REPORT DOCUMENTATION PAGE

Form Approved OMB NO. 0704-0188

The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggesstions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA, 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any oenalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.
PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
25-10-2016	Final Report	15-Jul-2013 - 14-Jul-2016

4. TITLE AND SUBTITLE
Final Report: Mathematical Models and Advanced Numerical Methods for Complex Flows and Structures

5a. CONTRACT NUMBER
W911NF-13-1-0249

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
611102

6. AUTHORS
James Glimm, Roman Samulyak, Xiangmin Jiao

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES
Research Foundation of SUNY at Stony Brc
W-5510 Melville Library

Stony Brook, NY 11794 -3362

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS (ES)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSOR/MONITOR'S ACRONYM(S)
ARO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
63848-MA.41

12. DISTRIBUTION AVAILIBILITY STATEMENT
Approved for Public Release; Distribution Unlimited

13. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT
Our results pertain to two complex flow problems: Brittle fracture and turbulence. Our results are partly numerical and algorithmic and partly theoretical, in formulation of improved models, equations and physical formulation. A new mass conservative mesoscale model for the simulation of brittle fracture of solid materials has been developed. In our previous work, we represented the solids by spring networks, using an algorithm based on the energy minimization of the network of triangular springs, with critical strain and splitting of over stressed bonds and connected to them nodes ensuring the conservation of mass during the crack evolution. The applicability of the

15. SUBJECT TERMS
Brittle Fracture, Failure Waves, Numerical Geometry of Surfaces, Turbulence and Mixing

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			James Glimm
UU	UU	UU	UU		19b. TELEPHONE NUMBER: 631-632-8355

Standard Form 298 (Rev 8/98)
Prescribed by ANSI Std. Z39.18

Report Title

Final Report: Mathematical Models and Advanced Numerical Methods for Complex Flows and Structures

ABSTRACT

Our results pertain to two complex flow problems: Brittle fracture and turbulence. Our results are partly numerical and algorithmic and partly theoretical, in formulation of improved models, equations and physical formulation.

A new mass conservative mesoscale model for the simulation of brittle fracture of solid materials has been developed. In our previous work, we represented the solids by spring networks, using an algorithm based on the energy minimization of the network of triangular springs, with critical strain and splitting of over stressed bonds and connected to them nodes ensuring the conservation of mass during the crack evolution. The applicability of the network-based brittle fracture method is limited by its compatibility with established engineering codes for solid dynamics, which are primarily based on finite element methods. In the second phase of the project, we focused on the adaptation of the brittle fracture method to the finite element framework in 2D and 3D, thus extending capabilities of finite element methods for the description of fracture dynamics. While the core of our new method is still based on energy minimization, our new code is based on a simplified version of the ALE (Arbitrary Lagrangian Eulerian) formulation to allow large displacement and arbitrary mesh motion, coupled with mesh adaptivity to improve accuracy and stability. The adoption of the finite element method allows the new software to be interoperable with standard FEM codes widely used for material studies, and the new approach also improves the accuracy of simulation of real materials. We have also developed a formulation of the contact mechanics using collision detection and the method Lagrange multipliers. The code has been parallelized for the use on multi core CPU machines. The core FEM code (with the fracture algorithms turned off) has passed several verification tests. The FEM code with fracture has been applied to the simulation of numerous fracture regimes associated with slow deformations and fast failure waves.

For theoretical models of brittle fracture, we propose (for work still in progress a fundamental framework based on thermodynamic principles, reformulated constitutive laws for fractured material, and a consistent time dependent evolution of the fracture progress.

For turbulence modeling (for work still in progress), we realize the turbulent dissipation rate and the related square of the strain matrix as solutions of a log normal stochastic random field equation. This result extends the Kolmogorov-Obukhov 1962 theory. Parameterization of the equation is accomplished through new scaling laws.

Enter List of papers submitted or published that acknowledge ARO support from the start of the project to the date of this printing. List the papers, including journal references, in the following categories:

(a) Papers published in peer-reviewed journals (N/A for none)

Received		Paper
08/07/2014	8.00	Abhishek Murthy, Ezio Bartocci, Flavio H. Fenton, James Glimm, Richard A. Gray, Elizabeth M. Cherry, Scott A. Smolka, Radu Grosu. Curvature Analysis of Cardiac Excitation Wavefronts, IEEE/ACM Transactions on Computational Biology and Bioinformatics, (03 2013): 323. doi: 10.1109 /TCBB.2012.125
08/07/2014	2.00	Tongfei Guo, Shuqiang Wang, Roman Samulyak. Sharp Interface Algorithm for Large Density Ratio Incompressible Multiphase Magnetohydrodynamic Flows, Procedia CS, (01 2013): 511. doi: 10.1016/j.procs.2013.05.215
08/07/2014	3.00	David Sharp, James Glimm, Gianluca Iaccarino. Quantification of margins and uncertainties using multiple gates and conditional probabilities, Reliability Engineering & System Safety, (06 2013): 99. doi: 10.1016/j.ress.2012.11.026
08/07/2014	4.00	Cao Lu, Xiangmin Jiao, Nikolaos Missirlis. A hybrid geometric?+?algebraic multigrid method with semi-iterative smoothers, Numerical Linear Algebra with Applications, (03 2014): 221. doi: 10.1002/nla.1925
08/07/2014	5.00	J. Melvin, R. Kaufman, H. Lim, T. Kaman, P. Rao, J. Glimm. Macro and micro issues in turbulent mixing, Science China Technological Sciences, (09 2013): 2355. doi: 10.1007/s11431-013-5340-0
08/07/2014	6.00	J. Melvin, P. Rao, R. Kaufman, H. Lim, Y. Yu, J. Glimm, D.H. Sharp. Atomic scale mixing for inertial confinement fusion associated hydro instabilities, High Energy Density Physics, (06 2013): 288. doi: 10.1016/j.hedp.2013.01.007
08/07/2014	7.00	J. Melvin, P. Rao, R. Kaufman, H. Lim, Y. Yu, J. Glimm, D. H. Sharp. Turbulent Transport at High Reynolds Numbers in an Inertial Confinement Fusion Context, Journal of Fluids Engineering, (07 2014): 912061. doi: 10.1115/1.4027382
08/27/2015	13.00	R. Samulyak, H. Wei. Mass-conservative network model for brittle fracture, JOURNAL OF COUPLED Systems and Multiscale Dynamics, (08 2014): 0. doi: 10.1166/jcsmd. 2014.1046
08/27/2015	14.00	Ezio Bartocci, Elizabeth M. Cherry, Flavio H. Fenton, James Glimm, Scott A. Smolka, Radu Grosu, Abhishek Murthy, Md. Ariful Islam. Model-order reduction of ion channel dynamics using approximate bisimulation, Theoretical Computer Science, (04 2014): 0. doi: 10.1016/j.tcs.2014.03.018
08/27/2015	15.00	J. Melvin, V. Rana, B. Cheng, J. Glimm, D. H. Sharp, H. Lim, D. C. Wilson. Sensitivity of inertial confinement fusion hot spot properties to the deuterium-tritium fuel adiabat, Physics of Plasmas, (02 2015): 0. doi: 10.1063/1.4908278

TOTAL: 10

Number of Papers published in peer-reviewed journals:

(b) Papers published in non-peer-reviewed journals (N/A for none)

Received	Paper
08/27/2015 16.00	Jeremy Melvin, Hyunkyung Kim, Verinder Rana, Baolian Cheng, James Glimm, David H. Sharp, Doug C. Wilson. Effects of Prehet on the Thermodynamics of the ICF Hot Spot, Journal of Physical Science and Application, (01 2015): 0. doi:
TOTAL:	**1**

Number of Papers published in non peer-reviewed journals:

(c) Presentations

1. J. Glimm Oct 7, 2015, Computational Physics with Sharp Discontinuities and Steep Gradients, Physics Department Lecture Los Alamos National Laboratory.

2. J. Glimm April 7-12, 2016, ”Calibration of velocity moments from experimental, theoretical and LES data” Mini Symposium talk, SIAM UQ conference, Lausanne, Switzerland. Los Alamo NM.

3. J. Glimm July 27, 2016 ”An Introduction to Theories of Turbulence” Mathematics Department Lecture, Oxford University, Oxford, UK.

4. J. Glimm July 27, 2016, ”Statistical Models of Turbulent Flow” Mathematics Department Colloquium, Oxford University, Oxford, UK.

5. J. Glimm August 15, 2016, ”A new scaling law extending Kolmogorov 1962 ideas” Physics seminar Los Alamos National Laboratory, Los Alamos, NM.

6. J. Glimm September 10, 2016, ”A Computational Platform for Risk Models in High Frequency Trading” Conference on Risk in Financial Markets, Stony Brook University, Stony Brook.

7. H. Liu and X. Jiao, Weighted Least Squares Based Essentially Non-Oscillatory Schemes on Unstructured Meshes, 8th International Congress on Industrial and Applied Mathematics - ICIAM 2015, Beijing, China, July 2015.

8. T. Delaney, X. Jiao, and R. Conley, Overcoming Element-Quality Dependency with Adaptive Extended-Stencil Finite Element Method, 8th International Congress on Industrial and Applied Mathematics - ICIAM 2015, Beijing, China, July 2015.

9. R. Samulyak, Computational Methods for Complex Systems and Applications to Laser Ablation and Exploding Wire Experiments, Army Research Laboratory Seminar, Aberdeen, MD, November 12, 2015

Number of Presentations: 9.00

Non Peer-Reviewed Conference Proceeding publications (other than abstracts):

Received	Paper

TOTAL:

Number of Non Peer-Reviewed Conference Proceeding publications (other than abstracts):

Peer-Reviewed Conference Proceeding publications (other than abstracts):

Received	Paper	
08/07/2014	9.00	POOJA RAO[L],, JEREMY A. MELVIN[L],, WENLIN HU[L], , RYAN KAUFMAN[L], , HYUNKYUNG LIM[L], , JAMES G. GLIMM[L]. Predictive Simulations for Problems with Solution Non-Uniqueness, 11th World Congress on Computational Mechanics (WCCM XI). 20-JUL-14, . : ,
10/10/2016	1.00	V. Dyedov, N. Ray, D. Einstein, X. Jiao, T.J. Tautges. AHF: Array-Based Half-Facet Data Structure for Mixed-Dimensional and Non-manifold Meshes, 22nd International Meshing Roundtable. 13-OCT-13, Orlando, Florida. : ,

TOTAL: **2**

Number of Peer-Reviewed Conference Proceeding publications (other than abstracts):

(d) Manuscripts

Received	Paper
08/07/2014 10.00	J. Glimm, B. Plohr, D. Sharp. LARGE EDDY SIMULATION, TURBULENT TRANSPORT ANDTHE RENORMALIZATION GROUP, Stony Brook University Preprint SUNYSB-AMS-12-02 (09 2013)
08/27/2015 19.00	W. Hu, H. Lim, J. Glimm, D. H. Sharp. Statistical Moments in MIxing Flows, SUNYSB-AMS-15-02 (03 2015)
10/10/2016 22.00	Xiangmin Jiao, Hongxu Liu. WLS-ENO: Weighted-least-squares based essentially Non-Oscillatory Schemes for Finite Volume Methods on Unstructured Meshes, Journal of Computational Physics (03 2015)
10/11/2016 12.00	H. Wei,, R. Samulyak. Mass-conservative network model for brittle fracture, Journal of Coupled Systems and Multiscale Dynamics (01 2014)
10/17/2016 17.00	James Glimm, Hyunkyung Lim, Ryan Kaufman, Wenlin Hu. Euler equation existence, nonuniqueness and mesh converged statistics, Stony Brook University Preprint SUNYSB-AMS-14-04 (10 2014)
10/17/2016 21.00	Ying Xu, Xiaoxue Gong, Vinay Mahadeo, Tulin Kaman, Johan Larsson, James Glimm. Mesh Convergence For Turbulent Combustion, SUNYSB-AMS-15-04 (06 2015)
10/17/2016 18.00	R Kaufman, H Lim, J Glimm. Conservative front tracking: the algorithm, the rationale and the API, SUNYSB-AMS-15-01 (03 2015)
10/19/2016 11.00	J. Melvin,, H. Lim,, V. Rana,, B. Cheng,, J. Glimm,, D.H. Sharp,, D.C. Wilson. Sensitivity of ICF Hot Spot Properties to the DT Fuel Adiabat, Physics of Plasmas (03 2014)
10/19/2016 20.00	Hyunkyung Lim, James Glimm, Flavio H. Fenton, Elizabeth M. Cherry, Shuai Xue. Sharp Boundary Electrocardiac Simulations, SIAM J. SCI. COMPUT. (05 2015)
TOTAL:	**9**

Number of Manuscripts:

Books

Received Book

TOTAL:

Received Book Chapter

TOTAL:

Patents Submitted

Patents Awarded

Awards

Graduate Students

NAME	PERCENT_SUPPORTED	Discipline
Wei Li	1.00	
Hongxu Liu	0.50	
Cao Lu	0.50	
Verinder Rana	0.21	
Hamid Said	0.00	
Rebecca Conley	0.00	
Tristan Delaney	0.00	
Vinay Mahadeo	0.00	
FTE Equivalent:	**2.21**	
Total Number:	**8**	

Names of Post Doctorates

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Names of Faculty Supported

NAME	PERCENT_SUPPORTED	National Academy Member
James Glimm	0.00	Yes
Xiangmin Jiao	0.09	
Roman Samulyak	0.12	
FTE Equivalent:	**0.21**	
Total Number:	**3**	

Names of Under Graduate students supported

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Student Metrics

This section only applies to graduating undergraduates supported by this agreement in this reporting period

The number of undergraduates funded by this agreement who graduated during this period:	0.00
The number of undergraduates funded by this agreement who graduated during this period with a degree in science, mathematics, engineering, or technology fields:.....	0.00
The number of undergraduates funded by your agreement who graduated during this period and will continue to pursue a graduate or Ph.D. degree in science, mathematics, engineering, or technology fields:.....	0.00
Number of graduating undergraduates who achieved a 3.5 GPA to 4.0 (4.0 max scale):.....	0.00
Number of graduating undergraduates funded by a DoD funded Center of Excellence grant for Education, Research and Engineering:.....	0.00
The number of undergraduates funded by your agreement who graduated during this period and intend to work for the Department of Defense	0.00
The number of undergraduates funded by your agreement who graduated during this period and will receive scholarships or fellowships for further studies in science, mathematics, engineering or technology fields:	0.00

Names of Personnel receiving masters degrees

NAME
Total Number:

Names of personnel receiving PHDs

NAME
Cao Lu
Wei Li
Total Number: **2**

Names of other research staff

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Sub Contractors (DD882)

Inventions (DD882)

Scientific Progress

Technology Transfer

We have made considerable progress with tech transfer, both with Army applications and (using leveraged funding), with other government laboratories.

Army Laboratory tech transfer

We have given two research seminars at the ARL, Aberdeen, MD:

- R. Samulyak, New Conservative Models and Numerical Algorithms for Brittle Fracture, Army Research Laboratory Seminar, Aberdeen, MD, June 10, 2014.

- R. Samulyak, Computational Methods for Complex Systems and Applications to Laser Ablation and Exploding Wire Experiments, Army Research Laboratory Seminar, Aberdeen, MD, November 12, 2015

Our latest brittle fracture software package based on finite elements is compatible with engineering codes used at ARL. The brittle fracture software has been transferred to ARL.

In the last visit, we have also discussed ARL experiments on laser ablation for the characterization of explosive materials. Preliminary multiscale simulations using FronTier have been recently performed in a simplified regime (without reacting chemistry) for the purpose of verification and validation. The simulations achieved agreements in the characterization of major processes with publishes works.

Following last year's visited ARL, to discuss brittle fracture, we started modeling effort to treat the boundary of the fracture zone with equations similar to those of reactive chemistry, building on ideas of Michael Greenfield of ARL. Recently, we have proposed that the outer boundary of the fracture zone has the properties of a detonation wave within the shear wave mode of the nonlinear elastic equations. To preserve the brittle regime, we assume the nonlinearity is stiff, not soft, as can result from work hardening.

0.1 Technology Transfer to Other Government Laboratories

The brittle fracture is of interest to groups at BNL, from the standpoint of characterization of materials at extreme conditions and safety of nuclear reactors, whose fuel shell casings become brittle over time due to radiation damage.

Our work on turbulent mixing has been applied to modeling of inertial confinement fusion experiments, based on leveraged funding from Los Alamos National Laboratory. For situations with limited concentration diffusion of a miscible mixture, our codes appear to be unique in an ability to predict the microstructure of the mixing flow. The mixture microstructure is an input into the nuclear reaction calculations for such devices.

Our work with ORNL on microstructures of fluid mixtures continues. Specifically, we are adding a new force term (disjuncture pressure, familiar in the sense of lubrication theory) to moderate the spacing between closely spaced interfaces in an immiscible, highly stirred environment.

With LANL, we have analyzed the inertial confinement fusion experiments at LLNL. These are a challenge to the simulation and modeling community as currently the best simulations do not fully duplicate the experiments. We found a strong sensitivity for untracked Eulerian simulations at all to the most extremely fine grids, a fact which could be a major problem for LANL in that that laboratory prefers to use Eulerian codes. We found various hydro instability sensitivities for the simulations arising from combined instability mechanisms, but the single effect instabilities did not seem to be a problem for this experiment.

