\lambda(\ln J)^2 - \mu \ln J + \tfrac{1}{2}\mu(\text{trace}(\mathbf{F}^{\mathrm{T}}\mathbf{F}) - 2) \tag{31}$$

where μ and λ are standard Lamé constants. Then the stress in the current configuration is defined to be

$$\sigma = \frac{\lambda \ln J}{J}\mathbf{I} + \frac{\mu}{J}(\mathbf{F}\mathbf{F}^{\mathrm{T}} - \mathbf{I}) \tag{32}$$

For the sake of stability a time step size is chosen to ensure that information cannot travel more than a characteristic distance in a single time step. However, the integration regions produced from cell subdivision cannot be used as a characteristic distance because several of them are guaranteed to be very small at any time within the problem. Experience has consistently shown that the cell size of the background grid represents a reliable characteristic distance, which is confirmed by the temporal convergence measurements of the first example to follow.

The maximum wave speed for a 3D isotropic elastic solid is defined as

$$v_{\max} = \sqrt{\frac{\lambda + 3\mu}{\rho}}. \tag{33}$$

Note that $v_{\max}$ approaches infinity as Poisson's ratio approaches $\frac{1}{2}$.

An adaptive time step size may be set throughout the simulation by

$$\Delta t(x,t) = \text{CFL}\frac{\min(h)}{v_{\max} + \max(|\mathbf{v}_p|)} \tag{34}$$

where $0 < \text{CFL} < 1$; see Figure 6(b).

A nominal time step size may be chosen at the beginning of the simulation by assuming that material properties are constant in space and time and that object velocities are well below wave speeds

$$\Delta t_0(x,t) = \text{CFL}\frac{\min(h)}{v_{\max}} \tag{35}$$

For comparison purposes an explicit non-linear FEM code based on linear triangles with single Gauss point integration is constructed according to typical designs; see, for example, the text of Belytschko *et al.* [48]. The discrete equations are listed in Appendix A.

DOI: 10.1002/nme

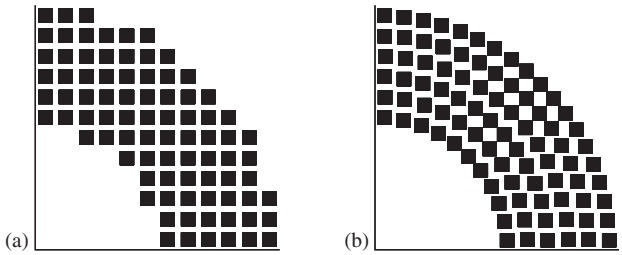

Figure 4. Alternate particle arrangements: (a) Cartesian particle placement used with GIMP and (b) radial particle placement used with WLS.

5.1. Oscillating ring

In previous work [32] a manufactured solution is developed for a dynamic ring made of a neo-Hookean material. The full development of the solution is not repeated here but its key element is the radial displacement of a ring as a function of time and position:

$$u(R) = A\cos(c\pi t)(c_1 R + c_2 R^2 + c_3 R^3) \tag{36}$$

where

$$c_1 = \frac{-6R_I}{R_O(R_O - 3R_I)}, \quad c_2 = \frac{3(R_O + R_I)}{R_O^2(R_O - 3R_I)}, \quad c_3 = \frac{-2}{R_O^2(R_O - 3R_I)}, \tag{37}$$

A is a user-specified amplification factor, $c = \sqrt{E/\rho}$, E is the Young's modulus, t is time, R is radius in the reference configuration, R_I is the inner ring radius, and R_O is the outer ring radius. The manufactured solution is used to measure the accuracy properties of the WLS method described in this work in comparison to an existing algorithm, GIMP, and to a FEM code.

For PIC methods there is some freedom in the way in which particles are arranged within the cell structure. The GIMP method typically works best when particles are arranged in a cartesian manner, because gaps and overlaps between particles are minimized. The cartesian arrangement of particles is used to discretize the ring for GIMP; see Figure 4(a).

