Material Point Method Calculations with Explicit Cracks

J. A. Nairn[1]

Abstract: A new algorithm is described which extends the material point method (MPM) to allow explicit cracks within the model material. Conventional MPM enforces velocity and displacement continuity through its background grid. This approach is incompatible with cracks which are displacement and velocity discontinuities. By allowing multiple velocity fields at special nodes near cracks, the new method (called CRAMP) can model cracks. The results provide an "exact" MPM analysis for cracks. Comparison to finite element analysis and to experiments show it gets good results for crack problems. The intersection of crack surfaces is prevented by implementing a crack contact scheme. Crack contact can be modeled using stick or sliding with friction. All results are two dimensional, but the methods can be extended to three dimensional problems.

keyword: Material point method, cracks, fracture, numerical methods, contact

1 Introduction

The material point method (MPM) has recently been developed as a numerical method for solving problems in dynamic solid mechanics [Sulsky, Chen, and Schreyer (1994), Sulsky, Zhou, and Schreyer (1995), Sulsky and Schreyer (1996), Zhou (1998)]. In MPM, a solid body is discretized into a collection of points much like a computer image is represented by pixels. As the dynamic analysis proceeds, the solution is tracked on the material points by updating all required properties such as position, velocity, acceleration, stress state, *etc.*. At each time step, the particle information is extrapolated to a background grid which serves as a calculational tool to solve the equations of motions. Once the equations are solved, the grid-based solution is used to update all particle properties. This combination of Lagrangian and Eulerian methods has proven useful for solving solid mechanics problems including those with large deformations or rotations and involving materials with history dependent properties such as plasticity or viscoelasticity effects [Sulsky, Chen, and Schreyer (1994), Sulsky, Zhou, and Schreyer (1995), Sulsky and Schreyer (1996), Zhou (1998)]. MPM is amendable to parallel computation [Parker (2002)], implicit integration methods [Guilkey and Weiss (2002)], and alternative interpolation schemes that improve accuracy [Bardenhagen and Kober (2003)].

Although MPM uses a background grid and is frequently compared to finite element methods, a new derivation of MPM [Bardenhagen and Kober (2003)] presents it as a Petrov-Galerkin method that has similarities with meshless methods such as Element-Free Galerkin (EFG) methods [Belytschko, Lu, and Gu (1994)] and Meshless-Local Petrov-Galerkin (MLPG) methods [Atluri and Shen (2002a), Atluri and Shen (2002b), Atluri and Zhu (1998)]. The "meshless" aspect of MPM, despite the use of a grid, derives from the fact that the body and the solution are described on the particles while the grid is used solely for calculations. The body can translate through the grid. Furthermore the grid can be discarded each time step and redrawn which makes MPM suitable to adaptive mesh methods. It is essential for any extension to MPM, such as presented here, to preserve the separation between the grid and the particles. MPM, EFG, and MLPG differ in their methods used to derive shape functions and in their selection of test functions during numerical implementation [Bardenhagen and Kober (2003), Atluri and Shen (2002a)].

One potential application of MPM is as a tool in dynamic fracture modeling. It was recently shown that MPM can accurately calculate fracture parameters such as energy release rate [Tan and Nairn (2002)], but those results were for a crack at a symmetry plane and thus the crack could be described by symmetry conditions alone. Conventional MPM is not capable of handling explicit, internal cracks. The problem is that conventional MPM methods extrapolate particle information to a single velocity

[1] Material Science and Engineering, University of Utah, Salt Lake City, Utah 84112, USA

field on the background grid. A property of the background grid, which is analogous to finite element analysis grids, is that all displacements in the single velocity field are continuous. Because representation of cracks requires displacement discontinuities, MPM can not represent cracks.

This paper describes a modified material point method labeled as CRAMP for ”CRAcks” with ”Material Points.” The following list gives the essential differences between CRAMP and conventional MPM:

1. Cracks are described in 2D as a series of line segments. The end-points of the line segments can be additional mass-less particles to make it easy to add crack descriptions to standard MPM data structures.
2. Each node in the background grid is allowed to have multiple velocity fields. If the node is far from any crack, the node will have a single velocity field as in conventional MPM. For nodes near a crack, however, each node will have separate velocity fields for information interpolated from particles on opposite sides of a crack.
3. Most calculations in the MPM algorithm needed to be adjusted to account for the possibility that a particular node might have multiple velocity fields.
4. When nodes have separate velocity fields and displacement continuities, it is possible for the two sides of the crack to cross over each other. To prevent non-physical crossing, all calculations at nodes with multiple velocity fields must implement contact methods. The algorithm in this paper can model crack contact by stick or by sliding with friction.
5. A modified scheme for updating stresses and strains was added which appears to improve energy calculations. An important application of crack calculations is to do fracture predictions. Because fracture work requires accurate energy calculations, it was important to optimize the MPM energy results. This last correction can be applied to conventional MPM as well as to CRAMP.

All results in this paper are for 2D calculations. In most cases, the extension to 3D is simple and obvious. The one exception is the crack description. In 3D, the crack needs to be described by connected surfaces instead of line segments. The algorithm presented here does stress analysis calculations for existing, internal cracks. The subject of evaluating fracture parameters at crack tips and predicting crack propagation will be in future work. The crack description used, however, is very flexible. It is a trivial matter to extend the crack during a dynamic analysis once the harder problem of deciding when and where to extend the crack has been solved. Finally, the CRAMP algorithm is efficient. Comparison between conventional MPM and CRAMP show the crack calculations to be only about 10% slower. The most time consuming part of the CRAMP algorithm is determining which particles interpolate to which velocity field at each node.

2 MPM With Explicit Cracks — CRAMP

This section describes the CRAMP algorithm for including explicit cracks in MPM calculations. The features of the algorithm are given here; the full algorithm is provided in the Appendix. The first task is to describe the internal cracks. One simple way to introduce displacement discontinuities into MPM would be to introduce cracks in the background grid. Although this approach can handle certain problems, it severely limits the flexibility of MPM. The background grid is supposed to serve as a calculational tool and not as a device to carry information about the solution or about the solid. It would also be difficult or impossible to translate the crack along with the body during large displacement calculations. In CRAMP, the crack is instead described as a series of line segments. For compatibility with MPM data structures, the endpoints of the line segments are massless material points. By translating the crack segments along with the solution, it is possible to track cracks in moving bodies. A problem can contain any number of cracks.

2.1 Multiple Velocity Fields

The influence of cracks on the MPM solution is that they influence the velocity fields at some nodes in the background grid. In conventional MPM, the first step in the algorithm is to extrapolate the particle momenta and masses to the background grid. The equations are [Sulsky, Zhou, and Schreyer (1995)]:

$$\mathbf{p}_i^k = \sum_{p=1}^{n_p} m_p \mathbf{v}_p^k S_{i,p}^k \qquad m_i^{Dk} = \sum_{p=1}^{n_p} m_p S_{i,p}^k \tag{1}$$

where $\mathbf{p}_i^k$ is nodal momentum, $\mathbf{v}_p^k$ is particle velocity, m_p is particle mass, $S_{i,p}^k$ is the shape function for node i evaluated at the current position of particle p, and m_i^{Dk} is the nodal mass in the lumped (or diagonal) mass matrix. The superscript k indicates these terms apply to the k^{th} MPM step. In this approach, each nodal point has a single momentum and displacement discontinuities are not allowed. To allow displacement discontinuities, CRAMP allows each node to have three types of velocity fields - one for particles on the same side of all cracks as the node (0), one for particles above a crack relative to the node (1), and one for particles below a crack relative to the node (2). The first step in crack calculations is thus to examine each particle-node combination (with non-zero shape function) and determine the appropriate velocity field denoted by

$$\nu(p,i) = 0,\ 1,\ \text{or}\ 2 \tag{2}$$

This determination is done by a line-crossing algorithm. First, a line is drawn from particle p to node i. If the line does not cross any crack, the velocity field is 0; if it crosses a crack from above, the velocity field is 1; if it crosses a crack from below, the velocity field is 2. The field determination is the most time consuming part of CRAMP and must be done efficiently; an efficient algorithm is given in the Appendix. Once the velocity field's are determined, the modified, initial MPM extrapolations become

$$\left.\begin{aligned} \mathbf{p}_{i,j}^k &= \sum_{p=1}^{n_p} m_p \mathbf{v}_p^k S_{i,p}^k \delta_{j,\nu(p,i)} \\ m_{i,j}^{Dk} &= \sum_{p=1}^{n_p} m_p S_{i,p}^k \delta_{j,\nu(p,i)} \end{aligned}\right\} j = 0,\ 1,\ 2 \tag{3}$$

Each node may have one to three velocity fields denoted by index j on $\mathbf{p}_{i,j}^k$ and $m_{i,j}^{Dk}$. Each velocity field interpolates only from particles contributing to that field as determined by the Kronecker delta function $\delta_{j,\nu(p,i)}$. Although three velocity fields are defined, no node should ever have more that two velocity fields — one each for particles on the two sides of a crack.

