Modeling Multiple Crack Propagation in the Material Point Method by J-Integral Methods Accounting for other Cracks Intersecting the J Contour

John A. Nairn

Wood Science and Engineering, Oregon State University, Corvallis, OR 97330, USA
Email: John.Nairn@oregonstate.edu

Yamina E. Aimene

L3MA UR4_1, UFR STE, Université des Antilles, Schoelcher 97233, FRANCE
Email: Yamina.Aimene@univ-antilles.fr

Abstract

Numerical crack propagation modeling of multiple, explicit cracks requires methods that can resolve interacting, and potentially intersecting, cracks as well as methods to calculate crack-tip parameters with sufficient accuracy for predicting when and where cracks propagate. These problems were solved using the material point method (MPM). First, the MPM method for modeling a single explicit crack known as CRAMP method was extended to account for two interacting explicit cracks by tracking four instead of original two crack velocity fields needed to resolve one explicit crack. Second, crack propagation and propagation direction were found using crack-tip J integral with partitioning into mode I and mode II stress intensity factors. J calculations, however, must be augmented whenever other cracks intersect the J contour for a propagating crack. If the intersecting crack's tip is inside the J contour, the contour must be adapted to avoid that tip. A robust J calculation algorithm with intersecting-crack corrections is provided. Several examples show the new corrections are accurate. Selected crack propagation examples show the new methods have sufficient accuracy for interacting crack propagation calculations without any need to remesh or highly refine crack-tip regions.

Keywords: A. Material Point Method; B. J Integral; C. Numerical Crack Propagation; D. Intersecting Cracks

1. Introduction

Many real-world phenomena and industrial applications involve multiple cracks. Some examples include interactions between intra- and inter-laminar cracks in aerospace [1, 2, 3] or wood composites [4], interactions between fractures in geomechanics, and multiple cracking in biaxially-loaded coatings [5]. The goal of this work was to derive a material point method (MPM) approach to modeling multiple propagating explicit cracks including situations with closely-spaced cracks (CSC), which includes intersecting cracks.

The conditions required for crack propagation, especially in elastic materials, are predicted well by fracture mechanics [6, 7, 8]. Although the most direct numerical approach to using fracture mechanics

Nomenclature

$c_{i,j}$, $v(p,i)$	Crack velocity field j on node i and for material point p on node i
m_i, $\boldsymbol{p}_i$, $\boldsymbol{v}_i$, $\boldsymbol{f}_i$	Nodal mass, momentum, velocity, and force
$\boldsymbol{x}_p$, $\boldsymbol{v}_p$, $\boldsymbol{a}_p$	Material point position, velocity, and acceleration
S_{ip}, $\boldsymbol{G}_{ip}$, S^*_{is}	GIMP shape functions, their gradients, and renormalized for crack surface position s
$N_i(\boldsymbol{x})$, $\nabla N_i(\boldsymbol{x})$	Grid shape functions and their gradients
m_p, $\boldsymbol{b}_p$, $\boldsymbol{T}_p$	material point mass, body force, and traction
$\boldsymbol{a}_{g\to p}$, $\tilde{\boldsymbol{v}}_{g\to p}$	Acceleration and velocity extrapolated from grid to particle p
$\boldsymbol{a}_{g\to s}$, $\tilde{\boldsymbol{v}}_{g\to s}$	Acceleration and velocity extrapolated from grid to crack surface position s
$\boldsymbol{F}^{(n)}_{i,j}$, $\boldsymbol{F}^{(n)}_{s}$	Crack surface tractions on node i and on crack surface position s
W, K	Strain energy and kinetic energy
$\boldsymbol{u}$, $\boldsymbol{\sigma}$, $\hat{\boldsymbol{n}}$	Displacement, stress, and unit normal vector
J, J_k, G	J integral, J integral components and energy release rate
dJ_k	Increment of J_k for one segment of intersecting crack
Γ_i	Partial J integral along one section of the J contour
J_T	Contribution of crack tractions to J integral
P	Pressure applied to crack surfaces
K_I, K_{II}	Mode I and II stress intensity factors
E, ν, ρ	Material modulus, Poisson's ratio, and density.

is to work with explicit cracks, that approach faces two challenging tasks — 1. numerically resolving multiple explicit cracks; 2. predicting when and where each crack propagates. Task 1 can be solved in finite element analysis (FEA) by creating a mesh that follows initial crack geometry. Because propagating such meshed cracks is constrained by mesh lines, however, a better approach is to use off-mesh methods. In FEA, the extended finite element method (XFEM) represents off-mesh cracks by adding enriched shape functions [9, 10]. In the material point method (MPM), the CRAMP method (for CRAcks in MPM) represents cracks with a collection of massless points that are independent of MPM's background grid [11].

Task 2 can be solved by using crack-tip parameters to predict both propagation and direction of that propagation. Stress-state parameters are typically based on fracture mechanics parameters. For example, FEA can find mode I and mode II stress intensity factors of explicit cracks using virtual crack closure methods (VCCT) [12, 13]. Although VCCT, often provides accurate results even on coarse meshes, it requires cracks on mesh lines and sufficiently accurate crack-tip stresses. Consequently, using VCCT for crack propagation may need both remeshing and adaptive refinement of crack-tip regions. A better approach for propagating off-mesh cracks is to use domain methods such as to find J integral by integrating a contour around the crack tip [14]. J integral methods need enough resolution for accurate stresses on the contour, but do not rely as much as VCCT on highly-resolved crack tip stresses. Ideally, J methods can be accurate enough to model crack propagation without remeshing or refinement. XFEM adopts this approach through domain calculations [15], which are similar to J integral methods, to model crack propagation [10, 16]. MPM can model 2D and 3D crack propagation with J integral methods [17, 18, 19, 20].

This paper describes a new MPM approach to modeling multiple explicit cracks by extending CRAMP [11] to handle multiple cracks and revising J integral methods to handle "J Contours Intersected by other Cracks" — abbreviated here as JCICs. In original CRAMP [11], each node in MPM's background

grid is allowed to have two crack velocity fields corresponding to particles on the same side of the crack as the node or particles on the opposite side of the crack. This algorithm is only valid when no node interacts with particles on opposite sides of more than one crack (*i.e.*, no node "sees" more than one crack). The "Task 1" section explains CRAMP extensions to resolve two closely-spaced cracks (CSC), which corresponds to either two intersecting cracks or two cracks where any portions of their surfaces are separated by less than the extent of shape functions being used (typically two or three cells in MPM's background grid). In brief, two CSCs are modeled by allowing each node that "sees" both cracks to have up to four velocity fields corresponding to particles on the same side of both cracks as the node, on the opposite side of one crack or the other, or on the opposite sides of both cracks. This extension was implicitly referenced in prior publications [21, 22] and is available on GitHib [23]. This paper elaborates on the approach and the Appendix provides a detailed multi-crack CRAMP algorithm including the most recent improvements.

The "Task 2" section explains fracture mechanics methods for predicting crack propagation using J integral calculations [14]. J integral values in MPM can be calculated by integrating along a contour along grid lines surrounding the crack tip [17, 18, 19, 20]. As long as a problem contains no CSCs, standard J integral methods remain valid even with multiple cracks. But when a problem starts with CSCs or when propagation causes CSCs, the J contours will inevitably be intersected by other cracks causing JCIC configurations. The solution to task 2 here was to revise J integral methods to remain valid when JCICs occur. The new methods were validated as accurate. Thus, crack propagation in any direction and including crack interactions and intersections can proceed without needing remeshing or refinement. Some demonstration examples for propagating and interacting cracks are given.

Another approach to extending CRAMP for multiple cracks by Adibaskoro *et al.* [24] used many "grids" rather then the maximum of four velocity fields implemented here. The authors predicted crack growth using crack-tip stress magnitude [24] rather than fracture mechanics used here. That approach caused mesh dependence (because crack-tip stress magnitude changes with resolution) and produced chaotic crack propagation. Fan *et al.* [25] used CRAMP style for describing explicit cracks but then partitioned material points into domains on different sides of cracks. Once material points are partitioned, the role of cracks was changed to using MPM contact methods [26]. These authors predicted crack growth using energy release derived from a crack-tip domain integral [15] but did not explain how to correct that calculation when a second crack intersects that domain. The J integral approach used here avoids the problem of relying on crack tip stresses and solves the issue of cracks intersection other crack tip domains.

Because modeling of explicit cracks is challenging, others have proposed methods that can potentially handle real-world, interacting crack problems without the overhead of discretizing explicit cracks. Some examples include building failure into material constitutive models [27, 28], using smeared cracks or damage mechanics [29, 30, 31, 32, 33], variational phase-field fracture mechanics [34, 35, 36, 37], and field gradient methods [38]. The question remains — is multiple, explicit crack modeling worthwhile? We claim the answer is "yes" for two reasons. First, when a problem has a small number of interacting cracks and is 2D (extending J integral to 3D is challenging), modeling of explicit cracks using fracture mechanics is likely realistic modeling and a justifiable approach. Second, an explicit fracture mechanics tool would be a valuable tool for validating alternative methods that avoid explicit cracks. Although alternate methods often claim they are equivalent to using fracture mechanics, few interacting crack problems are available to validate those claims. Side-by-side numerical simulations using these new explicit crack methods *vs.* any alternate failure model could be used to justify the use of that alternate method. Such justification could provide confidence in using that method for problems where explicit crack calculations are not viable.

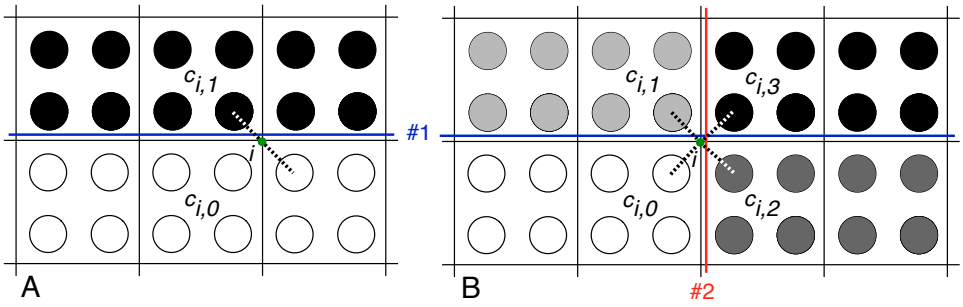

Figure 1: Crack velocity fields illustrated in 2D. A. Nodes near a single crack, labeled #1, may have two crack velocity fields. B. Nodes near two interacting cracks, labeled #1 and #2, may have up to four crack velocity fields. Particle shading indicates which particles extrapolate to specific crack velocity fields, $c_{i,j}$, on node i for j from 0 to 3. Dashed lines from node i to selected particles show example ray tracing lines used to determine crack velocity fields.

2. Numerical Methods

2.1. Task 1: Modeling Closely-Spaced Cracks (CSCs)

Problems with multiple isolated cracks can be modeled with original CRAMP as long as resolution is high enough to keep cracks separated on the background grid. But high resolution is "expensive" and no resolution can separate intersecting cracks. The enhanced CRAMP described here can model two CSCs including intersecting CSCs. The basic principle is to allow each node, i, to have up to four *crack velocity fields*, $c_{i,j}$ for $j = 0$ to 3. Each MPM time step starts by calculating $v(p,i) = 0$ to 3, to define crack velocity field on node i when interacting with particle p. Explicit cracks are defined by a collection of massless particles that define the crack and can track its surfaces. In 2D, crack particles are connected by line segments along the crack [11]. In 3D, crack particles are connected into triangular patches to define the crack surface [20, 39].

The $v(p,i)$ velocity fields are determined by tracing a line from particle p to each node i with non-zero shape function and using line-crossing (2D) or surface-crossing (3D) methods to see if it crosses zero cracks (field $j = 0$), one crack ($j = 1$ or 2) or two cracks ($j = 3$). Figure 1 illustrates: A. two velocity fields for node i near a single crack; B. four possible velocity fields for node i near two intersecting CSCs. An algorithm for finding $v(p,i)$ is given in the appendix, along with comments on performance and parallelization. The algorithm is limited to two cracks (or four crack velocity fields) on each node. Problems with nodes that see three or more cracks can either increase resolution until such nodes are eliminated or choose an approximation based on analyzing just two of the cracks that see that node. Because propagating cracks tend to avoid other cracks, three-crack nodes are rare in simulations unless they are present prior to any propagation. One approximation to handling three-node cracks is given in the appendix. None of the examples in this paper needed that approximation.

2.1.1. MPM With Multiple Crack Velocity Fields

Given $v(p,i)$, remaining MPM tasks are similar to standard MPM [40, 41] except that they must solve equations in each crack velocity field, resolve crack contact situations at nodes with more than one crack velocity field, and update crack position. These steps are analogous to multimaterial MPM methods [26, 42] except that a particle's static material type is replaced by a particle's dynamically-

determined crack side using $v(p,i)$. Revised MPM equations for each crack velocity field become:

$$\frac{d\boldsymbol{p}_{i,j}^{(n)}}{dt} = \boldsymbol{f}_{i,j}^{(n)} + \boldsymbol{f}_{i,j,T}^{(n)} \tag{1}$$

where

$$\begin{aligned}
\boldsymbol{p}_{i,j}^{(n)} &= \sum_p \boldsymbol{p}_p^{(n)} S_{ip}^{(n)} \delta_{jv(p,i)} \\
\boldsymbol{f}_{i,j}^{(n)} &= \sum_p \left(-m_p \frac{\boldsymbol{\tau}_p^{(n)} \cdot \boldsymbol{G}_{ip}^{(n)}}{\rho_0} + m_p S_{ip}^{(n)} \boldsymbol{b}_p \right) \delta_{jv(p,i)} \\
\boldsymbol{f}_{i,j,T}^{(n)} &= \delta_{jv(p,i)} \int_{S_T} N_i(\boldsymbol{x}) \boldsymbol{T}_p \, dS
\end{aligned}$$

where $\boldsymbol{p}$ is momentum, $\boldsymbol{f}$ is force, $\boldsymbol{f}_T$ is force due to particle traction $\boldsymbol{T}_p$, m_p is particle mass, $\boldsymbol{\tau}$ is Kirchhoff stress, $\boldsymbol{b}$ is body force, $N_i(\boldsymbol{x})$ are grid shape functions, and S_{ip} and $\boldsymbol{G}_{ip}$ are generalized interpolation MPM shape functions (GIMP) and their gradients [40]:

$$S_{ip}^{(n)} = \int_{\Omega_p} N_i(\boldsymbol{x}_p^{(n)}) \, d\boldsymbol{x} \quad \text{and} \quad \boldsymbol{G}_{ip}^{(n)} = \int_{\Omega_p} \nabla N_i(\boldsymbol{x}_p^{(n)}) \, d\boldsymbol{x} \tag{2}$$

These are found by integrating grid shape functions and their gradients over the particle's domain Ω_p. Subscripts (i,j) indicate a property for crack velocity field j on node i; subscript p indicates a particle property; superscript (n) indicates these quantities refer to time step n. Kronecker deltas, $\delta_{jv(p,i)}$, insure that crack velocity field j only represents particles that use that field (*i.e.*, particles that have $v(p,i) = j$).