In contrast to GIMP, the 'particles' of WLS do not represent volume in the algorithm; they can be treated as sample points only. Therefore it has been found that performance is improved by using a body-fitted arrangement of particles as shown in Figure 4(b). Generally speaking the arrangement of particles at or near the surface has a significant effect on convergence, but the arrangement of interior particles is less important, provided enough particles are present to avoid an ill-conditioned moment matrix. For the results that follow, the arrangement of particles is chosen that produces the best accuracy for each method.

Error is measured on each particle as the magnitude of the difference between computed and exact displacements:

$$\delta_p = ||(\mathbf{x}_p - \mathbf{X}_p) - \mathbf{u}_{\text{exact}}(\mathbf{X}_p, t)||. \tag{38}$$

The manufactured solution is always smooth in space and time, therefore a strict definition of error may be used: the maximum error from all particles and all time steps $L_\infty = \max(\delta_p)$. This definition demands that error over all particles be reduced for a problem to be considered more accurate, instead of merely reducing error for a majority of the particles.

The Poisson's ratio used for the ring must be zero, else the manufactured solution cannot be expressed in a closed form. In 2D the number of particles per cell is four with the cartesian arrangement, and approximately four in the radial particle arrangement. Young's modulus is 10 000, initial density is one, and the amplication factor is 0.1. The ring's initial outer radius is one and initial inner radius is 0.5. The CFL is 0.3, except for the measurements of temporal convergence. The relatively low CFL was found to be important for spatial convergence at the finest mesh sizes. Representative plots of the time history are shown in Figure 5.

DOI: 10.1002/nme

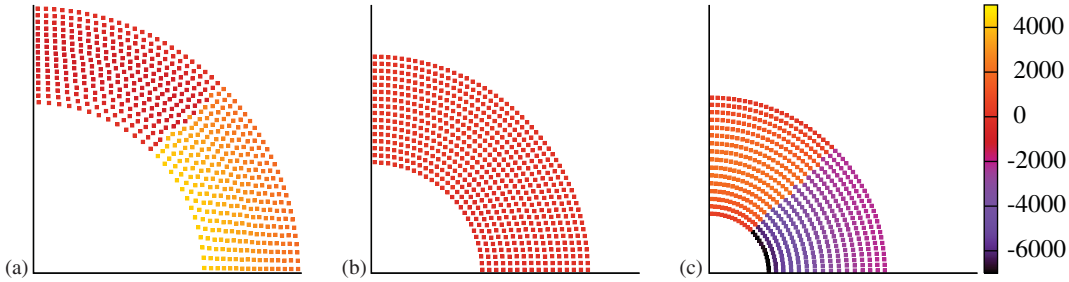

Figure 5. Radial (top) and hoop (right) stress (Pa) for oscillating ring: (a) time = 0 ms; (b) time = 5 ms; and (c) time = 10 ms.

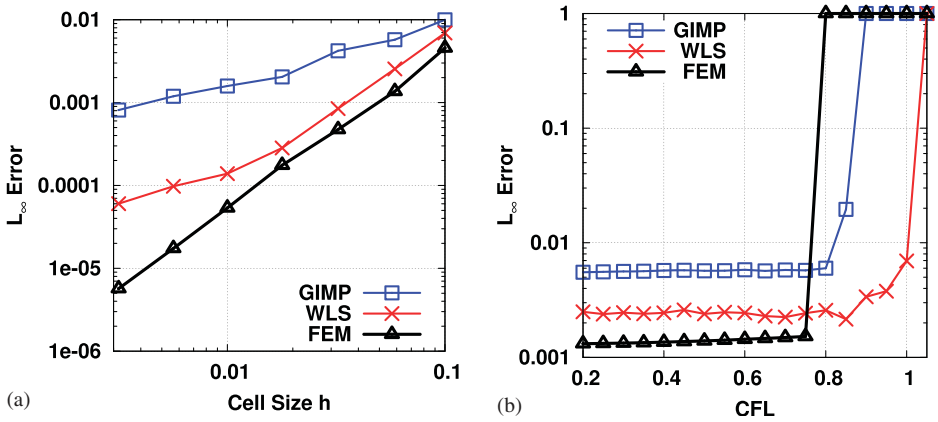

Figure 6. Convergence for the oscillating ring: (a) spatial and (b) temporal.

