THRUST AREA REPORT • UCRL-ID-129202

Computational Mechanics

Peter J. Raboin,
Thrust Area Leader

This is an informal report intended primarily for internal or limited external distribution. The opinions and conclusions stated are those of the author and may or may not be those of the Laboratory.

Work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.

January 1998

Disclaimer

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial products, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

This report has been reproduced
directly from the best available copy.

Available to DOE and DOE contractors from the
Office of Scientific and Technical Information
P.O. Box 62, Oak Ridge, TN 37831
Prices available from (615) 576-8401, FTS 626-8401

Available to the public from the
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Rd.,
Springfield, VA 22161

THRUST AREA REPORT • UCRL-ID-129202

Computational Mechanics

Peter J. Raboin,
Thrust Area Leader

Reprinted from Engineering Research, Development and Technology
FY 97 UCRL 53868-97

January 1998

Peter J. Raboin, Thrust Area Leader

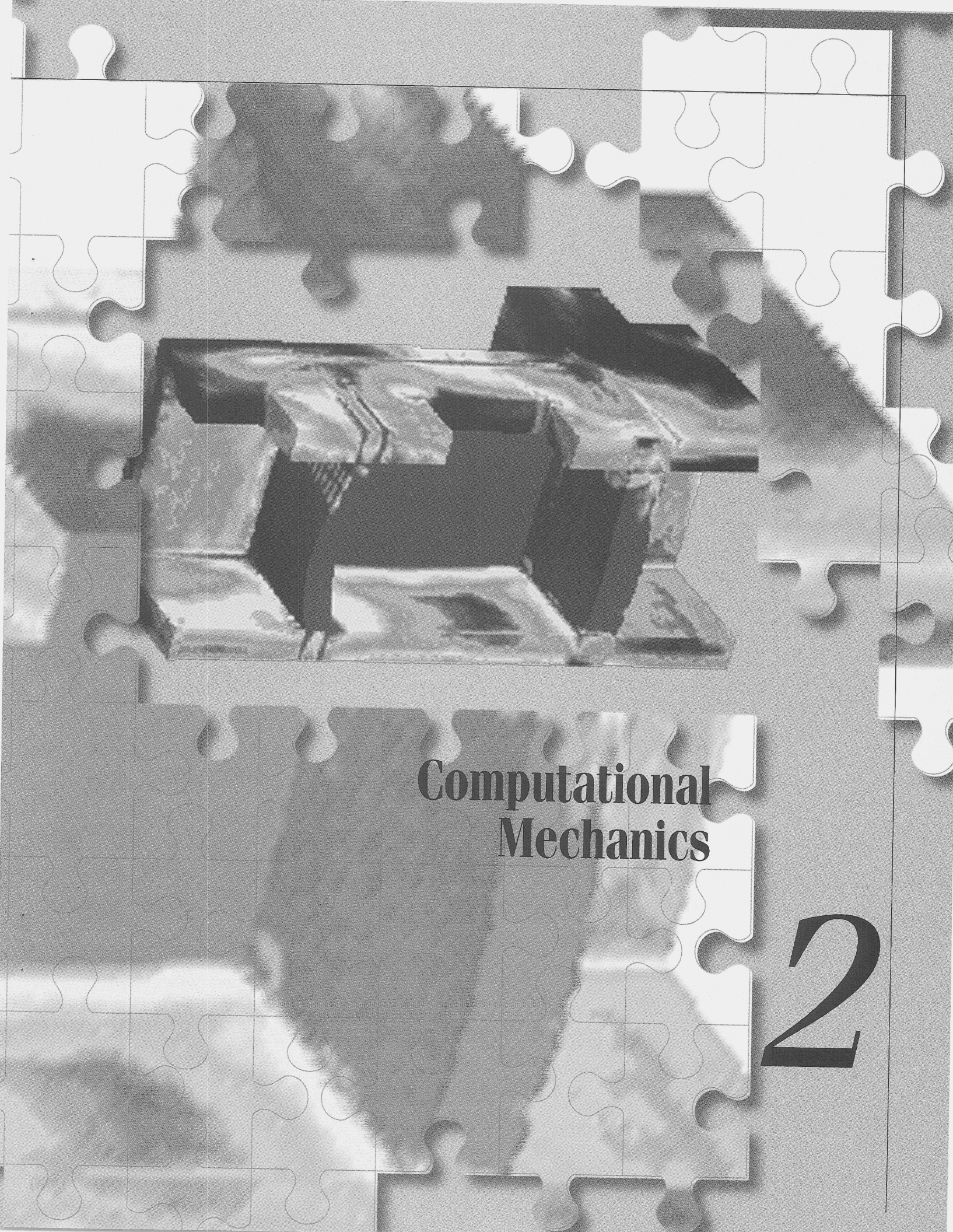
The Computational Mechanics thrust area is a vital and growing facet of the Mechanical Engineering Department at Lawrence Livermore National Laboratory (LLNL). This work supports the development of computational analysis tools in the areas of structural mechanics and heat transfer. Over 75 analysts depend on thrust area-supported software running on a variety of computing platforms to meet the demands of LLNL programs.

Interactions with the Department of Defense (DoD) High Performance Computing and Modernization Program and the Defense Special Weapons Agency are of special importance as they support our ParaDyn project in its development of new parallel capabilities for DYNA3D. Working with DoD customers has been invaluable to driving this technology in directions mutually beneficial to the Department of Energy.

Other projects associated with the Computational Mechanics thrust area include work with the Partnership for a New Generation Vehicle (PNGV) for "Springback Predictability" and with the Federal Aviation Administration (FAA) for the "Development of Methodologies for Evaluating Containment and Mitigation of Uncontained Engine Debris."