In terms of numerical methods, we have explored more applications of high-order geometry algorithms. In particular, we have incorporated it as part of the framework for parallel uniform mesh refinement with mesh-based database MOAB at Argonne National Laboratory. Under framework, we are also building efficient multigrid solvers to utilize these uniformally refined meshes.

Front tracking technologies have been used in new software libraries for complex mag- netohydrodynamic flows. The software has been extensively used in DOE applications in the area of high energy and accelerator physics, in particular in the design of high power accelerator targets and high energy density plasma devices. This work supports research programs at BNL, LANL, Fermilab, and ORNL.

Final Report: Mathematical Models and Advanced Numerical Methods for Complex Flows and Structures

Principal Investigator

James Glimm
Distinguished Professor

Co-Principal Investigators

Xiangmin Jiao — Associate Professor
Roman Samulyak — Professor

Department of Applied Mathematics and Statistics
SUNY at Stony Brook
Stony Brook, NY 11794-3600

Name	Email	Phone (631-632-): Voice	Fax
James Glimm	glimm@ams.sunysb.edu	8355	8490
Xiangmin Jiao	jiao@ams.sunysb.edu	4408	8490
Roman Samulyak	rosamu@ams.sunysb.edu	8353	8490

Submitted to

Director
U. S. Army Research Office
ATTN: AMXRO-ICA
4300 South Miami Boulevard
P.O. Box 12211
Research Triangle Park, NC 27709-2211

Contents

Project Description

1 Introduction

Computational study of brittle fracture networks are of great interest to engineers who work with material which breaks before it can undergo plastic‘ deformation. This can occur when the material is in a “pre-stressed” configuration, where small deformations and strains can lead the material to surpass its internal critical stress. Of great importance to engineers is the prediction of crack nucleation and overall dependence of the resulting crack pattern on material properties.

Computational methods are of growing importance in modeling brittle fracture in materials. Due to the complex nature of fracture mechanics, several phenomenological methods have been proposed. Such methods include generalized and extended finite element methods (X-FEM) [12, 14], cohesive element (CE) models [7], and spring models [4].

The extended FEM (X-FEM) [14] and the generalized FEM (GFEM) [15], which are closely related and both belong to the partition of unity methods (PUM) [12], enrich the traditional FEM function space with families of discontinuous shape functions, which can model the displacements of either the crack tip or opposite sides of the crack plane. The main advantage lies in the fact that the interface can be studied within each individual element without having to constantly remesh near the crack tip. The types of enrichment functions can a variety of engineering problems the crack tip including branching [10], material interfaces, and soft discontinuities.

Cohesive zone models (CZM) were first introduced by Barenblatt [7] and have been incorporated into commercial FEM codes. The CZM is intriguing because it explicitly avoids the creation of stress singularities by modeling inter-element traction-displacement relationships. Elements become separated when their tractions exceed a critical threshold, and the location of the “cohesive elements” (CE) can generate complex fracture networks. Cohesive elements have been used in traditional finite element codes by Xu and Needleman [23] and Ortiz and Camacho [9]. Additionally, cohesive zone models have been incorporated into other finite element frameworks such as X/G-FEM [14], meshless methods [6], and isogeometric analysis [20]. While CZMs may provide complex fracture structures, it was noted in [11] that CZMs are dependent on aspects of the mesh.

Spring models were introduced in [8, 13] and have the advantage of very simple implementation of solid mechanics and fracture mechanics. For ex-

ample, Meakin [13] modeled fracture using a two-dimensional network of springs with a critical tension parameter. Over-strained springs were removed to simulate the propagation of fracture. Beale added random defects and perturbations to a spring model to investigate their effects on the propagation of the crack surface [8]. However, both of these models implemented fracture mechanisms by removing springs; thereby losing mass conservation and "obliterating" material under compression.

Recently, Wei et al. [4] investigated the use of mass conservative spring models, and were able to reproduce complex fracture networks in two and three dimensions. The method used nonlinear optimization of the global energy functional and split vertices adjacent to springs which were strained past a pre-defined threshold. The advantage of this method was that it conserved mass and produced rich crack networks throughout the material. The work demonstrated complex fracture patterns which qualitatively change due to variations in material properties.

The spring network model of [4] is difficult to integrate into pre-existing finite element software. The intent of this paper is to apply the fracture mechanisms of [4] to existing finite element codes. In this work, we describe the fracture mechanism incorporated into a finite element solid mechanics code as well as collision detection algorithms to prevent inter-element penetration.

The remainder of the report is divided as follows. In Section 2, the principles of continuum mechanics and their implementation within the finite element method are briefly discussed. In Section 3, the fracture mechanism is described, and particular emphasis is placed on the detection of intersecting elements and their resolution through the introduction of Lagrange multipliers. In Section 5, we present results for some test cases using our fracture mechanism.

Our main computational approach to brittle fracture is quasi-static in nature. The dynamic simulation of fracture with the resolution of relevant temporal and spatial scales will be performed in the future. As a first step towards this goal, we have developed a dynamic theory of brittle fracture. The summary of this work is given in Section 6.

The final component of this research program, theoretical work on turbulence, is presented in Section 7. A brief summary and conclusions are given in Section 9.

2 Finite Element-Based Brittle Fracture Model and Algorithms

2.1 Principle of Virtual Work

Finite element analysis is an important tool in the study of solid mechanics and fracture mechanics in particular. Linear elastic fracture mechanics (LEFM) has been well-studied using methods such as X-FEM and CZM. However, the range of applications of LEFM is limited by the assumptions of linear elasticity which is valid for small displacements and small strains. In many applications involving brittle fracture, large displacements and rotations may occur while the strain exhibited within a material may still be in the elastic regime. Therefore it is important to take into account these geometric nonlinearities in the continuum formulation. In particular, we use a quasi-static updated Lagrangian description of the physical system following in [2].

For any solid body of some material, denoted at time τ as ${}^{\tau}V$, the material is in energetic equilibrium when the principle of virtual work is satisfied, namely that the body is in a minimum energy configuration. The principle of virtual work for a body at some future time $t+\Delta t$ can be written as [2]

$$\int_{{}^{t+\Delta t}V} {}^{t+\Delta t}\tau_{ij}\delta_{t+\Delta t}e_{ij}\,{}^{t+\Delta t}dv = {}^{t+\Delta t}\mathscr{R}, \tag{1}$$

where

$${}^{t+\Delta t}\mathscr{R} = \int_{{}^{t+\Delta t}A} {}^{t+\Delta t}_{t+\Delta t}t_k\delta u_k\,{}^{t+\Delta t}da + \int_{{}^{t+\Delta t}V} {}^{t+\Delta t}\rho^{t+\Delta t}_{t+\Delta t}f_k\delta u_k\,{}^{t+\Delta t}dv \tag{2}$$

is the external virtual work, in which δu_k is a virtual variation in the current displacement components ${}^{t+\Delta t}u_k$, and

$$\delta_{t+\Delta t}e_{ij} = \delta\frac{1}{2}({}_{t+\Delta t}u_{i,j} + {}_{t+\Delta t}u_{j,i}) \tag{3}$$

is the corresponding virtual variation in strains.

This equation cannot be solved directly, since the configuration at time $t+\Delta t$ is unknown. However, a solution can be obtained through referring all variables to a known previously calculated equilibrium configuration. In practice, the approach basically lies between the total Lagrangian formulation and the updated Lagrangian formulation. The method we adopted is the updated Lagrangian formulation.

2.2 Updated Lagrangian Approach

In the updated Lagrangian formulation, all variables in the equation of principle of virtual work are referred to the configuration at time t, i.e. the updated configuration of the body.

The volume integral of the Cauchy stress times variations in infinitesimal strains can be transformed to give:

$$\int_{t+\Delta t V} {}^{t+\Delta t}\tau_{ij}\delta_{t+\Delta t}e_{ij}\,{}^{t+\Delta t}dv = \int_{tV} {}^{t+\Delta t}_{t}S_{ij}\delta^{t+\Delta t}_{t}\varepsilon_{ij}\,{}^{t}dv, \tag{4}$$

so that the formulation in this case is transformed to:

$$\int_{tV} {}^{t+\Delta t}_{t}S_{ij}\delta^{t+\Delta t}_{t}\varepsilon_{ij}\,{}^{t}dv = {}^{t+\Delta t}\mathscr{R}, \tag{5}$$

where

$${}^{t+\Delta t}_{t}S_{ij} = \frac{{}^{t}\rho}{{}^{t+\Delta t}\rho}\,{}^{t}_{t+\Delta t}x_{i,s}\,{}^{t+\Delta t}\tau_{sr}\,{}^{t}_{t+\Delta t}x_{j,r} \tag{6}$$

is the Cartesian components of the second Piola-Kirchhoff stress tensor, and

$$\delta\,{}^{t+\Delta t}_{t}\varepsilon_{ij} = \delta\frac{1}{2}({}^{t+\Delta t}_{t}u_{i,j} + {}^{t+\Delta t}_{t}u_{j,i} + {}^{t+\Delta t}_{t}u_{k,i}\,{}^{t+\Delta t}_{t}u_{k,j}) \tag{7}$$

is the Cartesian components of the Green-Lagrange strain tensor from configuration at time t to the configuration at time $t+\Delta t$ and referred to the configuration at time t. The incremental stress decomposition used in this case is

$${}^{t+\Delta t}_{t}S_{ij} = {}^{t}\tau_{ij} + {}_{t}S_{ij}, \tag{8}$$

where ${}^{t}\tau_{ij}$ is the Cartesian components of the Cauchy stress tensor, and ${}_{t}S_{ij}$ is the Cartesian components of the second Piola-Kirchhoff stress increment tensor referred to the configuration at time t.

The incremental strain decomposition is

$${}^{t+\Delta t}_{t}\varepsilon_{ij} = {}_{t}\varepsilon_{ij} \tag{9}$$

$${}_{t}\varepsilon_{ij} = {}_{t}e_{ij} + {}_{t}\eta_{ij} \tag{10}$$

where

$${}_{t}e_{ij} = \frac{1}{2}({}_{t}u_{i,j} + {}_{t}u_{j,i}) \tag{11}$$

$$ {}_t\eta_{ij} = \frac{1}{2}\, {}_t u_{k,i}\, {}_t u_{k,j} \tag{12} $$

The constitutive relation between stress and stain increments used is

$$ {}_t S_{ij} = {}_t C_{ijrs}\, {}_t\varepsilon_{rs}, \tag{13} $$

and the virtual work equation can be rewritten as

$$ \int_{{}_tV} {}_t C_{ijrs}\, {}_t\varepsilon_{rs}\delta\, {}_t\varepsilon_{ij}\, {}^t dv + \int_{{}_tV} {}^t\tau_{ij}\delta_t\eta_{ij}\, {}^t dv = {}^{t+\Delta t}\mathscr{R} - \int_{{}_tV} {}^t\tau_{ij}\delta_t e_{ij}\, {}^t dv, \tag{14} $$

which is a nonlinear equation in the incremental displacements u_i.

2.3 Constitutive Relation

Calculation of the constitutive tensors is an important part for solving the non-linear problems. The constitutive relation defines the stress-strain matrices in the finite element evaluations.

2.3.1 Linear elasticity

In the updated Lagrangian formulation, the constitutive relation is

$$ {}^t\tau_{ij} = {}^t_t C_{ijrs}\, {}^t\epsilon_{rs} \tag{15} $$

in which ${}^t\tau_{ij}$ is the Cauchy stress tensor, ${}^t\epsilon_{rs}$ is the Euler-Almansi strain tensor, and ${}^t_t C_{ijrs}$ is the material property tensor at time t.

This constitutive relation is used in the evaluation of the element stress matrices and stress vectors, i.e. total second Piola-Kirchhoff and Cauchy stresses are calculated directly from total Green-Lagrange and Euler-Almansi strains respectively.

For linear elasticity, the relation considered for the updated Lagrangian formulation is

$$ {}_t S_{ij} = {}_t C_{ijrs}\, {}_t\epsilon_{rs} \tag{16} $$

in which

$$ {}_t C_{ijrs} = {}^t_t C_{ijrs}. \tag{17} $$

2.4 Linearization

Approximate solutions to (14) can be obtained by assuming $\delta\, {}_t\epsilon_{ij} = \delta\, {}_te_{ij}$and ${}_tS_{ij} = {}_tC_{ijrs}\, {}_te_{rs}$, so that the equation becomes

$$\int_{{}_tV} {}_tC_{ijrs}\, {}_te_{rs}\delta\, {}_te_{ij}\, {}^tdv + \int_{{}_tV} {}^t\tau_{ij}\delta\eta_{ij}\, {}^tdv = {}^{t+\Delta t}\mathscr{R} - \int_{{}_tV} {}^t\tau_{ij}\delta e_{ij}\, {}^tdv. \quad (18)$$

This linearization is necessary because (14) is nonlinear with respect to nodal displacements.

3 Crack Formation and Propagation Algorithm

The cracks form and propagate when some part of the brittle material exceeds its critical strain. In the network based model, the overs-trained bonds are split [4]. In the FEM based approach, we split over strained nodes and generate new edges to introduce cracks. In order to determine the over strained nodes, we calculate the nodal strain tensor by averaging the value of strains from the neighboring elements. The principal strain is the largest eigenvalue of the strain tensor, and its eigenvector is the maximum strain direction. We collect a set of over strained nodes by comparing their principal strains with the fracture critical strain threshold. For each over strained node, we split edges that are most orthogonal to the maximum strain direction. Depending on the location of the node, we propose some rules:

1. If a node u_1 is an interior node, we split two edges with minimum angles to the orthogonal direction of the maximum strain. u_1 is the splitting end of the two edges.

   (a) If the two edges belong to the same element, we will discard one with a larger angle and pick a different edge.

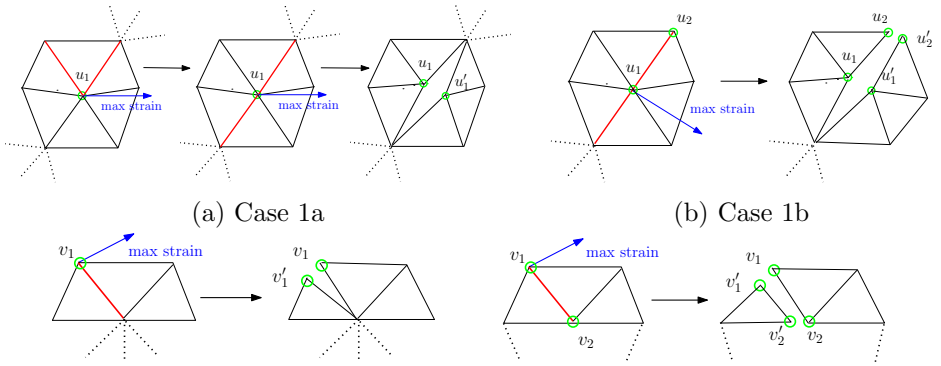
   (b) If the other end of a chosen edge, vertex u_2, is on the boundary or crack surface, u_2 will also be a splitting end. This is to prevent the case where two chunks are connected by a single vertex.

2. If a node v_1 is a boundary node, we select one edge with the minimum angle to the orthogonal direction of the maximum strain.

   (a) If the chosen edge is an interior edge, v_1 will be the splitting end. Otherwise, no edge will be split.

(b) Similar to case 1 if the other end of the chosen edge, vertex v_2, is on the boundary or crack surface, v_2 will also be a splitting end.

After each node is processed, we update the element connectivity list and other related data structures, as is shown in Figure 1.



(a) Case 1a (b) Case 1b

(c) Case 2a (d) Case 2b

Figure 1: Illustrations of the mesh splitting process.

3.1 Collision Detection

When cracks form and the mesh breaks into fragments, the mesh may fold and the fragments may overlap, creating unphysical states. Introduction of proper constraints resolves this issue. It provides a physics-based interaction model between fragments and the cracked material and leads to the propagation of cracks.

Before elimination, the overlapping and folding should be detected first. A standard way is to do pairwise triangles collision detection, in which triangles have at least one edge on crack surface. In order to reduce the number of pairwise comparison, the computational field is divided into coarse grids, and elements are assigned to each grid based on where the coordinates of their center fall into.

The detection process sweeps from upper left corner grid block towards lower right corner grid block, and only elements in right, lower or right lower diagonal adjacent grid would be detected pairwise for each grid block.

For each pair of triangles, first of all, every vertex of each element is examined if it is inside of the other one. This step checks 6 vertices using relative coordinates within each triangle, and would preclude succeeding examinations in most overlapping situations. However, there are cases that

triangles are just intersecting each other by edges, thus, we need to check edge-to-edge intersections pair-wisely, and this line segments intersection checking happens between 9 pairs.

In cases of stiff material, both the deformation and the overlapped parts are very small. The round-off error makes it difficult to assert whether these elements are overlapped. Presetting a "threshold" value helps to resolve this problem. By adding this controlling parameter to the algorithm, we adjust the precision of the overlap detection. This is crucial for the simulation because adding too few constraints would impair the correct physical system we would like to obtain, while adding too many constraints would result in an over-determined state of the system, eventually leading to a failure.

3.2 Contact Constraints

For frictionless contact problems, the contact condition, known as a Kuhn-Tucker condition, can be written as

$$g_N \leq 0,\ p_N \leq 0 \ \text{ on } \Gamma_c,\ g_N p_N = 0, \tag{19}$$

where g_N is the normal component of the gap function, and p_N is the normal component of contact stress.