Results for temporal and spatial convergence of WLS are presented in Figure 6 in comparison to FEM and GIMP. Temporal convergence measurements are made using the nominal time step size of Equation (35).

For this problem WLS is an improvement over GIMP. The method behaves in a second-order manner for coarser meshes, in contrast to the first-order (at best) behavior of GIMP. This suggests that the method is suitable for use with geometrically simple objects, such as pressure vessels and beams, for which the accuracy of FEM is customarily expected. The convergence plot reflects that there is a small first-order error that is negligible until fairly high grid resolution is reached (50 cells over a unit disk); at this time, analysis has not been performed to identify the source.

Temporal convergence is not displayed for any of the methods used to solve the ring problem because spatial error dominates temporal error over the region of interest. Temporal convergence occurs for higher values of CFL used with implicit solvers, but the central difference time integration scheme used here is explicit. An important and in-depth analysis of time integration for the MPM family of methods has recently been completed by Steffen *et al.* [35].

5.2. *Single disk impact*

Although a contact model is not developed in this paper, the problems for which GIMP and WLS may be used frequently involve contact, impact, and inter-penetration scenarios, where energy conservation is a high priority. This aspect of the algorithm is assessed via the frictionless impact of a disk against a wall as shown in Figure 7.

The Poisson's ratio is 0.3, the number of particles per cell is about 4, and Young's modulus and initial density are both 1000. The disk is 0.4 units in diameter and is initially located within a unit square of 48×48 cells at (0.3,0.3). The disk has initial particle velocities of $(0.2, -0.2)$ with $CFL = 0.4$. The kinetic energy of the disk is expected to reduce by half during impact, because

DOI: 10.1002/nme

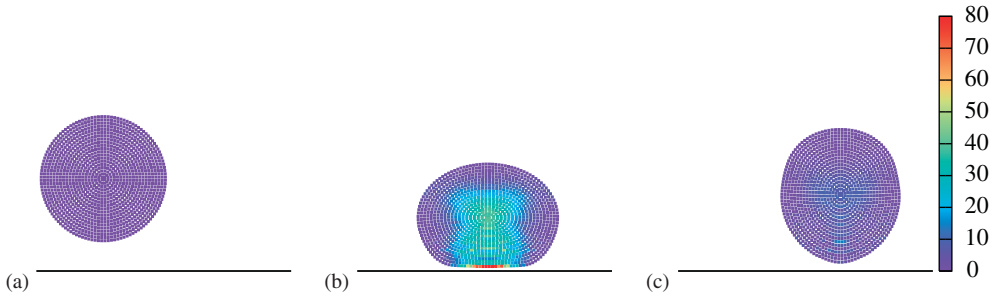

Figure 7. Strain energy per volume for single disk: (a) time = 0 s; (b) time = 2 s; and (c) time = 3 s.

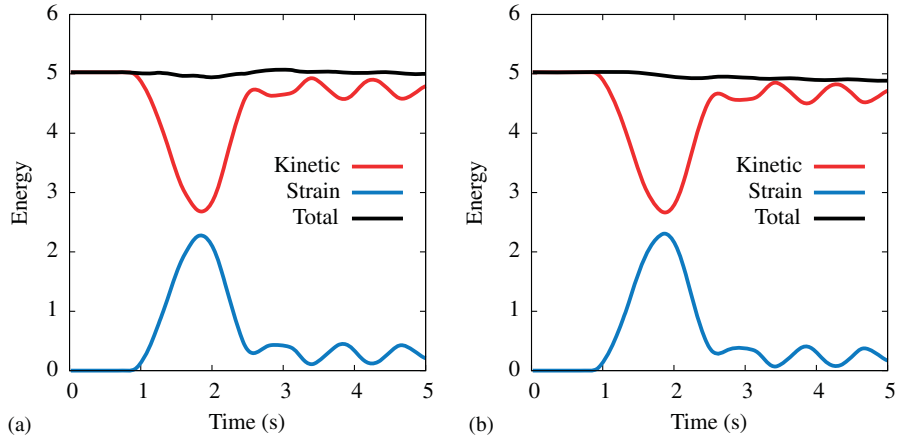

Figure 8. Energy for single-disk impact: (a) WLS and (b) GIMP.

the y-component of velocity will approach zero during impact, while the x-component will remain about the same.