Updated momentum on the grid (indicated with a tilde) is:

$$\tilde{\boldsymbol{p}}_{i,j} = \boldsymbol{p}_{i,j} + (\boldsymbol{f}_{i,j} + \boldsymbol{f}_{i,j,T})\Delta t$$

Note that superscript (n) on grid values are now dropped for simplicity (unless needed). Standard MPM updates (called FLIP updates [43]) to particle velocity and position become:

$$\begin{aligned}
\boldsymbol{v}_p^{(n+1)} &= \boldsymbol{v}_p^{(n)} + \boldsymbol{a}_{g\to p}\Delta t \\
\boldsymbol{x}_p^{(n+1)} &= \boldsymbol{x}_p^{(n)} + \tilde{\boldsymbol{v}}_{g\to p}\Delta t - \frac{1}{2}\boldsymbol{a}_{g\to p}(\Delta t)^2
\end{aligned}$$

where $\boldsymbol{a}_{g\to p}$ and $\tilde{\boldsymbol{v}}_{g\to p}$ are accelerations and updated velocities extrapolated from the grid to the particle using appropriate velocity fields:

$$\boldsymbol{a}_{g\to p} = \sum_i \frac{\boldsymbol{f}_{i,v(p,i)} + \boldsymbol{f}_{i,v(p,i),T}}{m_{i,v(p,i)}} S_{ip}, \quad \tilde{\boldsymbol{v}}_{g\to p} = \sum_i \frac{\tilde{\boldsymbol{p}}_{i,v(p,i)}}{m_{i,v(p,i)}} S_{ip}, \quad m_{i,j} = \sum_p m_p S_{ip} \delta_{jv(p,i)}$$

These particle updates can be extended to various damping schemes or to enhance stabilization using XPIC(k) [44, 45] or FMPM(k) [46]. The reader is referred to other MPM papers for more details on these basic equations [11, 40, 41, 44]. The following sections concentrate on changes needed to handle explicit cracks.

2.1.2. Updating Crack Particles in CRAMP

The massless particles that define each crack must move along with other material points. Prior CRAMP [11] only moved crack plane particles. A better approach is for each crack particle to separately track crack surface positions above and below the crack plane and to move those surface particles on each time step. Their update starts by extrapolating updated grid velocity to surface particle locations, $\boldsymbol{x}_s$. A standard extrapolation would be:

$$\tilde{\boldsymbol{v}}_{g\to s} = \sum_i \tilde{\boldsymbol{v}}_{i,v(s,i)} N_i(\boldsymbol{x}_s) \quad \text{where} \quad \tilde{\boldsymbol{v}}_{i,v(s,i)} = \frac{\tilde{\boldsymbol{p}}_{i,v(s,i)}}{m_{i,v(s,i)}} \tag{3}$$

where $v(s,i)$ is the appropriate crack velocity field when extrapolating from node i to surface particle s. It is found using the methods to find $v(p,i)$ (see the appendix). Unlike material points, CRAMP particles have no domain, which is why this extrapolation uses ordinary grid shape functions, $N_i(\boldsymbol{x}_s)$, rather than GIMP shape functions in Eq. (2).

Another surface particle difference is that they do not extrapolate *to* the grid. Consequently, the extrapolation in Eq. (3) might encounter nodes with zero mass but $N_i(\boldsymbol{x}_s) \neq 0$. To account for such effects on extrapolations, Eq. (3) should replace $N_i(\boldsymbol{x}_s)$ with mass-weighted, renormalized shape functions defined by:

$$S_{is}^{*(n)} = \frac{m_{i,v(s,i)} N_i(\boldsymbol{x}_s)}{\sum_{i\in\phi_s} m_{i,v(s,i)} N_i(\boldsymbol{x}_s)} \quad \text{where} \quad \phi_s = \{i : m_{i,v(s,i)} \neq 0\} \quad \text{and} \quad \sum_i S_{is}^{*(n)} = 1 \tag{4}$$

The final extrapolations are clarified by casting them as

$$\tilde{\boldsymbol{v}}_{g\to s} = \frac{\tilde{\boldsymbol{p}}(\boldsymbol{x}_s)}{m(\boldsymbol{x}_s)} \quad \text{where} \quad \tilde{\boldsymbol{p}}(\boldsymbol{x}_s) = \sum_{i\in\phi_s} \tilde{\boldsymbol{p}}_{i,v(s,i)} N_i(\boldsymbol{x}_s) \quad \text{and} \quad m(\boldsymbol{x}_s) = \sum_{i\in\phi_s} \boldsymbol{m}_{i,v(s,i)} N_i(\boldsymbol{x}_s) \tag{5}$$

An unweighted renormalization was also tried, but was prone to instability caused by low-mass nodes with inaccurate velocities. Mass-weighting resolves such instabilities.

Given $\tilde{\boldsymbol{v}}_{g\to s}$, a second order position update using standard MPM FLIP methods [44] is

$$\boldsymbol{x}_s^{(n+1)} = \boldsymbol{x}_s^{(n)} + \tilde{\boldsymbol{v}}_{g\to s}\Delta t - \frac{1}{2}\boldsymbol{a}_{g\to s}(\Delta t)^2$$

where $\boldsymbol{a}_{g\to s}$ is acceleration extrapolated to surface particles by analogous methods used for $\tilde{\boldsymbol{v}}_{g\to s}$:

$$\boldsymbol{a}_{g\to s} = \frac{\boldsymbol{f}(\boldsymbol{x}_s)}{m(\boldsymbol{x}_s)} \quad \text{and} \quad \boldsymbol{f}(\boldsymbol{x}_s) = \sum_{i\in\phi_s} \left(\boldsymbol{f}_{i,v(s,i)} + \boldsymbol{f}_{i,v(s,i),T}\right) N_i(\boldsymbol{x}_s) \tag{6}$$

A first-order FLIP update omits the acceleration term and can skip its extrapolation. While FLIP updates are acceptable, more stable results are possible by switching to PIC updates [44], which should be based on XPIC(1) updates from Ref. [45]:

$$\boldsymbol{x}_s^{(n+1)} = \boldsymbol{x}_s^{(n)} + \left(\boldsymbol{v}_{g\to s} + \frac{1}{2}\left(\tilde{\boldsymbol{v}}_{g\to s} - \boldsymbol{v}_s^{(n)}\right)\right)\Delta t = \boldsymbol{x}_s^{(n)} + \left(\frac{3\tilde{\boldsymbol{v}}_{g\to s} - \boldsymbol{v}_s^{(n)}}{2} - \boldsymbol{a}_{g\to s}\Delta t\right)\Delta t$$
$$\boldsymbol{v}_s^{(n+1)} = \tilde{\boldsymbol{v}}_{g\to s}$$

The first position update is in terms of $\boldsymbol{v}_{g\to s}$ for extrapolation at the start of the time step, but that extrapolation is never done. It is eliminated in the second position update using

$$\tilde{\boldsymbol{v}}_{g\to s} = \boldsymbol{v}_{g\to s} + \boldsymbol{a}_{g\to s}\Delta t$$

XPIC(1) updates are characterized as second order updates where acceleration is found from change in velocity (which must be tracked as indicated) rather than from extrapolated acceleration (although acceleration is used to avoid extrapolating $\boldsymbol{v}_{g\to s}$). Although XPIC(1) dissipates energy, that issue is not a concern because massless crack particles have no energy. A fourth update option is FMPM(1) [46], but it was found less reliable on some problems. XPIC(1) update methods are recommended.

Once crack surfaces are moved, crack particles can be moved by two methods. The simplest method is to keep them between the surfaces by moving the plane by sum of the two surface movements. Prior CRAMP moved crack particles using a center-of-mass velocity extrapolated from the grid [11]. This approach, however, suffers from dwindling, or even disappearing, grid information for particles at the mid plane of widely-opened cracks. Moving crack planes by the average of crack surface movements is the preferred approach.

2.1.3. Crack Surface Contact

When cracks close, the algorithm must handle crack surface contact to prevent interpenetration. The process is to examine each node with more than one crack velocity field. For any pair of crack velocity fields, a calculation is done to determine if they are in contact. When contact is detected, momenta in the two fields are changed by $\Delta\boldsymbol{p}_{i,j}$, where possible changes can model contact by various contact laws [26, 47] or as an imperfect interface [48, 49]. Accurate contact detection requires an accurate crack surface normal vector [48, 50, 51]. This normal is calculated during the algorithm to find $v(p,i)$ by averaging normals to the crack plane at the point where traced lines cross the crack (see the appendix).

Accurate contact calculations require two evaluations of contact effects in each time step [47]. The first is done after extrapolating momenta to the grid leading to revised momenta $\boldsymbol{p}_{i,j}+\Delta\boldsymbol{p}_{i,j}$. The second ($\Delta\tilde{\boldsymbol{p}}_{i,j}$) is done after updating grid momenta to get final momenta updates of:

$$\tilde{\boldsymbol{p}}_{i,j} = \boldsymbol{p}_{i,j} + \Delta\boldsymbol{p}_{i,j} + \Delta\tilde{\boldsymbol{p}}_{i,j} + (\boldsymbol{f}_{i,j} + \boldsymbol{f}_{i,j,T})\Delta t$$

When updating particles, the acceleration on contact nodes is found using:

$$\boldsymbol{a}_{g\to p} = \sum_i \frac{1}{m_{i,v(p,i)}}\left(\frac{\Delta\tilde{\boldsymbol{p}}_{i,j}}{\Delta t} + \boldsymbol{f}_{i,v(p,i)} + \boldsymbol{f}_{i,v(p,i),T}\right)S_{ip}$$

Note that acceleration is adjusted by contact effects done after the momentum update ($\Delta\tilde{\boldsymbol{p}}_{i,j}$) but not done before ($\Delta\boldsymbol{p}_{i,j}$). This approach is needed because otherwise cracks in contact with a perfect interface would not correctly revert to a simulation with no cracks [47].

Basic crack contact calculations for $\Delta\boldsymbol{p}_{i,j}$ and $\Delta\tilde{\boldsymbol{p}}_{i,j}$ are given in [11], but two changes are given here. First, contact detection can be improved by considering both the velocity and position of those fields. This change is described in [48]. Second, contact calculations need modification for CSCs. For example, Fig. 1B shows a node with CSCs that has four contacting surfaces corresponding to contact between crack velocity fields 0 and 1, 2 and 3, 0 and 2, and 1 and 3. The first two are contact on crack #1 and the second two are for crack #2. A node with CSCs having less than four velocity fields would have fewer contacting surfaces. Contact calculations must repeat for each pair of potentially contacting velocity fields present on the node, being sure to use the appropriate crack's normal.

2.1.4. Traction Loaded Cracks and Cohesive Laws

Because CRAMP tracks crack surface positions, it can model crack traction laws that depend on crack-opening displacements, such as cohesive traction laws [52]. Traction laws are implemented by adding a traction force $\boldsymbol{F}_{i,j}^{(n)}$ to nodal forces $\boldsymbol{f}_{i,j}^{(n)}$, where

$$\boldsymbol{F}_{i,j}^{(n)} = \sum_s \pm \boldsymbol{F}_s^{(n)} S_{is}^{*(n)} \delta_{jv(s,i)} \tag{7}$$

is a sum over crack surface particles, $\pm\boldsymbol{F}_s^{(n)}$ is traction force on surface particle s above (+ sign) or below (− sign) the crack plane, $S_{is}^{*(n)}$ is a renormalized shape function for surface particle s (see Eq. (4)), and the Kronecker δ function selects the appropriate crack velocity field for the surface particle s–node i pair as $j = v(s,i)$.

The traction force on surface particles is given by:

$$\pm\boldsymbol{F}_s^{(n)} = \mp A_k \big(T_n(\boldsymbol{\delta}_k), T_t(\boldsymbol{\delta}_k) \big) \cdot (\hat{\boldsymbol{n}}_k, \hat{\boldsymbol{t}}_k) \tag{8}$$

where k is crack plane particle associated with the surface particles, A_k is crack area associated with that particle, $T_n(\boldsymbol{\delta}_k)$ and $T_t(\boldsymbol{\delta}_k)$ are normal and tangential tractions as a function of crack opening displacement vector, $\boldsymbol{\delta}_k$, and $\hat{\boldsymbol{n}}_k$ and $\hat{\boldsymbol{t}}_k$ are normal and tangential unit vectors to the crack. Although Eq. (8) allows implementation of any traction law, the only traction law used in examples below is to model pressure loaded cracks such that $T_t(\boldsymbol{\delta}_k) = 0$ and $T_n(\boldsymbol{\delta}_k) = -P$ where P is constant pressure on crack surfaces.

2.1.5. CRAMP vs. *Other Options*

Both CRAMP and XFEM define off-grid cracks with a collection of points. Unlike XFEM [9, 10], CRAMP needs no changes to shape functions, needs no assumptions about crack tip stress states, and is independent of material properties.

Compared to other MPM methods, the Adibaskoro *et al.* [24] use of multiple grids appears similar to our partitioning into multiple velocity fields. They often needed a very large number of grids while we find four velocity fields to be sufficient (the Appendix provides an efficient crack hashing algorithm for determining the velocity fields).

The MPM approach by Fan *et al.* [25] that converts crack analysis to material zones that interact by MPM contact calculations is ill-advised and hampered by use of out-dated MPM contact methods [26, 42]. Updated MPM contact methods show that the main source of errors in contact calculations is determination of the contact surface normal and that MPM is not capable of finding multiple normals that would be needed to handle zones created by intersecting cracks [51]. These issues are resolved by retaining crack velocity fields and exploiting geometric information in tracked explicit cracks. An explicit crack surface clearly defines the contacting surface including accurate determination of the contact surface normal. When two cracks intersect, explicit crack tracking can find the two normals needed for contact calculations on each crack (see intersecting crack contact example in Fig. 9B). Furthermore, by keeping cracks and materials separate, CRAMP and multimaterial MPM methods [26, 42] can be combined by allowing each crack velocity field to have multiple material velocity fields (all examples in this paper used a single material in each crack velocity field.)

Some other particle methods model cracks using "node visibility" rules that discount effects of particles on the opposite side of a crack determined by tracing rays between particles and nodes [53, 54] or by a diffraction analysis [55]. Although CRAMP similarly traces rays, no particles are discounted — every particle in the domain of a node contributes to that node. The ray tracing determines which crack velocity field to use.

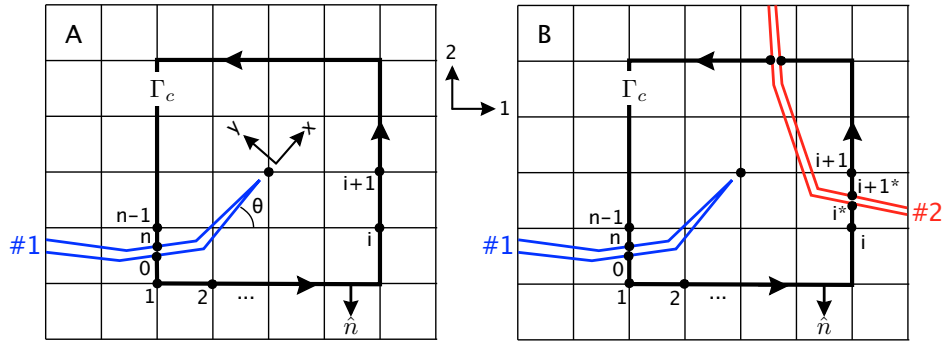

Figure 2: A. A contour, Γ_c, around a crack tip for a single crack #1. B. A contour, Γ_c, around a crack #1's tip that intersects a second crack #2. A crack propagation direction axis system has x-y coordinates rotated by θ from the 1-2 grid axes. $\hat{\boldsymbol{n}}$ is an outward-directed normal for the contour. Nodes along the contour are labeled from 0 to n. Material points around the crack tip are omitted for clarity.

2.2. Task 2: J Integral for J Contours Intersected by other Cracks (JCICs)

Crack propagation predictions can use crack-tip fracture parameters such as stress intensity factors or energy release rate. This section describes energy release rate calculations for crack growth by evaluating dynamic J integral [14]. Although task 1 to discretize cracks includes both 2D and 3D cracks, the results here for task 2 are limited to 2D problems.