The possible existence of multiple velocity fields carries through the remainder of the algorithm. Each conventional MPM calculation must consider all velocity fields. For example, the total nodal forces (with damping) are

$$f_{i,j}^{tot} = f_{i,j}^{int} + f_{i,j}^{ext} - \kappa \mathbf{p}_{i,j}^k \quad j = 0,\ 1,\ 2\ (\text{as needed}) \tag{4}$$

where $f_{i,j}^{int}$ and $f_{i,j}^{ext}$ are internal and external forces at a node and κ is a damping coefficient. The updated nodal momenta become

$$\mathbf{p}_{i,j}^{k'} = \mathbf{p}_{i,j}^k + \Delta t\, f_{i,j}^{tot} \qquad j = 0,\ 1,\ 2\ (\text{as needed}) \tag{5}$$

where Δt is the time step. When updating particle positions, velocities, stresses, and strains, the equations only use the velocity field appropriate to each particle/node pair. For example, updating of position becomes

$$\mathbf{x}_p^{k+1} = \mathbf{x}_p^k + \Delta t \sum_{i=1}^{n_n} \frac{\mathbf{p}_{i,\nu(p,i)}^{k'} S_{i,p}^k}{m_{i,\nu(p,i)}^{Dk}} \tag{6}$$

Above are some examples of the modified equations; all the required modifications are detailed in the Appendix.

If the body is translating, all cracks need to translate along with it. This task is accomplished by calculating the center of mass velocity of each node with multiple velocity fields:

$$\mathbf{v}_{i,cm}^k = \frac{\sum_{j=1}^3 \mathbf{p}_{i,j}^{k''} \varphi_{i,j}}{\sum_{j=1}^3 m_{i,j}^{Dk} \varphi_{i,j}} \tag{7}$$

where $\varphi_{i,j}$ is 1 or 0 depending on whether or not velocity field j is present at node i. Once all nodal velocities are reduced to a single field, the mass-less particles that define the crack can move using standard MPM methods for updating particle position.

2.2 Crack Surface Contact

Several times during each MPM step, the nodal momenta and velocities are updated. These updates may occur as a consequence of boundary conditions or when implementing the equations of motion. Whenever the nodal momenta change, it is essential to verify that the change corresponds to a physically allowed change which is defined here as meaning that opposite sides of cracks do not cross over each other. To prevent non-physical changes, the CRAMP algorithm includes contact methods. The methods are based on the contact methods develop by Bardenhagen [Bardenhagen, Brackbill, and Sulsky (2000), Bardenhagen, Guilkey, Roessig, Brackbill, Witzel, and Foster (2001)], but there are two key differences. First, the methods in Bardenhagen, Brackbill, and Sulsky (2000) and Bardenhagen, Guilkey, Roessig, Brackbill, Witzel, and Foster (2001) were for contact between dissimilar materials; here contact is within the same material but on two sides of a crack. Second, their methods used to identify contact did not work well for internal cracks; they would either identify contact too soon or identify it unreliably. This section describes the contact methods in CRAMP.

The identification of crack surface contact is based mostly on nodal volume at crack nodes (*i.e.*, nodes with multiple velocity fields). The total nodal volume (which is a nodal area in 2D calculations) is calculated during each MPM step using

$$V_i^k = \sum_{p=1}^{n_p} V_p^k S_{i,p}^k \tag{8}$$

where V_p^k is the volume of particle p or an area in 2D calculations defined by

$$V_p^k = (1+\varepsilon_{p,xx}^k)(1+\varepsilon_{p,yy}^k)\frac{m_p}{\rho_p t_p} \tag{9}$$

where ρ_p and t_p are the density and thickness of the 2D particle. Whenever momenta change, the nodal volumes at all nodes with multiple velocity fields are normalized by the undeformed volume. For regular grids the undeformed volume is the volume of each element in the background grid; for irregular grids, the undeformed volume includes a portion of each element containing that node. The relative volume is defined as

$$V_{rel} = \frac{V_i^k}{V_i^{undef}} \tag{10}$$

Two critical relative volumes are preselected as V_{sep} (less than 1) and $V_{contact}$ (greater than 1). If $V_{rel} < V_{sep}$, the crack surfaces are assumed to be separated and no changes in momenta are needed. If $V_{rel} > V_{contact}$, the cracks surfaces are assumed to be in contact and momenta are adjusted as explained below. The region $V_{sep} < V_{rel} < V_{contact}$ is a gray area and a second method is used to decide whether or not contact is present.

Several second methods are possible, but the calculations here used relative velocity of the two crack surfaces [Bardenhagen, Brackbill, and Sulsky (2000), Bardenhagen, Guilkey, Roessig, Brackbill, Witzel, and Foster (2001)]. Once a crack node with $V_{sep} < V_{rel} < V_{contact}$ is found, the velocities above and below the crack are calculated from

$$\mathbf{v}_{i,a}^k = \frac{\mathbf{p}_{i,a}^k}{m_{i,a}^{Dk}} \quad \mathbf{v}_{i,b}^k = \frac{\mathbf{p}_{i,b}^k}{m_{i,b}^{Dk}} \tag{11}$$

where a and b indicate the velocity fields corresponding to particles above and below the crack. By examining the angle between the relative velocities and the crack surface normal, the cracks are assumed to be separated if they are moving apart as defined by

$$(\mathbf{v}_{i,a}^k - \mathbf{v}_{i,b}^k)\cdot\hat{\mathbf{n}} \le 0 \tag{12}$$

where $\hat{\mathbf{n}}$ is the crack surface normal. If the crack surfaces are moving towards each other, the cracks are assumed to be in contact and the momenta are adjusted. The crack surface normal can be calculated earlier in the algorithm during the line-crossing algorithm. Whenever a line from a particle to a node crosses a crack, the normal to that crack segment is saved for that node. The normal at a given node is the average of all such normal vectors. The normal is defined as directed from above the crack to below the crack.

Once contact is identified (by $V_{rel} > V_{contact}$ or by crack surfaces moving towards each other), the momenta are adjusted by two alternate methods. The simplest contact method is contact by *stick* conditions. The *stick* method simply reverts to conventional MPM where the momenta above and below the crack are set equal to each other and equal to the center-of-mass momenta. The required momenta changes to implement *stick* conditions are

$$\Delta\mathbf{p}_{i,a}^k = \frac{m_{i,a}^{Dk}\mathbf{p}_{i,b}^k - m_{i,b}^{Dk}\mathbf{p}_{i,a}^k}{m_{i,a}^{Dk}+m_{i,b}^{Dk}} \quad \Delta\mathbf{p}_{i,b}^k = -\Delta\mathbf{p}_{i,a}^k \tag{13}$$

These changes conserve total momentum.