This system is modeled with WLS and GIMP, and the strain and kinetic energies are plotted in Figure 8 as a function of the time.

It can be seen that energy is handled correctly in WLS. The transfer of energy from kinetic to strain, then back again, occurs smoothly and without significant fluctuation or instability. The corresponding energy plot is also shown for GIMP which displays the same desirable trends.

For multi-body impact it should be straightforward to implement contact in the manner commonly used for GIMP, where a separate computational grid is created for each object, and the contact is handled with a method such as that described by Bardenhagen *et al.* [42].

5.3. Dynamic hole in plate

A square plate with a circular hole is suddenly subjected to a body force in the horizontal direction. The body force is modeled with a force vector $(5 \times 10^6 X, 0)$ in the reference configuration that increases with X-position but remains constant in time. The solution involves a large deformation, dynamic simulation with a neo-Hookean constitutive model; therefore no exact answer is available. The Poisson's ratio is 0.3, the number of particles per cell is about 4, Young's modulus is 210×10^9, and initial density is 7850. The plate is one unit high and wide and the hole radius is 0.5. The CFL is 0.4. The progression of the simulation is shown in Figure 9.

In order to compare WLS to GIMP and FEM, a critical area of the problem—the surface of the hole—is examined in Figure 10 at the final time of 696 μs. For the sake of stress analysis it must be found what values of stress develop on the hole's surface so that a determination can be made about whether the plate is strong enough to withstand the forces on it.

DOI: 10.1002/nme

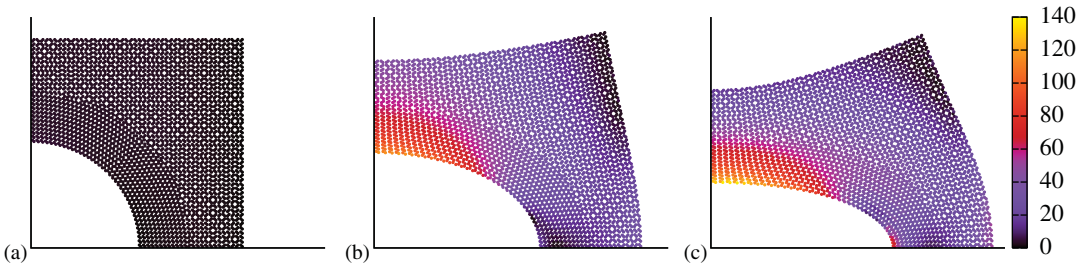

Figure 9. Von Mises stress (GPa) of plate with hole: (a) time = 0 μs; (b) time = 464 μs; and (c) time = 696 μs.

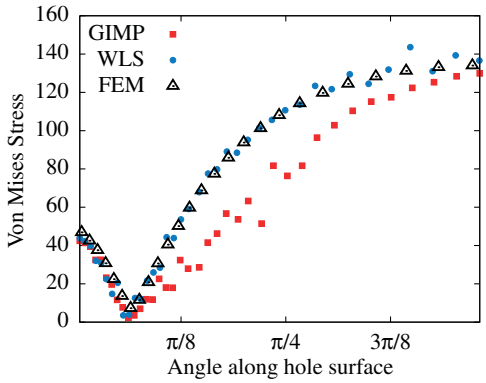

Figure 10. Comparison of Von Mises stress (GPa) at hole surface.

In Figure 10 the particles or FEM Gauss points that are initially located within a distance of $0.35h$ from the hole surface, where h is the WLS cell size, are compared for FEM, WLS, and GIMP. The Von Mises stress is plotted with respect to the angle along the hole surface. The angle of the bottom face of the bar is zero, and the angle of the left face is $\pi/2$.