Spurred on by the increasing demand for computational analyses and process simulation tools, code development activities increased this past year as parallel DYNA3D entered into the "production" environment for the first time. With access to ASCI MPP computers, ever larger and more complex simulations were completed. Analysis times have been dramatically reduced as parallel processing on the IBM SP2 has scaled up the compute speed of DYNA3D by as much as a factor of 27 over the Cray YMP. Typical simulations within the last year included weapon transport safety, earthquake analyses of bridges, buildings and laser-support systems, knee-injury modeling, blast effects calculations, automobile crashworthiness, underground shock simulations and metal forming manufacturing.

In this report for FY-97, there are five articles, detailing three code development activities and two projects that synthesized new code capabilities with new analytic research in damage/failure and biomechanics. The articles this year are 1) Energy- and Momentum-Conserving Rigid-Body Contact for NIKE3D and DYNA3D; 2) Computational Modeling of Prosthetics: A New Approach to Implant Design; 3) Characterization of Laser-Induced Mechanical Failure Damage of Optical Components; 4) Parallel Algorithm Research for Solid Mechanics Applications Using Finite Element Analysis; and 5) An Accurate One-Step Elasto-Plasticity Algorithm for Shell Elements in DYNA3D.



Computational Mechanics

2

2. Computational Mechanics

Overview
Peter J. Raboin, Thrust Area Leader

Energy- and Momentum-Conserving Rigid-Body Contact for NIKE3D and DYNA3D
Michael A. Puso and Edward Zywicz 2-1

Computational Modeling of Prosthetics: A New Approach to Implant Design
Karin Hollerbach, Scott A. Perfect, and Harry E. Martz 2-5

Characterization of Laser-Induced Mechanical Damage of Optical Components
Douglas R. Faux 2-11

Parallel Algorithm Research for Solid Mechanics Applications Using Finite Element Analysis
Carol G. Hoover, Daniel C. Badders, Anthony J. De Groot, and Robert J. Sherwood 2-17

An Accurate One-Step Elasto-Plasticity Algorithm for Shell Elements in DYNA3D
Jerry I. Lin 2-25

Energy- and Momentum-Conserving Rigid-Body Contact for NIKE3D and DYNA3D

Michael A. Puso
Defense Technologies Engineering Division
Mechanical Engineering

Edward Zywicz
New Technologies Engineering Division
Mechanical Engineering

During long-time simulations of rigid body dynamics, especially with contact, conservation of momentum and energy is directly related to time-step size for both implicit and explicit formulations. For example, numerical simulation of a rigid ball bouncing inside a rigid box will always eventually go unstable if the time step is too large, or come to a stop if the time step is appropriately small. Consequently, we have implemented algorithms for rigid body dynamics with contact that can exactly conserve linear momentum, angular momentum, and energy, independent of the time step. The algorithms are inherently implicit and have been implemented in NIKE3D and DYNA3D. In addition, rigid-body smoothing has been incorporated into NIKE3D and DYNA3D for spherical rigid-body contact.

Introduction

The rigid material idealization is often exploited in finite element analyses to save computational time, since rigid materials do not require constitutive evaluations and vastly reduce the number of degrees of freedom in the problem. Yet, the algorithms for integrating rigid-body equations of motion can be somewhat more complicated than those for deformable solid elements, due to the presence of angular momentum. Furthermore, in an explicit setting, no critical time step naturally arises to maintain stability, unlike in the case of deformable elements.

In the past, with DYNA3D the user would have to guess at a particular time step for problems with rigid bodies. In NIKE3D, the rigid-body dynamics algorithms were not unconditionally stable. The inclusion of contact in these rigid-body problems would exacerbate the time step problem since contact penetrations have a tendency to drive energy (algorithmically) into the system.

Even with a stable time-step size, contact problems will dissipate energy (algorithmically) and cause free-body contact problems to eventually lose all momentum and energy. To solve these problems, an exact energy- and momentum-conserving algorithm has been developed and implemented into DYNA3D and NIKE3D for rigid-body dynamics. This algorithm can be shown to be unconditionally stable in the energy sense.[1]

In addition to the new conserving integration algorithm, a new method has been implemented to represent rigid spherical components in NIKE3D and DYNA3D. It allows spherical surfaces, defined by rigid materials, to roll or slide smoothly instead of "bumping," as is usually encountered with discrete faceted surfaces and traditional contact algorithms. The energy-conserving algorithms along with spherical smoothing can be used for understanding rolling problems, such as ball bearings.

Progress

A rigid-body dynamics algorithm,[2] has been implemented into NIKE3D and DYNA3D. This required the modification of the time integration schemes already used.[3,4] This integration method was extended in our work to include the effects of contact resulting in an exact energy- and momentum-conserving algorithm for rigid-body dynamics with contact. In fact, two different formulations have been developed.

The method implemented into NIKE3D is a penalty-based formulation which is similar to the current contact algorithms, in that the contact force

is proportional to the gap distance. This method enforces the so called "consistency condition," in that it mitigates the gap distance between the slave node and master surface. By enforcing the consistency condition, relatively large time steps can be used. Consequently, this algorithm is well suited for NIKE3D.

Since conservation algorithms use the midpoint rule, NIKE3D was modified for the generalized Hilber-Hughes-Taylor integration algorithm (also known as the *alpha* method). This method is a generalization of NIKE3D's Newmark integration rule. The *alpha* method can perform mid-point integration for energy conservation, or apply algorithmic damping by appropriate specification of integration parameters.

The method implemented into DYNA3D is a Lagrange multiplier method which uses a predictor step to anticipate contact, and a corrector step to apply forces that enforce the appropriate gap velocity. Because only the gap velocity is considered, the velocity constraint should be activated at or near the point of incipient contact. This requires small time steps and is well suited for DYNA3D.

The new contact algorithms were also modified for the rigid spherical contact idealization. This will allow them to capture the effect of smooth rolling of spherical bodies. Currently, only the spherical contact algorithm is available in the energy- and momentum-conserving form for NIKE3D. DYNA3D has the new algorithms implemented for both general and spherical contact, and also includes the effects of friction.[5]

Formulation

The basic theory[2] was generalized to address contact. This required the appropriate definitions and time integration of contact forces. First, the notions of discretized energy and momentum conservation are defined, then the individual algorithms for NIKE3D and DYNA3D that satisfy these conservation principles are presented.