A frictional sliding contact method follows the approach of Bardenhagen, Guilkey, Roessig, Brackbill, Witzel, and Foster (2001). In physical terms, this method adjusts the velocity above the crack to be

$$\tilde{\mathbf{v}}_{i,a}^k = \mathbf{v}_{i,a}^k - \Delta v_n\left(\hat{\mathbf{n}} + \mu'\hat{\mathbf{t}}\right) \tag{14}$$

where $\hat{\mathbf{t}}$ is a unit normal tangential to the crack surface in the direction of sliding and μ' is an *effective* coefficient of friction defined by

$$\mu' = \min\left(\mu, \frac{\Delta v_t}{\Delta v_n}\right) \tag{15}$$

Here μ is the actual coefficient of friction, and Δv_n and Δv_t are the components of the relative crack face velocities normal and tangential to the crack:

$$\Delta v_n = \left(\mathbf{v}_{i,a}^k - \mathbf{v}_{i,b}^k\right)\cdot\hat{\mathbf{n}} \quad \Delta v_t = \left(\mathbf{v}_{i,a}^k - \mathbf{v}_{i,b}^k\right)\cdot\hat{\mathbf{t}} \tag{16}$$

When μ' reduces to $\Delta v_t/\Delta v_n$, the surfaces are sticking due to friction or the velocities are adjusted to equal the center of mass velocities. When μ' reduces to μ, the contact is by friction. In the limit of frictionless sliding ($\mu = 0$), μ' is always zero, the crack surface velocities normal to the surfaces are adjusted to be equal and the tangential velocities remain unchanged. All CRAMP algorithm steps are in terms of nodal momenta instead of velocities. The above velocity equations in terms of momenta correspond to adjusting the momenta above and below the crack by

$$\begin{aligned} \Delta\mathbf{p}^k_{i,a} &= \left[\left(\frac{m^{Dk}_{i,a}\mathbf{p}^k_{i,b} - m^{Dk}_{i,b}\mathbf{p}^k_{i,a}}{m^{Dk}_{i,a}+m^{Dk}_{i,b}}\right)\cdot\hat{\mathbf{n}}\right](\hat{\mathbf{n}} - \mu'\hat{\mathbf{t}}) \\ \Delta\mathbf{p}^k_{i,b} &= -\Delta\mathbf{p}^k_{i,a} \end{aligned} \tag{17}$$

where $\hat{\mathbf{t}}$ is now a unit vector tangential to the crack surface that is in the same direction as the crack line segments and μ' is a signed quantity given by

$$\mu' = \begin{cases} -\mu & \text{if } \dfrac{\Delta v_t}{\Delta v_n} < -\mu \\ +\mu & \text{if } \dfrac{\Delta v_t}{\Delta v_n} > +\mu \\ \dfrac{\Delta v_t}{\Delta v_n} & \text{otherwise} \end{cases} \tag{18}$$

Finally, the normal and tangential velocity components can be calculated from

$$\begin{aligned} \Delta v_n &= \frac{m^{Dk}_{i,a}\mathbf{p}^k_{i,b} - m^{Dk}_{i,a}\mathbf{p}^k_{i,b}}{m^{Dk}_{i,a}}\cdot\hat{\mathbf{n}} \\ \Delta v_t &= \frac{m^{Dk}_{i,a}\mathbf{p}^k_{i,b} - m^{Dk}_{i,a}\mathbf{p}^k_{i,b}}{m^{Dk}_{i,a}}\cdot\hat{\mathbf{t}} \end{aligned} \tag{19}$$

One difference between this frictional contact method and the one in Bardenhagen, Guilkey, Roessig, Brackbill, Witzel, and Foster (2001) is that the same normal is used for both crack surfaces (it is defined from the crack line segments) and thus momentum is exactly conserved. In Bardenhagen, Guilkey, Roessig, Brackbill, Witzel, and Foster (2001), the two contacting materials had separate normal vectors and momentum was only conserved when those normals were equal and opposite.

2.3 Modified Method to Update Particle Stresses and Strains

One part of each MPM step involves updating the particle stresses and strains. If this task is not done optimally, there can be numerical difficulties and inaccuracies in energy calculations. Because accurate energy results are essential for fracture calculations, the updating methods were examined and slightly revised from previous methods in the literature [Sulsky, Chen, and Schreyer (1994), Sulsky, Zhou, and Schreyer (1995), Sulsky and Schreyer (1996), Zhou (1998), Bardenhagen (2002)].

As explained in the appendix, updating the stresses and strains on the particles involves calculating the strain increment for the current step and then using a constitutive law to determine the stress increment. The strain increment is a function of the current strain rate which is calculated from the current nodal velocities (see Subtask 2 for updating stresses and strains in the Appendix). There are several alternatives for which nodal velocities to use for the updating process. In an early MPM paper [Sulsky, Chen, and Schreyer (1994)], the nodal velocities were calculated *after* updating the nodal momenta. This approach, referred to as the "Update Stress Last" or USL, has serious numerical difficulties which are revealed by considering a node which interacts with only a single particle. Following through the algorithm in the Appendix, the nodal velocity used for strain rates would at such a node would be

$$\mathbf{v}_i = \mathbf{v}^k_p + \Delta t\left(-\frac{\overline{\sigma}^k_p\cdot\mathbf{G}^k_{i,p}}{S^k_{i,p}} + \overline{\mathbf{b}_\mathbf{p}} + \frac{f^k_p}{m_p}\right) \tag{20}$$

For simplicity, this analysis only considers a single velocity field and ignores damping. Only a single velocity field is considered, because only one is possible when there is only a single particle. As a consequence, all discussion in this section applies to both conventional MPM and to CRAMP. The first term in the brackets causes a problem. When the one particle is on the opposite side of the element from the node, the shape function ($S^k_{i,p}$) will approach zero, but its gradient ($\mathbf{G}^k_{i,p}$) will not. The first term is thus unstable.

There are two solutions to the USL dilemma. The first solution was given by Sulsky, Zhou, and Schreyer (1995). Their approach was to adopt a momentum based algorithm similar to the approach in the Appendix. It was not the use of momentum that improved the algorithm, but rather the way nodal velocities were calculated before updating particle stresses and strains. In their approach, referred to as "Modified Update Stress Last" or MUSL, the updated particle momenta are extrapolated to the grid a second time before calculating the nodal velocities (see Task 6c in the appendix). Tracing a node with a single particle, the nodal velocities used for strain rates become

$$\mathbf{v}_i = \mathbf{v}_p^k + \Delta t \sum_{i=1}^{n_n} \frac{f_i^{tot} S_{i,p}^k}{m_i^{Dk}} \tag{21}$$

Although some nodal masses in the denominator (m_i^{Dk}) may approach zero, whenever the mass is close the zero, there will be a corresponding $S_{i,p}^k$ identically close to zero to cancel it out. This approach greatly improves the stability of MPM. An alternate fix is to update strains *before* updating momentum [Bardenhagen (2002)]. In this approach, referred to as “Update Stress First” or USF, the nodal velocities (for a single-particle node) used for strain rates are simply

$$\mathbf{v}_i = \mathbf{v}_p^k \tag{22}$$

which clearly have no numerical difficulties.

MUSL and USF are nearly identical. MUSL finds velocities for momenta extrapolated at the end of an MPM step while USF finds velocities from the same momenta by extrapolating at the beginning of the next time step. The only mathematical difference between these two approaches is the shape functions used in the extrapolation. MUSL using $S_{i,p}^k$ while USF uses $S_{i,p}^{k+1}$. In numerous calculations, both give greatly improved energy calculations compared to USL methods. MUSL tends to slowly dissipate energy while USF tends to slowly increase in energy [Bardenhagen (2002)]. This observation leads to an obvious compromise which combines MUSL and USF by updating stresses and strains both before updating nodal momenta and after updating (and re-extrapolating) nodal momenta. In this approach, referred to as “Update Stress Averaged” or USAVG, the strain increment at each updating is divided by 2 to use the two methods equally. In numerous calculations, MPM results using USAVG conserves energy nearly exactly. Sample calculations comparing USAVG to USL, MUSL, and USF are given in the next section.