Dynamic J-integral components, J_1 and J_2, for a 2D crack are [56, 57]:

$$J_k = \int_{\Gamma_c} \left[(W+K)n_k - \sigma_{ij} n_j \frac{\partial u_i}{\partial x_k} \right] d\Gamma + \int_{V(\Gamma_c)} \rho \left(\frac{\partial^2 u_i}{\partial t^2} \frac{\partial u_i}{\partial x_k} - \frac{\partial u_i}{\partial t} \frac{\partial^2 u_i}{\partial t \partial x_k} \right) dV \tag{9}$$

where Γ_c is a contour around the crack tip and $V(\Gamma_c)$ is volume contained by that contour. The integrand terms are W for strain energy, K for kinetic energy, $\hat{\boldsymbol{n}} = (n_1, n_2)$ for outward normal to Γ_c, $\boldsymbol{\sigma}$ for Cauchy stress, $\boldsymbol{u} = (u_1, u_2)$ for displacement, and ρ for density. The first term is the usual J integral for static problems (but with added kinetic energy term) while the second term is needed for contour-independence in dynamic problems [56, 57]. The 1-2 axes refer to the background grid while x-y axes refer to crack propagation direction (see Fig. 2A). J_{total}, or energy release rate for self-similar crack growth in the x direction, is derived from J_k using:

$$J_{total} = J_1 \cos\theta + J_2 \sin\theta \tag{10}$$

where θ is angle between the 1 axis and the crack-growth x direction.

J can be calculated in MPM by constructing a contour along grid lines surrounding the crack tip. Using grid-based paths is convenient because particle values needed for J are easily extrapolated to the grid and normals to the contour are either (1,0) or (0,1). Figure 2A shows a grid-based contour centered on the node nearest the crack tip and defined by nodes 0 to n along the contour. General rectangular contours on the grid can be defined by $(+h, +v, -h, -v)$ for distances from the center node to the contour in the ±horizontal and ±vertical grid directions. When all distances are the same, such as the (2,2,2,2) contours in Fig. 2, the square contours are referred to by their total side length, such as "4-cell" contours or by physical length of their sides, such as "2 mm" contours.

The contour integral term in Eq. (9) sums integrations between adjacent nodes along the contour. The calculations use displacement field for particles on the same side of the crack as each contour node

(*i.e.*, data from particles near node i having $v(p,i) = 0$). Here "displacement field" means all information needed by J (W, K, $\boldsymbol{\sigma}$, and $\partial u_i/\partial x_k$) and that information is extrapolated to velocity fields on the grid before J calculations using standard MPM extrapolation methods. The contour interval that crosses the crack between contours nodes 1 and $n-1$ in Fig. 2 is an exception. For this interval, the contour is split at the crack intersection and two *virtual* nodes 0 and n are created. The displacement field on node 0 interpolates displacement fields for $v(p,1) = 0$ on node 1 and for $v(p,n-1) = 1$ on node $n-1$. This later displacement field is for particles on the opposite side of the crack as node $n-1$, which means it is for particles on the same side of the crack as node 1. Analogously, the displacement field on node n interpolates displacement fields for $v(p,n-1) = 0$ on node $n-1$ and for $v(p,1) = 1$ on node 1. Once virtual nodes are evaluated, the two node-to-node intervals are added to the contour integral. The volume integral in Eq. (9) can use material points enclosed in the contour as integration points with integrand terms derived from particle data. Finally, if the crack has tractions, that loading is added to J integral by methods in Ref. [18].

For non-interacting cracks, this J method is independent of the contour in dynamic problems [17]. It can model traction-loaded crack surfaces [18] and can partition J into mode I (K_I) and mode II (K_{II}) stress intensity factors for modeling mixed-mode crack propagation. For problems with CSCs, however, new J methods are needed whenever contours are intersected by other cracks (*e.g.*, the JCIC in Fig. 2B). The reason is that J theory relies on the contour enclosing one discontinuity (the target crack) within an elastic continuum [14]. A second crack within the contour creates another discontinuity that breaks up the elastic continuum. Any numerical model that integrates either JCICs or crack-tip domains [10, 15, 16] without modifying the calculations for other cracks that disrupt the elastic continuum would be invalid.

To assess *unmodified* J calculations, we included a method labeled as PIJC for "Partially Integrated J Contour." PIJC simply identifies J contour segments that cross other cracks. Each crossed segment splits the contour path at two more virtual nodes by the same methods used for virtual nodes 0 and n (see extra splits in Fig. 2B). J calculations then integrate along the rectangular contour always using the velocity field for particle on the same side of the crack as node i, but the calculations are otherwise not modified to account for other cracks within the contour. PIJC is illustrated by the solid lines inscribed in the J contours in Fig. 3. PIJC is clearly an invalid J calculation because it does not account for other discontinuities within the contour. It is included here to quantify improvements in the new methods, which are defined in the next section, and as a stand-in for evaluating J integral or domain methods by MPM or non-MPM methods that ignore intersecting cracks.

2.2.1. Fully Integrated J Contour Method

The new method to account for intersecting cracks is labeled as FIJC for "Fully Integrated J Contour" because the integration path includes surfaces of intersecting cracks. FIJC first finds intersecting cracks as in PIJC and evaluates the PIJC result, but then adds J terms for integration along both sides of any contour intersecting cracks (as illustrated by the dashed lines in Fig. 3). Crack path integrations only need the first term in Eq. (9) because no particles are allowed between crack surfaces. Strain and kinetic energy terms are found by extrapolating grid results from proper crack velocity fields to crack surface positions. Contour normal comes from crack normal vectors. Accounting for tractions on intersecting crack surfaces is explained in section 2.2.2.

The solid and dashed lines in Fig. 3A show that FIJC splits the calculation into two J contours. The right contour should be zero because it completely encloses an elastic region with no discontinuities, while the left contour is a valid J contour for a single crack. Similarly, Fig. 3B has two regions on the left that should have $J = 0$ and a region on the right to find J for a single crack tip. One might be tempted to

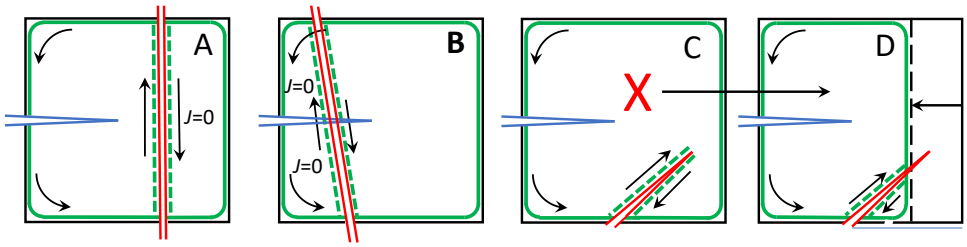

Figure 3: Three JCIC examples with a single crack intersecting the J contour. The solid line inscribed in the rectangular path is J integration for PIJC method that ignores intersecting cracks. The dashed lines shows additional J integrations along both surfaces of intersecting cracks in FIJC. A. Intersecting crack passes through the contour. B. Intersecting crack crosses the primary crack. C. Two crack tips with the J contour that requires an adapted contour. D. FIJC after adapting the contour by moving one edge the intersecting crack's tip.

identify isolated regions such as those in Fig. 3A and B with expected $J = 0$ and eliminate those terms, but that approach is not recommended. First, it complicates evaluation of the volume integral in Eq. (9). Second, the examples in Fig. 3 show only a single crack intersecting the J contour, but this approach extends to additional intersecting cracks simply by integrating the surfaces of each one. Locating $J = 0$ regions in all possible configurations would be wasted effort simply to avoid adding terms that should sum to zero.

Figure 3C shows an intersecting crack whose tip is within the contour. Although integration along crack surfaces appears to remove that crack, that approach is invalid because the second crack tip creates a second singularity and integration along its surfaces does not account for that singularity. The best solution is to adapt the contour to omit the crack tip as shown in Fig. 3D. When FIJC encounters a JCIC that needs adapting, one edge of the contour is moved to remain on a grid line but omit the second crack's tip. If the second crack still crosses the adapted contour (as in Fig. 3D), those intersections are handled like the intersecting cracks in Fig. 3A or B.

2.2.2. JCIC Algorithm Description

A J integral algorithm when contours are intersected by other cracks follows:

1. Extrapolate particle state to displacement fields needed by J calculations for all crack velocity fields on grid nodes.

2. Find the node nearest the target crack tip. Find all node-to-node intervals for a rectangular path centered on that node by following grid lines. The algorithm typically starts with a square, N-cell contour. For non-square contours, such as those that result when adapting the contour to omit a second crack's tip, trace the rectangular shape defined by that contour.

3. Check each node-to-node interval and identify those crossed by a crack. The interval crossed by the target crack (and there should only be one) defines *virtual* nodes 0 and n as described above. For each interval that crosses a different crack, create two more *virtual* nodes within that segment and interpolate grid results to those nodes using velocity fields defined by the crossing crack (see i^* and $i+1^*$ *virtual* nodes in Fig. 2B). Store a list of all *virtual* node pairs caused by intersecting cracks. A robust algorithm must handle the uncommon case where two cracks cross the same interval. If found, insert two more virtual nodes in the interval, and interpolate grid results using velocity fields defined by the CSCs. Finally, the algorithm should handle cracks that enter and exit a contour on the same contour interval.

4. Evaluate the first term in Eq. (9) for J_1 and J_2 along each node-to-node interval including virtual nodes in the path (midpoint rule integration works well). *Real* nodes use extrapolated displacement fields from particles with $v(p,i) = 0$. *Virtual* nodes use data from interpolated displacement fields found in step 3.

5. Return to each *virtual* node pair created by an intersecting crack. For each crack surface, extrapolate grid displacement fields to that crack's surfaces using the same methods used to extrapolate crack-surface velocities (Eq. (5)). W and K come from those extrapolations. The traction term is found as follows: 1. if the crack surfaces are in contact, set $T_i = \sigma_{ij} n_j$ using extrapolated stresses and crack surface normal, $\hat{\boldsymbol{n}}$; 2. if the crack is opened, set the traction to zero, but 3. if an open intersecting crack has a traction law, use the traction determined by that law (*i.e.*, Eq. (8) divided in crack segment area A_k). Starting from the crack point just before the crack enters the contour, integrate each crack segment until reaching a crack point outside the contour or reaching the crack tip. The J integration by midpoint rule from crack point j to $j+1$ is:

$$dJ_k = \begin{cases} \frac{(1-\phi)ds}{2}\left((1-\phi)I_k^{(j)} + (1+\phi)I_k^{(j+1)}\right) & \text{first segment} \\ \frac{\phi ds}{2}\left((2-\phi)I_k^{(j)} + \phi I_k^{(j+1)}\right) & \text{last segment (if not a crack tip)} \\ \frac{ds}{2}\left(I_k^{(j)} + I_k^{(j+1)}\right) & \text{all other segments} \end{cases}$$

   where $I_k^{(i)}$ is integrand for first term in Eq. (9) for crack point j on the surface being integrated and ds is length of the crack segment. For the first and last segments, ϕ is fraction of ds from crack point j to the J contour. The integration must include both surfaces adjusting sign to account for crack surface side and integration direction, and, obviously, only do calculations for one of the two *virtual* node pairs caused by cracks that enter the contour at one interval and leave at another.

   An uncommon, albeit not rare, special case is when a segment crosses through the corner of a contour such that both crack point j and $j+1$ are outside the contour. This case will result in two *virtual* node pairs for the same intersecting crack that both start from crack point j. The J integration for this special case is

$$dJ_k = dJ_k(\phi_1) - dJ_k(\phi_2)$$

   where ϕ_1 and ϕ_2 are fraction of ds from crack point j to the two contour crossings before crack point $j+1$ and $dJ_k(\phi)$ is the "first segment" result given above. The first term integrates from the first crossing point to crack point $j+1$ while the second subtracts the portion of the first term that was outside the contour.

6. If any integration in step 5 reaches a crack tip, adjust the rectangular contour to omit that crack tip. Find the distance (in grid cells) between the two crack tips and move one contour edge to omit the intersecting crack's tip (adapt the edge that can move the least). For accuracy and whenever possible, keep the new edge at least 2 cells from each crack tip [17]. After moving one edge, return to step 2 using the adapted contour.

   Note 1: Experience reveals that accuracy is improved by also adapting contours when the second crack's tip is outside the contour but in a cell adjacent to the contour. This extra adaptation is not based on J theory by rather on practical numerical issues of integrating along a path that is too close to any crack tip [17].

Note 2: If the two crack tips are in the same cell, no adapted contour could omit the second crack. In other words, it is not possible to find J integral for one crack tip when another crack tip is within the same cell. One option is to increase resolution until the crack tips are in separate cells. We found that a practical solution is to print a warning (as an alert that higher resolution might be needed) but then continue with the crack surface integration found in step 5. The propagation example in Section 3.5 used this method and correctly stopped the propagation when the two crack tips entered the same cell. In other words, proceeding with step 5 resulted in reduced or zero J corresponding to two crack tips in the same cell effectively linking into a single crack.

7. Evaluate the integrals over $V(\Gamma_c)$ using all material points within the rectangular contour as integration points.

8. Find target crack's total J integral using Eq. (10).

9. If all, or part, of the target crack's surfaces within the contour rectangle are loaded with tractions, J calculations must add a term J_T, which is defined below, for integration on the target crack's surface within the contour. Note that contributions due to intersecting-crack tractions are done in step 5; J_T is only for target-crack tractions.

The above nine steps define the adaptive FIJC algorithm. To test consequences of omitting contour adaptation, skip step 6. To test the consequences of relying on methods that ignore intersecting cracks, skip steps 5 and 6 (*i.e.*, use the PIJC method). Note that this method to account for intersecting cracks fails if a second crack is entirely within the J contour and therefore does not intersect the contour. Problem resolution needs be selected such that contours are smaller than any crack length in the problem.

If the target crack has tractions and is *not* crossed by another crack within the contour, step 9 adds a crack traction term by integrating on the crack surfaces [18, 58]:

$$J_T = \int_{x_0}^{x_{tip}} \boldsymbol{T} \cdot \frac{d\boldsymbol{\delta}}{dx} dx = \int_{0}^{\boldsymbol{\delta}(x_{root})} \boldsymbol{T} \cdot d\boldsymbol{\delta} = W_Z(x_{root}) \tag{11}$$

This integration is along x in crack coordinates (see Fig. 2A), x_0 is where the crack enters the rectangular contour, x_{tip} is at the crack tip, $\boldsymbol{T}$ is crack traction (relative to the lower surface), and $\boldsymbol{\delta} = \boldsymbol{u}_u - \boldsymbol{u}_l$ is crack opening displacement between upper and lower surfaces. Reducing the x_0 to x_{tip} integral in Eq. (11) to traction integral at x_{root} is only possible when the crack surface displacements are continuous. For this special case, $W_Z(x_{root})$ uses $x_{root} = x_0$ if the entire crack path within the contour has tractions or x_{root} if only x_{root} to x_{tip} has tractions (*e.g.*, x_{root} at trailing edge, or root, of a cohesive zone around the crack tip) [18]. $W_Z(x_{root})$ is potential energy due to tractions at x_{root}. When the tractions are modeling a elastic cohesive zone with damage, $W_Z(x_{root})$ is remaining potential energy after damage [52].