Consider the free-body motions (that is, no applied forces) of N rigid bodies with contact. Using the mid-point rule, the discretized equations of motion, in momentum form, for the i-th body are given by:

$$\boldsymbol{p}^i_{n+1} - \boldsymbol{p}^i_n = \boldsymbol{f}^i_{n+1/2}\Delta t \quad \boldsymbol{h}^i_{n+1} - \boldsymbol{h}^i_n = \overline{\boldsymbol{R}}^i \times \boldsymbol{f}^i_{n+1/2}\Delta t \quad (1)$$

where $\boldsymbol{p}^i_n = M^i \boldsymbol{v}^i_n$ is the linear momentum at time t_n; $\boldsymbol{h}^i_n = \boldsymbol{I}^i_n \cdot \omega^i_n$ is the angular momentum; v is the velocity of the center of mass; ω is the rotation rate; $\boldsymbol{f}$ is the contact force; and $\overline{\boldsymbol{R}}$ is the appropriately defined moment arm, measured from the center of mass. Conservation of the discretized system linear and angular momentum is represented by:

$$\sum_{i=1}^{N} \boldsymbol{p}^i_n = \boldsymbol{P}, \qquad \sum_{i=1}^{N} \boldsymbol{X}^i_n \times \boldsymbol{p}^i_n + \boldsymbol{h}^i_n = \boldsymbol{H} \qquad (2)$$

where $\boldsymbol{P}$ and $\boldsymbol{H}$ are constants, and $\boldsymbol{X}$ is the location of the center of mass. Energy conservation is given by:

$$T_n + U_n = E,$$

$$\text{where} \quad T_n = \frac{1}{2}\sum_{i=1}^{N} \boldsymbol{p}^i_n \cdot \boldsymbol{v}^i_n + \boldsymbol{h}^i_n \cdot \omega^i_n \quad . \qquad (3)$$

The constant E is the total system energy, and U is some stored potential energy (U is non-zero when contact is made in the penalty method, but is always zero in the Lagrange multiplier method).

In general, time-stepping methods satisfy **Eq. 1**, but **Eqs. 2** and **3** are only satisfied in the limit as $\Delta t \to 0$. By using "algorithmic" definitions for $\boldsymbol{f}$ and $\overline{\boldsymbol{R}}$ and the integration rules given in Reference 2, the conservation laws, **Eqs. 2** and **3**, can be satisfied to some tolerance, yielding an unconditionally stable algorithm.

For the penalty method, an algorithmic definition of the gap is used in place of the exact gap. The algorithmic gap is given by the recursion:[6]

$$g^d_{n+1} = g^d_n + \overline{\boldsymbol{v}} \cdot \left(\left(\boldsymbol{x}^s_{n+1} - \boldsymbol{x}^s_n \right) - \left(\boldsymbol{x}^m_{n+1} - \boldsymbol{x}^m_n \right) \right) \qquad (4)$$

where $\overline{\boldsymbol{v}}$ is the contact normal; $\boldsymbol{x}^s$ represents the position of the slave node; and $\boldsymbol{x}^m$ is the closest point projection of the slave node onto the master surface. This "dynamic" gap is a second order approximation to the actual gap. The contact force is given by

$$\boldsymbol{f} = p\overline{\boldsymbol{v}} \quad ,$$

$$\text{where} \quad p = \frac{1}{2}\frac{U_{n+1} - U_n}{g^d_{n+1} - g^d_n}$$

$$\text{and} \quad U = \begin{cases} k\left(g^d\right)^2 & \text{if } g < 0 \\ 0 & \text{otherwise} \end{cases} \quad . \qquad (5)$$

The nodal points on both the slave and master surfaces of the rigid bodies are calculated at a modified position given by:

$$\overline{\boldsymbol{x}} = \frac{1}{2}\left(\boldsymbol{X}_{n+1} + \boldsymbol{X}_n\right) + \frac{1}{2}\frac{\tan\left(|\theta| \ /2\right)}{|\theta| \ /2}\left(\boldsymbol{R}_{n+1} + \boldsymbol{R}_n\right) \qquad (6)$$

where $\boldsymbol{X}$ is the position of the center of mass; $\boldsymbol{R}$ is

the position to the node from the center of mass; and θ is the compound rotation from the n to n+1 position. From the nodal positions given by **Eq. 6**, the contact normal, $\bar{\boldsymbol{v}}$, and the moment arm, $\bar{\boldsymbol{R}}$, are calculated.

In the Lagrange multiplier method, an explicit "predictor" step is taken, in which no contact forces are applied. From the predictor step nodal positions, penetrated nodes are identified, and moment arms, $\boldsymbol{R}_p$, and contact normals, $\bar{\boldsymbol{v}}$, are calculated. The contact force is now:

$$\boldsymbol{f} = \lambda \bar{\boldsymbol{v}} \tag{7}$$

where λ is the unknown Lagrange multiplier. Using the predicted normals, the following algorithmic moment arms for respective slave and master nodes are used to conserve angular momentum:

$$\bar{\boldsymbol{R}}^s = \boldsymbol{R}_p^s + \left(\left(\boldsymbol{X}_n^s - \boldsymbol{X}_n^m\right) + \left(\boldsymbol{R}_p^s - \boldsymbol{R}_p^m\right)\right) /2$$

$$\bar{\boldsymbol{R}}^m = \boldsymbol{R}_p^m - \left(\left(\boldsymbol{X}_n^s - \boldsymbol{X}_n^m\right) + \left(\boldsymbol{R}_p^s - \boldsymbol{R}_p^m\right)\right) /2. \tag{8}$$