The results displayed in the figure suggest that WLS is able to provide a surface stress which is nearly as accurate as FEM. But GIMP shows some difficulty in predicting surface stress, and the GIMP result has more scatter. While it would be difficult to determine a reliable failure point for the surface of the hole using GIMP, the WLS result reflects realistic stresses at the hole's surface.

6. DISCUSSION

The WLS method introduced in this paper achieves the objective of improving the accuracy of solid mechanics simulations that are performed within a PIC framework. The method takes advantage of the concepts of weighted least squares surface estimation and implicit surface definition to more precisely define the region of integration. The method is motivated by two general requirements. It needs to provide the accuracy of finite elements, which are well-known and are trusted for predictive results. And it needs to be as easy to initialize as other PIC methods, while avoiding nearest neighbor searches and handling contact and inter-penetration scenarios with ease. The method of this paper represents a compromise between the demands of these two families of methods and affords several benefits of both.

The method achieves accuracy that is significantly better than GIMP, but somewhat less than FEM. Its PIC form allows it to be used within an existing MPM/GIMP implementation while avoiding nearest neighbor searches. However, the complexity of implementation and initialization are both greater than required for FEM or MPM/GIMP. In addition to the programming effort required for GIMP, WLS also requires implementation of least squares and marching cubes routines.

DOI: 10.1002/nme

Problems are initialized with the interior particles of GIMP, but also require surface particles, which may be calculated from three-dimensional image data with additional effort.

The PIC structure of the method allows it to be used side-by-side with GIMP in space or time to provide additional accuracy for certain objects. The interior particles of WLS may be located at the same positions as GIMP particles and have the same initial volumes; surface particles have zero volume. This enables a one-way transfer of algorithm from WLS to GIMP if a problem begins to display behaviors for which WLS is ill-suited, such as rupture of a surface. For example, an over-pressurized tank can be modeled up until its point of rupture with WLS, then GIMP may assume control of the solution using the same particles and continue simulation of the disintegration of the tank.

APPENDIX A: DISCRETE EQUATIONS FOR THE COMPARISON FINITE ELEMENT CODE

An explicit non-linear FEM code based on linear triangles with single Gauss point integration is constructed following the text of Belytschko *et al.* [48]. The discrete momentum equation at each node i is

$$\mathbf{a}_i = \mathbf{f}_i^{\text{int}}/M_i + \mathbf{b}(\mathbf{X}_i, t) \tag{A1}$$

Mass lumping is defined from the reference configuration as

$$M_i = \frac{1}{3}\rho^0 \sum_e A_e^0 \tag{A2}$$

where the index e loops over the elements surrounding node i. The internal forces are

$$\mathbf{f}_i^{\text{int}} = -\sum_e \sigma(\mathbf{F}_e)\nabla\phi_e(\mathbf{x}_i) \tag{A3}$$

where the deformation gradient on each element is

$$\mathbf{F}_e = \left(\mathbf{I} - \sum_i^3 \mathbf{u}_i \nabla\phi_e(\mathbf{x}_i)\right)^{-1} \tag{A4}$$

The updates of nodal velocity and position are

$$\mathbf{v}_i^{n+1/2} = \mathbf{v}_i^{n-1/2} + \mathbf{a}_i^n \Delta t \tag{A5}$$

$$\mathbf{x}_i^{n+1} = \mathbf{x}_i^n + \mathbf{v}_i^{n+1/2}\Delta t \tag{A6}$$

REFERENCES

1. Lucy LB. A numerical approach to the testing of the fission hypothesis. *Astronomical Journal* 1977; **82**: 1013–1024.
2. Monaghan JJ. An introduction to sph. *Computer Physics Communications* 1998; **48**:89–96.
3. Libersky LD, Petschek AG. Smooth particle hydrodynamics with strength of materials. *Advances in the Free Lagrange Method*. Lecture Notes in Physics, vol. 395/1991. Springer: Berlin, 1990; 248–257. DOI: 10.1007/3-540-54960-9_58.
4. Randles PW, Libersky LD. Smoothed particle hydrodynamics: some recent improvements and applications. *Computer Methods in Applied Mechanics and Engineering* 1996; **139**:375–408.
5. Swegle JW, Hicks DA. Smooth particle hydrodynamics stability analysis. *Journal of Computational Physics* 1995; **116**:123–134.
6. Johnson GR, Beissel SR. Normalized smoothing functions for sph impact computations. *International Journal for Numerical Methods in Engineering* 1996; **39**(16):2725–2741.
7. Dilts GA. Moving least squares particle hydrodynamics. II: Conservation and boundaries. *International Journal for Numerical Methods in Engineering* 2000; **48**(10):1503–1524.