We adopt the method of Lagrangian multipliers to resolve the contact problem. The contact contribution and its variation can be written as:

$$\Pi_c = \int_{\Gamma_c} \lambda_N g_N \, dA,\ \ C_c = \int_{\Gamma_c} (\lambda_N \delta g_N + \delta\lambda_N g_N) \, dA, \ , \tag{20}$$

where λ_N denotes the Lagrangian multiplier which can be identified as the contact pressure p_N, and δg_N is the variation of the normal contact gap function [21]. The Lagrangian multipliers add constraints contribution to the weak formulation of the solids in contact, and prevent further penetration between overlapping elements and push already overlapped triangles back to separated status.

Equation (20) can be generally discretized by introducing discretization for the gap function in each contact area Γ_c^i, that

$$g_N^i = \boldsymbol{C_i}^T \boldsymbol{u}, \qquad \boldsymbol{C} = [\boldsymbol{C}_1 | \boldsymbol{C}_2 | ... | \boldsymbol{C}_{n_s}], \qquad \boldsymbol{\Lambda} = (\lambda_1, \lambda_2, ..., \lambda_{n_s}), \tag{21}$$

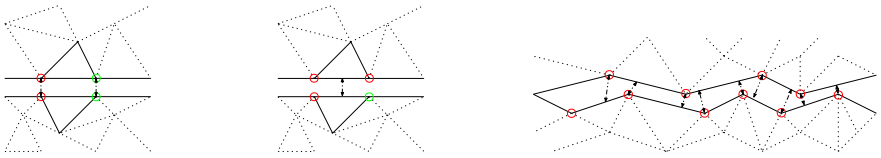
which $\boldsymbol{C}$ depends on the choice of discretization, thus

$$\int_{\Gamma_c} \lambda_N g_N \, dA \ \rightarrow\ \boldsymbol{\Lambda}^T \boldsymbol{C}^T \boldsymbol{u},\ \int_{\Gamma_c} \lambda_N \delta g_N \, dA \ \rightarrow\ \delta \boldsymbol{u}^T \boldsymbol{C} \boldsymbol{\Lambda},\ \int_{\Gamma_c} \delta\lambda_N g_N \, dA \ \rightarrow\ \delta \boldsymbol{\Lambda}^T \boldsymbol{C} \boldsymbol{u}, \tag{22}$$

The constraints can be arranged into the KKT matrix as

$$\begin{bmatrix} \left({}^{t+\Delta t}_{t}\boldsymbol{K}_L + {}^{t+\Delta t}_{t}\boldsymbol{K}_{NL}\right) & \boldsymbol{G}^T & \boldsymbol{C}^T \\ \boldsymbol{G} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{C} & \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{\lambda}_D \\ \boldsymbol{\lambda}_c \end{bmatrix} = \begin{bmatrix} \mathscr{R} - {}^{t}_{t}\boldsymbol{F} \\ \boldsymbol{b} \\ \boldsymbol{g_c} \end{bmatrix}, \quad (23)$$

There are several different approaches to discretize the gap function (21) along contact surfaces. For example, one could use node-to-node contact, node-to-segment contact, or segment-to-segment contact [22], as is shown in Figure 2.



(a) Node-to-node. (b) Segment-to-segment. (c) Node-to-segment.

Figure 2: Different contact constraint discretizations.

In the situation for brittle material, the crack surfaces are complicated, thus searching for all contact constraints on surfaces becomes a difficult task. In order to enhance time efficiency, we utilize features of *half-edge* to detect pairwise collision between elements, and collect contact constraint between each pair. Due to possible variety of collision scenarios, we adopt node-to-segment contact relation exclusively to adapt to different situations.

To obtain the smallest yet effective set of node-to-segment constraints, we propose the following rules for choosing master segment and slave node, as is shown in Figure 3:

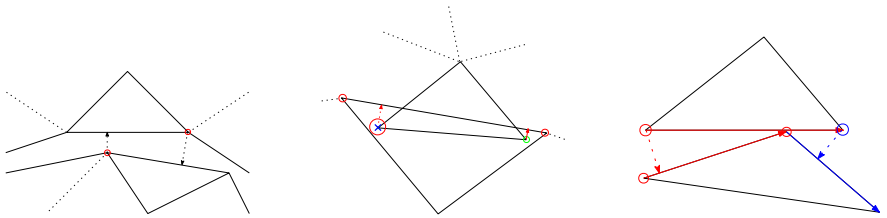
1. Each pair of overlapped elements could have no more than 2 node-to-segment contact constraints; each element could have no more than 1 contact constraints in which it acts as master segment or slave node.

2. The master segment must be a boundary edge. The slave node must have the smallest distance towards the opposite element among all feasible nodes, and that distance must not exceeds the preset detection precision threshold, where the distance is negative in case of penetration.

3. If there are more than one feasible contact constraint pair, we compare the angle between the master segment and two outgoing edges of slave

node, and choose the pair of node-segment with the smallest angle, and the angle must not exceed the preset threshold.

4. If all above rules does not satisfy, there will be no constraints subscribed for the given pair of overlapped elements.



(a) 2 constraints per pair. (b) Choose the smallest gap. (c) Choose the smallest angle.

Figure 3: Rules on choosing contact.

The Lagrangian Multipliers are obtained in a predict-and-correct manner, which use predicted position without contact constraints for both node x_b and segment (x_a, x_c) to calculate the surface norm $\boldsymbol{n}$ and the gap function g_N, which are calculated in the following steps:

1. Project slave node x_b on the master edge (x_a, x_c) as x_b' and calculate ξ

$$x_b' = \xi x_a + (1-\xi)x_c, \qquad (x_b' - x_b) \cdot (x_a - x_c) = 0. \tag{24}$$

2. Calculate the normal gap distance as a linear relation for displacement u_a, u_b, u_c

$$g_N = (\xi(x_a + u_a) + (1-\xi)(x_c + u_c) - (x_b + u_b)) \cdot \boldsymbol{n}d. \tag{25}$$

3. Assemble the coefficient of $\boldsymbol{u}$ into $\boldsymbol{C}$ and constant into $\boldsymbol{g_c}$ in (23).

As is shown in Figure 4, this node-to-segment contact will reduce to a node-to-node contact if ξ is 0 or 1 in some cases.

Even if we apply these rules to how contact constraints are added, there could still be numerous duplicate or redundant contact constraints. For example, a duplicate constraint occurs when adding degenerated node-to-segment contact, which is essentially a node-to-node contact, that the slave node is exactly the same vertex sharing by adjacent elements. Redundant constraints occur when the mesh breaks up totally at one vertex, when 3 or

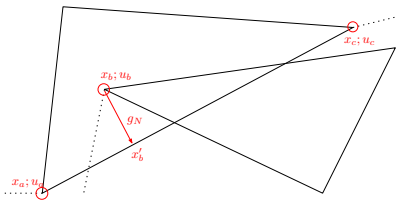


Figure 4: Contact constraint calculation.

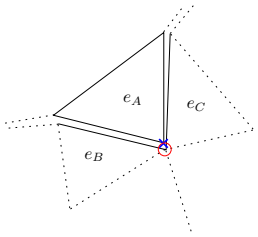


Figure 5: Redundant constraints.

more elements are in node-to-node contact with one another, and vertices constraints between v_a and v_c may be derived from constraints between v_a and v_b, and constraints between v_b and v_c, as shown in Figure 5

Redundant constraints would result in a singular or badly conditioned KKT system. To resolve the problem, we firstly collect all contact constraints and apply QR factorization to extract the largest linear independent rows from the contact constraints matrix. After that, we add the deducted contact constraint matrix $\boldsymbol{C}$ to the KKT system.

4 Software Development and Verification

4.1 Finite Element Implementation

We describe how the continuum mechanics are formulated using the method of finite elements. We follow an updated Lagrangian approach similar to the one proposed by Bathe, et. al., in [2]. In particular, we describe the mesh data structures, construction of the linear system of equations, linear solvers for the global system of equations.

4.1.1 Finite Element Computations

The focus of the finite element method is to discretize (18) over a finite dimensional family of appropriate displacements. We take our notation from [2] and use it directly in the MATLAB environment.

The position and displacement of a point in an element T can be parameterized in terms of a family of finite element shape functions $\{\phi_k\}_{k=1}^{3}$

as

$$ {}^{0}x_i = \sum_{k=1}^{3} \phi_k(\xi,\eta)\ {}^{0}x_i^k, \qquad i = 1,2; \tag{26} $$

$$ u_i = \sum_{k=1}^{3} \phi_k(\xi,\eta)\ {}^{0}u_i^k, \qquad i = 1,2. \tag{27} $$

Equation (18) can be put into matrix form

$$ {}^{t}_{t}\boldsymbol{K}\boldsymbol{u} = \boldsymbol{f}, \tag{28} $$

where ${}^{t}_{t}\boldsymbol{K}$ is the stiffness matrix associated with the body ${}^{t}V$, $\boldsymbol{u} = \left[u_1^1, u_2^1, \ldots, u_1^N, u_2^N\right]^T$ is a vector of nodal displacements, and $\boldsymbol{f}$ is the out-of-balance load vector taken from the right side of (18).

4.1.2 Stiffness Matrix Computation

The linear component of the Green-Lagrange strain in (11) can be computed using the relation

$$ {}_{t}\boldsymbol{e} = {}^{t}_{t}\boldsymbol{B}_L\boldsymbol{u}, \tag{29} $$

where

$$ {}_{t}\boldsymbol{e} = \left[e_{11}, e_{22}, 2e_{12}\right]^T, \tag{30} $$

$$ \boldsymbol{u} = \left[u_1^1, u_2^1, u_1^2, u_2^2, u_1^3, u_2^3\right]^T, \tag{31} $$

$$ {}^{t}_{t}\boldsymbol{B}_L = \begin{bmatrix} {}_{t}\phi_{1,1} & 0 & {}_{t}\phi_{2,1} & 0 & {}_{t}\phi_{3,1} & 0 \\ 0 & {}_{t}\phi_{1,2} & 0 & {}_{t}\phi_{2,2} & 0 & {}_{t}\phi_{3,2} \\ {}_{t}\phi_{1,2} & {}_{t}\phi_{1,1} & {}_{t}\phi_{2,2} & {}_{t}\phi_{2,1} & {}_{t}\phi_{3,2} & {}_{t}\phi_{3,1} \end{bmatrix}. \tag{32} $$

The element stiffness matrix is then computed via the matrix equation

$$ {}^{t}_{t}\boldsymbol{K}_L = {}^{t}_{t}\boldsymbol{B}_L^T\ {}^{t}_{t}\boldsymbol{C}\ {}^{t}_{t}\boldsymbol{B}_L. \tag{33} $$

Using the fact that

$$ \delta_t\eta_{ij} = \frac{1}{2}\left({}_{t}u_{k,i}\delta_t u_{k,j} + {}_{t}u_{k,j}\delta_t u_{k,i}\right), \tag{34} $$

the nonlinear contribution to the stiffness matrix at time t can be written as

$$ {}^{t}_{t}\boldsymbol{K}_{NL} = {}^{t}_{t}\boldsymbol{B}_{NL}^T\ {}^{t}\boldsymbol{\tau}\ {}^{t}_{t}\boldsymbol{B}_{NL}, \tag{35} $$

where

$$ {}^{t}\boldsymbol{\tau} = \begin{bmatrix} {}^{t}\tau_{11} & {}^{t}\tau_{12} & 0 & 0 \\ {}^{t}\tau_{12} & {}^{t}\tau_{22} & 0 & 0 \\ 0 & 0 & {}^{t}\tau_{11} & {}^{t}\tau_{12} \\ 0 & 0 & {}^{t}\tau_{12} & {}^{t}\tau_{22} \end{bmatrix}, \tag{36} $$

$$ {}^{t}_{t}\boldsymbol{B}_{NL} = \begin{bmatrix} {}_{t}\phi_{1,1} & 0 & {}_{t}\phi_{2,1} & 0 & {}_{t}\phi_{3,1} & 0 \\ {}_{t}\phi_{1,2} & 0 & {}_{t}\phi_{2,2} & 0 & {}_{t}\phi_{3,2} & 0 \\ 0 & {}_{t}\phi_{1,1} & 0 & {}_{t}\phi_{2,1} & 0 & {}_{t}\phi_{3,1} \\ 0 & {}_{t}\phi_{1,2} & 0 & {}_{t}\phi_{2,2} & 0 & {}_{t}\phi_{3,2} \end{bmatrix}. \tag{37} $$

The matrix $\boldsymbol{\tau}$ is constructed from the Cauchy stresses of the body ${}^{t}V$. The Cauchy stresses of each element are computed from (15) using the Euler-Almansi strain tensor referring to the initial configuration ${}^{0}V$. The process for determining these stresses is detailed in section 4.2.2.

The total element stiffness matrix is given by

$$ {}^{t}_{t}\boldsymbol{K} = {}^{t}_{t}\boldsymbol{K}_{L} + {}^{t}_{t}\boldsymbol{K}_{NL}. \tag{38} $$

4.1.3 Load Vector Computation

The load vector $\boldsymbol{f}$ can be computed from the external virtual work and an internal virtual work due to being in the current configuration ${}^{t}V$, which is constant with respect to the nodal displacements. Oftentimes, it is sufficient to consider the external virtual work as being deformation independent, in which case (2) becomes

$$ {}^{t+\Delta t}\mathscr{R} = \int_{{}^{0}V} {}^{0}\rho \, {}^{t+\Delta t}_{0}f_k \delta u_k \, {}^{0}dv + \int_{{}^{0}S} {}^{t+\Delta t}_{0}t_k \delta u_k \, {}^{0}ds. \tag{39} $$

The final term included in the load vector is due to virtual work associated with being in the configuration. It is constant, depending only on the current configuration of the body, and plays an important roll in the iterative scheme discussed in section 4.1.5. The contribution of each element to the out-of-balance internal virtual work is given by

$$ {}^{t}_{t}\boldsymbol{F} = {}^{t}_{t}\boldsymbol{B}_{L}^{T} \, {}^{t}\hat{\boldsymbol{\tau}}, \tag{40} $$

$$ {}^{t}\hat{\boldsymbol{\tau}} = \left[{}^{t}\tau_{11}, \, {}^{t}\tau_{22}, \, {}^{t}\tau_{12} \right]. \tag{41} $$

The global system of equations can then be written as

$$ \left({}^{t+\Delta t}_{t}\boldsymbol{K}_{L} + {}^{t+\Delta t}_{t}\boldsymbol{K}_{NL} \right) \boldsymbol{u} = \mathscr{R} - {}^{t}_{t}\boldsymbol{F} \tag{42} $$

4.1.4 Dirichlet and Gliding Boundary Conditions

In this section, we discuss how boundary conditions are enforced. The method we use is based on the use of Lagrange multipliers, as discussed in [1] to incorporate "gliding" boundary conditions, where displacements of certain nodes are constrained to move along a line or a plane (in 3D). Gliding boundary conditions are enforced using a set of constraints of the form

$$\boldsymbol{G}\boldsymbol{u} = \boldsymbol{b}. \tag{43}$$

The constraints can be added to (42) by forming the KKT matrix

$$\begin{bmatrix} \left({}^{t+\Delta t}_{t}\boldsymbol{K}_L + {}^{t+\Delta t}_{t}\boldsymbol{K}_{NL}\right) & \boldsymbol{G}^T \\ \boldsymbol{G} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathscr{R} - {}^{t}_{t}\boldsymbol{F} \\ \boldsymbol{b} \end{bmatrix}. \tag{44}$$

Dirichlet boundary conditions which are not "gliding" conditions (that is the position of the node is predetermined) are not included (44). Rather, the contributions from these degrees of freedoms in the element stiffness matrices (38) are placed in the load vector of (42) or (44), and the rows associated with those virtual displacements are not considered.

4.1.5 Iterative Solution of Nonlinear System

Equation (14) is a nonlinear with respect to nodal displacements, and therefore we need an iterative scheme to determine a solution at each step. In [2], a modified Newton-Ralphson method was proposed along with different techniques for acceleration. We try to follow in the same spirit, although the structure of our matrix is inherently different and requires different solvers for the iteration to converge. We solve (14) iteratively by solving for several incremental displacements of the form

$${}^{t+\Delta t}u_i^{(k+1)} = {}^{t+\Delta t}u_i^{(k)} + \Delta u_i^{(k+1)}, \tag{45}$$

$${}^{t+\Delta t}u_i^{(0)} = {}^{t}u_i. \tag{46}$$

The iterative scheme can be incorporated into (44) as

$$\begin{bmatrix} \left({}^{t+\Delta t}_{t}\boldsymbol{K}_L + {}^{t+\Delta t}_{t}\boldsymbol{K}_{NL}\right) & \boldsymbol{G}^T \\ \boldsymbol{G} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \Delta\boldsymbol{u}^{(k)} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathscr{R} - {}^{t+\Delta t}_{t+\Delta t}\boldsymbol{F}^{(k-1)} \\ \boldsymbol{b}^{(k-1)} \end{bmatrix}, \tag{47}$$

where ${}^{t+\Delta t}_{t+\Delta t}\boldsymbol{F}^{(k-1)}$ is computed from (40) over an incrementally deformed body ${}^{t+\Delta t}V^{(k-1)}$. The vector $\boldsymbol{b}^{(k-1)}$ is an updated form of the gliding boundary conditions, (43), used to ensure that each incremental displacements

enforce the correct boundary conditions at each increment. It is computed via

$$\boldsymbol{b}^{(k-1)} = \boldsymbol{b} - \boldsymbol{G}^{\,t+\Delta t}\boldsymbol{u}^{(k-1)}. \tag{48}$$

At each increment, (47) is solving using an incomplete LU-preconditioned GMRES solver, which is available in MATLAB and PETSc. The iteration stops once

$$\frac{\left\|\Delta \boldsymbol{u}^{(k)}\right\|_2}{\left\|\Delta \boldsymbol{u}^{(0)}\right\|_2} < \text{tol}. \tag{49}$$

4.2 Material Properties

We describe how material properties and tensors are computed in our implementation.