The Lagrange multiplier is calculated such that the following gap velocity constraint is enforced:

$$\bar{\boldsymbol{n}} \cdot \left(\Delta \boldsymbol{v}^m - \Delta \boldsymbol{v}^s\right) = 0 \tag{9}$$

where $\Delta \boldsymbol{v}$ is the algorithmic velocity jump of the slave/master node from time n to n + 1, and is given by:

$$\Delta \boldsymbol{v} = \boldsymbol{V}_{n+1/2} + \boldsymbol{V}_{n-1/2} +$$

$$\left(\exp(\theta)\omega_{n+1/2} + \omega_{n-1/2}\right) \times \bar{R}. \tag{10}$$

Here $\boldsymbol{V}$ is the velocity of the center of mass, and the matrix exp is the exponential map.[2] In the limit as $\Delta t \rightarrow 0$, the rotation, exp (θ), tends toward the identity matrix, and the continuum result is recovered for an elastic collision.[7]

The above algorithms are particularized to spherical contact by appropriately modifying the contact gap and normal calculations for the spherical surfaces.

Examples

In the first example, NIKE3D is used with rigid spherical smoothing. A 0.274-kg/m^3 solid ball is placed within a 0.743-kg/m^3 spherical shell **(Fig. 1)** and given an initial velocity of 1 m/s in the y direction, such that the inner ball rolls smoothly in the y-z plane. Because the system is entirely free, linear and angular momentum of the system is conserved (see **Fig. 2**). In fact, the energy and angular momentum are conserved within machine precision even after 2000 iterations.

In the second example, the Lagrange multiplier method in DYNA3D is used to model two slender (50 m × 1 m × 1 m) rigid bars impacting. The initial x velocity of the left bar is 1 m/s, and the right bar is at rest. **Figure 3** shows the rigid bars

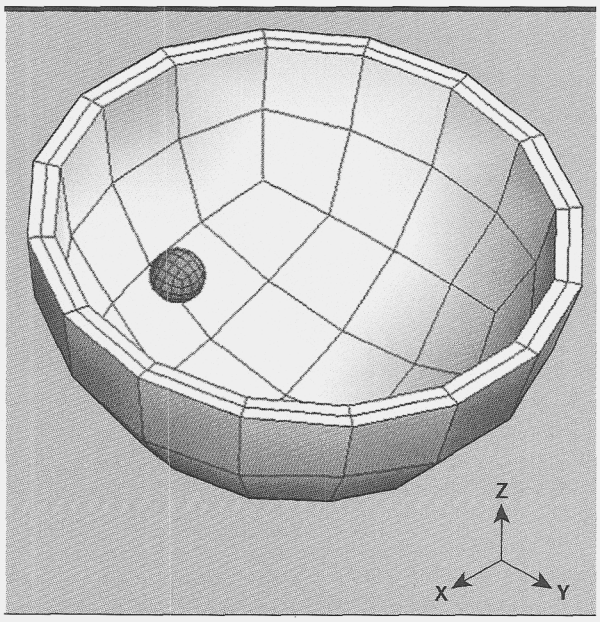


Figure 1. Ball inside spherical shell. Only the bottom half of the shell is shown.

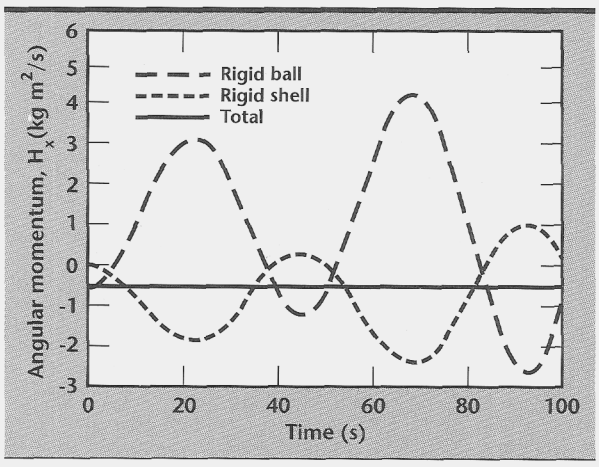


Figure 2. Angular momentum in the x direction for rigid ball and rigid shell.

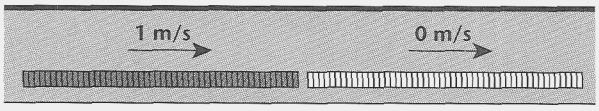


Figure 3. Initial conditions for impacting rigid rods.

separated by 0.01 m. **Figure 4** shows the displacement vs time of the rigid bars. Unlike the penalty method, the Lagrange multiplier method eliminates penetration, and is generally superior to the penalty method.

Future Work

It remains to implement the energy-conserving contact algorithms into the general contact algorithms for NIKE3D (rigid and deformable). Continued testing and work is needed for both the DYNA3D and NIKE3D implementations to increase robustness.

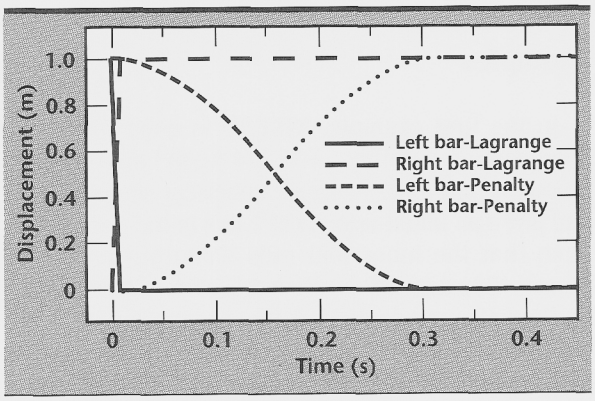


Figure 4. Displacement of impacting rigid rods.

References

1. Richtmyer, R. D., and K. W. Morton (1967), "Difference Methods for Initial Value Problems, Second Edition," *Interscience*, New York, NY.