DOI: 10.1002/nme

8. Nayroles B, Touzot G, Villon P. Generalizing the finite element method: diffuse approximation and diffuse elements. *Computational Mechanics* 1992; **10**:307–318.
9. Belytschko T, Lu YY, Gu L. Element free Galerkin methods. *International Journal for Numerical Methods in Engineering* 1994; **37**:229–256.
10. Lu YY, Belytschko T, Gu L. A new implementation of the element free Galerkin methods. *Computer Methods in Applied Mechanics and Engineering* 1994; **113**:397–414.
11. Belytschko T, Gu L, Lu YY. Fracture and crack growth by element free Galerkin methods. *Modelling and Simulation in Materials Science and Engineering* 1994; **115**:277–286.
12. Liu WK, Adee J, Jun S. Reproducing kernel particle methods for elastic and plastic problems. *Advanced Computational Methods for Material Modeling*. vol. AMD 180 and PVP 268. ASME: New York, 1993; 175–190.
13. Liu WK, Jun S, Li S, Adee J, Belytschko T. Reproducing kernel particle methods for structural dynamics. *International Journal for Numerical Methods in Engineering* 1995; **38**:1655–1679.
14. Liu WK, Jun S, Zhang Y. Reproducing kernel particle methods. *International Journal for Numerical Methods in Fluids* 1995; **20**:1081–1106.
15. Jun S, Liu WK, Belytschko T. Explicit reproducing kernel particle methods for large deformation problems. *International Journal for Numerical Methods in Engineering* 1998; **41**:137–166.
16. Atluri SN, Zhu T. A new meshless local Petrov-Galerkin (mlpg) method. *Computer Modeling in Engineering and Sciences* 1998; **22**:117–127.
17. Belytschko T, Krongauz Y, Organ D, Fleming M, Krysl P. Meshless methods: an overview and recent developments. *Computer Methods in Applied Mechanics and Engineering* 1996; **139**:3–47.
18. Fries TP, Matthies HG. Classification sand overview of meshfree methods. *Technical Report*, Technical University Braunschweig, 2004.
19. Liu GR. *Mesh Free Methods*: *Moving Beyond The Finite Element Method*. CRC Press LLC: Boca Raton, 2003.
20. Duarte CA, Oden JT. Hp clouds—A meshless method to solve boundary-value problems. *Technical Report 95-05*, Texas Institute for Computational and Applied Mathematics, University of Texas at Austin, 1995.
21. Melenk JM, Babuska I. The partition of unity finite element method: basic theory and applications. *Computer Methods in Applied Mechanics and Engineering* 1996; **139**:289–314.
22. Liu WK, Li S, Belytschko T. Moving least-square reproducing kernel methods (i) methodology and convergence. *Computer Methods in Applied Mechanics and Engineering* 1997; **143**:113–154.
23. Fernández-Méndez S, Huerta A. Computer methods in applied mechanics and engineering. *International Journal for Numerical Methods in Engineering* 2004; **193**:1257–1275.
24. Rabczuk T, Belytschko T, Xiao SP. Stable particle methods based on lagrangian kernels. *Computer Methods in Applied Mechanics and Engineering* 2004; **193**:1035–1063.
25. Moës N, Dolbow J, Belytschko T. A finite element method for crack growth without remeshing. *International Journal for Numerical Methods in Engineering* 1999; **46**:131–150.
26. Sulsky D, Chen Z, Schreyer H. A particle method for history dependent materials. *Computer Methods in Applied Mechanics and Engineering* 1994; **118**:179–196.
27. Sulsky D, Zhou S, Schreyer H. Application of a particle-in-cell method to solid mechanics. *Computer Physics Communications* 1995; **87**:236–252.
28. Sulsky D, Schreyer H. Axisymmetric form of the material point method with applications to upsetting and Taylor impact problems. *Computer Methods in Applied Mechanics and Engineering* 1996; **139**:409–429.
29. Brackbill J, Ruppel H. Flip: a low-dissipation, particle-in-cell method for fluid flows in two dimensions. *Journal of Computational Physics* 1986; **65**:314–343.
30. Harlow F. The particle-in-cell computing method for fluid dynamics. *Methods in Computational Physics* 1963; **3**:319–343.
31. Bardenhagen S, Kober E. The generalized interpolation material point method. *Computer Modeling in Engineering and Sciences* 2004; **5**:477–495.
32. Wallstedt P, Guilkey J. An evaluation of explicit time integration schemes for use with the generalized interpolation material point method. *Journal of Computational Physics* 2008; **227**:9628–9642.
33. Bardenhagen S. Energy conservation error in the material point method for solid mechanics. *Journal of Computational Physics* 2002; **180**:383–403.
34. Sulsky D, Schreyer H, Peterson K, Kwok R, Coon M. Using the material point method to model sea ice dynamics. *Journal of Geophysical Research* 2007; **112**, DOI: 10.1029/2005JC003329.
35. Steffen M, Kirby RM, Berzins M. Decoupling and balancing of space and time errors in the material point method (MPM). *International Journal for Numerical Methods in Engineering* 2010; **82**(10):1207–1243. DOI: 10.1002/nme.2787.
36. York AR, Sulsky DL, Schreyer HL. The material point method for simulation of thin membranes. *International Journal for Numerical Methods in Engineering* 1999; **44**:1429–1456.
37. York AR, Sulsky DL, Schreyer HL. Fluid-membrane interaction based on the material point method. *International Journal for Numerical Methods in Engineering* 2000; **48**:901–924.
38. Guilkey JE, Weiss JA. Implicit time integration for the material point method: quantitative and algorithmic comparisons with the finite element method. *International Journal for Numerical Methods in Engineering* 2003; **57**:1323–1338.
39. Sulsky D, Kaul A. Implicit dynamics in the material-point method. *Computer Methods in Applied Mechanics and Engineering* 2004; **193**:1137–1170.