3 Results and Discussion

3.1 Opening Crack in Double Cantilever Beam

The top of Fig. 1 shows the results of a CRAMP calculation on a double cantilever beam specimen (DCB) with a crack half way through the specimen at the mid-plane. The sample was end-loaded at time zero and damped to have the results converge to the static solution. The simulation was continued until the stress stabilized. The numerical details are given in the figure caption. As expected, the crack opens, there are tensile bending stresses on the inner surfaces of the DCB arms and compression stresses on the outer surfaces.

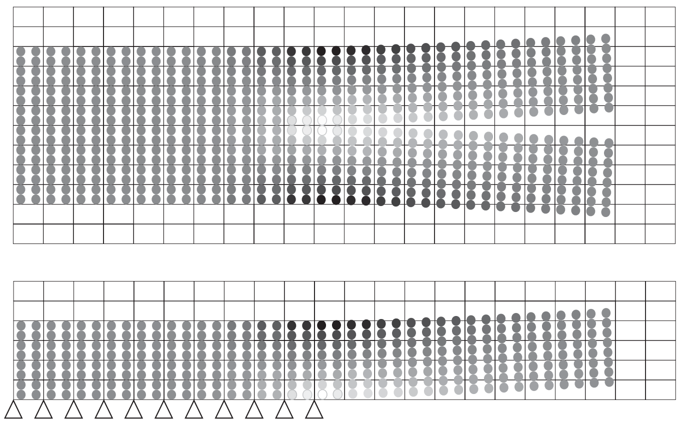
Figure 1 : CRAMP analysis (top) and conventional MPM analysis (bottom) of an end-loaded double cantilever beam. The shades of gray indicate magnitude of the tensile stress in the x direction (black compression to white tension, but the whitest particles are outlined for clarity). The material had properties $E = 0.1$ MPa, $\nu = 0.33$, and $\rho = 1.5$ g/cm^3. The full specimen was $100\times36\times1$ (in mm); the crack length was 50 mm. The end loads were 0.4 mN.

Although the above DCB calculation was a full calculation with an explicit crack, the same problem can be solved by symmetry without the need to model explicit cracks. The method is to analyze half the specimen and to fix the grid points on the left half of the lower edge of the specimen to have zero y-direction displacement (or zero y-direction velocity) throughout the analysis. The results, given in the bottom of Fig. 1, show that a symmetry analysis gives numerically identical results to the full analysis, thus confirming the crack algorithm correctly accounts for the presence of a crack. Of course, this result was a goal of CRAMP — to develop an algorithm that gives the “exact” MPM result in the presence of cracks. Here “exact” means that the MPM calculations for the top half of the full specimen are exactly the same as the calculations done when considering only half the specimen (bottom of Fig. 1). Similarly, the calculations in the bottom half of the full specimen are exactly the same as calculations that would be used in analysis of a lower-half specimen. The complication in a full specimen is that both halves need to use the mid-plane nodes for normal MPM calculations. This simultaneous use of the mid-plane nodes is accomplished naturally in the CRAMP algorithm by those nodes having two velocity fields.

An alternate approach to handling explicit cracks in MPM is to modify the calculations using node-visibility criteria. In node-visibility methods, a line is drawn from each particle to each node. If that line crosses a crack, than that node no longer influences the calculations for that particle. Node visibility [Belytschko, Lu, and Gu (1994)] and the related diffraction criteria [Organ, Fleming, and Belytschko (1996)] have been the most common choices for implementing cracks in EFG [Belytschko and Tabbara (1996)] and MLPG [Ching and Batra (2001), Batra and Batra (2002)] methods. Although node visibility can also implement cracks in MPM, it leads to less accurate results than the CRAMP method. The above definition of "exact" MPM could be rephrased as a "crack patch" test in which the results of any crack algorithm applied to a symmetric problem with an explicit crack are compared to a standard analysis with no crack algorithm that includes the crack by symmetry conditions alone. An algorithm passes the test if the results of the two analysis are numerically identical. The CRAMP method passes the "crack patch" test while MPM with node visibility does not. Similarly, because node visibility and diffraction criteria in EFG and MLPG modify the shape functions differently than when the crack is defined only by symmetry conditions, those methods also would not pass a "crack patch" test.

For an additional check, the MPM results were compared to static finite element analysis results (FEA). The FEA analysis used the same grid of rectangular, 4-noded elements that was used for the MPM calculations. Figure 2 plots the *x* direction stress along the mid-plane of the specimen. To find the MPM stresses at the mid-plane, the particle stresses were extrapolated to the grid by the same methods used to extrapolate momenta to the grid. The FEA and MPM results are very close. Although 4-node elements are not ideal for bending problems and the element size was large, the results show the accuracy of MPM to be similar to that of FEA and provide further evidence that CRAMP is getting the correct solution in the presence of cracks.

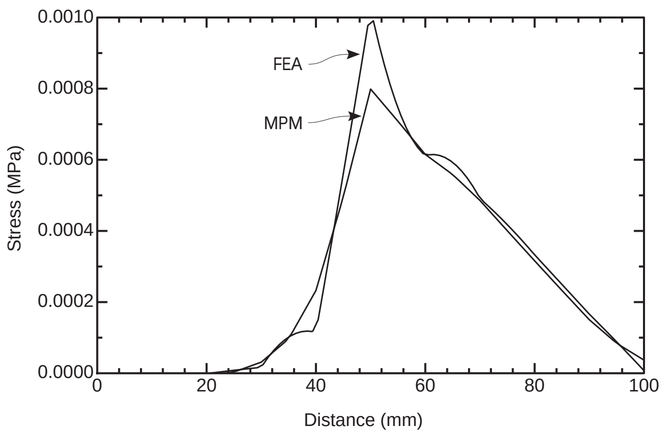

Figure 2 : Comparison of FEA results to CRAMP results. The plot is for σ_{xx} as a function of position along the mid-plane of the specimen. The peak stress is at the crack tip. The material properties are given in the caption of Fig. 1

3.2 Cracks Experiencing Contact

The DCB analysis was for an opening crack. The results in this section give a simulation when crack contact is important. The simulation is for two disks moving towards each other, contacting, and then bouncing apart. The disk on the left has a horizontal crack in the middle of the disk. The length of the crack is equal to one third the diameter of the disk. The disk on the right has the same size crack but oriented in the vertical direction. When the disks first make contact, the axial compression of the disk on the left causes mode I loading of the crack and the crack opens [Shetty, Rosenfield, and Duckworth (1987)]. The transverse compression to the disk on the right causes the cracks faces to contact and the contact algorithm keeps the surfaces from crossing. After the impact event, the two disks begin to vibrate and the cracks open and close. The contact algorithm keeps the solution proceeding correctly.

The plots in Fig. 3 show four snapshots of the solution. The cracks are indicated by a black line and the shades of gray for the material points indicate tensile stress in the *y* direction. Frame **a** shows the initial conditions with closed cracks and zero stress; the disks have initial velocities towards each other. Frame **b** shows a moment soon after contact. The diametrical compression has opened the crack in the left disk and white zones show crack tip stress concentrations in the *y*-direction normal stress.