2. Simo, J. C., and K. K. Wong (1991), "Unconditionally Stable Algorithms for Rigid Body Dynamics That Exactly Preserve Energy and Momentum," *Int. J. Num. Meth. Eng.*, **31**.

3. Puso, M. A. (1996), "Implicit Rigid Body Static and Dynamic Analysis with NIKE3D," *Thrust Area Report*, University of California, Lawrence Livermore National Laboratory, Livermore, CA, (UCRL-ID-125471).

4. Benson, D. J., and J. O. Hallquist (1986), "A Simple Rigid Body Algorithm for Structural Dynamics Programs," *Int. J. Num. Meth. Eng.*, **22**.

5. Zywicz, E., and M. A. Puso (1997), *A General Predictor-Corrector Solver for Explicit Finite-Element Contact*, University of California, Lawrence Livermore National Laboratory, Livermore, CA, (UCRL-JC-128410).

6. Armero, F., and E. Petocz (1997), *Formulation and Analysis of Conserving Algorithms for Dynamic Contact Impact Problems*, University of California, Berkeley, CA, (UCB/SEMM-96/10).

7. Kane, T. R., and D. A. Levinson (1985), *Dynamics: Theory and Applications*, McGraw Hill, San Francisco, CA.

Computational Modeling of Prosthetics: A New Approach to Implant Design

Karin Hollerbach
Institute for Scientific Computing Research
Center for Applied Scientific Computing
Computations

Scott A. Perfect
New Technologies Engineering Division
Mechanical Engineering

Harry E. Martz
Manufacturing and Materials Engineering Division
Mechanical Engineering

We are applying finite element techniques to computer-aided design (CAD) models of prosthetic implants, to 1) calculate implant kinematics and regions of high stress within certain implant materials; and 2) make design recommendations to improve performance and reduce the likelihood of failure. In the process of working with industry to achieve these goals, we have developed finite element models of several human joints, as well as prosthetic joint implants (or joint replacements).

Introduction

We are applying finite element analysis to the design and evaluation of joint replacement prostheses. Total joint replacement has become a major industry and will expand dramatically as our population ages and our life expectancy increases. The present joint replacement market is served by an inadequate product, frequently requiring revision surgeries. Today's implants may fail to reproduce normal joint kinematics, or they may exhibit material failure at the articular interface or at the bone-implant interface. Implant failure frequently results from a lack of understanding of human joint biomechanics and an inability to quantitatively assess the effects of implant geometry and material modifications on the biomechanics of the implanted joint.

Finite element modeling of joints with implants offers an improved approach to implant design and an opportunity to evaluate designs before they are manufactured or surgically implanted. This is an attractive alternative to current methods that use only expensive, time-consuming experimental testing, where retrieved implants serve as a post-failure demonstration of poor design strategies.

Non-linear 3-D finite element modeling and analysis are powerful tools for the evaluation and design of joint replacement prostheses. We are using this technology to model several different implant designs and to perform non-linear, 3-D, large-displacement analysis that replicates loads that would be experienced by the prosthetic implant *in vivo*.

Material failure is common in implant designs containing polyethylene components. Wear and fatigue are correlated with the pattern and magnitude of stresses experienced by the material. Orthopedic implant materials, especially polyethylene, have non-linear stress/strain relationships. Polyethylene wear occurs primarily in sub-surface regions, which cannot be adequately modeled in 2-D analyses. In addition, implants are 3-D and experience forces in all dimensions in the patient. Fully non-linear 3-D, modeling tools can produce more accurate results than the linear 2-D, analyses.

Progress

In our implant analyses, we are applying finite element techniques to computer-aided design (CAD) models of prosthetic implants, to 1) calculate

implant kinematics and regions of high stress within certain implant materials; and 2) make design recommendations to improve performance and reduce the likelihood of failure. In the process of working with industry to achieve these goals, we have developed finite element models of several human joints, as well as prosthetic joint implants (or joint replacements).[1]

The analysis code used in our research is Lawrence Livermore National Laboratory's (LLNL) NIKE3D, a non-linear, implicit, 3-D finite element code developed for studying dynamic, finite deformations. Articular contact algorithms in NIKE3D are based on a master/slave approach that provides general multi-body contact capability, including sliding with contact, gaps, and friction, as needed. Contact algorithms, a key feature in orthopedic modeling where the primary areas of interest are near the contacting articular surfaces and near the bone-implant interface, use the penalty method, in which a penalty is generated when interpenetration between contacting surfaces is calculated.

In analyzing the prosthetic implants, we have created finite element models of three thumb carpometacarpal joint prostheses and one knee prosthesis. From Initial Graphics Exchange Specification (IGES) we obtained format descriptions generated by the orthopedic industry using CAD tools. Using TrueGrid (XYZ Scientific Applications, Inc.), we meshed the 3-D surfaces of the implants into a set of eight-node solid (hexahedral) elements that closely approximated the surface descriptions. Mesh generation was performed manually, to allow for fine control of mesh quality, especially near contacting surfaces.

Data available in the orthopedic literature provided us with properties for the implant materials in the prosthetic models. The preliminary material model used for the polyethylene pieces was an elastic-plastic model. Numerical data obtained from Bartel[2] and De Heer[3] are shown in **Table 1**.

The thumb implant designs that were modeled include 1) a ball-and-socket joint with a polyethylene socket and a CoCr ball and stem; 2) a ball-and-socket joint with a Ti-backed polyethylene socket and a CoCr ball and stem; 3) an asymmetric saddle-shaped joint, in which the metacarpal component consists of a Ti stem with a polyethylene articular surface and a CoCr trapezial component; and 4) a similar but symmetric saddle-shaped joint.

Boundary conditions, defining net carpometacarpal joint forces for a specified set of commonly used grasps, were calculated based on simulations derived from a virtual five-link model of the thumb.[4]

Carpo-metacarpal joint reaction forces predicted by the five-link model are, for an applied load of 10 N: 188.5 N for key pinch, 241.1 N for screwdriver grasp, and 61.5 N for tip pinch. The joint reaction forces in pinch and grip are not perpendicular to the metacarpal, but are directed dorsally and radially (**Fig. 1**).