40. Love E, Sulsky DL. An unconditionally stable, energy–momentum consistent implementation of the material-point method. *Computer Methods in Applied Mechanics and Engineering* 2006; **195**:3903–3925.
41. Love E, Sulsky DL. An energy-consistent material-point method for dynamic finite deformation plasticity. *International Journal for Numerical Methods in Engineering* 2005; **65**:1608–1638.
42. Bardenhagen S, Guilkey J, Roessig K, Brackbill J, Witzel W, Foster J. An improved contact algorithm for the material point method and application to stress propagation in granular material. *Computer Modeling in Engineering and Sciences* 2001; **2**:509–522.
43. Nairn JA. Material point method calculations with explicit cracks. *Computer Modeling in Engineering and Sciences* 2003; **4**:649–663.
44. Ma J, Lu H, Komanduri R. Structured mesh refinement in generalized interpolation material point method (gimp) for simulation of dynamic problems. *Computer Modeling in Engineering and Sciences* 2006; **12**:213–227.
45. Wallstedt P, Guilkey J. Improved velocity projection for the material point method. *Computer Modeling in Engineering and Sciences* 2007; **19**:223–232.
46. Belytschko T, Parimi C, Moës N, Sukumar N, Usui S. Structured extended finite element methods for solids defined by implicit surfaces. *International Journal for Numerical Methods in Engineering* 2003; **56**:609–635.
47. Lorensen WE, Cline HE. Marching cubes: a high resolution 3d surface construction algorithm. *ACM SIGGRAPH. Computer Graphics* 1987; **21**(4):163–169.
48. Belytschko T, Liu WK, Moran B. *Nonlinear Finite Elements for Continua and Structures*. Wiley: New York, 2000.

DOI: 10.1002/nme