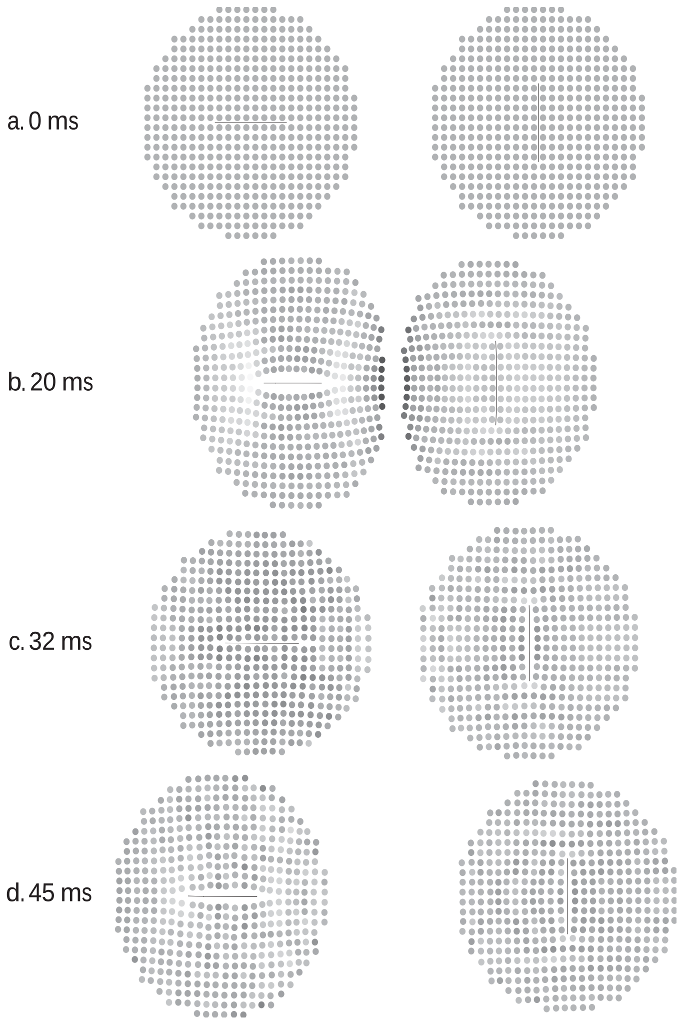

Figure 3 : Four snapshots for two disks with cracks colliding and separating. The shades of gray indicate y direction tensile stress (black minimum to white maximum) The material properties were $E = 0.1$ MPa, $\nu = 0.33$, and $\rho = 1.5$ g/cm^3. The disks were 30 mm in diameter and 1 mm thick. The cracks were centrally located and 10 mm long. The crack surfaces were frictionless. At the beginning the disks were moving towards each other, each with a speed of 1000 mm/sec.

The crack on the right has closed and the contact algorithm kept the surfaces from crossing each other. In frame **c**, the disk vibrations have caused the crack on the left to close and the contact algorithm prevents cross over. The crack on the right has opened slightly. By frame **d**, the crack on the left has opened again.

The contact algorithm handles crack surface contact well, but it was essential to implement the volume method to determine contact rather than rely on other methods such as relative velocity [Bardenhagen, Guilkey, Roessig, Brackbill, Witzel, and Foster (2001)] or normal traction methods [Bardenhagen, Brackbill, and Sulsky (2000)]. When other contact methods were used, the crack surfaces would think they were in contact long before visual evidence indicated there were actually in contact. For example, the contact between the two disks is handled by conventional MPM. In conventional MPM, two particles will interact (*i.e.*, be in contact) whenever they both interact with a particular node. The contact space between the disks in frame b is a typical example of MPM contact where the closest approach is determined by the mesh density. When analyzing internal cracks, it is important to have new contact methods that allow closer approach before numerical contact. The volumetric scheme used here worked well for allowing realistic contact. The critical volumes used for these soft disks were $V_{sep} = 0.9$ and $V_{contact} = 1.1$. These critical values can work for any materials, but stiffer materials might be handled more efficiently and more accurately by using critical values closer to one.

The volumetric method is robust for internal cracks, but has deficiencies for edge cracks. When there are edge cracks, the relative volume at a node might be less then one even when the cracks are in contact. The above algorithm might mistakenly consider such cracks as separated. The problem is that the undeformed volume is calculated from the element area while the true undeformed volume should account for the node being near the edge of a material. This problem did not affect the above edge-crack DCB results because the cracks were always separated. To better handle edge cracks in contact, enhanced volumetric methods are needed. One approximate approach is to normalize the nodal volume to the total number of particles interacting with that node. This approach improves the contact detection at edge nodes, but sometimes gives invalid contact detection at internal nodes. This problem will be the subject of future work.

3.3 Comparison to Experiment

This example considers a much larger problem (58017 material points), stiffer materials, and comparison to experiments. The configuration, illustrated Fig. 4 shows five transparent disks (poly methylmethacylate (PMMA)) constrained by a jig to remain planar and aligned. At time zero, the left side is impacted (at 6 m/sec) and high speed photography was used to record photoelastic fringes [Roessig and Foster (2001), Roessig (2001)]. To study crack effects, the central disk contained a crack aligned with the loading direction having a length equal to half the diameter of the disk. The top half of Fig. 4 shows experimental results for one particular time. The stress concentrations at the crack tip are evident by the increased number of fringes. The front of the stress wave has just reached the last disk.

The MPM simulations modeled the five disks by an approach similar to that described in Bardenhagen, Guilkey, Roessig, Brackbill, Witzel, and Foster (2001). The disk material was set to a high modulus material ($E = 174860$, $\nu = .214$, $\rho = 1.90$). A high modulus material was used instead of actual PMMA modulus because the analysis involved less total displacement. The simulations and experiments were compared by normalizing to the wave speeds in the different materials [Bardenhagen, Guilkey, Roessig, Brackbill, Witzel, and Foster (2001)]. Simulations with lower moduli that had larger displacements developed noise in regions where particles crossed element boundaries. The methods in Bardenhagen and Kober (2003) can solve some or all of such noise issues, but those methods were not available in the CRAMP code. The impactor was modeled as a material with much higher density ($E = 17486$, $\nu = 0.214$, $\rho = 190$) to emulate the impact event. The photoelasticity fringes were calculated by taking a periodic function of the principle stress difference. The equation used was

$$\cos\left(k_f\sqrt{\left(\sigma_{xx}-\sigma_{yy}\right)^2+4\tau_{xy}^2}\right) \tag{23}$$

where k_f is physically a fringe constant for the material. By comparison to experiments, k_f was determined to be 0.0889 (for stresses in MPa). There were 30 elements or 60 material points across the diameter of each disk. The simulation results in the bottom of Fig. 4 are for the simulation time closest to the experimental time. The predicted fringes agree well with experimental results. In particular the fringe patterns around the crack show that the CRAMP code is correctly modeling the presence of the crack.

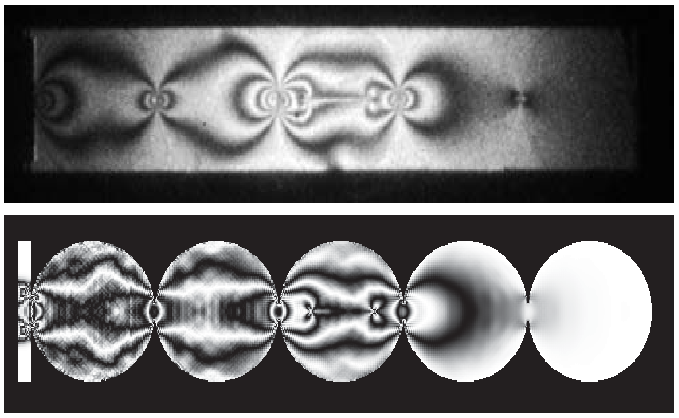

Figure 4 : Five poly methylmethacylate disks impact loaded from the left side. the central disk has a crack parallel to the loading direction of length equal to half the diameter of the disk. The top figure gives the experiment birefringence pattern at one particular time. The bottom figure gives the calculated birefringence pattern.

One might think that this problem is symmetric along the mid-plane or that the analysis could be done by considering half the sample with fixed displacements along the mid-plane except for no constraints on the crack surface. Examination of the full results, however, reveals the the crack surfaces experience contact as the stress waves pass by the crack. It was thus necessary to do a full analysis and use the crack contact methods to handle contact. An analysis of half the specimen develops non-physical crack surface displacements as the crack surface displacements extend past the mid-plane.