When another crack crosses a traction-loaded target crack within the contour (see Fig. 3B), J_T integration in Eq. (11) must be modified to account for surface displacement discontinuities. The integration from x_0 to x_{tip} is replaced by integrations over each continuous section of the crack surfaces leading to:

$$J_T = W_Z(x_{root}) - \sum_{j \in \text{crossings}} \big(W_Z(x_j) - W_Z(x_{j+1}) \big). \tag{12}$$

where "crossings" is the set of crack particles such that another crack intersects the target crack between crack particle j and $j+1$. The summation finds potential energy on the two sides of the displacement discontinuity at the crack intersection. An example below shows the consequence of ignoring crack-crossing corrections to crack traction terms.

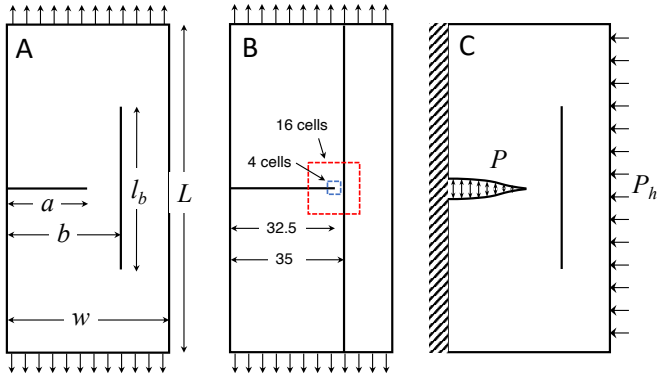

Figure 4: A. Single-edge-notched specimen loaded in tension (SENT) with a horizontal and a vertical crack. The lengths of the two cracks are a and l_b. The vertical crack is located a distance b from the left edge. B. An SENT specimen with the vertical crack extended the entire specimen length. C. A center-notched specimen (CN) with crack surfaces loaded by pressure P. The specimen's right edge is optionally loaded with a horizontal pressure P_h. The left edge is a symmetry plane.

3. Results and Discussion

3.1. SENT with Interacting Cracks

Example 1 in Fig. 4A is a $w = 50 \times L = 100$ mm^2, single edge notched specimen loaded by tensile strain (SENT) with a horizontal or H crack half way through the specimen ($a/w = 0.5$). A second vertical or V crack with length $l_b = 50$ mm is located at b/w. When $b > a$, the cracks do not intersect and the results model J as an H crack propagates up to a V crack (as $a \leftarrow b$). When $b < a$, the V crack crosses the H crack and the results model J after an H crack that has propagated through a V crack. These quasi-static calculations for decreasing b test J accuracy for modeling propagation up to and through other cracks.

To validate calculations, the new J calculations were compared to reference FEA calculations of energy release rate, G, using a virtual crack-closure technique (VCCT) [12, 13]. The static, plane-stress, linear-elastic FEA meshed just the specimen's top half with the bottom of the mesh at the specimen mid plane. The bottom left half was stress free to model the H crack while the bottom right half was fixed at zero displacement. The entire top was displaced by 0.5 mm (1% strain). The V crack was explicitly modeled by splitting the mesh along a vertical crack path of semi-length 25 mm. All elements were 8-node quadrilateral elements. The material properties were $E = 2500$ MPa and $\nu = 0.33$. The results in Fig. 5 are plotted as G/G_0 where $G_0 = 13,402$ J/m^2 is the converged VCCT energy release rate without the second crack.

The first FEA calculations used either 1 mm or 0.5 mm equally-sized elements in an orthogonal grid (a "regular grid"). Results at different resolutions matched within 1% except for $0.5 < b/w < 0.58$ (or distance between cracks < 4 mm; see dashed lines in Fig. 5A). The problem in this range is that the gap from H crack tip to the V crack is small and a regular grid has few elements in that gap. Furthermore, the smallest gap that can be evaluated is when the gap has only one element (*i.e.*, $e = b - a$ where e is element size in the grid). Although an advantage of VCCT is that it can be accurate with coarse meshes, a crack tip near another crack or CSCs, is an example of when VCCT needs a refined mesh for accurate results. Convergence checks showed that FEA needed at least 10 elements between the H crack tip and the V crack. This added mesh refinement allowed accurate G results for very small

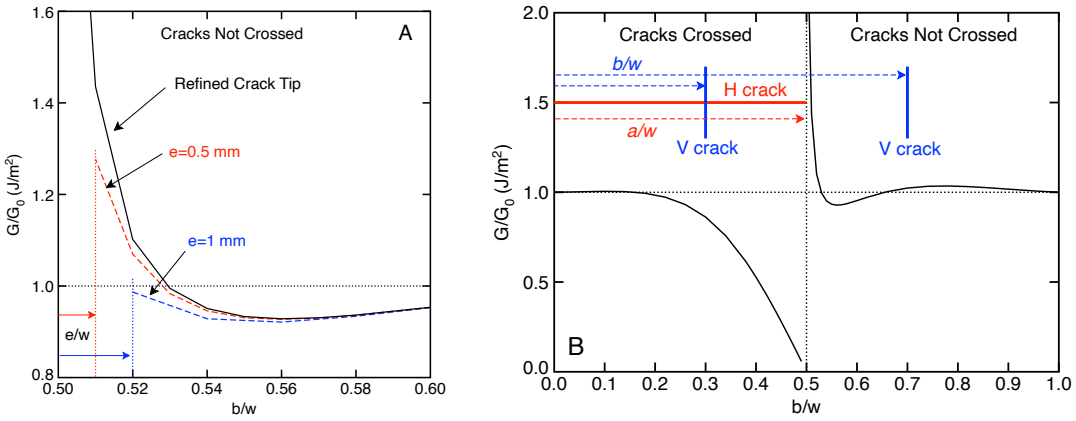

Figure 5: Calculation of G using VCCT in FEM normalized to G_0 for a single crack. A. Results for $0.5 < b/w < 0.6$ using regular meshes with $e = 1$ mm or 0.5 mm elements (dashed lines) or after refining the mesh near the crack tip (solid line). B. Full VCCT results. When needed, the mesh in the gap between the H and V cracks was refined for accurate calculations. When $b/w < 0.5$, the vertical crack crosses the horizontal crack; when $b/w > 0.5$, the cracks are not crossed.

gaps. The solid curves in Fig. 5 show converged VCCT calculations. As the V crack got far from the H crack tip (*i.e*, b/w much smaller or larger than a/w), G approached G_0 for a single crack. For "Cracks Crossed" ($b/w < 0.5$), G approached zero for b near a and increased as b/w decreased. This behavior is analogous to a short edge crack (when $b \lesssim a$) getting longer (as b decreases) causing G to increase. For "Cracks Not Crossed" ($b/w > 0.5$), G increased as the gap between the cracks got smaller emulating expected results for a crack approaching the edge of a finite-sized specimen.

2D plane-stress MPM calculations for J used either 1 mm or 0.5 mm regular grids with 4 material points per cell. To minimize dynamic effects, strain was ramped with velocity $V_0 \sin(\pi t/t_r)$ where V_0 was set to 0.2% of the material's wave speed and ramp time t_r was time to reach 0.5 mm displacement (1% strain). MPM calculations used the approximate full-mass matrix method of order 1 or FMPM(1) [46]. After t_r, the strain was held fixed for $0.5t_r$ and some grid damping was added. Finally, J was given by its final steady-state value. The J contour was set to an 8 mm square (8-cell or 16-cell contours for 1 mm or 0.5 mm cells, respectively). Although J calculations are more efficient with smaller contours, large contours were used to cause more JCIC situations for validation tests.

Sensitivity of J to method (FIJC *vs.* PIJC) and resolution (1 mm *vs.* 0.5 mm cells) was evaluated by plotting J/G_0 as a function of b/w. Figure 6A plots FIJC results. The boxes and circles are for 1 mm and 0.5 mm cells, respectively. For this, and all subsequent figures, open symbols indicate contours without crack intersections while filled symbols indicate JCIC calculations. For contours without intersections, J matched VCCT results for both resolutions. For JCICs, both resolutions showed the expected rise in J as the gap between non-crossed cracks got smaller ($0.5 \leftarrow b/w$). Higher-resolution results (0.5 mm cells) were closer to VCCT than lower-resolution results (1 mm cells). Crossed-crack results ($b/w < 0.5$) also agreed well with VCCT calculations. Although MPM results differed slightly from VCCT, MPM got reasonable accuracy with a regular grid while VCCT needed refinement to handle crack tips close to another crack.

Figure 6B plots PIJC results to quantify inaccuracies of ignoring intersecting cracks. For contours without intersections PIJC was valid, but for JCICs, PIJC had large errors. In the region where the second crack approaches the crack ($0.50 < b/w < 0.58$), J decreased as the gap got smaller rather than

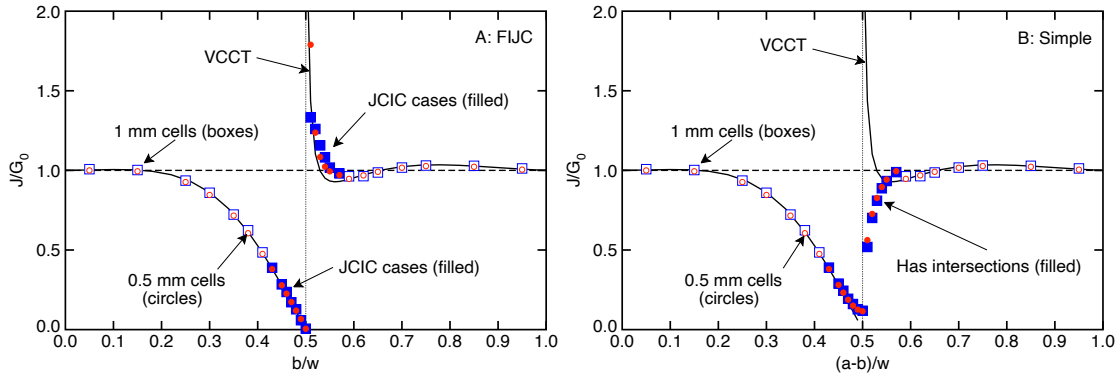

Figure 6: J calculations using 8 mm × 8 mm square contours for 1 mm (boxes) or 0.5 mm (circles) regular grids. Open symbols are calculations where the second crack did not intersect the J contour; filled symbols are JCIC cases. Results were normalized to G_0 for a single crack found using VCCT. Solid lines are converged VCCT results. A. J calculated using FIJC. B. J calculated using PIJC.

increase as expected. In the region where the second crack intersects the first crack ($0.42 < b/w < 0.50$), the results were closer to VCCT but higher and did not approach zero as $b \rightarrow a$. PIJC was expected to be inaccurate and these results reveal large errors in methods that ignore intersecting cracks.

Sensitivity of FIJC J calculations to both contour size and resolution was evaluated by plotting J/G_0 for 2 mm (circles), 4 mm (diamonds), or 8 mm (boxes) contours. Fig. 7A plots results using 1 mm cells (therefore 2-, 4-, and 8-cell contours). Symbols are drawn with larger size for larger contours to help visualization. Both 8 mm and 4 mm contour results were close to path independent, with 4 mm results slightly closer to VCCT calculations. 2 mm contour results were too low, most likely caused by insufficient resolution around the crack tip rather than by J calculation method. A 2 mm contour with 1 mm cells contains only 4 cells and 16 material points. Finding J with that contour relies too much on particles close to the crack tip that may insufficiently resolve crack tip stresses. The situation is analogous to VCCT results with a regular grid. When VCCT relied on too few elements near the crack tip, the results were too low (see Fig. 5A) and the mesh had to be refined for acceptable results. In contrast, J calculations can get good results with a regular grid as long as the contour is not too close to the crack tip.

Figure 7B plots results at higher resolution (0.5 cells) corresponding to 4-, 8-, and 16-cell contours; all contours gave similar results. Noting that J with large contours that encounter JCICs (filled symbols) matched calculations with smaller contours without intersections further emphasizes that FIJC correctly accounts for intersecting cracks. The best results used the smallest 2 mm contour. At this higher resolution, this contour surrounds 16 elements and 64 material points. Comparing resolutions suggests that 4-cell contours (which have 2 cells in all directions from the crack tip) gave accurate J at either resolution.

3.2. Interpreting Differences Between FIJC and PIJC

Consequences of ignoring intersecting cracks (*e.g.* PIJC errors in Fig. 6B) can be understood by the "patch" test problem in Fig. 4B for an SENT specimen with a 32.5 mm horizontal crack ($a/w = 0.65$) and a vertical crack at 35 mm spanning the entire specimen length. Under uniform tensile strain, such a V crack has no contact, which divides the specimen into two independent regions. The left region

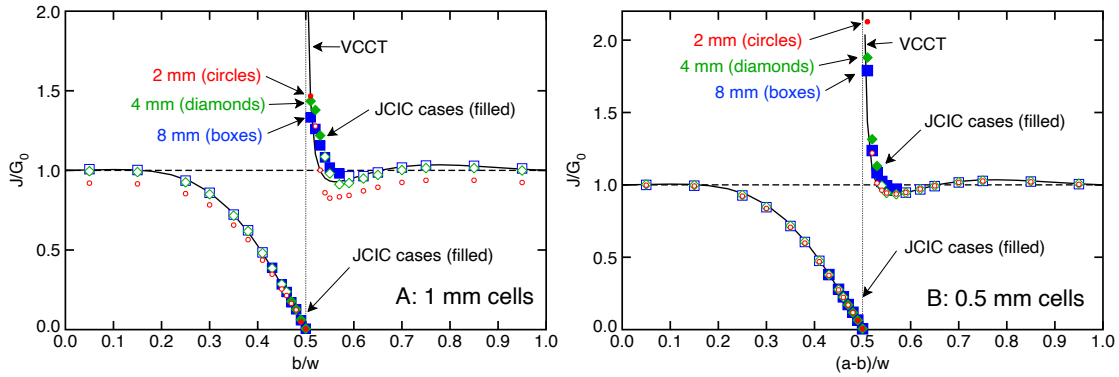

Figure 7: J calculations using 8 mm (boxes), 4 mm (diamonds), or 2 mm (circles) square contours. Open symbols are calculations where the second crack did not intersect the J rectangle; filled symbols are JCIC cases. Results were normalized to G_0 for a single crack found using VCCT. Solid lines are converged VCCT results. A. Calculations for a regular grid with 1 mm cells. B. Calculations for a regular grid with 0.5 mm cells.

deforms as a $w' = 35$ mm wide SENT specimen with a single crack having $a/w' = 0.929$. The right region deforms in uniform tension. The advantage of this patch test is that FIJC with contours that cross into the right region (*i.e.*, JCICs) can be validated by comparing to smaller contours entirely within the SENT region and to VCCT results for a $w' = 35$ mm SENT specimen. Another advantage is that explicit J contour calculations can explain the errors in PIJC compared to FIJC methods.