4.2.1 Constitutive Relation

The rank 4 material stiffness tensor of 15 is generally defined from material parameters (Young's modulus, bulk modulus, shear modulus, etc.). In the direct computation, these material parameters are converted into the Lam parameters (assuming plane stress condition $\tau_{33} = \tau_{31} = \tau_{32} = 0$)

$$\lambda = \frac{E}{1-\nu^2} \tag{50}$$

$$\mu = \frac{E}{2\,(1+\nu)} \tag{51}$$

where E is the Young's modulus of the material and ν is the Poisson ratio. The stiffness tensor in direct index notation is given by

$$C_{ijkl} = \lambda\delta_{ij}\delta_{kl} + 2\mu\,(\delta_{ik}\delta_{jl} + \delta_{il}\delta_{jk}) \tag{52}$$

the stress-strain relation is given by the generalized Hooke's Law

$$\tau_{ij} = C_{ijkl}\epsilon_{kl}. \tag{53}$$

For computation of stiffness matrices and load vectors from computed stresses and strains, we follow the Voigt notation for symmetric rank 2 tensors [3], such as stress and strain. In particular, a general strain tensor,

ϵ_{ij}, and a general stress tensor, τ_{ij}, are represented as 3-vectors

$$\epsilon_{ij} = \begin{bmatrix} \epsilon_{11} & \epsilon_{12} \\ \epsilon_{21} & \epsilon_{22} \end{bmatrix} \Longrightarrow \boldsymbol{\epsilon} = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \end{bmatrix} = \begin{bmatrix} \epsilon_{11} \\ \epsilon_{22} \\ 2\epsilon_{12} \end{bmatrix}, \tag{54}$$

$$\tau_{ij} = \begin{bmatrix} \tau_{11} & \tau_{12} \\ \tau_{21} & \tau_{22} \end{bmatrix} \Longrightarrow \boldsymbol{\tau} = \begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix} = \begin{bmatrix} \tau_{11} \\ \tau_{22} \\ \tau_{12} \end{bmatrix}. \tag{55}$$

The stress-strain relation of (15) can then be represented in a symmetric 3-by-3 matrix form

$$\boldsymbol{C} = \begin{bmatrix} \lambda + 2\mu & \lambda & 0 \\ \lambda & \lambda + 2\mu & 0 \\ 0 & 0 & \mu \end{bmatrix}, \tag{56}$$

and the constitutive relation (15) can be written as

$$\boldsymbol{\tau} = \boldsymbol{C}\boldsymbol{\epsilon}. \tag{57}$$

4.2.2 Strain Tensors

The computation of valid strain tensors is necessary in the updated Lagrangian formulation because of the presence of finite deformations and large displacements (including rigid body rotations). The two strain tensors required for computation are the Green-Lagrangian strain tensor at time $t + \Delta t$ referring to the configuration at time t,

$$ {}^{t+\Delta t}_{t}\varepsilon_{ij} = \frac{1}{2}\left(\frac{\partial^{t+\Delta t}x_k}{\partial^t x_j}\frac{\partial^{t+\Delta t}x_k}{\partial^t x_i} - \delta_{ij}\right), \tag{58}$$

$$= \frac{1}{2}\left({}_t u_{i,j} + {}_t\, u_{j,i} + {}_t\, u_{k,i}\, {}_t u_{k,j}\right) \tag{59}$$

which is energy-conjugate with the second Piola-Kerchoff stress tensor and used in the computation of the stiffness matrix in (18). Equation (59) is included because it can be simply computed in terms of the finite element basis functions.

The other strain tensor, the Euler-Almansi strain tensor at time t, is given by

$$ {}^{t}\varepsilon_{ij} = \frac{1}{2}\left(\delta_{ij} - \frac{\partial^0 x_k}{\partial^t x_i}\frac{\partial^0 x_k}{\partial^t x_j}\right), \tag{60}$$

which is energy-conjugate with the Cauchy stress of the material at time t. The Euler-Almansi strain is related to the Cauchy stress by (15). The computation of (60) requires extra care because it has a nonlinear relationship with the nodal positions/displacements cannot be computed as straightforward as a matrix product as the Green-Lagrange strain tensor. We compute the Euler-Almansi strain defined on a physical element via a composition mappings between three elements

1. A master element, $\hat{T}$, with coordinates $\boldsymbol{u} = (u_1, u_2)$.
2. An undeformed element, ${}^0T$, from the original configuration, with coordinates ${}^0\boldsymbol{x} = \left({}^0x_1,\ {}^0x_2\right)$.
3. A deformed element, ${}^tT$, in the current configuration, with coordinates ${}^t\boldsymbol{x} = \left({}^tx_2,\ {}^tx_2\right)$.

Additionally we define two isoparametric mappings from the master element to the physical elements based on the linear finite element shape functions. Let $\boldsymbol{\phi} : \hat{T} \rightarrow {}^0T$ and $\boldsymbol{\psi} : \hat{T} \rightarrow {}^tT$ be diffeomorphisms from the master element to the undeformed element and the deformed element, respectively. The maps define a composite mapping from the undeformed element in its original configuration to the deformed element in the current configuration,

$$\boldsymbol{\omega} \equiv \boldsymbol{\psi} \circ \boldsymbol{\phi}^{-1} : {}^0T \rightarrow {}^tT. \tag{61}$$

The partial derivatives in (60) can then be computed from the Jacobian of $\boldsymbol{\omega}^{-1}$,

$$\frac{\partial\left({}^0x_1,\ {}^0x_2\right)}{\partial\left({}^tx_1,\ {}^tx_2\right)} \equiv \nabla_{{}^t\boldsymbol{x}}\left(\boldsymbol{\omega}^{-1}\right) = \left(\nabla_{\boldsymbol{u}}\boldsymbol{\phi}\right)\left(\nabla_{\boldsymbol{u}}\boldsymbol{\psi}\right)^{-1} = \frac{\partial\left({}^0x_1,\ {}^0x_2\right)}{\partial\left(u_1, u_2\right)}\frac{\partial\left(u_1, u_2\right)}{\partial\left({}^tx_1,\ {}^tx_2\right)}, \tag{62}$$

where $\nabla_{\boldsymbol{x}}$ denotes partial derivatives taken with respect to coordinate system $\boldsymbol{x}$. Once the Jacobian has been constructed, it can be used to directly compute ${}^t\varepsilon_{ij}$.

4.3 Mesh Data Structures

In general, to formulate a physical problem in the language of finite elements, one needs to specify

1. Geometry
2. Material Properties

3. Boundary Conditions

We will discuss in detail the data structures used and how the linear system of equations are constructed. Currently, the whole program is implemented in MATLAB, because of the ease power of its matrix computations in determining tensor properties (stresses/strains, stiffness matrices, etc.) and integration of linear solvers in one programming environment. However, the data structures described may also be implemented in any programming language.

4.3.1 Geometry

In this section, we describe primarily the data structures used to store the discretized mesh. Currently, the meshes are generated using the Triangle unstructured mesh generator [17], which is used to generate quality Delaunay triangulations with flexibility of determining minimum angle among the triangles and region-specific maximum element sizes. Triangle takes as input a planar straight-line graph (PSLG), which is given as a collection of vertices desired in the triangulation along with edges which are desired in the final mesh. The inputs, in most cases, represent different features (boundaries, cracks, etc.) which are desired in the reference configuration of the undeformed mesh.

From the PSLG, Triangle generates the most basic of unstructured mesh data structures

1. A double array of vertex positions in (x, y) coordinates.

2. An integer array of vertex indices associated with each element, ordered counter-clockwise.

These two arrays are passed back to MATLAB, and are used to assemble the linear system of equations to be solved in (47).

Additional mesh information about the edges is constructed and is used for crack propagation and assignment of external tractions on a surface. Specifically, we use a *half-edge* data structure to construct node-to-node connectivity. Briefly, a half-edge is a directed connection between two nodes on the same element. The half-edges inside an element are numbered consecutively starting from the "local" id of the node located at the tail of the half-edge. An integer matrix of the same size as the element array is constructed with each entry holding the index of an "opposite" half-edge which is located in an adjacent element connected across that edge. If there is no element opposite of a half-edge (i.e. a boundary), then the opposite

half-edge index is assigned to be zero, indicating that the edge lies on the boundary of the mesh.

4.3.2 Material Properties

The main material properties stored are the Lam constants of the material and the associated stiffness tensor as a 3-by-3 matrix in Voigt notation.

4.3.3 Boundary Conditions

Both Dirichlet and Neumann boundary conditions are considered in our computations. Each are stored in separate arrays and are used in different parts of the linear system of equations.

Dirichlet boundary conditions specify desired nodal displacements. They are stored in MATLAB as a $NDBC$-by-3 matrix, where $NDBC$ is the number of Dirichlet boundary conditions specified. Each row of the matrix corresponds to a single constraint given in the form

$$au_1 + bu_2 = g,$$

and each node can have up to two constraints assigned to it. Additionally, an integer index array of size $N + 1$, where N is the number of nodes of the mesh, is also stored with each nonzero entry corresponding to the index of the first constraint assigned to that node. The $(N + 1)$th entry of the associated index array is automatically assigned to be $NDBC + 1$.

Neumann boundary conditions specify external tractions assigned on parts of the mesh boundary, and are used in the computation of (2). Neumann boundary conditions are assigned to be constant tractions along each boundary half-edge, which also puts them into assignment with one node along the boundary. External tractions are stored in a matrix of size $NNBC$-by-2, where $NNBC$ is the number of Neumann boundary conditions assigned. Like Dirichlet boundary conditions, an index array of size $N + 1$ is also stored with each nonzero entry pointing to the location of the associated traction assigned to that node's boundary half-edge. The $(N + 1)$st index is stored as $NNBC+1$ to indicate the number of Neumann boundary conditions.

4.4 Collision Detection

In this section we introduce the techniques for detecting collision between elements. Different from approximated disk-inside-element detecting method

in [4], which is compatible with the penalty and energy minimization method, we implemented an accurate and exact collision detection, which would eliminate any slight penetration, and could be applied compatibly through Lagrange Multipliers. This accurate collision detection works well with stiff materials when the deformation is small and elements basically moves tiny space between each simulation steps.

4.4.1 Grid-based domain decomposition

Due to large numbers of elements in simulation, we could not check collision and overlapping between each pair. However, under preconditions that the deformation of each simulation steps is small, we could have the conclusion that collision will only occur within a small range of each element's neighbourhood, and will only be possible between boundary elements, thus a great amount of work would be saved if we slice the computational domain into step-size and element-size related cubes and only detect potential elements within each cube's neighbours, see figure6 for demonstration, and figure7 for more complicated examples.

4.4.2 Pair-wise detection

The detection method is implemented in two steps. Firstly, we slice the domain into cubes, determine all boundary elements and store them in a three-dimensional matrix which the first and second indices indicates the cubes, and the third dimension stores the element number. Secondly, we perform a two-step check between any pair of elements:

1. Check if any vertices of an element falls into the other element. If any satisfies, break the checking procedure.

2. Check if any pair of edges from both elements intersects.

The first step for checking vertices are implemented using the Barycentric coordinate. Let $\boldsymbol{x_1}, \ldots, \boldsymbol{x_n}$ be a simplex in space $\boldsymbol{A}$, where $\boldsymbol{x_1} = (1, 0, \ldots, 0), \ldots, \boldsymbol{x_n} = (0, 0, \ldots, 1)$, and a point p in $\boldsymbol{A}$ could be expressed as:

$$p = \lambda_1 \boldsymbol{x_1} + \ldots + \lambda_n \boldsymbol{x_n} \tag{63}$$

If the simplex is a convex hull, and all $\lambda_1, \ldots, \lambda_n$ falls in $[0, 1]$, we could conclude that p is within the simplex.

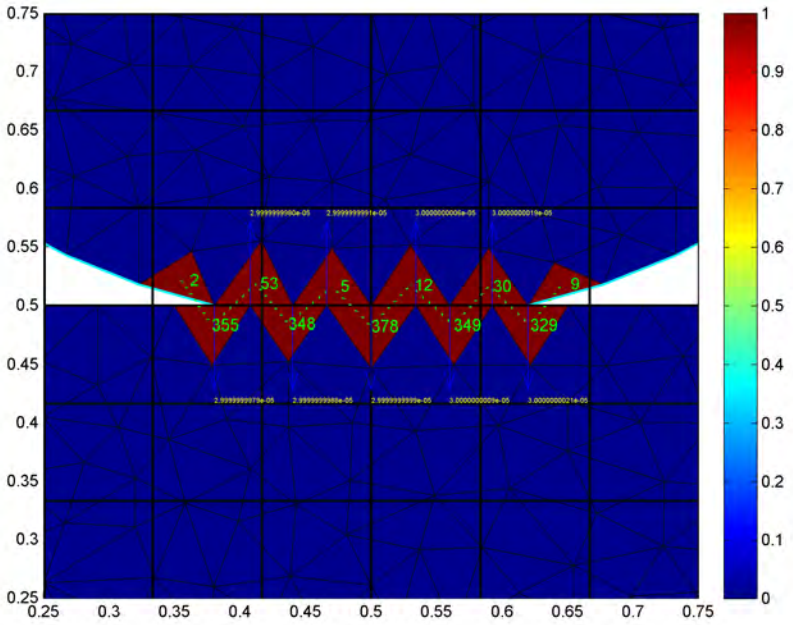


Figure 6: A demonstration for collision detection, where black solid line shows cube division, blue indicates non-collision triangles and red indicates triangles that collide, green line between each pair shows the pair of elements that collide, blue arrow shows the contact surface normal and green number shows the distance between them.

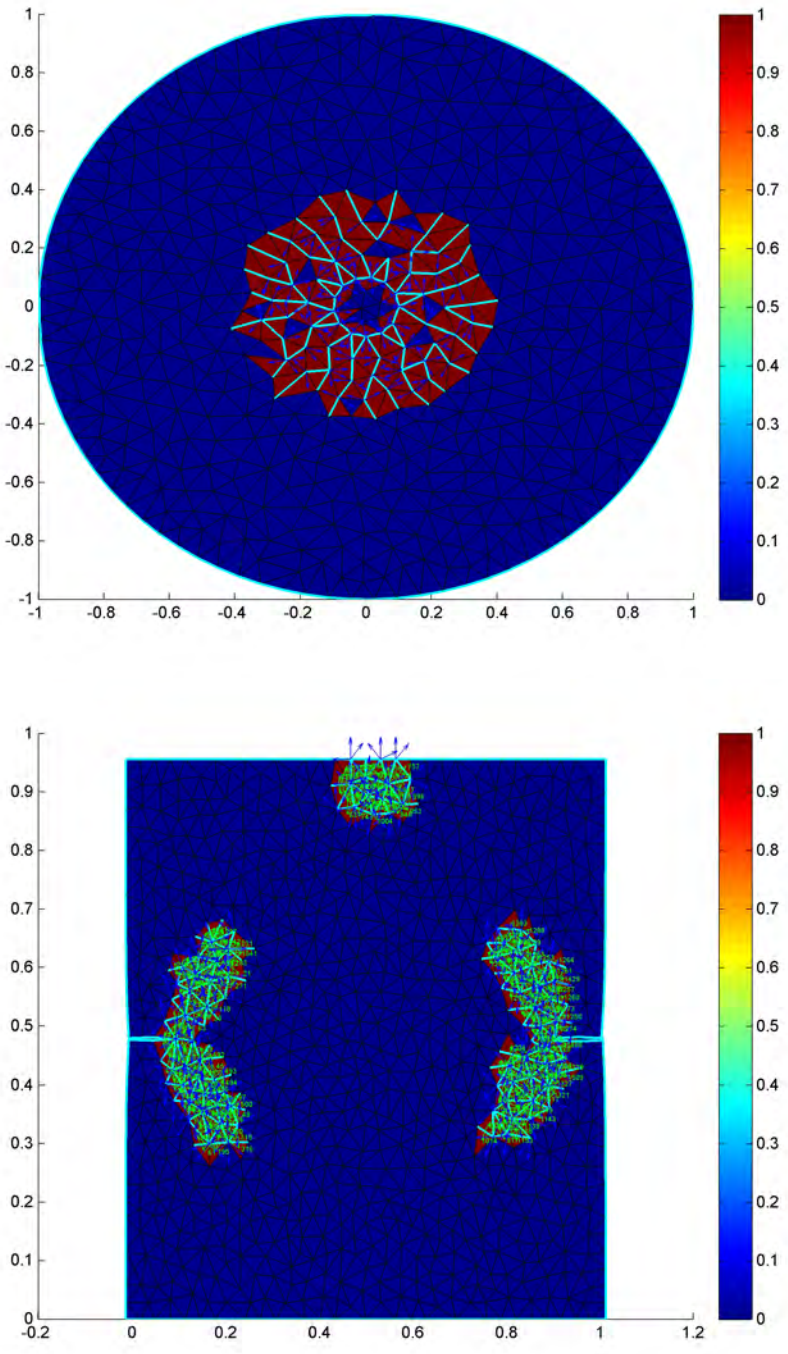


Figure 7: More complicated demonstrations in various simulations.