3.4 Energy Calculations

The final example is not a crack problem, but an examination of the energy results using the four methods for updating stresses - USL, MUSL, USF, and USAVG. The problem, illustrated in Fig. 5, is for transverse impact on a polymer specimen. The simulation mimics experimental results in Nairn (1989). The beam (of dimensions 88×6×12.95 mm with span of 68 mm) was Delrin polyoxymethylene polymer ($E = 2900$ MPa, $\nu = 0.33$, $\rho = 1.5$). The impactor was given a modulus 10 times higher, a higher density, and a thickness chosen to have its' total mass match the experimental impact mass of 367 g. The impact velocity was set to the experimental

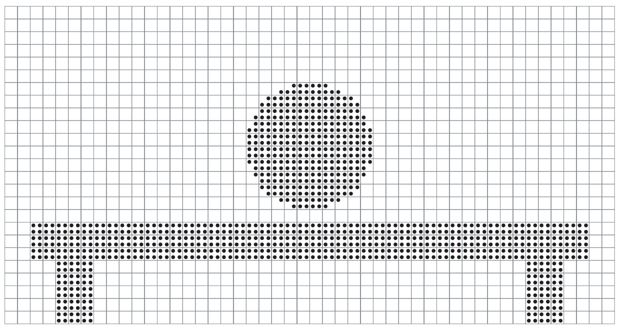

Figure 5 : Initial configuration and material point discretization for transverse impact of a disk on a polymer beam. The dimensions and material properties are given in text of the paper.

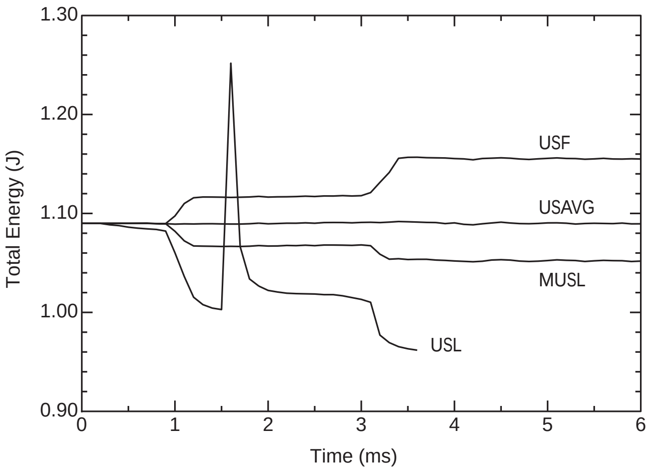

Figure 6 : Total energy (kinetic + strain energy) during impact of a disk on a polymer beam (see Fig. 5) by MPM using four different methods for updating stress and strain in each time step — USL, MUSL, USF, and USAVG. The methods are explained in the text of the paper.

result of 2.43 m/sec. No gravity was used and thus the sum of kinetic and strain energy should remain constant throughout the simulation.

Figure 6 shows total energy in the MPM simulation using each of the four stress updating methods. By tracking the kinetic energy in the impactor, the beam impact started at about 0.4 ms and continued until about 4.2 ms. The original MPM method, or USL, gave poor results. The simulation had to be stopped at about 3.6 ms because it became unstable and one of the material points left the grid. The three other methods gave stable results and better energy results. The MUSL approach slowly dissipated energy (with two step drops) while the USF approach slowly increased in energy (with two step increases). These observations are similar to results by these two methods in Bardenhagen (2002). The USAVG approach conserved energy. The total energy remained within $\pm 0.18\%$ of the average throughout the entire simulation. By the algorithm, USAVG is an average of the USF and MUSL methods, but it does not lead to results that are a simple average of the other methods. For example, because USF increased in energy more than MUSL decreased in the energy, their average increased in energy with time. The USAVG result, however, did not follow this average but rather gave nearly exact conservation of energy. Similar comparisons between USF, MUSL and USAVG were found in numerous other MPM results.

3.5 Algorithm Efficiency

Profiling of the CRAMP code showed that most additions due to crack effects add very little overhead to MPM code. The one exception is the need to examine particle-node pairs to determine the appropriate velocity field. This step involves determining whether or not a line from a particle to a node crosses a crack. It is important to implement this step as efficiently as possible. The Appendix gives one algorithm that works well. There are two key components. First, the line crossing algorithm should be efficient. Second, great improvement is possible by screening particle-node pairs and skipping the check entirely for those that can not cross a crack. A simple way to do this screening, which led to an order of magnitude efficiency increase in this step, is to keep track of the extent of each crack or the minimum and maximum x and y values for all end points of the segments of the crack. Then, when checking any particle node pair, the line crossing code can be skipped whenever the rectangle defined by the particle and node points has no intersection with the extent of the crack. With optimal line-crossing algorithm combined with screening using crack extents, the fracture calculations in this paper averaged only 10% longer than the comparable MPM calculation with no cracks.

4 Conclusions

This paper describes a CRAMP algorithm which extends MPM to naturally handle explicit cracks. When the fracture code is written efficiently, especially the new line crossing section, the CRAMP code achieves crack calculations with very little extra cost in calculation time. The results show that CRAMP gets the correct MPM solutions and comparisons to both FEA and experiments show that it gets good results for crack problems. To account for crack surface contact, there are checks for contact and both stick and sliding with friction can be handled. The method to detect crack contact is robust for internal cracks but may need some adjustment for problems involving edge cracks. Crack propagation is easily handled by simply moving the crack or adding crack segments at any time step. The important problem that remains is the calculation of crack tip or fracture parameters followed by prediction of crack propagation. This problem will be the subject of future work. Although this paper describes a 2D algorithm, extension to 3D is possible. In 3D, the crack line needs to be replaced by a crack surface described by planar elements instead line segments. The line crossing algorithm needs to be replaced by a 3D area crossing algorithm.

Acknowledgement: his work was support by a grant from the Department of Energy DE-FG03-02ER45914 and by the University of Utah Center for the Simulation of Accidental Fires and Explosions (C-SAFE), funded by the Department of Energy, Lawrence Livermore National Laboratory, under Subcontract B341493.

References

Atluri, S. N.; Shen, S. (2002): The Meshless Local Petrov-Galerkin (MLPG) method: A Simple & Less-Costly Alternative to the Finite Element and Boundary Element Methods. *CMES: Computer Modeling in Engineering & Sciences*, vol. 3, pp. 11–52.

Atluri, S. N.; Shen, S. P. (2002): *The Meshless Local Petrov-Galerkin (MLPG) Method*. Tech. Science Press.

Atluri, S. N.; Zhu, T. (1998): A New Meshless Local Petrov-Galerkin (MLPG) Approach in Computational Mechanics. *Computational Mechanics*, vol. 22, pp. 117–127.

Bardenhagen, S. G. (2002): Energy Conservation Error in the Material Point Method. *J. Comp. Phys.*, vol. 180, pp. 383–403.

Bardenhagen, S. G.; Brackbill, J. U.; Sulsky, D. (2000): The Material Point Method for Granular Materials. *Computer Methods in Applied Mechanics and Engineering*, vol. 187, pp. 529–541.

Bardenhagen, S. G.; Guilkey, J. E.; Roessig, K. M.; Brackbill, J. U.; Witzel, W. M.; Foster, J. C. (2001): An Improved Contact Algorithm for the Material Point Method and Application to Stress Propagation in Granular Material. *CMES: Computer Modeling in Engineering & Sciences*, vol. 2, pp. 509–522.

Bardenhagen, S. G.; Kober, E. M. (2003): The Generalized Interpolation Material Point Method. *CMES: Computer Modeling in Engineering & Sciences*, submitted.

Batra, R. C.; Batra, H.-K. (2002): Analysis of Elastodynamic Deformations Near a Crack/Notch Tip by the Meshless Local Petrov-Galerkin (MLPG) Method. *CMES: Computer Modeling in Engineering & Sciences*, vol. 3, pp. 717–730.

Belytschko, T.; Lu, Y. Y.; Gu, L. (1994): Element-Free Galerkin Methods. *Int. J. Num. Meth. Engrg.*, vol. 37, pp. 229–256.

Belytschko, T.; Tabbara, M. (1996): Dynamic Fracture Using Element-Free Galerkin Methods. *Int. J. Numer. Methods in Eng.*, vol. 39, pp. 923–938.

Ching, H.-K.; Batra, R. C. (2001): Determination of Crack Tip Fields in Linear Elastostatics by the Meshless Local Petrov-Galerkin (MLPG) Method. *CMES: Computer Modeling in Engineering & Sciences*, vol. 2, pp. 273–289.