Figure 8A shows 4-cell and 16-cell contours centered on a node nearest to the crack tip. With 0.5 mm cell resolution, the 2.5 mm space between the crack tip and the V crack has 5 cells. To assess FIJC *vs.* PIJC expectations, the 16-cell contour in Fig. 8A was divided into sections with Γ_1 to Γ_7 being J integrations along contour sections divided by the open circles. Under uniform tension on the right (and assuming quasi-static conditions), the partial J terms (which only need J_1 in Eq. (9) and can omit the second dynamic term) are:

$$\Gamma_2 = -\Gamma_4 = \sigma_{yy}^{(0)} \varepsilon_{yy}^{(0)} l_x^{(+)} \quad \text{and} \quad \Gamma_3 = -\Gamma_7 = W^{(0)} l_y^{(+)} = \frac{1}{2} \sigma_{yy}^{(0)} \varepsilon_{yy}^{(0)} l_y^{(+)}$$

where $\sigma_{yy}^{(0)}$, $\varepsilon_{yy}^{(0)}$, and $W^{(0)}$ are the constant stress, strain, and strain energy density due to uniform tension region on the right, $l_x^{(+)}$ is length of segments Γ_2 and Γ_4, and $l_y^{(+)}$ is length of segments Γ_3 and Γ_7. The total contour integral around the region on the right is $\Gamma_2 + \Gamma_3 + \Gamma_4 + \Gamma_7 = 0$, which implies FIJC should return the correct $J = \Gamma_1 + \Gamma_6 + \Gamma_5$, which is a standard J integral contour for the H crack. Because PIJC omits Γ_6 and Γ_7, the error in PIJC compared to FIJC is

$$\text{PIJC error} = J(\text{PIJC}) - J(\text{FIJC}) = -\Gamma_7 - \Gamma_6 = \left(W^{(0)} - \langle W_{edge}(y) \rangle\right) l_y^{(+)} < 0 \tag{13}$$

where $\langle W_{edge}(y) \rangle$ is average strain energy density along Γ_6. Because $\langle W_{edge}(y) \rangle > W^{(0)}$ due to crack-tip stress concentrations, the PIJC error is expected to be negative causing J(PIJC) to be too low. The magnitude of the PIJC error should increase as the contour size ($l_y^{(+)}$) increases.

Table 1 gives J results as a function of contour size by PIJC and FIJC. All results were normalized to VCCT calculations for a 35 mm wide specimen with a single crack loaded to 1% strain of J_{VCCT}=13,287 J/m^2. For 4-cell and 8-cell contours without JCICs, PIJC and FIJC were identical and

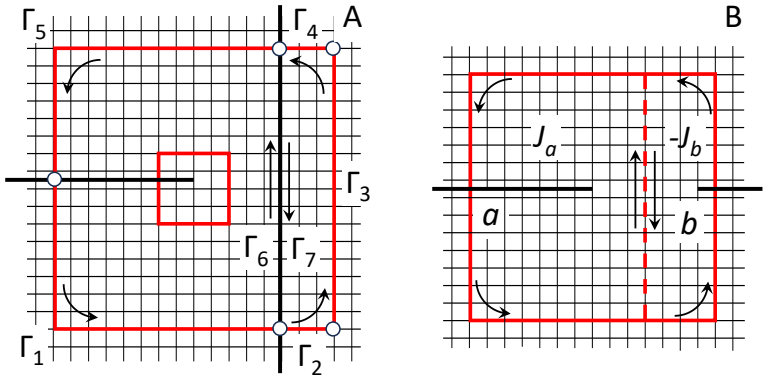

Figure 8: A. Crack tip region of the SENT specimen in Fig. 4B showing 4-cell and 16-cell J contours. Γ_i splits up the 16 cell contour into sections of J calculations separated by circles on nodes. B. Crack tip region for two collinear cracks separated by 6 cells. The thick line shows a 14-cell contour for crack a on the left. The dashed line splits the calculation into two J contours.

Table 1: J calculations for the specimen in Fig. 4B for various square J contour sizes and by either PIJC or FIJC. All results were normalized to VCCT calculation of J_{VCCT}=13,287 J/m^2.

J size (cells)	JCIC	J(PIJC)/VCCT	J(FIJC)/VCCT
4	No	1.002	1.002
8	No	1.004	1.004
16	Yes	0.943	1.064
32	Yes	0.773	1.061

matched VCCT results. Separate MPM results for a 35 mm wide specimen with single crack also matched full specimen results. For 16-cell and 32-cell contours, the contours are intersected by the V crack. As expected, PIJC results were too low and decreased for larger contours (*i.e.*, as $l_y^{(+)}$ increased). In contrast, FIJC results remained constant and reasonably close to VCCT results. The +6% differences likely reflect inaccuracies in crack-path integrations along Γ_6. Nevertheless, FIJC's reasonable accuracy and contour independence suggest it is a good method for crack propagation calculations that encounter JCICs.

If the V crack at 35 mm is shifted to intersect the first crack (*i.e.*, to $b < 32.5$ mm), the left region would be stress free because it is under uniform tension spanned entirely by the H crack; as a result $\Gamma_1 = \Gamma_5 = \Gamma_6 = 0$. Combined with stress-free crack surfaces the region on the right returns $J = \Gamma_2 + \Gamma_3 + \Gamma_4 + \Gamma_7$, which is a standard J integral contour for the H crack, and FIJC should return the correct result. The PIJC error now becomes

$$\text{PIJC error } (b < 32.5 \text{ mm}) = J(\text{PIJC}) - J(\text{FIJC}) = -\Gamma_7 - \Gamma_6 = -\Gamma_7 = \langle W_{edge}(y) \rangle l_y^{(+)} > 0 \qquad (14)$$

which is always positive and increases as contour size increases.

These patch test results for a full V crack explains PIJC observations for a partial V crack in Fig. 6A. When there is a gap between the crack tip and the V crack ($b/w > 0.5$), PIJC with JCICs is too low (by Eq. (13)). When the H and V cracks intersect ($b/w < 0.5$), PIJC with JCICs is too high (by Eq. (14)).

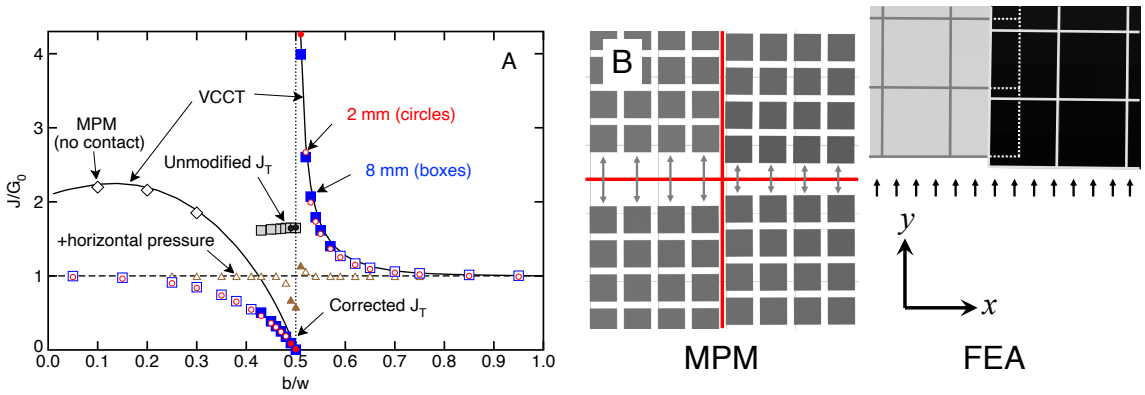

Figure 9: A. J calculations using 8 mm (boxes) or 2 mm (circles) square contours with 0.5 mm cells for CN specimens with pressure-loaded crack surfaces. The "Corrected J_T" symbols used Eq. (12); the "Unmodified J_T" symbols used Eq. (11). The triangles are for $P_h = 10$ MPa and frictional contact between crack surfaces. Open symbols are calculations where the second crack did not intersect the J rectangle; filled symbols are JCIC cases. Results were normalized to G_0 for a single crack found using VCCT. Solid lines are converged VCCT results. B. Small regions near a crack intersection when $b/w = 0.3$ by MPM and FEA. Shades of gray indicate x-direction displacement. MPM material points where scaled by 80% for visualization. FEA analyzed only upper specimen quadrant by symmetry.

3.3. Traction Loaded Crack

The next example tests step 9 in the J algorithm to account for tractions on the target crack surface. Figure 4C shows a center-notched specimen (CN) with a pressure-loaded crack and an optional horizontal pressure P_h. Crack pressure was applied by a traction law with pressure that depends on time but is independent of crack-opening displacement. A CN geometry (with reference to the H crack using symmetry plane on the left edge) was used because otherwise pressure when b was close to zero caused a small section of the specimen to leave the grid. To minimize dynamic effects, crack pressure was ramped up as $(P/2)\big(1-\cos(\pi t/t_r)\big)$ where P was set to 5 MPa. All other simulations details were the same as for example 1 with 0.5 mm cells. The results in Fig. 9A are plotted as J/G_0 where G_0 was found using new VCCT calculations for a single crack with 5 MPa pressure on the crack surfaces [13] as $G_0 = 1392$ J/m^2.

Figure 9A plots results for 2 mm (4 cell) and 8 mm (16 cell) contours when P_h=0. For $b > a$, the H crack has tractions, but it is not intersected by the V crack such that traction correction can use Eq. (11) [18, 58]. These results are similar to remote loading results (*e.g.*, Fig. 7) where J increases as $a \leftarrow b$ and are independent of contour size. For $b < a$, the V crack crosses the H crack. The filled symbols labeled "Corrected J_T" are results that account for intersections using Eq. (12). These results gave expected results where J is close to zero for $b = a$ and increases as b decreases. The accuracy of intersecting-crack traction corrections is confirmed by noting that 8 mm contour results, which were corrected for an intersecting second crack (filled boxes in Fig. 9), matched 2 mm contour results that did not need the corrections (open circles). The symbols labeled "Unmodified J_T" ignored the Eq. (12) intersecting crack terms needed when $b/w < 0.5$. Ignoring such intersections caused large errors.

Treating corrected MPM J results as valid, Fig. 9A shows that VCCT calculations in linear FEA (the solid lines) can have large errors. The problem is that pressure loading on the horizontal crack causes crack contact on the vertical crack resulting in mesh interpenetration. For $b/w > 0.5$, crack interpenetration was minor and VCCT was close to MPM J. For $b/w < 0.5$, however, interpenetration was significant. Figure 9B shows a region near a crack intersection when $b/w = 0.3$. MPM divided the

intersection region into four velocity fields that allowed the horizontal crack to open differently on the two sides of the vertical crack and the vertical crack to accurately model crack contact on both sides of the horizontal crack (shades of gray for material points are plotting x displacement and show continuity). In contrast, VCCT results show significant interpenetration on the V crack surface (shades of gray in elements are plotting x-direction displacement and show discontinuity). The claim that VCCT errors were caused by interpenetration was confirmed by a few MPM J calculations done without contact calculations; those results allowed interpenetration and agreed with VCCT results (see open diamonds labeled "MPM (no contact)" in Fig. 9A). VCCT results would improve by switching to non-linear FEA with contact methods and suitable mesh-tip refinement, but that was not part of this work. MPM, which is a non-linear analysis that handles intersecting crack contact, provides accurate results with a regular grid.

Both end loading (Fig. 7) and pressure loading (Fig. 9) show J increasing as b approaches a for $b/w > 0.5$ and then dropping to zero when $b = a$. These results imply a crack should accelerate as it approaches a second crack but stop when it reaches that crack. These examples, however, were for uniaxial loading and perpendicular cracks. Real-world problems likely have more complex loading and cracks at different orientations. To explore one loading effect, the CN example with pressure loaded cracks was repeated by ramping a horizontal pressure to $P_h = 10$ MPa (see Fig. 4C) and changing crack-contact modeling to use Coulomb friction with coefficient of friction $\mu = 0.6$ [47]. Results using a 2 mm contour and proper traction corrections are plotted as triangles in Fig. 9A. Both side pressure and friction diminish influence of the second crack. The increase in J leading up to intersection is reduced and the J after intersection is increased. In other words, J results approach results for a single crack which could enhance the chance that a crack approaching another crack might penetrate through that crack and keep propagating. Note that Fig. 9 does not include VCCT results with horizontal pressure. For accurate results, such VCCT calculations would need non-linear FEA with contact methods to prevent interpenetration on both cracks.

3.4. An Example with Two Crack Tips in the Contour

For an example that needs adapted contours, we modeled a 50 mm wide double-edge notched specimen in tension (DENT). With two collinear 23.5 mm cracks, the space between the cracks was 3 mm corresponding to 6 cells when using 0.5 mm cells. Any contour larger than 12 cells will contain both cracks tips. Figure 8B shows the crack tip region with a 14-cell contour centered on crack a. A non-adaptive FIJC calculation (*i.e.*, skip step 6 in Section 2.2.2) would integrate along the solid (red) contour and crack b's surface. Because crack b tractions are zero and $n_1 = 0$ on that crack's surfaces, integrating crack b's surfaces adds nothing to non-adaptive FIJC calculations. The errors in failing to adapt can be evaluated by splitting the contour into two contours at the dashed line in Fig. 8B. The left contour gives J_a or J for crack a. Accounting for orientation, the right contour gives $-J_b$ or $-J$ for crack b. Because integrations on the two shared edges cancel, non-adaptive calculations result in $J = J_a - J_b$. In this symmetric example, $J_a = J_b$ whenever the contour contains the second crack meaning that failure to adapt the contour incorrectly returns zero.

To demonstrate the effectiveness of adapting contours to fix two-crack-tip issues, Fig. 10A plots J/J_{ref} for this DENT specimen loaded to 1% strain as a function of J contour size. The results were normalized to J_{ref} taken as the J found for a 4-cell contour (which was 21,954 J/m^2). The solid line is FIJC that includes contour adaptation while the dashed line is non-adaptive FIJC. For contours 10 cells or less, the b crack tip does not intersect the J contour and all methods are identical. For 14 cells or larger, non-adaptive FIJC gives J/J_{ref} close to zero as expected while including adaptation restores the correct result. For 12 cells, the crack tip is adjacent to the contour, but outside the contour. Adapting

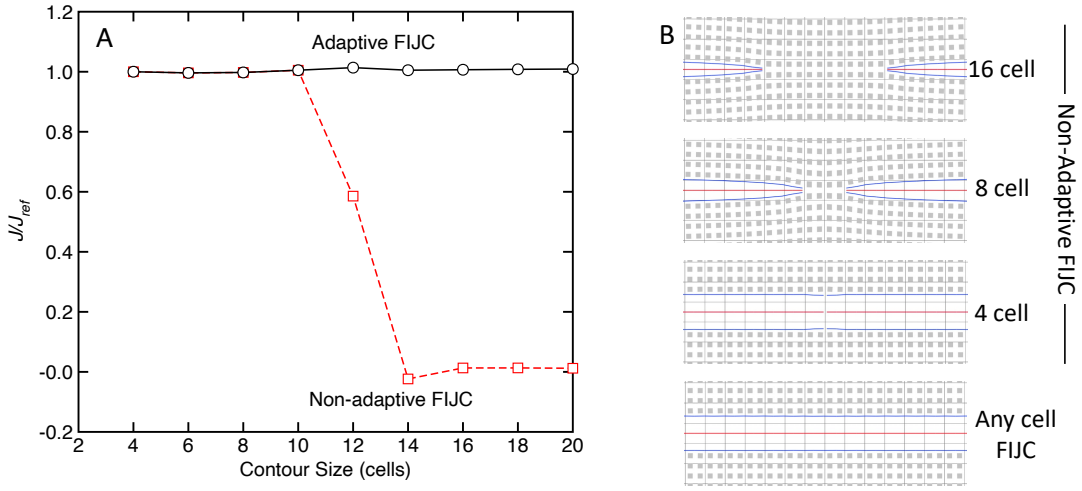

Figure 10: A. J/J_{ref} for two aligned cracks separated by 3 mm (or 6 cells) as a function of J contour size using FIJC (solid line) or non-adaptive FIJC (dashed line). B. Final crack tip region after crack propagation of two collinear cracks in a DENT specimen. These final states are when the crack propagation stopped. The three lines within the cracks indicate the tracked top surface, crack plane, and bottom surface positions, respectively.

this contours as well, as recommended in "Note 1" on step 6 in Section 2.2.2, gave excellent results. Failing to adapt such contours, would revert to the non-adaptive12-cell result that was 40% too loo.