For triangle elements A, where the Cartesian coordinates of the vertices are $(x_1, y_1), (x_2, y_2), (x_3, y_3)$,given point is (x, y), the Barycentric coordinate could be calculated from the following, see fig 8:

$$\lambda_1 = \frac{(y_2 - y_3)(x - x_3) + (x_3 - x_2)(y - y_3)}{(y_2 - y_3)(x_1 - x_3) + (x_3 - x_2)(y_1 - y_3)} \tag{64}$$

$$\lambda_2 = \frac{(y_3 - y_1)(x - x_3) + (x_1 - x_3)(y - y_3)}{(y_2 - y_3)(x_1 - x_3) + (x_3 - x_2)(y_1 - y_3)} \tag{65}$$

$$\lambda_3 = 1 - \lambda_1 - \lambda_2 \tag{66}$$

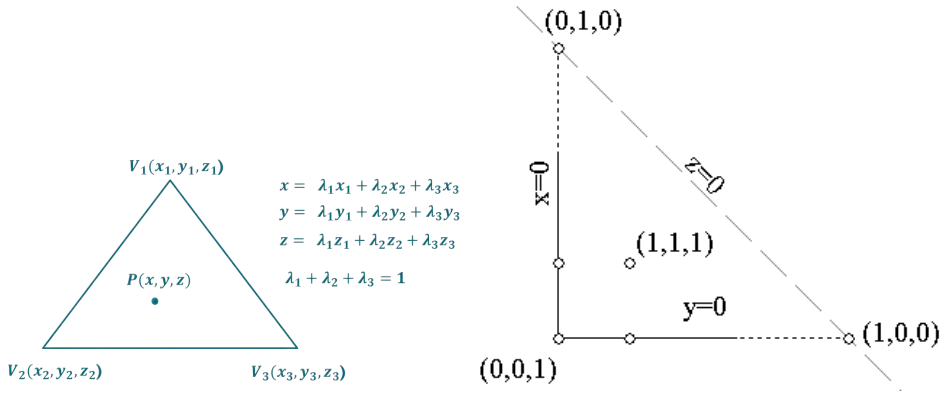


Figure 8: Barycentric Coordinate.

The second step for checking whether two edges $\boldsymbol{r}$ starting at point $\boldsymbol{p}$ and $\boldsymbol{s}$ starting at point $\boldsymbol{q}$, see fig 9, are performed by the following:

$$\begin{gathered} A_r = (\boldsymbol{q} - \boldsymbol{p}) \times \boldsymbol{r}, \qquad A_s = (\boldsymbol{q} - \boldsymbol{p}) \times \boldsymbol{s}, \qquad \Delta_{rs} = \boldsymbol{r} \times \boldsymbol{s} \\ u = A_r / \Delta_{rs}, \qquad t = A_s / \Delta_{rs} \end{gathered} \tag{67}$$

and the cases are:

1. If $\Delta_{rs} = 0$ and $A_r = 0$, in which case A_s is also 0, then the two lines are collinear. If an interval $[t_0, t_1]$ formed by $t_0 = (\boldsymbol{q} - \boldsymbol{p}) \cdot \boldsymbol{r} / (\boldsymbol{r} \cdot \boldsymbol{r})$ and $t_1 = t_0 + \boldsymbol{s} \cdot \boldsymbol{r} / (\boldsymbol{r} \cdot \boldsymbol{r})$ intersects with $[0, 1]$ then the segments are collinear and overlapping, otherwise they are disjoint.

2. If $\Delta_{rs} = 0$ and $A_r \neq 0$, then the two lines are parallel and non-intersecting.

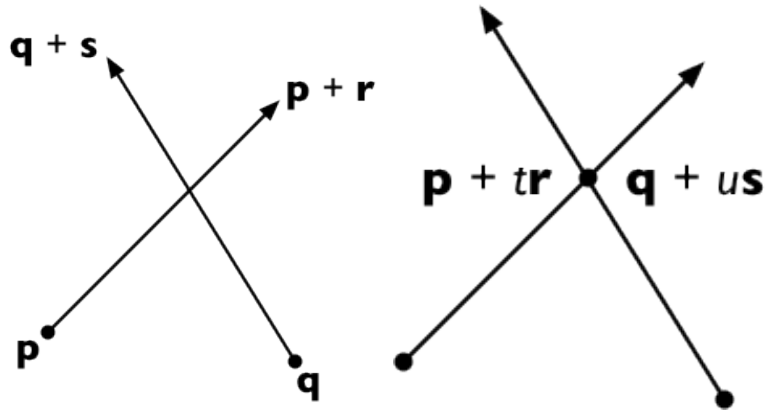


Figure 9: Barycentric Coordinate.

3. If $\Delta_{rs} \neq 0$, $0 \leq t \leq 1$ and $0 \leq u \leq 1$, then the two segments intersect at point $\boldsymbol{p} + t\boldsymbol{r} = \boldsymbol{q} + u\boldsymbol{s}$

4. Otherwise, segments do not intersect.

Especially for triangle elements, we need only to calculate 6 vertices and 9 pairs of edges.

4.5 Constraint Calculation

The constraints are needed in the system to push elements in collision away in the next step calculation. However, to determine the proper constraints for each pair of elements, and make sure overall constraints collected from all elements keep the KKT system stable is tricky. Because the shape, position, size, collision angle and distance may vary greatly for different elements, designing an universal algorithm that would fit in all different situations is difficult. In the development, the algorithm is modified numerous times in order to retrieve better results. The following paragraph will describe the ideas and how it is implemented in the code.

4.5.1 Corner Case Problem

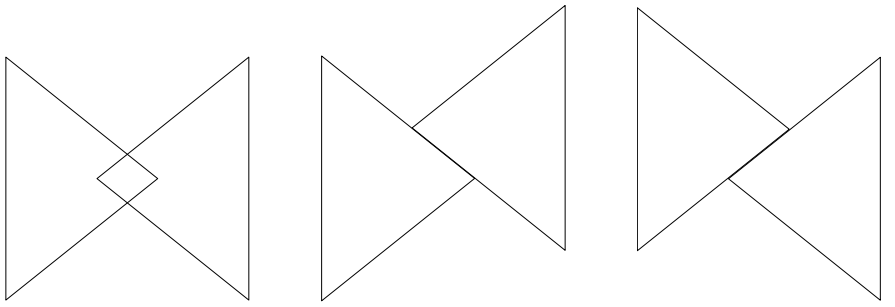
The algorithm, aimed at prescribing proper constraints for brittle material with complicated cracking surfaces, is not probable if we decided to track all cracks and compute constraints thereafter. Thus, element-pairwise calculation is a natural choice. However, the KKT system is sensitive to any

excessive constraints, or inadequate constraints, especially if the system is large, for a probable reason that the constraints try to restrict the elements in an exact place. Yet we are still able to obtain a stable set of constraints if designed carefully. The constraint calculation algorithm is mainly consist of the following two parts:

1. Decide proper point-to-edge constraints between given pair of 2 elements.

2. Remove redundant constraints due to sharing of edges and vertices between adjacent elements.

The first part is not trivial, even if the deformation and collision is tiny between steps. Think of the following situation, Which edge would the penetrated point be pushed to, especially for nearly point-to-point collisions, see figure 10,

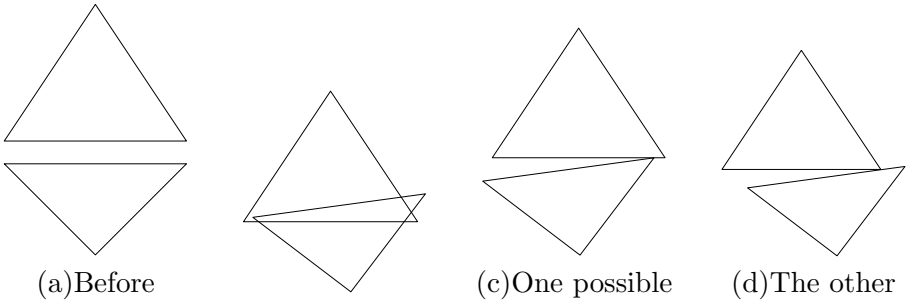


(a)Two elements collide. (b)One possible solution. (c)The other possible solution.

Figure 10: Ambiguous collision situations.

One possible solution is to use the virtual velocity, that is using the displacements between nodes and edges before and after collision to determine the direction the two elements move and penetrate each other, so that we can calculate a specific virtual time point t, at which the two elements exactly begin to collide, as well as to determine the correct edge and point from both elements that contribute to the constraint, see figure 11 (a) and (b).

However, because the virtual velocity is an linear estimation, the corner case problem is sensitive regarding rounding errors, which may result in different constraints and solutions, see figure 11 (c) and (d).



(a)Before collision. (b)After collision. (c)One possible solution. (d)The other possible solution.

Figure 11: Ambiguous collision situations.

For brittle materials with tiny deformation and collisions, the cases that two element segments penetrating each other at the corner or by the edge, as well as rounding error is prevalent. One feature we would need is rounding error control, the other is an algorithm to determine the correct edge and point for constraint calculation.

The rounding error control is performed though adding ϵ in equations , , and 67, and detect the $\bar{\lambda_i}$, $\bar{\boldsymbol{u}}$, and $\bar{\boldsymbol{t}}$ instead:

$$\begin{aligned} \bar{\lambda_i} &= \lambda_i + \epsilon,\ i = 1, 2, 3 \\ \bar{\boldsymbol{u}} &= \boldsymbol{u} + \epsilon, \qquad \bar{\boldsymbol{t}} = \boldsymbol{t} + \epsilon \end{aligned} \tag{68}$$

ϵ can be both positive and negative, which indicates the detect range around given element. In our code, we usually use $-10^{-3}|p|$, where $|p|$ is average element edge length. Here negative means we do not treat two elements as colliding if their penetrating depth is not more than $-10^{-3}|p|$.

The overall rules for algorithm determining the correct edge and point set is introduced previously. Besides, we introduced an random solution to resolve the ambiguous corner-to-corner collision in figure 10. Because pointed corner collision elements could glide to both sides, in which case, we could randomly choosing only one constraint from the two possible solutions.

4.5.2 Six Elements Test

Below is a test and demonstration of the constraint algorithm on six isolated elements.

The first part is pushing the top element against the six elements, see figure 12. In the process we could see it reached a corner to corner collision, and the algorithm chose the one on the right to calculate collision constraint

with. The bottom elements separated smoothly as the top one is pushed down through. If random selection algorithm is not applied, both collision constraint will keep the two elements in a strange point-to-point position.

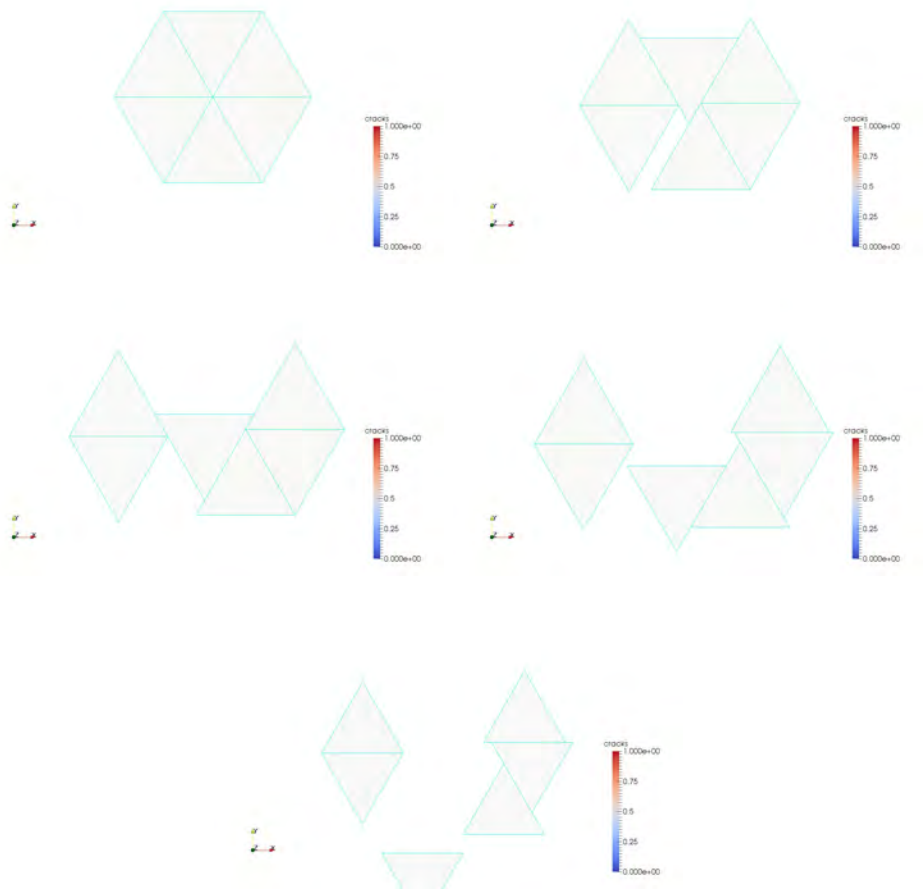


Figure 12: Push the top on six isolated elements.

The second part is rotating the top element, in which the inside vertex of the element is fixed and the total element is rotated around that vertex in a clockwise direction, see figure 13. There is also corner-to-corner collision case when 2 elements corner collide toward a third element's edge, and the random feature resolved the situation well.

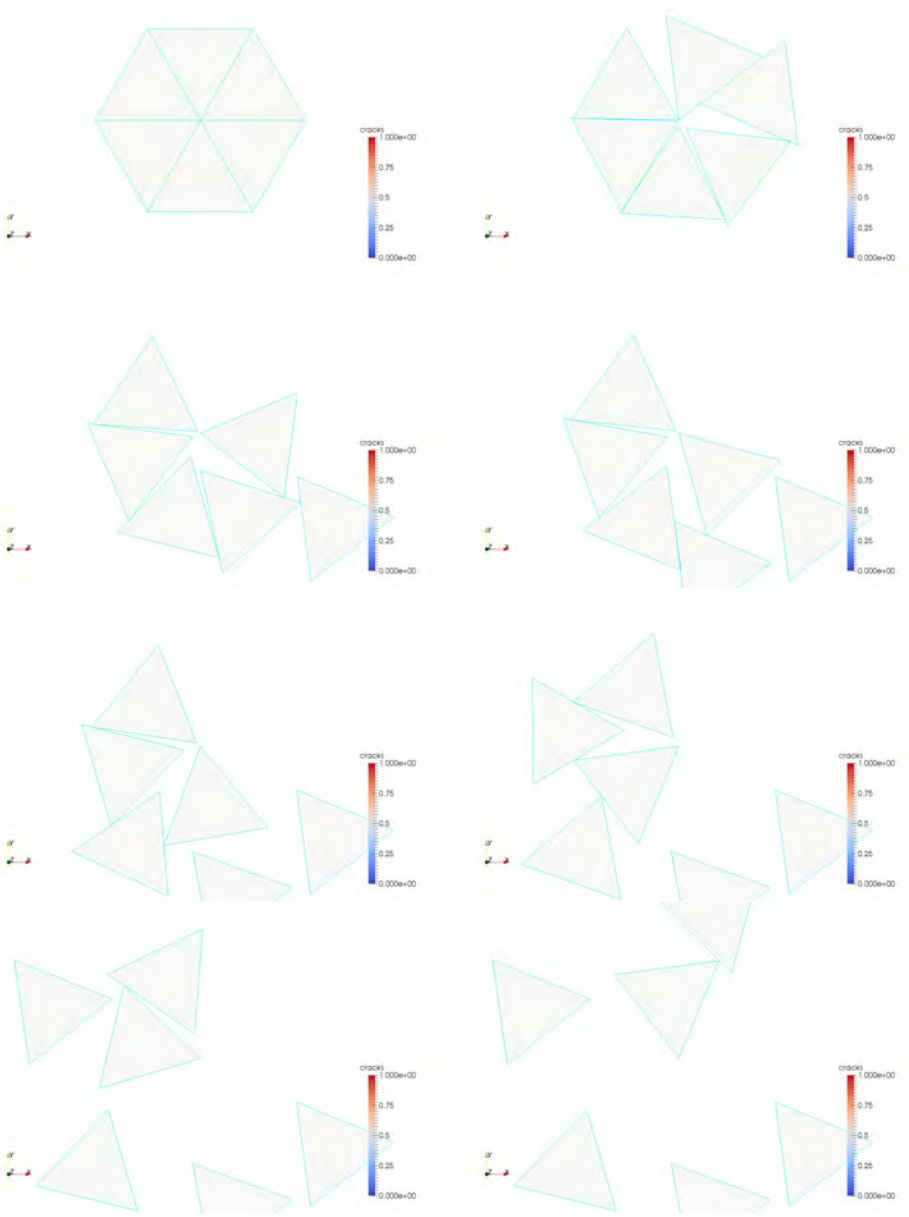


Figure 13: Rotate the top on six isolated elements.

4.6 Parallelazation

In order to achieve the maximum efficiency, apart from the MATLAB code which is more friendly for developing, the parallel code is implemented as a combination of C/C++ source code and various multithread C/C++ libraries.

Based on performance test of MATLAB code, the bottle necks are three parts, and we parallel them separately.