Guilkey, J. E.; Weiss, J. A. (2002): Implicit Time Integration for the Material Point Method: Quantitative and Algorithmic comparisons with the Finite Element Method. *Int J. Numer. Methods in Eng, vol 57, pp 1323-1338*.

Nairn, J. A. (1989): The Measurement of Polymer Viscoelastic Response During an Impact Experiment. *Polym. Eng. & Sci.*, vol. 29, pp. 654–661.

Organ, D. J.; Fleming, M.; Belytschko, T. (1996): Continuous Meshless Approximations for Nonconvex Bodies by Diffraction and Transparency. *Computational Mechanics*, vol. 18, pp. 225–235.

Parker, S. G. (2002): A Component-based Architecture for Parallel Multi-Physics PDE Simulation. In *International Conference on Computational Science (ICCS2002) Workshop on PDE Software*.

Roessig, K. M. (2001): 2001. Personal communication.

Roessig, K. M.; Foster, J. C. (2001): Experimental Simulations of Dynamic Stress Bridging in Plastic Bonded Explosives. APS Shock Compression of Condensed Matter Meeting, June 24-29, Atlanta Georgia, vol 620, pp 829-832.

Shetty, D. K.; Rosenfield, A. R.; Duckworth, W. H. (1987): Mixed-Mode Fracture in Biaxial Stress State: Application of the Diametral-Compression (Brazilian Disk) Test. *Eng. Fract. Mech.*, vol. 26, pp. 825–839.

Sulsky, D.; Chen, Z.; Schreyer, H. L. (1994): A Particle Method for History-Dependent Materials. *Comput. Methods Appl. Mech. Engrg.*, vol. 118, pp. 179–186.

Sulsky, D.; Schreyer, H. K. (1996): Axisymnmetric Form of the Material Point Method with Applications to Upsetting and Taylor Impact Problems. *Comput. Methods. Appl. Mech. Engrg*, vol. 139, pp. 409–429.

Sulsky, D.; Zhou, S.-J.; Schreyer, H. L. (1995): Application of a Particle-in-Cell Method to Solid Mechanics. *Comput. Phys. Commun.*, vol. 87, pp. 236–252.

Tan, H.; Nairn, J. A. (2002): Hierarchical Adaptive Material Point Method in Dynamic Energy Release Rate Calculations. *Comput. Meths. Appl. Mech. Engrg.*, vol. 191, pp. 2095–2109.

Zhou, S.-J. (1998): *The Numerical Prediction of Material Failure Based on the Material Point Method*. PhD thesis, Department of Mechanical Engineering, University of New Mexico, 1998.

Appendix A: CRAMP Algorithm

This appendix summarizes the CRAMP algorithm to include explicit cracks and some revisions to MPM to update stresses and strains by a method that minimizes dissipation of energy. The algorithm is for 2D calculations, but most vector results translate easily into a 3D algorithm. In the following, subscript p always refers to a particle property, subscript i always refers to a nodal property, bold face refers to a vector or a tensor, plain text refers to a scalar, and an over bar indicates a specific quantity (actual quantity divided by particle density). The following single algorithm includes four options for updating particle stresses and strains denoted as USL, MUSL, USF, and USAVG; these methods were described in the text of the paper.

Task 0: At the beginning of the $(k+1)^{th}$ MPM analysis step, all information for the numerical solution resides on the particles. The information relevant to the follow algorithm are particle position ($\mathbf{x}_p^k$), velocity ($\mathbf{v}_p^k$), specific stress ($\overline{\sigma}_p^k$), strain (ε_p^k), mass (m_p), density (ρ_p), thickness (t_p for 2D calculations), and any other material properties needed for constitutive law calculations. The shape functions and shape function gradients are defined from the current particle positions as

$$S_{i,p}^k = N_i(\mathbf{x}_p^k) \text{ and } \mathbf{G}_{i,p}^k = \nabla N_i(\mathbf{x}_p^k) \tag{24}$$

where $N_i(\mathbf{x})$ are the basic element shape functions. In practice, the shape functions and their gradients are not calculated and stored (which would require a large array), but anytime they are calculated during the algorithm, they refer to the particle position at the beginning of the $(k+1)^{th}$ step. The particle volume, which is an area in 2D calculations, is found from

$$V_p^k = (1+\varepsilon_{p,xx}^k)(1+\varepsilon_{p,yy}^k)\frac{m_p}{\rho_p t_p} \tag{25}$$

Task 1: Loop over the particles. For each particle, draw a line from that particle to each node in the element containing that particle and use a line-crossing algorithm to calculate $\nu(p,i) = 0$, 1, or 2 which determines the velocity field for particle p at node i. An optimized line-crossing algorithm is given below. Field 0 means the drawn line does not cross a crack; field 1 means the line crosses a crack and the particle is above the crack relative to the node; field 2 means the line crosses a crack and the particle is below the crack relative to the node. If $\nu(p,i) = 1$ or 2 and node i does not have that velocity field, then allocate a new velocity field for that node. Only nodes near cracks will have multiple velocity fields and except in unusual conditions, no node will ever have more than two velocity fields — one for particles on one side of the crack and one for particles on the other side.

In the same loop, calculate nodal momenta and lumped masses for all needed velocity fields using

$$\begin{aligned} \mathbf{p}_{i,j}^k &= \sum_{p=1}^{n_p} m_p \mathbf{v}_p^k S_{i,p}^k \delta_{j,\nu(p,i)} \quad j = 0,1,2 \\ m_{i,j}^{Dk} &= \sum_{p=1}^{n_p} m_p S_{i,p}^k \delta_{j,\nu(p,i)} \quad j = 0,1,2 \end{aligned} \tag{26}$$

where $\delta_{i,\nu(p,i)}$ is the Kronecker delta function. For use in the contact algorithm, calculate *total* nodal volumes using

$$V_i^k = \sum_{p=1}^{n_p} V_p^k S_{i,p}^k \tag{27}$$

Task 2: Apply any boundary conditions to nodal momenta calculated in the previous task and check for crack contact by the procedure listed below. For USF or USAVG methods only, update particle stresses and strains by the procedure listed below. The new particle stresses and strains are denoted $\overline{\sigma}_p^{k'}$ and $\varepsilon_p^{k'}$. For USL and MUSL methods, the stresses and strains are not updated and thus set $\overline{\sigma}_p^{k'} = \overline{\sigma}_p^k$ and $\varepsilon_p^{k'} = \varepsilon_p^k$.

Task 3: Loop over the particles again. For each particle calculate internal, external, and total forces for all velocity fields in use. The equations [Sulsky, Chen, and Schreyer (1994), Sulsky, Zhou, and Schreyer (1995)] now adjusted for multiple velocity fields are (each for $j = 0,\ 1,\ 2$):

$$\begin{aligned} f_{i,j}^{int} &= \sum_{p=1}^{n_p} \left(-m_p \overline{\sigma}_p^{k'} \cdot \mathbf{G}_{i,p}^k + m_p \overline{\mathbf{b}_\mathbf{p}} S_{i,p}^k \right) \delta_{j,\nu(p,i)} \\ f_{i,j}^{ext} &= \sum_{p=1}^{n_p} f_p^k S_{i,p}^k \delta_{j,\nu(p,i)} \\ f_{i,j}^{tot} &= f_{i,j}^{int} + f_{i,j}^{ext} - \kappa \mathbf{p}_{i,j}^k \end{aligned} \tag{28}$$

where $\overline{\mathbf{b}_\mathbf{p}}$ are any specific body forces on a particle, such as gravity, f_p^k are forces applied directly to particles, and κ is a damping constant which can be used to damp the analysis. If any nodes have fixed displacement in x or y directions, set the total forces for all velocity fields at that node to zero to prevent acceleration in the next task.

Task 4: Update nodal point momenta

$$\mathbf{p}_{i,j}^{k'} = \mathbf{p}_{i,j}^k + \Delta t\, f_{i,j}^{tot} \qquad j = 0,1,2 \tag{29}$$

Adjust any momenta for crack contact effects by the procedure listed below.