3.5. Evaluating FIJC During Propagation

This and subsequent examples modeled propagation of interacting cracks. In CRAMP, crack propagation is modeling by inserting a new crack segment ahead of the current crack tip segment. A new segment is added whenever a chosen fracture criterion is satisfied. The direction of that segment is determined by a chosen crack growth direction criterion. Discretization of initial cracks should use massless particles separated by distances comparable to material point diameter (d_p or half the cell size). Similarly, new segments inserted during crack propagation should match material point diameters. This approach limits crack propagation speeds to $d_p/\Delta t$ where Δt is the time step. Modeling supersonic crack propagation requires small time steps.

All crack propagation examples that follow were for isotropic materials and crack growth was predicted to be in the maximum hoop stress direction, which is also the direction that maximizes mode I energy release rate [59]. A unit vector $(\cos\theta, \sin\theta)$ in the maximum hoop stress direction by clockwise rotation θ from the current crack tip direction is:

$$\cos\theta = \frac{3R^2 + \sqrt{1+8R^2}}{1+9R^2} \quad \text{and} \quad \sin\theta = -R(3\cos\theta - 1) \quad \text{with} \quad R = \frac{K_{II}}{|K_I|}$$

Here K_I and K_{II} are found by partitioning J integral into mode I and more II stress intensity factors [17]. This angle varies from $\theta = 0$ when $K_{II} = 0$ (*i.e.*, self-similar crack propagation) to $\theta = \pm 70.53°$ when $K_I = 0$. Given this direction, the maximum mode I energy release rate is found from the maximum mode I stress intensity factor using:

$$K_I^{(max)} = K_I \cos^3\frac{\theta}{2} - \frac{3}{2}K_{II}\cos\frac{\theta}{2}\sin\theta$$

Figure 11: A. Observations of two offset cracks in concrete that avoided each other and then propagated toward each other. B. The same as A except the two cracks were offset be less distance. C. Two cracks observed in wood with the final tips changing direction and propagating toward the other crack. Author's photos are from concrete paths and a wooden structure in Corvallis, Oregon.

A crack propagates in the θ direction when $K_I^{(max)} \geq K_{Ic}$. This critical mode I stress intensity factor is related to mode I toughness by $K_{Ic} = \sqrt{J_{Ic}E^*}$ where E^* is effective modulus equal to E in plane stress or $E/(1-\nu^2)$ in plane strain.

The first propagation example modeled DENT specimens with two aligned edge cracks starting at $a/w = b/w = 0.25$. The plane stress calculations used $E = 2500$ MPa, $\nu = 0.33$, $J_{Ic} = 1000$ J/m^2, and $K_{Ic} = 1.58114$ MPa $\sqrt{\text{m}}$. Tension was applied by constant displacement rate slow enough to emulate quasi-static loading. The cracks started to propagate at 0.4% strain and all propagation was complete with less than 1% strain. Because static J increases as crack lengths increase at constant displacement, valid propagation modeling should show rapid crack growth (close to the material's wave speed) until the crack tips meet and form a single crack modeling complete sample failure.

Figure 10B shows the end result of simulating crack propagation using non-adaptive FIJC for three J contour sizes and adaptive FIJC, which was the same for all contour sizes. When omitting contour adaptation, the two cracks propagated rapidly but stopped propagating when their separation was 2 cells less than the contour radius. The crack stopped because the second crack tip in the JCICs caused non-adaptive J to incorrectly drop to zero (see Fig. 10A). The 4-cell contour linked the cracks because two cells less that the contour radius is zero. When using the recommended FIJC, crack propagation correctly continued rapidly until the cracks linked into one crack, and the results were independent of the contour size.

3.6. Two Interacting Offset Cracks

In the previous propagation example, two initially collinear cracks linked into a single crack. If that symmetry is broken by offsetting the cracks, the propagation changes. Observations and experiments show that two misaligned cracks approaching each other divert to avoid linking up [60]. Once the crack tips pass each other, they are shielded from tensile loading by the other crack causing crack propagation to slow or arrest. With enough continued loading, stress flow through the intact region between the cracks causes more crack growth. The new loading direction, however, is no longer simple tension. It changes to mixed-mode loading causing crack tips to divert toward the other crack. With enough loading, the cracks can intersect leaving a full crack with a "elliptical island" between the cracks. Some real-world examples of such crack growth in concrete or in wood are in Fig. 11.

To evaluate ability of CRAMP to model propagating, offset cracks, we modeled 50×100 mm specimens starting with two 5 mm edge cracks and using 0.5 mm cells in plane stress. The cracks were horizontal but offset from each other by 1, 2, 5, or 10 mm (*i.e.*, 2, 4, 10, or 20 cells). The isotropic

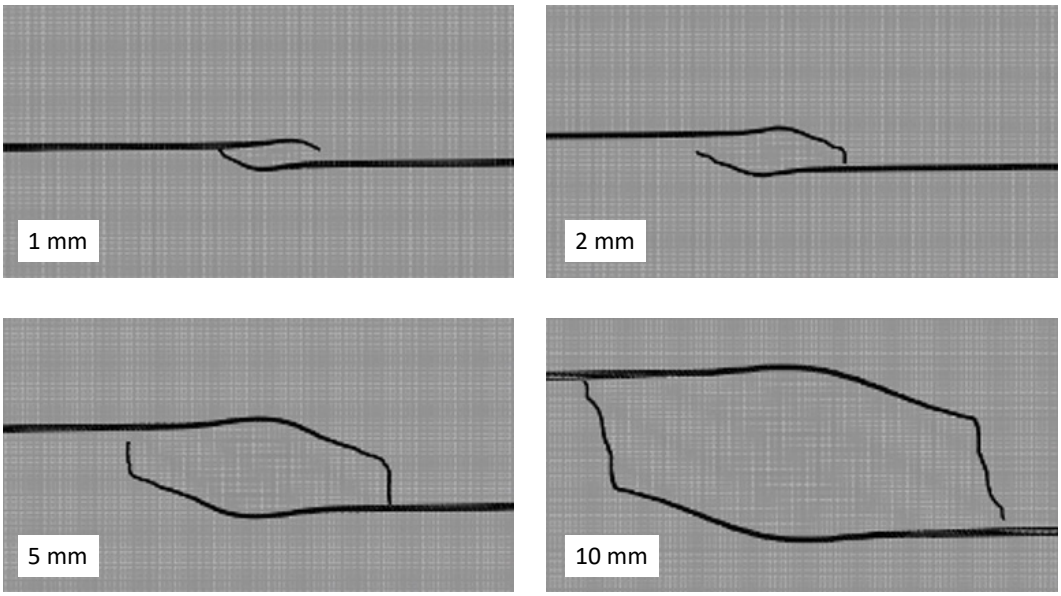

Figure 12: The final state after numerical crack propagation stopped in DENT specimens with two identical initial cracks. The initial cracks were 5 mm long and offset by 1, 2, 5, or 10 mm.

material properties were $E = 2500$ MPa, $\nu = 0.33$, $K_{Ic} = 0.5$ MPa $\sqrt{\text{m}}$, and $J_{Ic} = 100$ J/m^2. Tension was applied by constant displacement rate slow enough to emulate quasi-static loading. The cracks started to propagate at around 0.15% strain and all propagation was complete with less than 1% strain.

Figure 12 shows final state after crack propagation stopped. These simulations matched all expectations. After initiation, the cracks propagated rapidly (close to the material's wave speed) until each crack was about halfway through the specimen width. At this time, the cracks turned to avoid each other and crack growth slowed dramatically. The initial crack growth was pure mode I and self similar. After reaching the specimen mid plane, the interacting crack stress field induced mixed-mode character that resulted in crack deviation. The deviations were initially to avoid the other crack, but with continued loading and changing mixed-mode stress states, each crack turned toward the other crack. These diversions eventually led to crack intersection (or near intersection) and then all crack propagation stopped.

The above results matched real-world observations of crack paths surrounding an "elliptical island." The size of the island depends on initial offset of the two cracks. To visualize both crack speeds and eventual island sizes, Fig. 13A plots $g = 50 - \ell_1 - \ell_2$ where ℓ_i is path length for crack i. These plots start at $g = 40$ mm, which was the initial crack tip separation. As long as cracks remain straight, which they do until passing each other, g measures horizontal gap between the crack tips. After crack initiation at about 0.4 ms, both cracks rapidly propagated until they slightly passed the specimen middle (*i.e.*, g becomes negative). The maximum slope was 800 m/s for each crack for a material with shear wave speed of 967 m/s. After overlapping, shielding of each crack by the other crack caused crack growth to slow. Eventually all crack growth stopped and negative g indicates relative island sizes. Because the cracks curve after the tips pass each other, g overestimates island sizes, but still illustrates trends as the offset between the cracks increases.

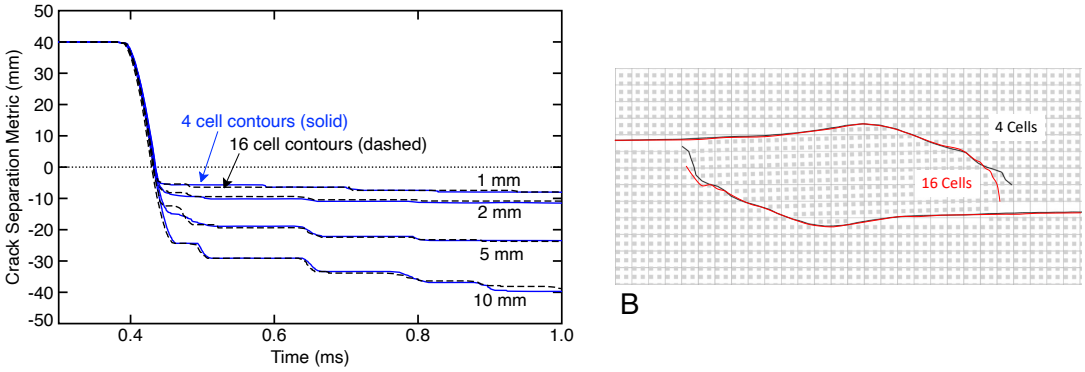

Figure 13: A. Plots of $g = 50 - \ell_1 - \ell_2$ as a function of time for DENT specimens with initial cracks offset by 1, 2, 5, and 10 mm. Solid lines used 4-cell J contours and dashed lines used 16-cell J contours. B. Final crack state for DENT specimens with initial cracks offset by 2 mm found using either 4-cell or 16-cell J contours.

The solid lines in Fig. 13A are calculations using 4-cell J contours. As a result, offset cracks did not encounter any JCICs until the approach of crack intersection meaning the new methods here were only crucial to correctly model final crack intersections. To verify crack-propagation modeling with JCICs, the calculations were repeated using 16-cell contours. These larger contours caused many JCICs and often enclosed the other crack tip meaning contour adaptation was needed. The dashed lines in Fig. 13A show that 16-cell crack propagation matched 4-cell propagation. Figure 13B shows the final crack paths for calculations using 4-cell or 16-cell contours and the results are nearly identical. Although using 16-cell contours is not recommended for general crack propagation, these results further validate FIJC for modeling interacting and intersecting cracks.

Two motivations for using MPM are for modeling explicit cracks and for modeling large deformation problems. To illustrate both capabilities, the 1 mm offset cracks example above was repeated by switching to a Neohookean, large-deformation, hyperelastic material [61, 62] and increasing the toughness 1000 fold such that the cracks did not start propagating until 5% strain instead of 0.15% strain. Because J integral is valid for any elastic material, it remains of viable option for predicting crack growth in large-deformation problems. Figure 14 shows the deformation state after the crack tips passed by each other. The black lines show the position of the tracked crack surfaces, which correctly responded to large-deformation crack opening displacement.

3.7. Multiple Interacting Cracks

For a demonstration example, we repeated the arbitrary, multi-crack problem introduced in Budyn *et al.* [63] and previously modeled by XFEM [63, 64] and MPM [24]. The problem is a 50.8×50.8 mm plate with 10 arbitrarily placed cracks (from figure in [63]). The material is a weak glass with $E = 689.4757$ MPa (10^5 psi), $\nu = 0.3$, $K_{Ic} = 0.87908$ MPa$\sqrt{\text{m}}$, and $\rho = 2.5$ g/cm^3. The MPM model used 128×128 cells corresponding cell size of about 0.40 mm, which matches resolution used in MPM simulation of this specimen in Ref. [24] and similar to resolution that was judged accurate in Section 3.1 for a similar specimen size. This section summarizes results using our new approach and then compares to other methods.

Figure 15 shows simulation results for “Slow” and “Fast” axial loading in the vertical direction (on the left) or biaxial loading (on the right). The slow loading used constant velocity equal to 0.02% of the

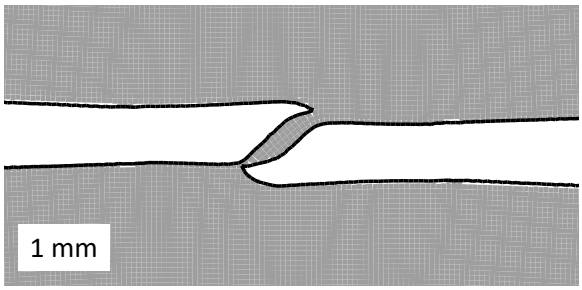

Figure 14: The final state after large-deformation numerical crack propagation stopped in DENT specimens with two identical initial cracks that were 5 mm long and offset by 1 mm. The material was modeled as a Neohookean, large-deformation, hyperelastic material. The black lines are the tracked crack surfaces.

material's wave speed (525 m/sec). During slow loading, all crack tips had rising stress intensity factor until one crack became critical and started to propagate. Because energy release rate typically increases with crack length, the critical crack remained critical until it encountered other cracks or until the sample failed. For axial loading, crack 1 is closest to horizontal and became critical first. It propagated until it intersected cracks 2 and 3. These cracks then propagated to the edges and the sample failed. Once a crack spanned the entire specimen, no further crack propagation occurred. For slow biaxial loading, crack 4 became critical first and propagated until a crack spanned the vertical direction. Crack 5 was the next to become critical. It propagated up to the vertical crack followed by crack 2 propagating to the edge to create a second crack spanning the horizontal direction. Once two cracks spanning vertical and horizontal directions were complete, no further crack propagation occurred. Seven cracks during axial loading and six cracks during biaxial loading never propagated.

The "Fast" loading increased the constant velocity 50-fold to 1% of the material's wave speed such that stress waves from the ends meet in the middle at 1% strain. This high rate caused significant changes in the failure processes. At high loading rate, multiple cracks can become critical and propagate before that propagation causes other cracks to become deactivated. For fast axial loading, all cracks except one have some crack propagation. The crack spanning the specimen at failure was similar to slow loading, but the final failure state had much more crack propagation at other cracks. For biaxial loading, every crack propagated and the final damage state corresponded to the specimen dividing into 11 fragments.

At other loading rates, the final biaxial damage state transitioned from two dominant cracks for loading rates below 0.02% of the material's wave speed to increasing damage at higher rates and finally to linking of all cracks into fragments at loading rates above 1% of the material's wave speed. The 128×128 spatial resolution was judged adequate because higher resolution did not change final damage states.