1. Collision pair-wise detection.
2. QR factorization solver to remove redundant constraints.
3. LU factorization solver to solve stiffness matrix.

The C/C++ parallel code is consist of modified source code generated from MATLAB-to-C compiler and stand alone parallel matrix solvers.

4.6.1 Collision Detection

In the previous section, we described a grid based detecting technique. Because the detecting is among local cubes, we could easily distribute the totally computation to different threads, given the overall domain information is stored and shared in memory. We implemented this parallel detecting part by automatically dividing and distributing all cubes work to a user prescribed number of total threads, with methods from pthread package.

4.6.2 Multithread QR Factorization Solver

The time cost for remove redundant constraints is huge if system becomes more complicated. We used the SuiteSparse multithread QR decomposition package to resolve the issue. To use the package, we implemented functions to transform the original matrix format into SuiteSparse QR package format back and forth. By calling methods in our remove redundant function, we could greatly reduce the QR factorization computation time.

4.6.3 Multicore LU Decomposition Solver

To make the multithread code more compatible with its other parts, we firstly would like to use a multithread LU solver, however, there are no well maintained or working multithread LU solver libraries available. Thus we used the PETSc multicore solver to solve the LU decomposition part.

Firstly, the whole program is distributed and started on a multicore machine, however, only one core is doing the multithread computation, including collision detection and QR factorization. After previous steps are done, we started distributing the stiffness matrix onto different cores and using PETSc library function to solve LU decomposition. The PETSc solver would collect solved result data from cores and assemble together to generate next step input on the main core.

4.7 Verification

In this section we introduce some verification tests for some problems linear elasticity. These tests verify that the finite element implementation correctly models the mathematical theory of continuum mechanics.

Consider a material body Ω with a global Cartesian frame with basis $\boldsymbol{e}_1$, $\boldsymbol{e}_2$, $\boldsymbol{e}_3$. Suppose there is a body force $\boldsymbol{f}(\boldsymbol{x}) = f_1\boldsymbol{e}_1 + f_2\boldsymbol{e}_2 + f_3\boldsymbol{e}_3$ which acts on the material at position $\boldsymbol{x}$ and possibly by a surface force $\boldsymbol{t}(\boldsymbol{x}) = t_1\boldsymbol{e}_1 + t_2\boldsymbol{e}_2 + t_3\boldsymbol{e}_3$ acting on a portion of the boundary Γ_N. The body deforms to a new configuration ${}^t\Omega$ with each new position given in the Cartesian frame of reference by ${}^t\boldsymbol{x} = \boldsymbol{x} + \boldsymbol{u}$, where $\boldsymbol{u} = (u_1, u_2, u_3)^T$ is the displacement of each material point in the Cartesian basis. There is another region of the boundary, Γ_D, such that the material point at $\boldsymbol{x} \in \Gamma_D$ is constrained by a set of m linear equations acting on its displacements written as $\boldsymbol{M}\boldsymbol{u} = \boldsymbol{b}$, where $m \leq 2$ in two dimensions and $m \leq 3$ in three dimensions. Then the displacement formulation of the Navier-Cauchy equations[1] are [5]

$$(\lambda + \mu)\, u_{j,ji} + \mu u_{i,jj} = -f_i, \quad \text{on } \Omega; \tag{69}$$

$$(\lambda u_{k,k}\delta_{ij} + \mu(u_{i,j} + u_{j,i}))\, n_j = t_i, \quad \text{on } \Gamma_N; \tag{70}$$

$$m_{ij}u_j = b_i, \quad \text{on } \Gamma_D. \tag{71}$$

We verify our finite element code by comparing the vertex displacements determined by the finite element discretization against problems with known solutions through a series of convergence studies described in the following sections. In particular, we show that the finite element code can be used for problems in linear elastic solid mechanics.

[1] It should be noted that the Navier-Cauchy equations are derived from the kinematic equations, material constitutive equations, and the displacement formulation of linear strains.

4.7.1 L-shaped Structure

The first verification test is of an L-shaped body, as shown in Figure 14, with corner vertices arranged in counterclockwise order as $(-1,-1)$, $(0,-2)$, $(2,0)$, $(0,2)$, $(-1,1)$, $(0,0)$. The displacement field is known in polar coordinates as

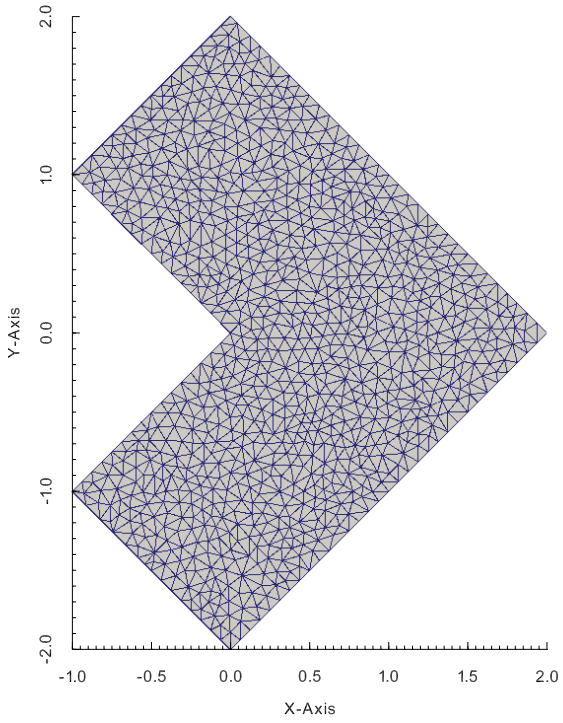


Figure 14: The initial material configuration for the L-shaped structure. This particular discretization contains 1003 vertices and 1878 elements.

$$
\begin{aligned}
u_r(r,\theta) &= \frac{1}{2\mu} r^{\alpha} \left[-(\alpha+1)\cos\left((\alpha+1)\theta\right) + \left(C_2 - (\alpha+1)\right) C_1 \cos\left((\alpha-1)\theta\right) \right], \\
u_\theta(r,\theta) &= \frac{1}{2\mu} r^{\alpha} \left[(\alpha+1)\sin((\alpha+1)\theta) + \left(C_2 + \alpha - 1\right) C_1 \sin((\alpha-1)\theta) \right].
\end{aligned}
\tag{72}
$$

where $\alpha \approx 0.544483737$ is the solution to the equation $\alpha \sin(2\omega) + \sin(2\omega\alpha) = 0$ with $\omega = 3\pi/4$, $C_1 = -(\cos((\alpha+1)\omega)) / (\cos((\alpha-1)\omega))$, and $C_2 = (2(\lambda+2\mu)) / (\lambda+\mu)$ [5]. The displacement solution (72) solves (69) with $f_i = 0$, and the boundary conditions are imposed as pure displacement conditions given by (72). This test case is very useful because its solution contains a strain/stress singularity at the reentrant corner located at $(0,0)$.

The problem was solved on several levels of refinement, using the displacement formula (72) to determine the value of the Cartesian components of the displacements for each of the boundary vertices. The linear stiffness matrix and associated boundary conditions are assembled into a linear system

$$\begin{bmatrix} \boldsymbol{A} & \boldsymbol{B}^T \\ \boldsymbol{B} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{b} \end{bmatrix}, \tag{73}$$

where $\boldsymbol{A}$ is a normalized stiffness matrix[1], $\boldsymbol{u}$ is a vector storing the Cartesian components of the vertex displacements, $\boldsymbol{\lambda}$ is a vector storing the Lagrange multipliers, $\boldsymbol{B}$ and $\boldsymbol{b}$ store the displacement constraints information in the form $\boldsymbol{B}\boldsymbol{u} = \boldsymbol{b}$. The linear system (73) is solved using GMRES with incomplete LU-factorization.

The convergence of relative L^2 displacement error is compared for several levels of refinement is shown in Figure 15 and shows linear convergence of the L^2 relative error. The linear convergence is indicative of the strain singularity which appears in the reentrant corner at $(0,0)$, as seen in Fig 16.

4.7.2 Verification by Method of Manufactured Solutions

Verification by the method of manufactured solutions (MMS) is a technique for verifying numerical codes by choosing a known solution, inserting into a homogeneous partial differential condition, and identifying the corresponding nonhomogeneous forcing function which the known solution solves [16]. In the case of the Navier-Cauchy equations, one is able to compute a body force $\boldsymbol{f}$ using (69) once given the Lamé parameters, λ and μ, and the known displacement field $\boldsymbol{u}$. Neumann and Dirichlet boundary conditions can also be determined from the displacement solution.

Verification by MMS has the advantage that numerical codes can be tested against the mathematical theory that they model without having

[1]The stiffness matrix is normalized by multiplying by the scalar parameter $(\lambda+2\mu)^{-1}$, which nondimensionalizes the stiffness matrix $\boldsymbol{A}$. This improves the condition number of the linear system and hence the convergence rate of the iterative methods used to solve the linear system.

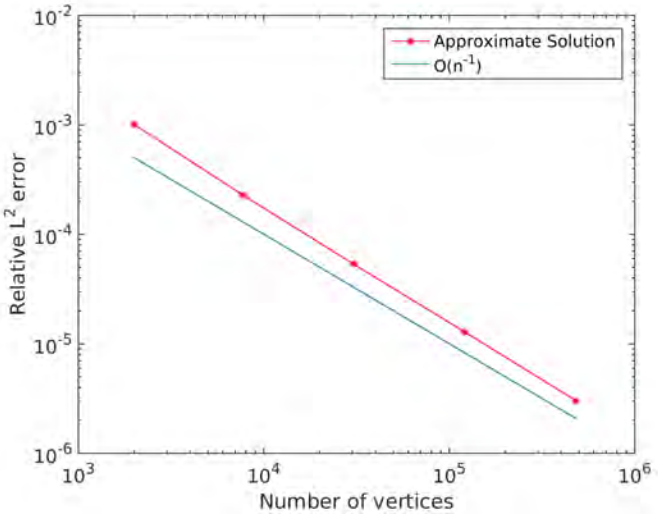


Figure 15: Convergence plot of the L-shape displacements compared to number of vertices.

to accurately model the physical behavior. In these next two examples, we examine solutions with displacements which are well outside the realm where linear elasticity reliably describes the behavior of real material. In both of these examples, we compute an expression for the body force from the chosen analytic displacement field $\boldsymbol{u}(\boldsymbol{x})$ and a variety of choices of Lamé parameters derived from real material properties. Specifically the Young's modulus, E, and Poisson ratio, ν, are chosen from real materials.

4.7.3 Plane stress verification

To verify the two dimensional code, a unit circle centered at $(0, 0)$ is deformed by a displacement field given by the vector-valued function

$$\boldsymbol{u}(x, y) = (\cos x - \sin y - 1)\boldsymbol{e}_1 + (\sin x + \cos y - 1)\boldsymbol{e}_2 \tag{74}$$

where $\boldsymbol{e}_i$, $i \in \{1, 2\}$, correspond to the Cartesian basis vectors in the x and y directions. From the known displacement solution (74) and Lamé parameters, the appropriate force function can be computed *analytically* as

$$\boldsymbol{f}(x, y; \lambda, \mu) = ((\lambda + 2\mu)\cos x - \lambda \sin y)\,\boldsymbol{e}_1 + (\lambda \sin x + (\lambda + 2\mu)\cos y)\,\boldsymbol{e}_2. \tag{75}$$

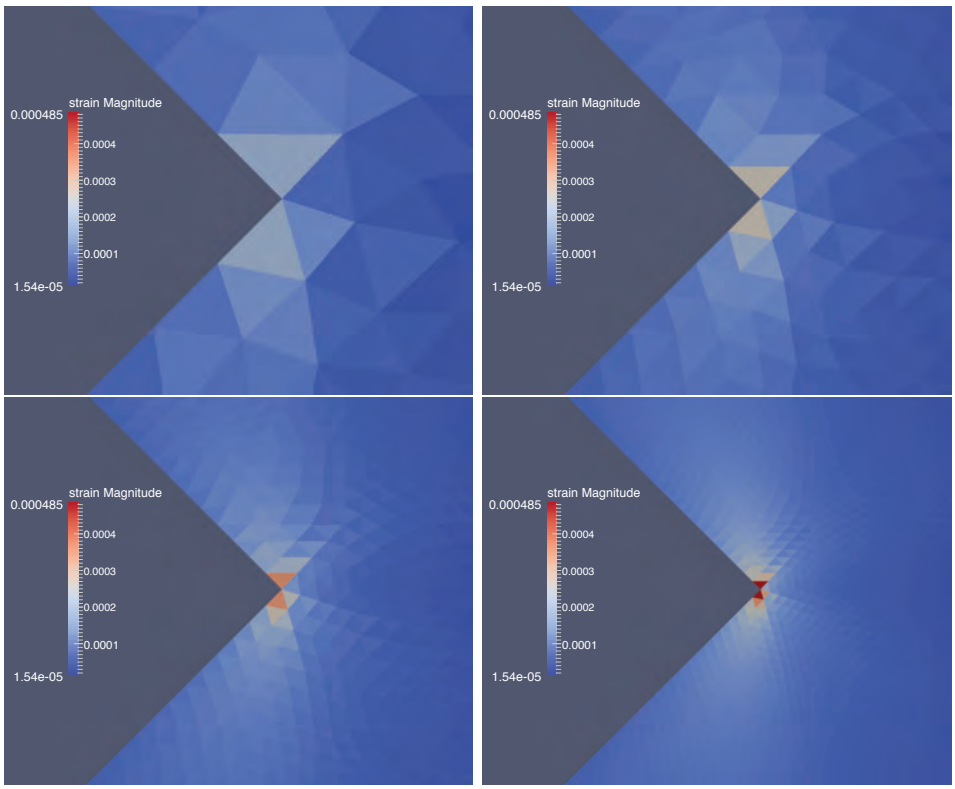


Figure 16: Close up view of reentrant corner and the Frobenius norm of the linear strain tensor for various levels of refinement.

The Lamé parameters are computed from the assumptions of plane stress, namely

$$\lambda = \frac{E\nu}{1-\nu^2}, \tag{76}$$

$$\mu = \frac{E}{2(1-\nu)}. \tag{77}$$

The Young's modulus and Poisson ratio were chosen to reflect four different material choices, as shown in Table 1.

Material Name (Symbol)	**Young's modulus (GPa)**	**Poisson Ratio**
Aluminum (Al)	70	0.30
Gold (Au)	83	0.44
Iron (Fe)	170	0.20
Nickel (Ni)	210	0.31

Table 1: Material properties in verification studies.

Boundary conditions are chosen to be pure Dirichlet boundary conditions where the components of the vertex displacements are determined by (74), and no external surface loads are assumed. The Dirichlet boundary conditions are enforced via Lagrange multipliers, resulting in a linear system similar to (73).

The unit circle was discretized using the Delaunay triangulation [18] for several levels of refinement varying from 122 vertices to 28,577 vertices. The final configuration for the coarsest mesh is shown in Figure 17.

Results of the convergence study are shown in Figure 18, and show that the finite element solution matches very well with the analytic solution at each of the vertices. As expected, the behavior of convergence is largely independent of the material properties chosen with the exception of very refined meshes.

4.7.4 Three dimensional verification

The final numerical study is performed on a three dimensional axis-aligned ellipsoid with semi-axes lengths 1, 2, and 3 along the x-, y-, and z-axis, respectively. The known displacement solution is given as

$$\boldsymbol{u}(x,y,z) = \left(0.05x^3 + 0.1xyz\right)\boldsymbol{e}_1 + \left(0.05y^3 + 0.1xyz\right)\boldsymbol{e}_2 + \left(0.05z^3 + 0.1xyz\right)\boldsymbol{e}_3 \tag{78}$$

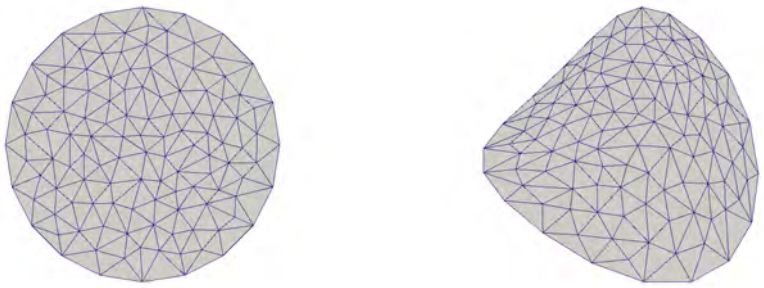


Figure 17: Original and deformed configuration of circle used in the verification tests.