Non-linear 3-D finite element analyses of these models show articular surface and sub-surface stress patterns when the implants are loaded with forces representative of common grasping modes, such as key pinch, tip pinch, and screwdriver grasps. These loads are static and eccentric to the joint components. Metal-backed and unbacked polyethylene socket models varied in stress patterns and levels of maximum stress. The implant stress analyses appear to correlate with material failures observed clinically in the ball-and-socket designs. Both ball-and-socket designs were also compared to the newer design with an asymmetric saddle-shape. With similar eccentric loading, stress patterns varied between implant designs, and maximal stresses were lower in the asymmetric implant.

Simulation results of the saddle-shaped implants (**Fig. 2**) correlate with clinical findings and help to validate the modeling approach. As a result of its high failure rate, the ball-and-socket implant is now

Table 1. Numerical data for polyethylene material model.

Bartel		De Heer	
Plastic strain	Stress (MPa)	Plastic strain	Stress (MPa)
0.0	14.07	0.0	10.7
0.0108	22.05	0.00925	15.86
0.0363	27.06	0.01797	18.31
0.0910	29.52	0.0502	20.69
		0.0801	22.75
		0.2325	31.15
Elastic modulus	1016 MPa	Elastic modulus	571.6 MPa
Poisson's ratio	0.46	Poisson's ratio	0.45

used less frequently in the thumb. However, a geometrically similar design is commonly used in the hip, which represents one of the largest joint replacement markets.

In addition to the thumb prostheses, we have also analyzed a knee replacement. The objectives of this study, static simulations of knee implant loading, were to assess the effect of knee flexion angle and varus rotation angle on the stresses in an all-polyethylene tibial component, as a function of flexion angle and varus rotation angle. We have completed all simulations for both the cruciate-retaining and posterior-stabilized designs. Some representative stress calculations are shown in **Fig. 3**.

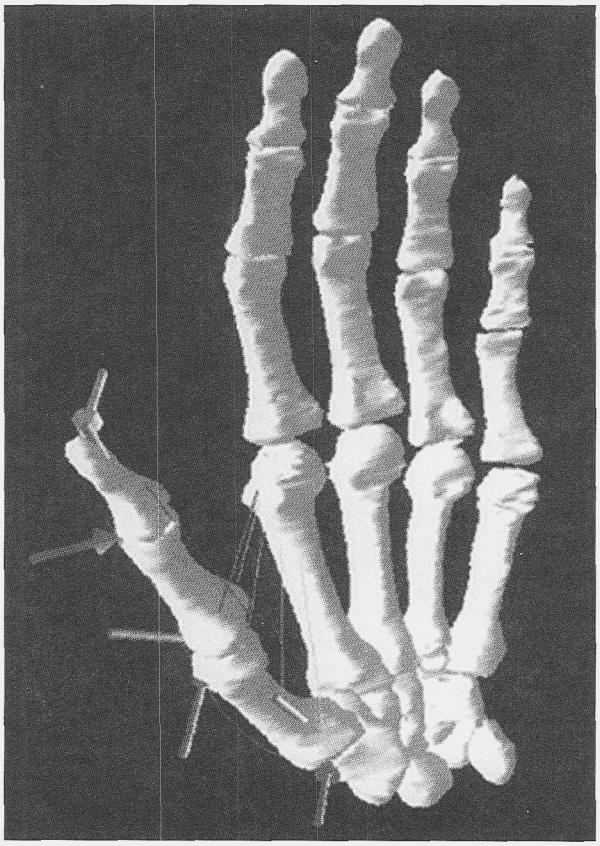


Figure 1. Biomechanics model used to calculate thumb carpo-metacarpal joint reaction forces for a key pinch and other grasps.

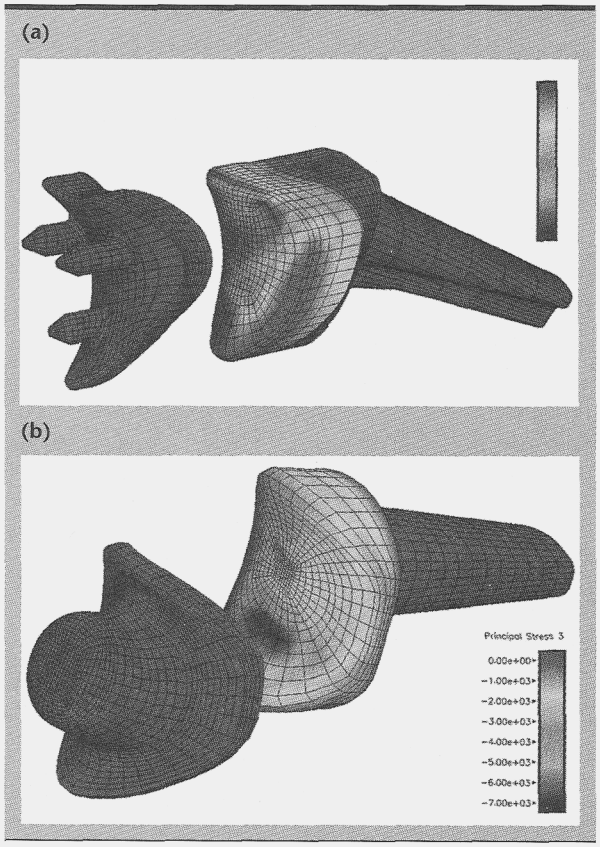


Figure 2. Finite element analyses of thumb carpo-metacarpal joint implants. Loading represents joint reaction forces experienced by the implants in a maximum key pinch grasp for (a) asymmetric saddle design, and (b) symmetric saddle design.