Budyn *et al.* [63] modeled only axial loading by XFEM using "crack length control" where each increment increased the length of active crack(s) and solved for stable state. The damage progression and final damage states by XFEM or MPM are nearly identical although XFEM showed more crack propagation on the right edge of crack 2. Figure 16 compares stress-strain curves in slow rate loading by our MPM results (solid line) to XFEM results (dashed line). The two methods are identical up to failure. At failure, XFEM displays "snap back" where both force and displacement decrease with increased crack lengths. This regime maps stable states but its occurrence implies crack growth in monotonic loading would be unstable. The explicit, dynamic CRAMP calculations fully model the expected unstable crack growth, which occurs at close to the material's wave speed once initiated. The load rapidly dropped

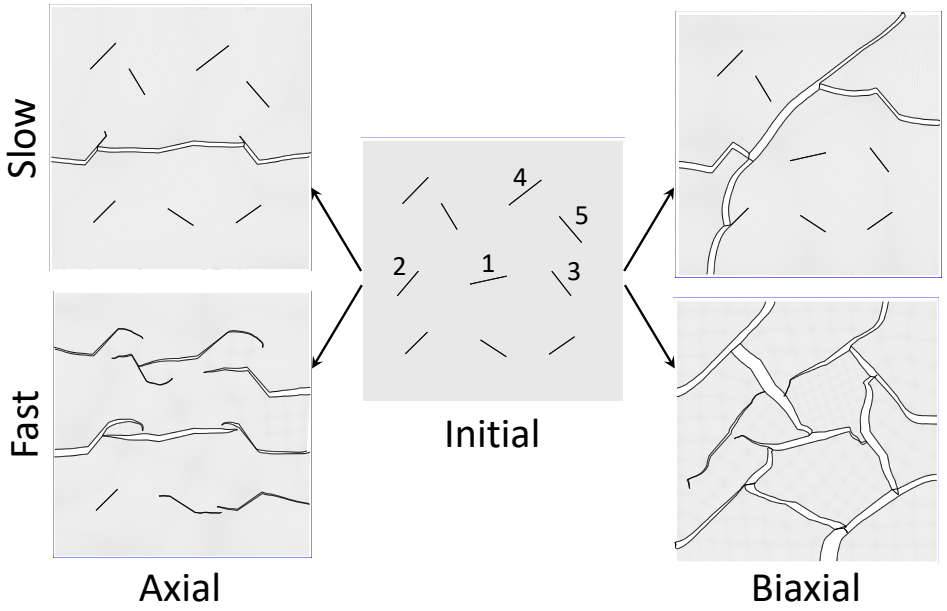

Figure 15: Failure of a specimen starting with multiple cracks (middle) by either axial loading in the vertical direction (on the left) or biaxial load (on the right) and by either "Slow" (top) or "Fast" loading rates. The dark lines show CRAMP algorithm tracking of crack surfaces.

to zero force followed by dynamic force oscillations. Both results are valid and would likely agree for configurations that do not have snap back response (unfortunately, all examples for comparison in [63] have snap back response).

Satula *et al.* [64] used XFEM to model biaxial loading. Despite their use of a static method, their results are very close to the fast, biaxial loading results in Fig. 15 rather than the slow, quasi-static damage they should match. This inconsistency is likely caused by their unrealistic assumption that "all cracks are allowed to grow at the same rate" [64] rather then the XFEM recommendation to only propagate "active cracks" [63]. Our slow simulations of biaxial loading show that six of the cracks do not propagate at all resulting in final damage state with two dominant cracks. Although no analytical or experimental results are available for this multiple crack example, our rate effects are consistent with other experiments that impacted soda-lime glass spheres at speeds ranging from 20–120 m/s (or from less than 1% to greater than 1% of estimated material wave speed) [65]. Those experiments had single Hertzian cracks at low speeds, increasing fragmentation at moderate speeds, and degradation into a powder at high speeds [65].

Adibaskoro *et al.* [24] modeled biaxial loading by explicit dynamic MPM with a final damage state similar to XFEM by Satula *et al.* [64] and the "Fast" biaxial loading in Fig. 15. Although they did not mention a loading rate, time stamps on their figures of damage propagation indicate slow loading. Nevertheless, their results match expectations for fast rather than slow loading. A possible explanation is their assumption that crack propagation occurs "when the maximum principal stress reaches the material strength, where the direction is perpendicular to the direction of the maximum principal stress" [24] rather than by J integral or energy release rate methods. A problem with crack tip stress methods is

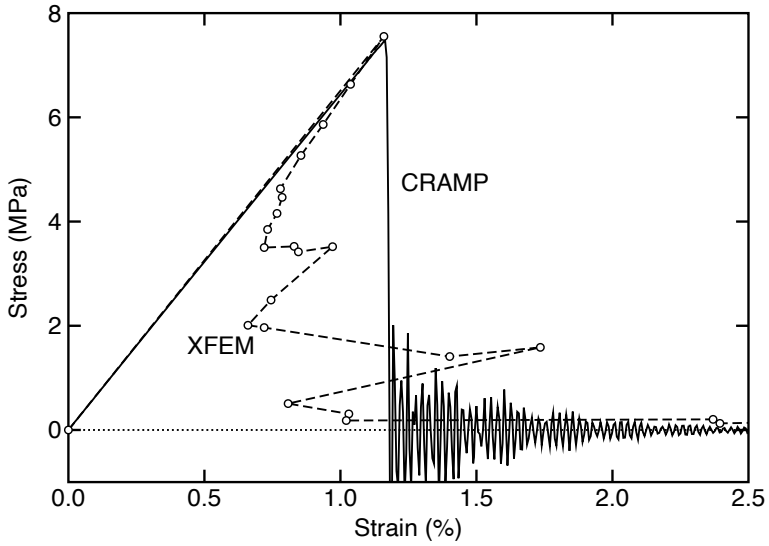

Figure 16: Stress-strain curves for axial loading on the multiple crack problem in Fig. 15 by CRAMP or by XFEM (from [63]). Our results used even slower rate of 0.005% of the material's wave speed, with no change except for frequency of post-failure oscillations when plotted *vs.* strain, and higher resolution of 200×200 to help resolve the sharp failure peak (a 3.5% difference from lower-resolution results).

that those stresses are prone to noise in dynamic crack propagation and their magnitude is affected by resolution. These observations probably explain why they got chaotic crack growth in simple, symmetric problems (J integral methods on similar problems give smooth crack growth) and why their failure loads are resolution dependent (because higher resolution finds higher stresses at crack tips; see Fig. 21 in [24]). Finally, the Adibaskoro *et al.* [24] model needed 51 grids to simulate this multicrack example. Our method needed only four crack velocity fields and the simulated "natural" propagation never resulted in a node interacting with more than two cracks. In other words, this example suggests that handling two explicit cracks on each node is sufficient for many practical calculations.

4. Conclusions

By allowing nodes to have four crack velocity fields, CRAMP can model two explicit cracks in MPM including closely spaced cracks, intersecting cracks, and even two cracks in the same background cell (all cases of CSCs). Crack propagation of multiple explicit cracks can be modeled by calculating J integral along with K_I and K_{II} stress intensity factors. These fracture mechanics calculations, however, must be modified when the J contour for one crack is intersected by one or more other cracks (JCICs). Intersecting cracks that cross the entire contour can be handled by integrating the two sides of the crack within the contour. If another crack tip is within the J contour or in any cell adjacent to the contour, the contour must be adapted to avoid the second tip. FIJC provides J calculations with sufficient accuracy, even for a regular mesh, meaning that MPM can be recommended as a numerical tool for explicit crack propagation simulations with multiple cracks.

One real-world example is modeling hydraulic fracturing in rock [21, 22, 66, 67]. When hydraulic fluid induces rock crack propagation that reaches other cracks or rock interfaces, does propagation stop

or does it penetrate the second crack and continue propagating? If propagation stops, does hydraulic fluid flow into the linked crack allowing for more crack propagation? Such modeling requires methods to handle intersecting cracks, crack tractions, background geological pressures, and contact mechanics. This topic will be the subject of a future publication.

Many real-world problems will likely exceed the capabilities of the explicit crack methods derived here. Examples include problems with too many cracks or problems that require modeling branching or initiation of new cracks that are not modeling by J integral methods. Researchers attacking such problems often turn to alternate methods that claim to implement concepts of fracture mechanics without the need to model explicit cracks. The explicit crack methods in this paper could be a tool for verifying such claims. The process would be to model problems that *can be done* with this explicit crack modeling and use those results as reference fracture mechanics results for validating alternate computational damage methods.

Acknowledgement

Funding for this projects was provided, in part, by FracGeo, LLC, USA. Cracking hashing methods (see the appendix), which are vital for efficiency when modeling many cracks, were implemented by Chad Hammerquist, a former Senior Researcher at FracGeo.

Appendix

This appendix only describes algorithms for determining crack velocity fields for particles, $v(p,i)$, and for crack surface positions, $v(s,i)$. Once these are determined, most MPM calculations are close to prior multimaterial MPM methods. In other words, nodes are divided into multiple velocity fields as needed, calculations proceed for each field, and contact is implemented in nodes that share velocity fields. A few special crack calculations and are given in text of the paper. The algorithm for finding $v(p,i)$ is enclosed by a loop over all particles and done during each time step's initialization phase. The main particle loop proceeds as follows:

1. Find element e containing particle p.
2. Find nodes i_k, where k is from 1 to number of nodes having non-zero shape functions for particle p.
3. If element e does not "see" any cracks, set all $v(p,i_k)=0$ and continue to next p (note: see below for prehashing method to determine which elements "see" cracks). If the element does see cracks, repeat the following steps for each i_k:
   (a) Set $\alpha=0$ to count number of cracks needing velocity fields on node i_k.
   (b) For each crack seen by element e (from a list determined in prehashing explained below):
      i. Trace a line from particle p to node i_k and return $s=0$, 1, or 2, meaning the line does not cross the crack, crosses it from above, or crosses it from below, respectively (use an efficient ray-tracing method and account for multiple crossings of the same crack).
      ii. Whenever $s=1$ or 2, increment α and set s_α, n_α, and $\hat{\boldsymbol{n}}_\alpha$, to returned s value, the crack number, and normal vector from crack geometry at the point of intersection.

iii. If α is now 2, exit this crack loop because this algorithm only handles two cracks on a single node.

(c) After checking all cracks, if $\alpha = 0$, then $v(p, i_k) = 0$ and continue to next i_k. But if $\alpha > 0$, use steps below to set $v(p, i_k)$ and to create crack velocity fields as needed.

When step 3.c above finds $\alpha = 1$ or 2, use the following steps to create all needed crack velocity fields:

A If $\alpha = 2$, try to fit the p-i_k pair into crack velocity field 3 (the only field with two cracks):

(a) If $c_{i_k,3}$ is empty, allocate crack velocity field 3 for node i_k, set $v(p, i_k) = 3$, and set $s^{(1)}_{i_k,3} = s_1$, $n^{(1)}_{i_k,3} = n_1$, $s^{(2)}_{i_k,3} = s_2$, and $n^{(2)}_{i_k,3} = n_2$. Here double subscripts (i, j) means s and n values for crack velocity field $c_{i,j}$ and superscripts (1) and (2) are for the two cracks in field 3.

(b) If $c_{i_k,3}$ was previously allocated, compare s_α and n_α to $s^{(m)}_{i_k,3}$ and $n^{(m)}_{i_k,3}$. If they match both cracks, set $v(p, i_k) = 3$. Failure to match means an unexpected crack geometry that can be handled as follows:

i. If n_α's match the two cracks, but s_α's match only one, Issue warning #1 (see below) and try to continue with one crossing cracking by setting s_1 and n_1 to the matched crack and jumping to the $\alpha = 1$ case.

ii. If n_α's match both cracks, but s_α's are wrong for both cracks, Issue warning #1 and set $v(p, i_k) = 0$.

iii. If n_α's match only one crack, the mismatched crack is a third crack for node i_k. Issue warning #2 and try to continue with one crossing crack by setting s_1 and n_1 to the matched crack and jumping to the $\alpha = 1$ case.

iv. If neither n_α matches a crack in field 3, node i_k "sees" four or more cracks. Issue warning #2 and set $v(p.i_k) = 0$.

B If $\alpha = 1$, either because only one crack was found or because the previous step deleted one of two found cracks, try to fit the p-i_k pair into crack velocity field 0, 1, or 2:

(a) If $c_{i_k,1}$ is empty, allocate crack velocity field 1, set $v(p, i_k) = 1$, and set $s^{(1)}_{i_k,1} = s_1$ and $n^{(1)}_{i_k,1} = n_1$.

(b) If $c_{i_k,1}$ was previously allocated and $n_1 = n_{i_k,1}$, do one of following two options:

i. If $s_1 = s_{i_k,1}$, set $v(p, i_k) = 1$.

ii. If $s_1 \neq s_{i_k,1}$, issue warning #1 and set $v(p, i_k) = 0$.

(c) If $c_{i_k,1}$ was previously allocated but $n_1 \neq n_{i_k,1}$, try to use crack velocity field 2:

i. If $c_{i_k,2}$ is empty, allocate crack velocity field 2, set $v(p, i_k) = 2$, and set $s^{(1)}_{i_k,2} = s_1$ and $n^{(1)}_{i_k,2} = n_1$.

ii. If $c_{i_k,2}$ was previously allocated and $n_1 = n_{i_k,2}$, do one of following two options:

A. If $s_{i_k,2} = s_2$, set $v(p, i_k) = 2$.

B. If $s_1 \neq s_{i_k,2}$, issue warning #1 and set $v(p, i_k) = 0$.

(d) If both $c_{i_k,1}$ and $c_{i_k,2}$ were previously allocated but n_1 matches neither crack, node i_k "sees" a third crack. Issue warning #2 and set $v(p, i_k) = 0$

C Whenever $v(p, i_k)$ is set, add $\hat{\boldsymbol{n}}_\alpha$ to $\hat{\boldsymbol{n}}_{i_k,j}$ in the velocity field. For field 3, add the two normal vectors to $\hat{\boldsymbol{n}}^{(1)}_{i_k,j}$ and $\hat{\boldsymbol{n}}^{(2)}_{i_k,j}$. The average of these normals is used in contact calculations. Because field 0 corresponds to p-i_k pairs where the line did not cross a crack, it does not have a normal.

Crack velocity field initialization is the most time consuming CRAMP task because it needs a ray tracing calculation for each particle–node pair possibly crossing each crack. Calculations can be greatly improved by parallelization and by crack hashing. Because particle calculations are mostly independent, the main particle loop can be made parallel. Because different particles may interact with the same node, however, changes to nodal values in that loop must be done in critical code (*i.e.*, one thread at a time) to avoid race conditions. The critical nodal calculations are allocation of crack velocity fields (steps A.a, B.a, and B.c.1 in the crack velocity field block) and tracking of crack normals (step C in the crack velocity field block). Crack hashing is done as follows:

- Mark all elements in the background grid as not seeing any cracks.
- Loop over all crack segments (or facets in 3D) in the problem and locate the extent of elements containing that segment (or facet). Expand that extent by one element in all directions to allow for generalized interpolation MPM shape functions.
- Add the crack containing that segment (or facet) to a list of cracks seen by each element within the segment's (or facet's) extent.

The resulting element-based lists of "seen" cracks are exploited in steps 3 and 3.b of the main particle loop to skip a vast majority (in most problems) of ray-tracing calculations needed to initialize crack velocity fields. Hashing methods are especially effective in problems with many cracks.

In well-behaved calculations, a node that sees only a single crack always terminates in steps B.a, B.b.i, B.c.i, or B.c.ii. and nodes seeing two cracks always terminates in A.a or A.b (first condition). All other cases are "unexpected" crack geometries or a node that sees three or more cracks (see A.b.ii, A.b.iii, A.b.iv, B.b.ii, B.c.ii.B, and B.d). The above algorithm handles all such "unexpected" cases by issuing a warning, implementing a rational alternative, and continuing. One could alternatively abort the simulations. Two classes of warnings are:

- Warning #1 means the algorithm detected lines that cross the same crack from both above and below. In most ray-tracing algorithms, this can only occur if an end point of the line lies exactly on the crack plane. The solution is to interpret one of those lines as if it did not cross the crack surface. The approach taken above is to interpret any mismatch as a non-crossing line, *i.e.*. to proceed as if the crack was not crossed. This approach fails if the erroneous cross occurs when the needed crack velocity field is empty, but otherwise is correct. Fortunately, the "otherwise" should be more common because erroneous crosses are uncommon.
- Warning #2 means the problem has three or more cracks interacting with the same node. Formally, this state means the algorithm has failed. The above approach, however, is to ignore excess cracks and proceed with the first two cracks found at each node.