Task 5: Loop over the particles to update particle position and velocity using Sulsky, Zhou, and Schreyer (1995):

$$\begin{aligned} \mathbf{x}_p^{k+1} &= \mathbf{x}_p^k + \Delta t \sum_{i=1}^{n_n} \frac{\mathbf{p}_{i,\nu(p,i)}^{k'} S_{i,p}^k}{m_{i,\nu(p,i)}^{Dk}} \\ \mathbf{v}_p^{k+1} &= \mathbf{v}_p^k + \Delta t \sum_{i=1}^{n_n} \frac{f_{i,\nu(p,i)}^{tot} S_{i,p}^k}{m_{i,\nu(p,i)}^{Dk}} \end{aligned} \tag{30}$$

Notice that the momentum, force, and mass used is this update come from the specific velocity field for each particle/node pair determined by the line crossing results for $\nu(p,i)$ evaluated in Task 1.

Task 6: a. For USL method only, update particle stresses and strains as described below to find $\overline{\sigma}_p^{k+1}$ and ε_p^{k+1} and set $\mathbf{p}_{i,j}^{k''} = \mathbf{p}_{i,j}^{k'}$ for later use.

b. For USF only, the stress and strain update was done before and thus set $\overline{\sigma}_p^{k+1} = \overline{\sigma}_p^{k'}$, $\varepsilon_p^{k+1} = \varepsilon_p^{k'}$. Also set $\mathbf{p}_{i,j}^{k''} = \mathbf{p}_{i,j}^{k'}$ for later use.

c. For MUSL or USAVG only, extrapolate the new particle velocities to the grid to get a revised set of nodal momenta using:

$$\mathbf{p}_{i,j}^{k''} = \sum_{p=1}^{n_p} m_p \mathbf{v}_p^{k+1} S_{i,p}^k \delta_{j,\nu(p,i)} \qquad j = 0,1,2 \tag{31}$$

Adjust any momenta for crack contact effects by the procedure listed below. Note that this extrapolation uses the new particle velocities but the shape functions and line crossing results from the original particle positions. This approach was found to give better results than one using updated information. Update particle stresses and strains using the new nodal momenta by the procedure listed below to find $\overline{\sigma}_p^{k+1}$ and ε_p^{k+1}.

Task 7: Calculate center of mass velocity for each node with multiple velocity fields:

$$\mathbf{v}_{i,cm}^k = \frac{\sum_{j=1}^3 \mathbf{p}_{i,j}^{k''} \varphi_{i,j}}{\sum_{j=1}^3 m_{i,j}^{Dk} \varphi_{i,j}} \tag{32}$$

where $\varphi_{i,j} = 1$ if there exists a p such that $\delta_{j,\nu(p,i)} = 1$, otherwise $\varphi_{i,j} = 0$. In other words, $\varphi_{i,j}$ is 1 or 0 depending on whether or not velocity field j is present for node i. Use this nodal velocity field to update the positions of all line segments that define the cracks by standard position updating methods.

Task 8: All information is now on the particles and if needed the cracks have translated. All nodal and mesh information can be discarded and then return to Task 0 to begin the next MPM calculation step.

Line Crossing Algorithm: The most time consuming, new calculation required for CRAMP is the line-crossing calculation in Task 2. It is important for this calculation to be optimal and precise. The only numerical difficulty occurs when a node lies very close to the crack path. In some line-crossing algorithms, numerical round off in this situation could result in two particles on the same side of the crack being labeled as being on opposite sides. The complementary problem of when a material point lines very close to a crack path never occurs because the contact methods keep particles from reaching the crack path.

A line-crossing algorithm in 2D based on signed areas of certain triangles solved the problem of nodes on cracks. For any three points, $\mathbf{x_1}$, $\mathbf{x_2}$, and $\mathbf{x_3}$, the signed area of the triangle with those vertexes is given by

$$\text{Area} = x_1(y_2-y_3)+x_2(y_3-y_1)+x_3(y_1-y_2) \tag{33}$$

This area is positive if the path from $\mathbf{x_1}$ to $\mathbf{x_3}$ is counter clockwise, negative if it is clockwise, and zero if the points are collinear. Using this signed area, the algorithm is as follows:

Subtask 1: Before doing any calculations, determine if the rectangle defined by the particle ($\mathbf{x_1} = \mathbf{x_p}$) and the node ($\mathbf{x_2} = \mathbf{x_n}$) under consideration intersects the extent of the segment endpoints in the crack. If it does not, the line does not cross the crack and rest of the algorithm can be skipped for that crack.

Subtask 2: For each crack segment with endpoints $\mathbf{x_3}$ and $\mathbf{x_4}$, calculate the sign of the areas of triangles (123), (124), (341), and (342) denoted as "+", "−" or "0".

Subtask 3: The particle is above the crack if the signs are $(-++-)$, $(-++0)$, $(0++0)$, or $(-0+0)$. The first case is the most common; the other three correspond to the node being on the crack segment, on the start point of the crack segment, or on the end point of the crack segment, respectively. The cases where the material point is on the crack segment can be ignored. Similarly, the particle is below the crack if the signs are $(+--+)$, $(+--0)$, $(0--0)$, or $(+0-0)$. All other combinations of signs indicate the line does not cross the line segment. In practice, many signed area calculations can be skipped. For example is the signs of (123) and (123) are $(++)$, there is no need to evaluate the signs of (341) and (342) because the line can not cross the segment.

Subtask 4: One complication is that a given $\mathbf{x_p}$ to $\mathbf{x_n}$ line might cross more than one segment in a single crack. In this situation, the crossing is ignored unless there are an odd number of crossings. To include this possibility, the previous two steps must check all segments in a crack before deciding if there is a crossing, but if Subtask 1 finds no intersection, the check for all segments in that crack can be skipped.

Contact Algorithm: Any time the nodal momenta are calculated (*i.e.*, in Tasks 2, 4, and 6c), the above algorithm must check cracks for contact. If contact is found, adjust nodal momenta for all velocity fields at nodes experiencing contact. If any momenta change in Task 4, back calculate the corresponding total nodal forces to match the new momenta. This recalculation is needed to insure particle velocities update correctly in Task 5. The algorithms for deciding on contact and for adjusting momenta are given in the *MPM With Explicit Cracks* section.

Updating Stresses and Strains: Because the above algorithm includes four different methods for updating stresses and strains (USL, MUSL, USF, and USAVG), it includes several locations for the updates (Tasks 2, 6a, and 6c). All stress and strain updates use the following procedure:

Subtask 1: Calculate nodal point velocities using the current nodal point momenta

$$\mathbf{v}_{i,j} = \frac{\mathbf{p}_{i,j}^{k*}}{m_{i,j}^{Dk*}} \qquad j = 0,\ 1,\ 2\ (\text{as needed}) \tag{34}$$

where $k*$ means the most recently calculated momenta and $\mathbf{v}_{i,j}$ is only calculated for active velocity fields ($\varphi_{i,j} = 1$).

Subtask 2: Loop over particles and find the strain increment for the current step from

$$\Delta\varepsilon_p^k = \Delta t^* \frac{\nabla\mathbf{v}_p^k + \nabla\mathbf{v}_p^{k\,T}}{2} \tag{35}$$

In two dimensions, the velocity gradient at the material points involves four terms calculated from the outer products

$$\nabla\mathbf{v}_p^k = \sum_{i=1}^{n_n} \mathbf{v}_{i,\nu(p,i)} \otimes \mathbf{G}_{i,p}^k \tag{36}$$

Here Δt^* is the time step for USL, MUSL, and USF, but half the time step for USAVG. Notice that USAVG updates stresses and strains twice during each step (in Tasks 2 and 6c); each update gives half the update for the current step.

Subtask 3: Input the strain increment ($\Delta\varepsilon_p^k$, and the individual components of the velocity gradient, $\nabla\mathbf{v}_p$, if needed) into a material constitutive law and update the particle stresses. Any constitutive law may be used.