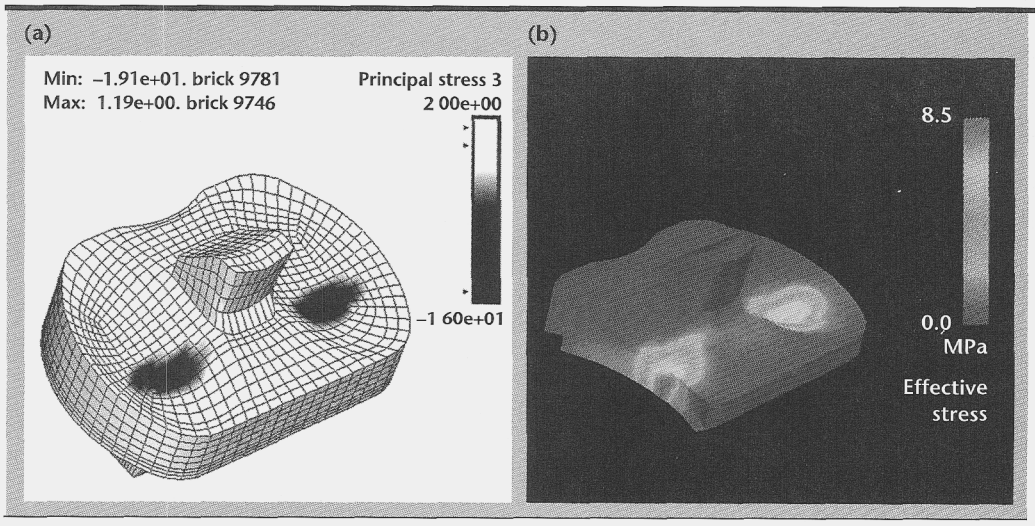


Figure 3. Finite element analysis of knee joint replacement with (a) principal stress at articular surface of polyethylene, and (b) cutaway view showing subsurface stress. Loading represents joint reaction force experienced by the implant in a straight leg stance.

Simulations of prosthetic joint implants without human joint models can provide predictive data on the mechanical performance of the implants. Integration of human joint with prosthetic joint models provides additional data for consideration in implant performance evaluation. With this in mind, we have developed several models of human joints, including a hand model and a knee model. In addition, we have developed a process by which we can acquire human joint data and process the data to create a new human joint model. The steps in this process are described in more detail below.

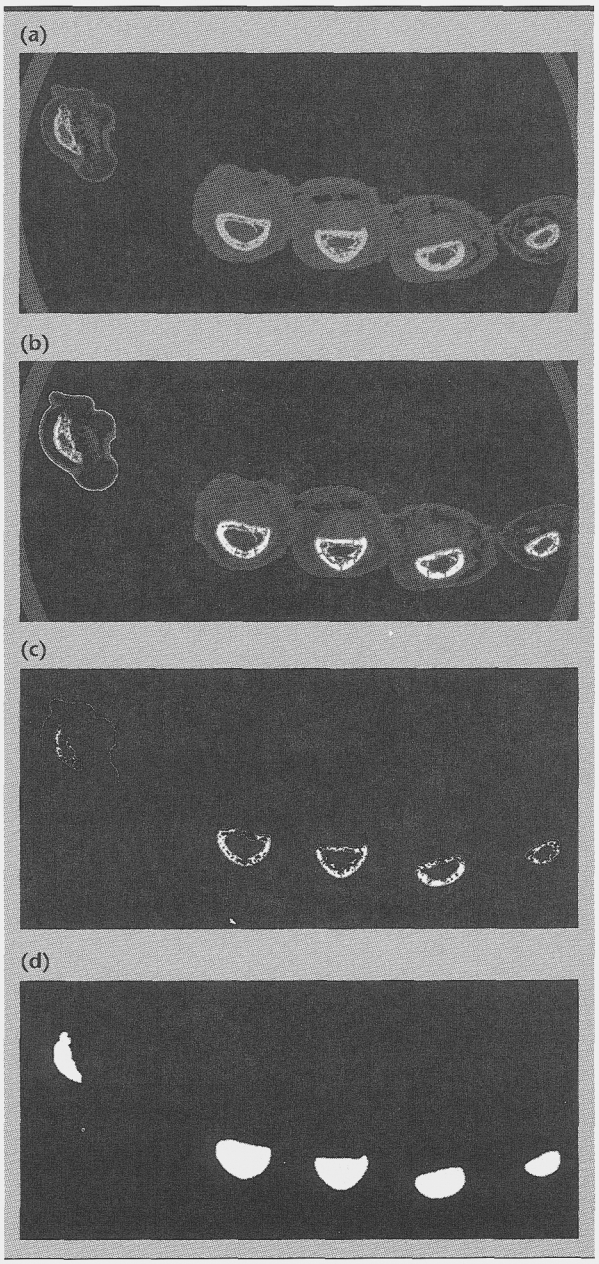


Figure 4. Segmentation of CT data: identification of bone tissues within a high-resolution, 3-D hand scan. A single slice is shown: (a) raw CT data; (b) bone outlines generated by automated edge detection; (c) partial manual correction of outlines; and (d) final bone identification overlaid on original CT data.

Developing a human joint model begins with scanning the joint to acquire geometrical data. Computed tomography (CT) scans are presently the best method to define the surfaces of bony material. CT scans are commonly used in medicine. However, scanners typically used in the medical field have a spatial resolution of up to 1 mm, which is not adequate for a precise definition of articular surfaces. We can achieve a higher resolution at LLNL, where we have scanned two cadaver hands and one knee, using the industrial scanners in the Non-destructive Evaluation group (NDE). Spatial resolutions of these scans have varied from 150 to 190 μm for the hand scans and approximately 280 μm for the knee scan.

While CT scans are good at defining bone surfaces, soft tissue geometries are more easily obtained using other scanning techniques, such as magnetic resonance imaging (MRI).