Warning #1 usually proceeds with the correct choice. To avoid them in the first few time steps, initial cracks positions should avoid being exactly on grid lines or passing through the centers of material points (any small difference is sufficient). Warning #2 means the calculations may suffer in accuracy, but an output warning can inform users of potential inaccuracy.

Implementation of the above algorithm for $n = 2$ reaches 4 valid fields and 10 cases issuing warnings. Extending this algorithm to $n > 2$ cracks would need to handle 2^n valid fields, but also need to handle an increasing number of unexpected cases. Implementing $n = 3$ would have 8 valid fields and 26 issuing warnings and $n = 4$ would have 16 valid fields and 42 issuing warnings. The coding for $n = 2$ is already complex. If most real-world problems can be solved with $n = 2$, the extra complexing for $n > 2$ can be avoided.

Finding $v(s, i)$ for extrapolating to (Eqs. (5) and (6)) or from (Eq. (7)) crack surface particles is similar to finding $v(p, i)$ except that it is not pre-calculated in velocity field initialization. A separate calculation is needed. The process is the same as the main particle loop above except material point particle p is replaced with surface particle s. For efficiency, before starting step 3 of the particle loop, the code can check for a field on node i appropriate for side s_s of crack n_s containing surface particle s. The inserted check is:

1. If n_s matches either $n_{i_k,1}$ or $n_{i_k,2}$ (let $j = 1$ or 2 be the one it matches), select the field as follows:
   (a) If s_s matches $s_{i_k,j}$, set $v(s, i_k) = j$. But, if a line from surface particle to node i crosses another crack, change $v(s, i_k)$ to 3, if available, or else proceed to step 3 below.
   (b) If s_s does not match $s_{i_k,j}$, set $v(s, i_k) = 0$. But, if a line from surface particle to node i crosses another crack, change $v(s, i_k)$ to $3 - j$, if available, or else proceed to step 3 below.
2. If n_s matches both crack and side for one of the cracks in $c_{i_k,3}$, check if line from surface particle to node i crosses the other crack. If it does, set $v(s, i_k) = 3$, or else proceed to step 3 below.
3. If previous checks did not find a crack velocity field, proceed with step 3 in the main particle loop. If an existing field is found, set that to $v(s, i_k)$

References

[1] A. S. D. Wang, F. W. Crossman, Initiation and growth of transverse cracks and edge delamination in composite laminates: Part 1. an energy method, J. Comp. Mat Supplement 14 (1980) 71–106.

[2] F. W. Crossman, W. J. Warren, A. S. D. Wang, J. G. E. Law, Initiation and growth of transverse cracks and edge delamination in composite laminates: Part 2. experimental correlation, J. Comp. Mat Supplement 14 (1980) 89–108.

[3] J. A. Nairn, S. Hu, The initiation and growth of delaminations induced by matrix microcracks in laminated composites, Int. J. Fract. 57 (1992) 1–24.

[4] J. A. Nairn, Predicting layer cracks in cross-laminated timber with evaluations of strategies for suppressing them, European Journal of Wood and Wood Products 77 (3) (2019) 405–419.

[5] Y. Leterrier, L. Boogh, J. Andersons, J.-A. E. Månson, Adhesion of silicon oxide layers on poly(ehtylene terepthalate). i: Effect of substrate properties on coating's fragmentation process, J. Polym. Sci., Part B: Polymer Physics 35 (1997) 1449–1461.

[6] A. G. Atkins, Y. W. Mai, Elastic and Plastic Fracture, John Wiley & Sons, New York, 1985.

[7] A. A. Griffith, The phenomena of rupture and flow in solids, Phil. Trans. 221A (1920) 163–198.

[8] M. F. Kanninen, C. H. Popelar, Advanced Fracture Mechanics, Oxford University Press, New York, 1985.

[9] T. Belytschko, T. Black, Elastic crack growth in finite elements with minimal remeshing, Int. J. for Numerical Methods in Engineering 45 (1999) 601–620.

[10] N. Moës, J. Dolbow, T. Belytschko, A finite element method for crack growth without remeshing, Int. J. for Numerical Methods in Engineering 46 (1999) 131–150.

[11] J. A. Nairn, Material point method calculations with explicit cracks, Computer Modeling in Engineering & Sciences 4 (2003) 649–664.

[12] E. F. Rybicki, M. F. Kanninen, A finite element calculation of stress intensity factors by a modified crack closure integral, Eng. Fract. Mech. 9 (1977) 931–938.

[13] J. A. Nairn, Generalized crack closure analysis for elements with arbitrarily-placed side nodes and consistent nodal forces, Int. J. Fracture 171 (2011) 11–22.

[14] J. R. Rice, A path independent integral and the approximate analysis of strain concentration by notches and cracks, J. Applied Mech. June (1968) 379–386.

[15] J. F. Yau, S. S. Wang, H. T. Corten, A mixed-mode crack analysis of isotropic solids using conservation laws of elasticity, Journal of Applied Mechanics 47 (1980) 335–341.

[16] C. Daux, N. Moës, J. Dolbow, N. Sukumari, T. Belytschko, Arbitrary branched and intersecting cracks with the extended finite element method, Int. J. for Numerical Methods in Engineering 48 (2000) 1741–1760.

[17] Y. Guo, J. A. Nairn, Calculation of J-integral and stress intensity factors using the material point method, Computer Modeling in Engineering & Sciences 6 (2004) 295–308.

[18] J. A. Nairn, Analytical and numerical modeling of r curves for cracks with bridging zones, Int. J. Fract. 155 (2009) 167–181.

[19] S. G. Bardenhagen, J. A. Nairn, H. Lu, Simulation of dynamic fracture with the material point method using a mixed j-integral and cohesive law approach, Int. J. Fracture 170 (2011) 49–66.

[20] Y. Guo, J. Nairn, Simulation of dynamic 3D crack propagation within the material point method, Computer Modeling in Engineering and Sciences 113 (4) (2017) 389–410.

[21] Y. E. Aimene, J. A. Nairn, Modeling multiple hydraulic fractures interacting with natural fractures using the material point method, in: SPE/EAGE European Unconventional Conference and Exhibition, Vienna, Austria, Feb. 25-27, 2014.

[22] Y. E. Aimene, A. Ouenes, Geomechanical modeling of hydraulic fractures interacting with natural fractures — validation with microseismic data and tracer data from the Marcellus and Eagle Ford., Interpretation 3 (3) (2015) 1–18.

[23] J. A. Nairn, Material point method (NairnMPM) and finite element analysis (NairnFEA) open-source software, http://osupdocs.forestry.oregonstate.edu/index.php/Main_Page (2017).

[24] T. Adibaskoro, S. Bordas, W. T. Solowski, S. Hostikka, Multiple dicreate crack initiation and propagation in material point method, Engineering Fracture Mechanics 301 (2024) 109918.

[25] S. Fan, W. Guilin, Z. Liang, W. Runqiu, C. Tianci, O. Xiaotian, Material point method for propagation of multiple branched cracks based on classical fracture mechanics, Comput. Methods Appl. Mech. Engrg 386 (2021) 114116.

[26] S. G. Bardenhagen, J. E. Guilkey, K. M. Roessig, J. U. Brackbill, W. M. Witzel, J. C. Foster, An improved contact algorithm for the material point method and application to stress propagation in granular material, Computer Modeling in Engineering & Sciences 2 (2001) 509–522.

[27] H. Schreyer, D. Sulsky, S.-J. Zhou, Modeling delamination as a strong discontinuity with the material point method, Computer Methods in Applied Mechanics and Engineering 191 (23–24) (2002) 2483 – 2507. doi:10.1016/S0045-7825(01)00409-1.

[28] J. Burghardt, R. Brannon, J. Guilkey, A nonlocal plasticity formulation for the material point method, Comput. Methods Appl. Mech. Engrg. 225–228 (2012) 55–64.

[29] L. Kachanov, Time of rupture process under creep conditions, Izv. Akad. Nauk SSR, Otd. Tekh. Nauk 8 (1958) 26–31.

[30] J. Chaboche, Le concept de contrainte effective appliqué à l'élasticité et à la viscoplasticité en présence d'un endommagement anisotrope, in: J.-P. Boehler (Ed.), Mechanical Behavior of Anisotropic Solids / Comportment Méchanique des Solides Anisotropes, Springer Netherlands, 1979, pp. 737–760.

[31] J. Oliver, Modelling strong discontinuities in solid mechanics via strain softening constitutive equations. part 1: Fundamentals, International Journal for Numerical Methods in Engineering 39 (21) (1996) 3575–3600. doi:10.1002/(SICI)1097-0207(19961115)39:21<3575::AID-NME65>3.0.CO;2-E.

[32] J. Oliver, Modelling strong discontinuities in solid mechanics via strain softening constitutive equations. part 2: Numerical simulation, International Journal for Numerical Methods in Engineering 39 (21) (1996) 3601–3623. doi:10.1002/(SICI)1097-0207(19961115)39:21<3601::AID-NME64>3.0.CO;2-4.

[33] J. A. Nairn, C. C. Hammerquist, Y. E. Aimene, Numerical implementation of anisotropic damage mechanics, Int. J. for Numerical Methods in Engineering 112 (12) (2017) 1846–1868. doi:DOI:0.1002/nme.5585.

[34] B. Bourdin, G. A. Francfort, J.-J. Marigo, The variational approach to fracture, J. Elasticity 91 (2008) 5–148.

[35] C. Miehe, M. Hofacker, F. Welschinger, A phase field model for rate-independent crack propagation: Robust algorithmic implementation based on operator splits, Computer Methods in Applied Mechanics and Engineering 199 (2010) 2765–2778.

[36] E. G. Kakouris, S. P. Triantafyllou, Phase-field material point method for brittle fracture, Int J Numer Meth Engng. 112 (2017) 1750–1776.

[37] E. G. Kakouris, S. P. Triantafyllou, Material point method for crack propagation in anisotropic media: a phase field approach, Arch Appl Mech 88 (2018) 287–316.

[38] M. Homel, E. B. Herbold, Field-gradient partitioning for fracture and frictional contact in the material point method, Int. J. for Numerical Methods in Engineering 109 (7) (2016) 1013–1044.

[39] Y. Guo, J. A. Nairn, Three-dimensional dynamic fracture analysis in the material point method, Computer Modeling in Engineering & Sciences 16 (2006) 141–156.

[40] S. G. Bardenhagen, E. M. Kober, The generalized interpolation material point method, Computer Modeling in Engineering & Sciences 5 (2004) 477–496.

[41] D. Sulsky, Z. Chen, H. L. Schreyer, A particle method for history-dependent materials, Comput. Methods Appl. Mech. Engrg. 118 (1994) 179–186.

[42] S. G. Bardenhagen, J. U. Brackbill, D. Sulsky, The material point method for granular materials, Computer Methods in Applied Mechanics and Engineering 187 (2000) 529–541.

[43] J. U. Brackbill, H. M. Ruppel, FLIP: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions, Journal of Computational Physics 65 (2) (1986) 314 – 343.

[44] J. A. Nairn, Numerical simulation of orthogonal cutting using the material point method, Engineering Fracture Mechanics 149 (2015) 262–275.

[45] C. C. Hammerquist, J. A. Nairn, A new method for material point method particle updates that reduces noise and enhances stability, Computer Methods in Applied Mechanics and Engineering 318 (2017) 724–738.

[46] J. A. Nairn, C. C. Hammerquist, Material point method simulations using an approximate full mass matrix inverse, Computer Methods in Applied Mechanics and Engineering 337 (2021) 113667.

[47] J. A. Nairn, S. G. Bardenhagen, G. S. Smith, Generalized contact and improved frictional heating in the material point method, Computational Particle Mechanics 5 (3) (2018) 285–296. doi:10.1007/s40571-017-0168-1.

[48] J. A. Nairn, Modeling of imperfect interfaces in the material point method using multimaterial methods, Computer Modeling in Eng. & Sci. 92 (3) (2013) 271–299.

[49] J. A. Nairn, Numerical implementation of imperfect interfaces, Computational Materials Science 40 (2007) 525–536.

[50] V. Lemiale, A. Hurmane, J. A. Nairn, Material point method simulation of equal channel angular pressing involving large plastic strain and contact through sharp corners, Computer Modeling in Eng. & Sci. 70 (1) (2010) 41–66.

[51] J. A. Nairn, C. C. Hammerquist, G. Smith, New material point method contact algorithms for improved accuracy, large-deformation problems, and proper null-space filtering, Computer Methods in Applied Mechanics and Engineering 362 (2020) 112859.

[52] J. A. Nairn, Y. E. Aimene, A re-evaluation of mixed-mode cohesive zone modeling based on strength concepts instead of traction laws, Engineering Fracture Mechanics 248 (2021) 107704.

[53] T. Belytschko, Y. Y. Lu, L. Gu, Element-free Galerkin methods, Int. J. Num. Meth. Engrg. 37 (1994) 229–256.

[54] T. Belytschko, M. Tabbara, Dynamic fracture using element-free Galerkin methods, Int. J. Numer. Methods in Engr. 39 (1996) 923–938.

[55] D. J. Organ, M. Fleming, T. Belytschko, Continuous meshless approximations for nonconvex bodies by diffraction and transparency, Computational Mechanics 18 (1996) 225–235.

[56] T. Nishioka, S. N. Atluri, Path-independent integrals, energy release rates, and general solutions of near-tip fields in mixed-mode dynamic fracture mechanics, Engr. Fract. Mech. 18 (1983) 1–22.

[57] T. Nishioka, Recent Developments in Computational Dynamic Fracture Mechanics, 1995, pp. 1–58.

[58] G. Bao, Z. Suo, Remarks on crack-bridging concepts, Appl. Mech. Rev. 45 (6) (1992) 355–366.

[59] F. Erdogan, G. C. Sih, On the crack extension in plates under plane loading and transverse shear, Journal of Basic Engineering, ASME 85 (1963) 519–527.

[60] A. S. Saleemi, J. A. Nairn, The plane-strain essential work of fracture as a measure of the fracture toughness of ductile polymers, Polym. Eng. and Sci. 30 (1990) 211–218.

[61] R. W. Ogden, Non-Linear Elastic Deformations, Ellis-Harwood, New York, 1984.

[62] O. C. Zienkiewicz, R. L. Taylor, The Finite Element Methods for Solid and Structural Mechanics, Elsevier Butterworth-Heinemann, Oxford, UK, 2000.

[63] E. Budyn, G. Zi, N. Moës, T. Belytschko, A method for multiple crack growth in brittle materials without remeshing, International Journal for Numerical Methods in Engineering 61 (10) (2004) 1741–1770.

[64] D. Sutula, P. Kerfriden, T. van Dam, S. P. Bordas, Minimum energy multiple crack propagation. part iii: Xfem computer implementation and applications, Engineering Fracture Mechanics 191 (2018) 257–276. doi:https://doi.org/10.1016/j.engfracmech.2017.08.004.

[65] A. Salman, D. Gorham, The fracture of glass spheres, Powder Technology 107 (2000) 179–185.

[66] R. L. Kranz, Microcracks in rocks: a review, Tectonophysics 100 (1983) 449–480.

[67] N. Warpinski, L. Teufel, Influence of geologic discontinuities on hydraulic fracture propagation, Journal of Petroleum Technology 39 (02) (1987) 209–220. doi:10.2118/13224-PA.