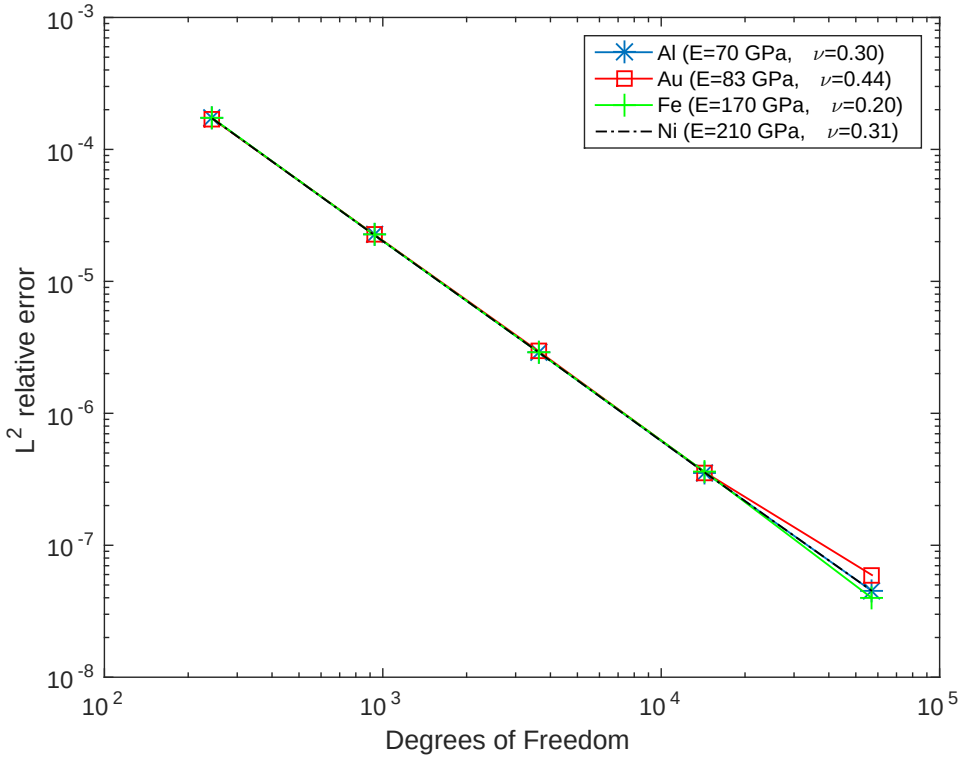


Figure 18: L^2 vertex displacement errors for several levels of refinement.

and the analytic force can be computed from the material coordinates and Lamé parameters as

$$\begin{aligned} \boldsymbol{f}(x,y,z;\lambda,\mu) = &-[0.3(\lambda+2\mu)x + 0.1(\lambda+\mu)(y+z)]\,\boldsymbol{e}_1 \\ &-[0.3(\lambda+2\mu)y + 0.1(\lambda+\mu)(x+z)]\,\boldsymbol{e}_2 \\ &-[0.3(\lambda+2\mu)z + 0.1(\lambda+\mu)(x+y)]\,\boldsymbol{e}_3. \end{aligned} \tag{79}$$

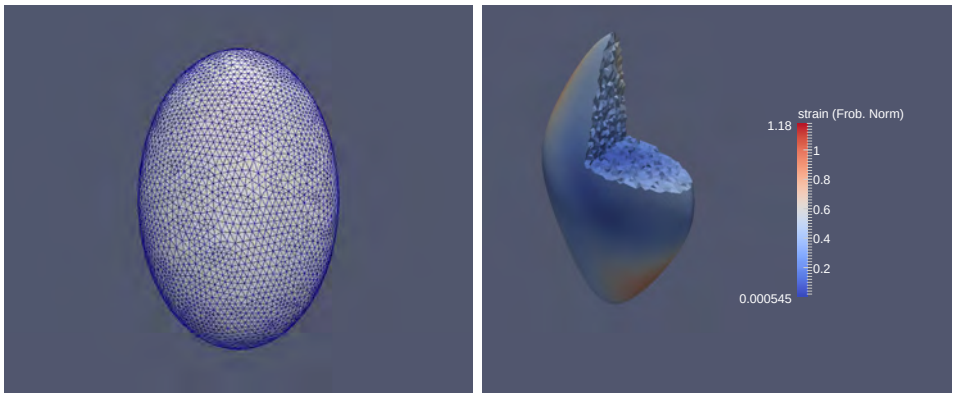


Figure 19: Initial configuration of ellipsoid (left) and a cut view of the final configuration with color map of the Frobenius norm of the strain tensor inside each element.

The ellipsoid was discretized using the TetGen [19] for several levels of refinement varying between 132 vertices and 25,202 vertices. Material properties were taken again from those in Table 1, and the Lamé parameters were computed using the formulas

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)} \tag{80}$$

$$\mu = \frac{E}{2(1+\nu)} \tag{81}$$

The convergence of the L^2 vertex displacement errors in Figure 20 indicates that the finite element solution does converge to the known analytic solution. The convergence is noticeably affected by the material properties, unlike in the plane stress case. Namely that the errors of gold ($E = 83$ GPa, $\nu = 0.44$) are slightly larger than the other three material choices. This is due to the fact that the Poisson ratio of gold is much closer to the theoretical upper bound of 0.5 than the other material.

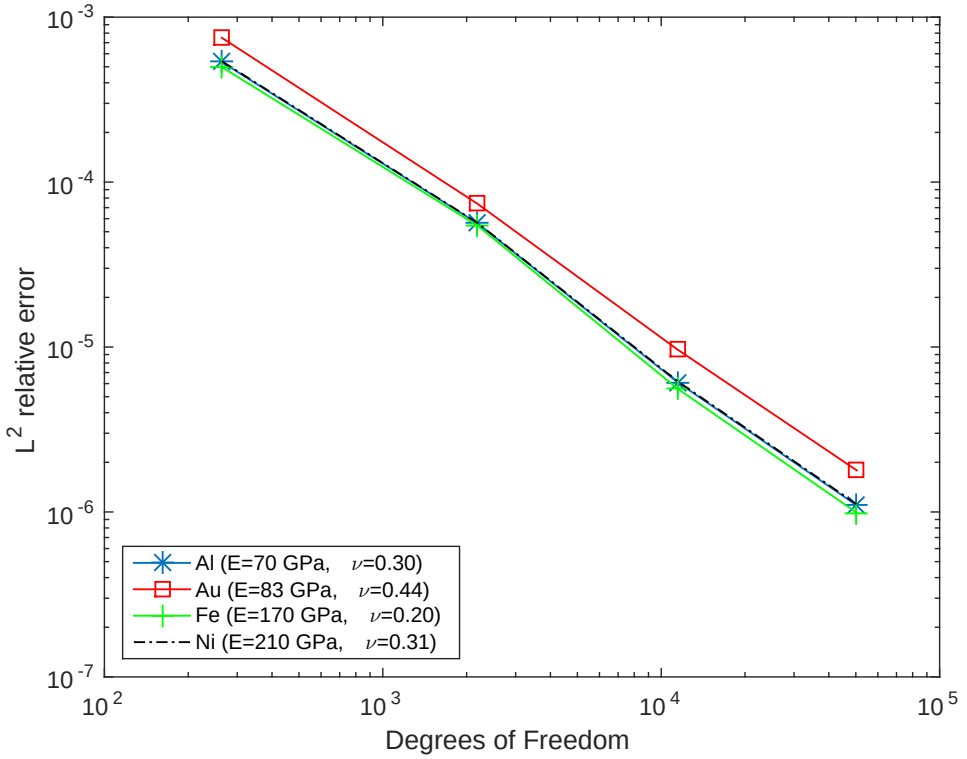


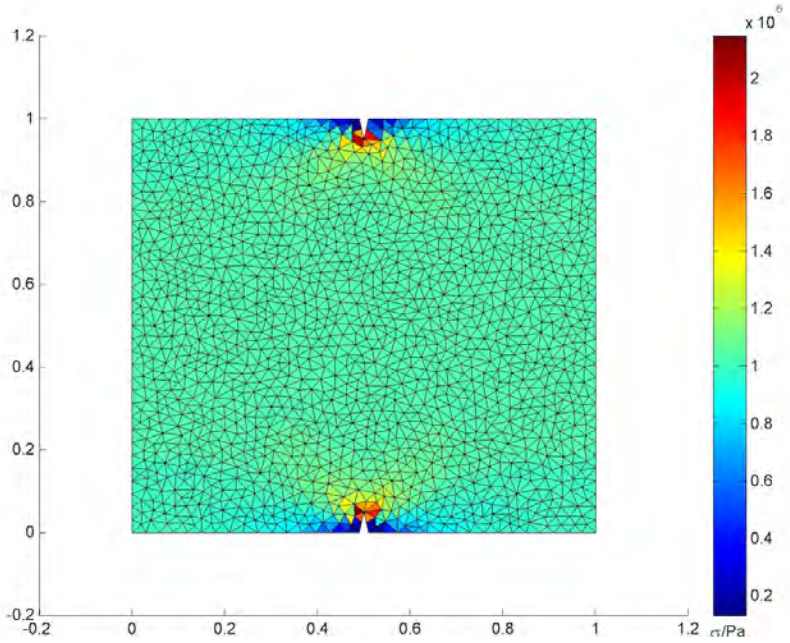
Figure 20: Convergence of L^2 vertex displacement errors for 3D verification study.

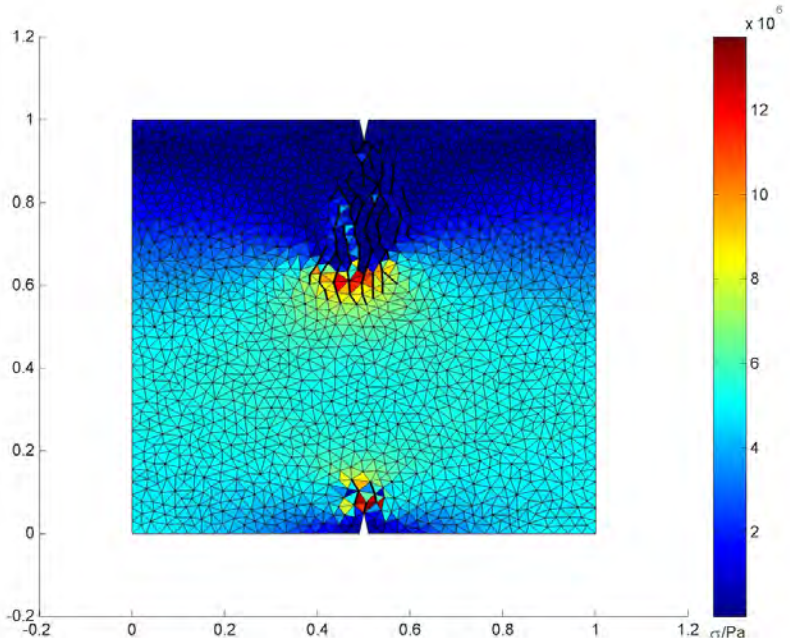
5 Simulation Examples

5.1 Stretch Brittle Defective Plate

To test the finite element fracture code, we simulate a square plate made of a brittle material (ceramic or glass), which has small defects on the top and bottom sides (see Figure 21). Exerting increasing deformation along the x direction on the left and right sides, we observe the formation and growth of cracks around the defects. In this simulation, the deformation displacement dx is varied based on the number of cracks at each step. If the number of cracks is larger than 2, we reduce the displacement dx. Otherwise, if there is no crack occurring in a series of steps, we increase the displacement dx. The physical parameters for this simulation are as follows: Young's Modulus is 5 GPa, the Poisson ratio is 0.18, and the critical breakdown strain is 1.3817×10^{-4}. We use a mesh whose minimum triangle area is 0.0005 and minimum triangle angle is $\pi/6$. The boundary conditions are gliding boundary conditions on the left and right along the y direction, which is a fixed dx for the x component while the y component is free, and the free boundary condition everywhere else. At each simulation step, we modify the gliding Dirichlet boundary condition for increased displacements on the left and right sides of the mesh. We then solve the updated-Lagrangian formulation for the nonlinear elasticity problem using the quasi- Newton method. A crack processing routine is applied to the mesh to open cracks around overs trained nodes, based on the principal direction of the stress tensor of the node, while the stress is an averaged value from the vertex's surrounding elements. To ensure the system's finite element discretized virtual work plus Lagrangian multiplier matrix to be non-singular or not badly scaled, we add extra boundary conditions for isolated elements. In our current case, because of lacking contact constraint of elements touching each other, we only placed two fixed points for each isolated piece of glass. In the end, we check if the crack number exceeds the maximum number of cracks prescribed. If so, we reduce the displacements dx at one step and re-run the procedure using the old data; otherwise we keep the deformation to be unchanged and run step 2 and 3 for a release of energy through crack formation until there are no more cracks occurring. Figure 21 shows some sample results of the fracture of a brittle plate under tension.

An example of fracture of a plate compressed by a sharp wedge is shown in Figure 22. As the material property of the wedge is the same as that of the plate, the wedge cracks first.


x 10^6
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
σ/Pa
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-0.2
0
0.2
0.4
0.6
0.8
1
1.2


x 10^6
12
10
8
6
4
2
σ/Pa
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-0.2
0
0.2
0.4
0.6
0.8
1
1.2

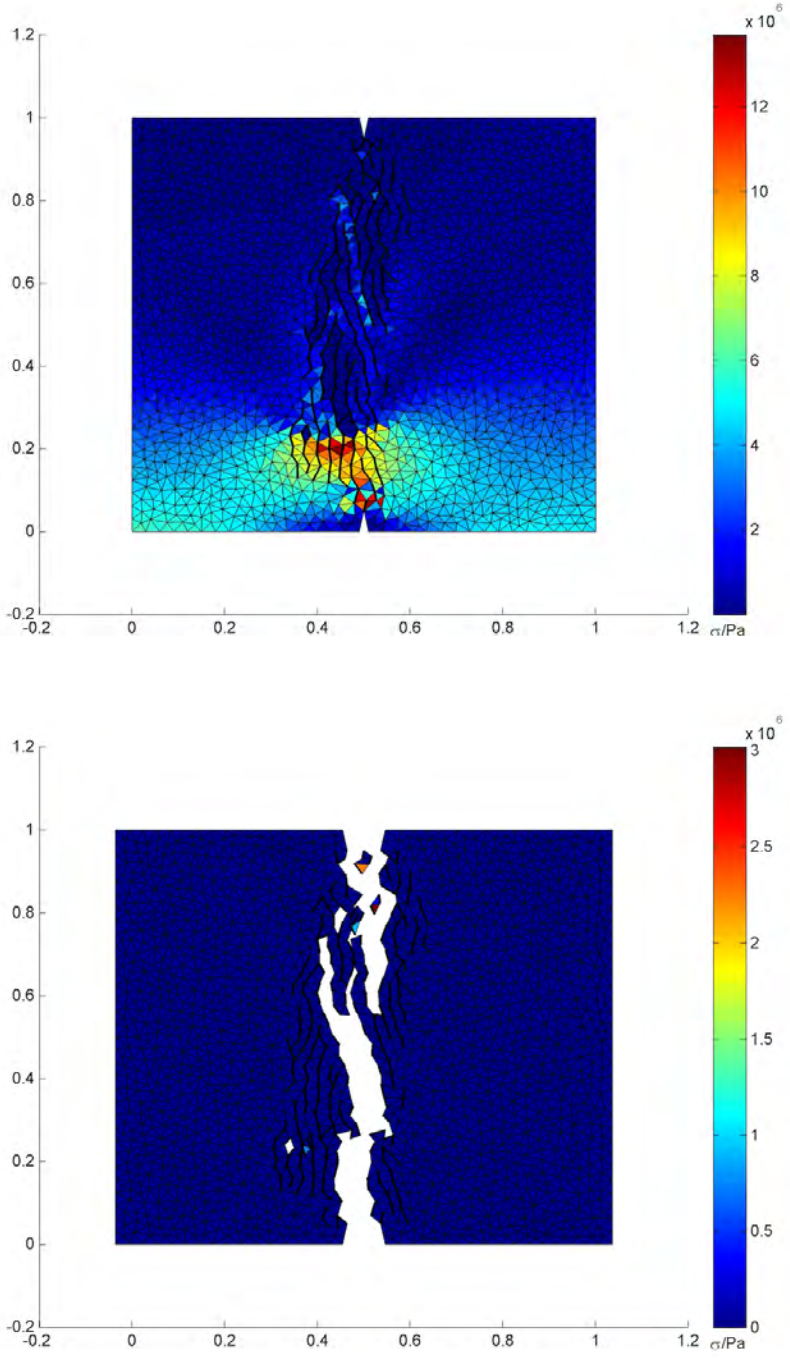


Figure 21: Fracture of brittle plate under tension.

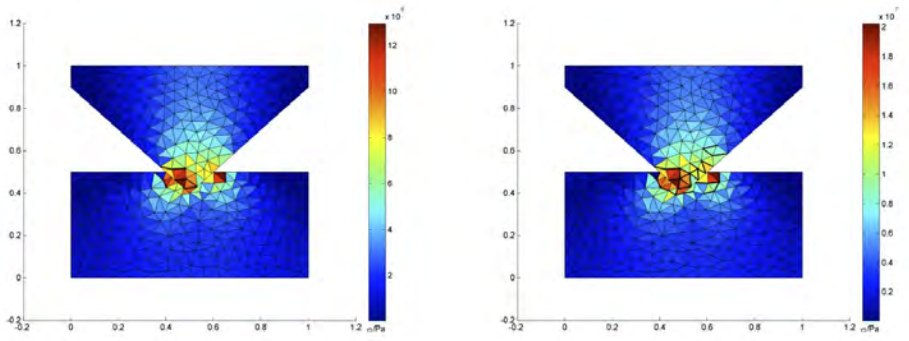


Figure 22: Fracture plate compressed by a wedge.

5.2 Fracture of Brittle Disks Hit by Projectiles

To test the robustness of the model, we simulate a brittle disk that has a hole in the center. The hole is in contact with an expanding softer material core, which is not displayed; see Figures 23. While exerting increasing expansion on the inner core, which is passed by contact condition towards the outer disk, cracks begin to form around the hole. The applied deformation of the inner core is such that the inner part of the disk completely disintegrates into grains, forming the so-called comminuted zone, while the outer part of the disk remains intact.

Figure 23: Fracture of brittle disk under core expansion.

Under increased deformation of the inner core, the disk undergoes complete disintegration into disjoint fragment (see Figures 24). This simulation, performed using a coarse initial mesh, gives results that qualitatively agree with previously published data. The robustness of the algorithm for the selection of constraints still need some improvements for for highly refined

meshes.

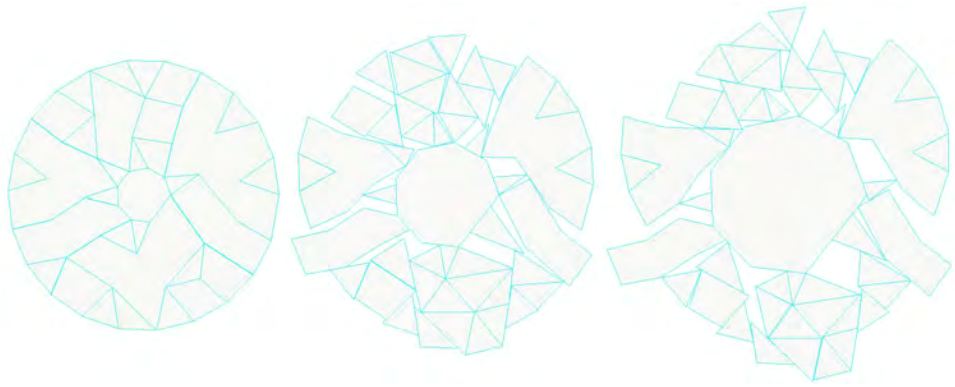


Figure 24: Complete disintegration of brittle disk under core expansion.

6 Dynamic theory of brittle fracture

The equations and physics modeling of failure waves in brittle fracture have been a concern for over 40 years, with partial and ad hoc which seem imperfectly connected to laws of physics.

Here we address this problem, and offer a solution (with a full development still in process). The closest of the previous work to our ideas is the contributions of Feng [30], who postulated a diffusion equation for material damage, but whose constitutive equations appear to be phenomenological rather than fundamental, and perhaps for this reason did not attract a large following.

We introduce concepts from thermodynamics as the foundation of our equations. These concept are partly fully standard and classical, and lead to the conservation laws of linear and nonlinear elasticity. They are also partly less standard, but not totally original, and here are based on minimization of an entropy rate. There is thus a double minimization in the thermodynamics derivation, and with the second minimization, we recover a diffusive term added to the conservation laws, for the propagation of the material damage.

The resulting equations include the model of Feng as a special case, but importantly, they allow and suggest a more fundamental formulation of the constitutive laws, which in Feng's treatment appear to be unsatisfactory, or preliminary.

Future plans are to bring this theoretical development to completion,

and to couple it with simulations, both at a continuum level and mesoscopic simulations, through use of the particle based fracture models described elsewhere in this report.

7 Theories of turbulence

We extend the modeling of Kolmogorov and Obukhov 1962 [24, 25] to characterize the space and time statistics of turbulent velocity gradients (turbulent fluctuations). The variance of these quantities is proportional to $(\partial u_i/\partial x_j)^2$, and is a component of the dissipation rate ϵ, and studied by Kolmogorov and Obukhov, who postulated log normal statistics for ϵ.

Our theory is formulated in the context of a Large Eddy Simulation (LES), which has both resolved scales, and also some number of unresolved but still turbulent scales. These are called subgrid scales, and require models and closure terms in an LES simulation.

Our results are based on a rescaling of space and time, to remove the nonuniversal character of the resolved scales from the subgrid scales. For the rescaled subgrid statistics we find a strong universality for the variance of $\ln \epsilon$ and related velocity gradients. This variance is universal within all inertial scale rescaled subgrid degrees of freedom, a fact that leads to new scaling laws for the associated variance $\Sigma^2 \sim k^{-1}$ and the fluctuation decay time scale $T \sim k^{-1}$. These two new scaling laws allow effective parameterization of both Σ and T, with corrections for the dissipative corrections at the end of the inertial region. The two quantities Σ^2 and T are the key inputs to a proposed stochastic random field equation to describe ϵ and $\ln \epsilon$

The resulting theory has limits in terms of the allowed range of unresolved subgrid scales to be modeled, and a renormalization group extension is suggested for the case of a large number of unresolved (subgrid) inertial scales.

The theory is applied to closures for particles transported in a turbulent flow, to predict particle clustering in such flows.

The entire program is still under development. See [26, 27, 28, 29].

8 Development on Discretization Technologies

In this project, we have developed the robust discretization methods. We have successfully used the WLS-based discretizations in the frameworks of finite difference, finite element, finite volume, and particle methods. We generalized these methods to unstructured meshes in an accurate and stable

fashion, and also applied them to solve various PDEs. We summarize the developments in three areas.

8.1 Robust Adaptive Finite Element and Finite Difference Methods

The finite element methods (FEM) are arguably one of the most important numerical tools for solving partial differential equations (PDE) over complex domains in engineering. They account for an overwhelming majority of the commercial and research code for modeling and simulations. However, the finite element methods depend heavily on element shape quality for stability and good performance, and as a result engineers spend significant time in generating and optimizing meshes.

In our recent work [31], we have introduced the *Adaptive Extended Stencil Finite Element Method* (AES-FEM) as a means for overcoming this dependence on element shape quality. Our method replaces the traditional basis functions with a set of *generalized Lagrange polynomial (GLP) basis functions*, which we construct using local weighted least-squares approximations. The method preserves the theoretical framework of FEM, and allows imposing essential boundary conditions and integrating the stiffness matrix in the same way as with the classical FEM. In addition, AES-FEM can use higher-degree polynomial basis functions than the classical FEM, while virtually preserving the sparsity pattern of the stiffness matrix.

The AES-FEM method is related to the generalized finite difference (GFD) methods. Earlier works on GFD, such as [32], could only achieve second order accuracy. By using the WLS framework, we can generalize the finite difference methods from structured meshes to unstructured meshes with arbitrarily high-order accuracy. In particular, for a given weighted stencil, the WLS formulation allows us to perform either *explicit differentiation* (i.e., to compute the differential operators from values at the stencil) or *implicit differentiation* (i.e., to express a differential operator as a formula of the values at the stencil, a.k.a. *finite difference formulae*). The former can be used in explicit solvers for differential equations, and the latter can be used in either explicit or implicit solvers. In GFD, the stencil can be very irregular, and hence can deliver accurate and stable solutions over unstructured meshes.

Both AES-FEM and GFD deliver high-order accuracy over unstructured meshes. Figure 25 shows numerical comparisons in solving for the Poison equation and a time-independent convection-diffusion equation. These results demonstrate that AES-FEM and GFD are both more accurate than

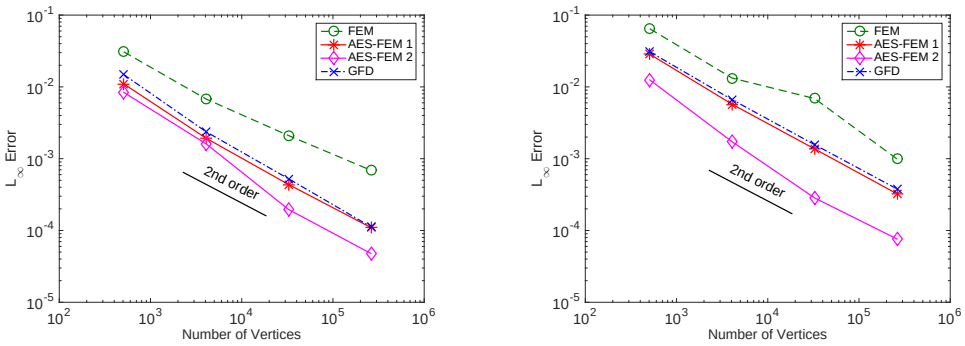


Figure 25: Comparison of errors for solving 3-D Poisson equation (a) and convection-diffusion equation (b) using classical FEM, two variants of AES-FEM, and GFD.

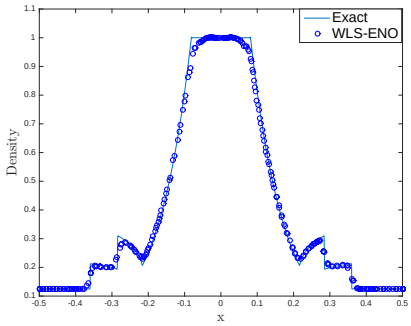
linear FEM, and they are also more efficient than FEM in terms of error versus runtime. In addition, they enable much better stability and faster convergence of iterative solvers than linear FEM over poor-quality meshes.

8.2 WLS-ENO Schemes for Hyperbolic Conservation Laws

The GFD and AES-FEM methods can effectively solve PDEs with smooth solutions. For hyperbolic problems, the solutions are only piecewise smooth, and robust techniques are needed to resolve discontinuities in the solution. ENO (Essentially Non-Oscillatory) and WENO (Weighted Essentially Non-Oscillatory) schemes are widely used techniques for suppressing oscillations for hyperbolic conservation laws. For structured meshes, these techniques can achieve high order accuracy for smooth functions while being non-oscillatory near discontinuities. For unstructured meshes, which are needed for complex geometries, similar schemes are required, but they are much more challenging to derive.

We have recently developed a new family of non-oscillatory schemes, called *WLS-ENO*, in the context of solving hyperbolic conservation laws using finite-volume methods over unstructured meshes [33]. WLS-ENO is derived based on Taylor series expansion and solved using a weighted least squares formulation. Unlike other non-oscillatory schemes, WLS-ENO does not require constructing sub-stencils, and hence it provides a more flexible framework and is less sensitive to mesh quality. We have developed rigorous analysis of the accuracy and stability of WLS-ENO, and conducted

(a) Cross section in xy plane.



(b) Solution along x axis vs. exact solution.

Figure 26: Numerical solution of 3-D explosion test by third-order WLS-ENO at $t = 0.1$.

numerical assessment in 1-D, 2-D, and 3-D for a number of benchmark problems, including wave equations, Burger's equation, and the Euler equations. Figure 26(a) shows an xy cross section of the numerical solution of a 3-D spherical explosion test of the third-order WLS-ENO at $t = 0.1$ [33]. Figure 26(b) shows the same numerical solution along the x axis, which agrees well with the exact solution [34, 35].

9 Summary and Conclusions

A numerical method for brittle fracture in a quasi-static approximation based on a nonlinear finite element approach has been developed, implemented, and tested. The method incorporates ideas from spring network models into the finite element framework. Geometric considerations, such as collision detection and the boundary conditions required to resolve them, are also discussed in depth. The fracture mechanism behaves independently of the underlying finite element function spaces used. The model has been applied to simulations of the formation and propagation of cracks in brittle materials, and the fracture and fragmentation of stretched and compressed materials. Future investigations will concentrate on extending the algorithm to dynamic simulations with the resolution of temporal and spatial scales. As a preliminary step in this direction, we have developed a dynamic theory of brittle fracture based on previous theoretical works. The report also describes our progress with the theory of turbulence.

References

[1] Jochen Alberty, Carsten Carstensen, Stefan A Funken, and Roland Klose. Matlab implementation of the finite element method in elasticity. *Computing*, 69(3):239–263, 2002.

[2] K.-J. Bathe, E. Ramm, and E. Wilson. Finite element formulations for large deformation dynamic analysis. *International Jounral for Numerical Methods in Engineering*, 9:353–386, 1975.

[3] Peter Helnwein. Some remarks on the compressed matrix representation of symmetric second-order and fourth-order tensors. *Computer Methods in Applied Mechanics and Engineering*, 190(22):2753 – 2770, 2001.

[4] Hongren Wei. *Mesoscale Models and Numerical Algorithms for Fractures in Solids*. PhD thesis, State University of New York at Stony Brook, 2012.

[5] J. Alberty, C. Cartensen, A. Funken, and R. Klose. Matlab implementation of the finite element method in elasticity. *Computing*, 69:239–263, 2002.

[6] Ettore Barbieri and Michele Meo. A meshless cohesive segments method for crack initialization and propagation in composites. *Advanced Composite Materials*, 18(1):45–63, 2011.

[7] Grigory Isaakovich Barenblatt. The mathematical theory of equilibrium cracks in brittle fracture. *Advances in Applied Mechanics*, 7(1):55–129, 1962.

[8] Paul D Beale and David J Srolovitz. Elastic fracture in random materials. *Physical Review B*, 37(10):5500, 1988.

[9] Godofredo T Comacho and Ortiz M. Computational modeling of impact damage in brittle materials. *International Journal of Solids and Structures*, 33(20):2899–2938, 1996.

[10] Christophe Daux, Nicolas Moës, John Dolbow, Natarajan Sukumar, and Ted Belytschko. Abritrary branched and intersecting cracks with the extended finite element method. *Int. J. Numer. Meth. Engrg.*, 48:1741–1760, 2000.

[11] René De Borst. Numerical aspects of cohesive-zone models. *Engineering Fracture Mechanics*, 70(14):1743–1757, 2003.

[12] Thomas-Peter Fries and Ted Belytschko. The extended/generalized finite element method: An overview of the method and its applications. *Int. J. Numer. Meth. Engrg.*, 84:253–304, 2010.

[13] Paul Meakin, G Li, LM Sander, E Louis, and F Guinea. A simple two-dimensional model for crack propagation. *Journal of Physics A: Mathematics and General*, 22(9):1393, 1989.

[14] Nicolas Moës and Ted Belytschko. Extended finite element method for cohesive crack growth. *Engineering Fracture Mechanics*, 69:813–833, 2002.

[15] JP Pereira, C Armando Duarte, and Xiangmin Jiao. Three-dimensional crack growth with hp-generalized finite element and face-offsetting methods. *Computational Mechanics*, 46(3):431–453, 2010.

[16] K. Salari and P. Knupp. Code verification by the method of manufactured solutions. Technical Report No. SAND2000-1444, Sandia National Labs., Albuquerque, NM (USA); Sandia National Labs., Livermore, CA(USA), 2000.

[17] J. R. Shewchuck. Triangle: Engineering a 2D quality mesh generator and delaunay triangulator. In *Applied Computational Geometry: Towards Geometric Engineering*, volume 1148, pages 203–222. Springer-Verlag, Berlin, 1996.

[18] Jonathan Richard Shewchuk. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In Ming C. Lin and Dinesh Manocha, editors, *Applied Computation Geometry: Towards Geometric Engineering*, volume 1148 of *Lecture Notes in Computer Science*, pages 203–222. Springer-Verlag, May 1996. From the First ACM Workshop on Applied Computational Geometry.

[19] Hang Si. Tetgen, a delaunay-based quality tetrahedral mesh generator. *ACM Trans. Math. Softw.*, 41(2):11:1–11:36, February 2015.

[20] Clemens V Verhoosel, Michael A Scott, René de Borst, and Thomas JR Hughes. An isogeometric approach to cohesive zone modelling. *Int. J. Numer. Meth. Engrg.*, 87(1-5):336–350, 2011.

[21] P. Wriggers. Finite element algorithms for contact problems. *Archives of Computational Methods in Engineering*, 2,4:1–49, 1995.

[22] P. Wriggers. *Computational Contact Mechanics*, chapter 6-9, pages 109–182. Springer-Verlag, Berlin, 2006.

[23] X-P Xu and Alan Needleman. Numerical simulations of fast crack growth in brittle solids. *Journal of the Mechanics and Physics of Solids*, 42(9):1397–1434, 1994.

[24] A. N. Kolmogorov, A refinement of previous hypotheses concerning the local structure of turbulence in a viscous incompressible fluid at high Reynolds number, *J. Fluid Mechanics*, 13 (1962), 82 – 85.

[25] A. M. Obukhov, Some specific features of atmospheric tubulence, *J. Fluid Mechanics*, 13 (1962), 77 – 81.

[26] J. Glimm and V. Mahadeo, The Spatial Statistics of Turbulent Fluctuations, 2016.

[27] J. Glimm and V. Mahadeo, Random Field Models Stochastic Equations for Turbulent Intensity, 2016.

[28] J. Glimm and V. Mahadeo, Stochastic Subgrid Models for Particle Clustering in Turbulent Flow, 2016.

[29] V. Mahadeo, A Random field subgrid scale model for particle laden turbulent flows, *Ph.D. Thesis, Stony Brook University*, 2016.

[30] R. Feng, Formation and propagation of failure in shocked glasses, *Journal of Applied Physics*, 87(4), 1693 – 1700, 2000.

[31] R. Conley, T. J. Delaney, and X. Jiao. Overcoming element quality dependence of finite elements with adaptive extended stencil fem (aesfem). *Int. J. Numer. Meth. Engrg.*, 2016. to appear.

[32] J. J. Benito, F. Ureña, and L. Gavete. The generalized finite difference method. In M. P. Álvarez, editor, *Leading-Edge Applied Mathematical Modeling Research*, chapter 7. Nova Science Publishers, Inc., 2008.

[33] H. Liu and X. Jiao. WLS-ENO: Weighted-least-squares based essentially non-oscillatory schemes for finite volume methods on unstructured meshes. *J. Comput. Phys.*, 2016. to appear.

[34] M. Lahooti and A. Pishevar. A new fourth order central WENO method for 3D hyperbolic conservation laws. *Applied Mathematics and Computation*, 218(20):10258–10270, 2012.

[35] M. Dumbser, O. Zanotti, A. Hidalgo, and D. S. Balsara. ADER-WENO finite volume schemes with space–time adaptive mesh refinement. *J. Comput. Phys.*, 248:257–286, 2013.