Collaborators at Massachusetts General Hospital (MGH) and the University of California at Davis (UCD) have provided several MRI datasets of patient knees (at MGH) and cadaver hands (at UCD), which we can use in conjunction with the CT data to help define exact attachment sites for soft tissues.

We have developed semi-automated CT processing software to convert CT images into 3-D surfaces. Segmentation of high resolution CT data is made difficult by the inhomogeneous trabecular structure of bones. Some level of human interaction is needed to fine-tune automatic segmentation results.

We have implemented tools that allow the user to visualize large data sets, manipulate color maps to display false colors, perform interactive thresholding, overlay the segmentation masks, save the corrected results, and create 3-D surfaces from the segmentation masks (**Fig. 4**). These 3-D surfaces are then meshed, and boundary conditions are applied, to set up the finite element simulations in the same way that the prosthetic implant analyses were performed. The loading (boundary) conditions and initial ligament tensions are determined from experiments performed in collaboration with our research partners at Louisiana State University Medical Center and from a model developed in collaboration with the GW Long Hansen's Disease Center.

Sample results from the human joint finite element analyses are shown in **Fig. 5**.

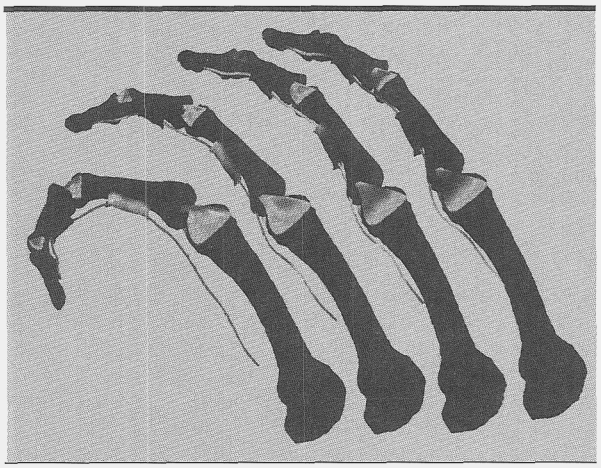


Figure 5. Finite element simulations of human joint articulation: flexion of index finger. Loading represents displacement of the tendon due to muscle contraction.

Future Work

Future work in this research project will be focused primarily on the integration of the human and prosthetic joint models, so that bone-implant interface analyses are possible. We will continue to work on the ball-and-socket and saddle designs for the thumb carpo-metacarpal joint and on the knee design. Most of our technical research progress in the upcoming year will be in developing these integrated models and in calculating the bone stresses resulting from the loads exerted by the implants on the bone.

In addition, model validation will form a key part of our work in the next year. The human knee and carpo-metacarpal joint models will be validated against experimental data. To maintain a manageable scope of work, our efforts are focused on specific aspects of the joint biomechanics that are relatively easy to access experimentally, such as joint kinematics (comparison with experiment and rigid body models), articular surface contact areas and stress patterns (comparison with experiment), and ligament stresses as a function of joint orientation (comparison with experiment).

For validation of implant models, we will compare contact stress calculations with experimentally determined values, where possible, and we will examine clinical data on the failure of one of the carpo-metacarpal joint implants. (Failure followed improper surgical insertion of the implant into the human joint).

Until now, we have worked with three orthopedic companies, (ExacTech, in Florida, ArthroMotion in Indiana, and Wright Medical in Tennessee) and with two orthopedic surgeons (Dr. A. Hollister, Louisiana State University Medical Center and Dr. K. Trauner, UCD). In the future, we expect to continue our research by expanding our interactions with these companies and with additional clinical and industrial partners.

Through clinical and industrial partnerships, we expect that our orthopedic research will become a self-sustaining research program at LLNL. To broaden our research interests and to augment contracts for analysis work with the orthopedic industry, we will also be pursuing external grant funding. Examples of related research application areas include simulations of fracture dislocations using the existing index finger model, population studies addressing ethnic-, gender-, and age-related differences in joint size and shape, and ergonomic simulations for tool design.

Additional future work involves the integration of our technology and experience into LLNL and other Department of Energy programs. For example, through NDE we are currently working with Knolls Atomic Power Laboratory to use our image processing, meshing, and modeling tools to analyze their systems. In this process, we have trained staff at NDE in the use of the segmentation and meshing codes, so that they may be used to perform inspections of CT scanned parts at Knolls. In addition, we will make our modeling results available to Health Services at LLNL and provide input on ergonomic device design teams, based on our hand simulations.

References

1. Hollerbach, K, S. Perfect, and E. Ashby (1997), "Computational Biomechanics: Finite Element Analyses of Human and Prosthetic Joints (video)," Lawrence Livermore National Laboratory, Livermore, CA, for the Department of Energy.

2. Bartel, D. L., J. J. Rawlinson, A. H. Burstein, C. S. Kanawat, and W. F. Flynn, Jr. (1995), "Stresses in polyethylene components of contemporary total knee replacements," *Clin. Orthop. and Related Research,* **317**, pp. 76–82.

3. De Heer, D. C (1992), *Stresses in Polyethylene Tibial Components for Total Knee Arthroplasty.* Master's thesis, Purdue University, School of Mechanical Engineering, West Lafayette, IN.

4. Giurintano, D. G, A. M. Hollister (1995), "Force Analysis of the Thumb for a Five-link System," *Med Eng Phys,* **17**, pp. 297–303.

Characterization of Laser-Induced Mechanical Damage of Optical Components

Douglas R. Faux
Defense Technologies Engineering Division
Mechanical Engineering

The goal of this research is to quantify by numerical techniques the effects of surface and sub-surface defects on damage initiation and growth in high-power laser optical components. The defects include laser absorbing spots (such as surface particulate contamination), and surface damage regions (such as micro-cracks and voids), which are present due to environmental exposure and fabrication processes. Our work here focused on three sources of particle contamination that can cause damage to optical components: front-surface; back-surface; and sub-surface. The DYNA2D code was used to model the growth of damage in the glass substrate.

The damage in the glass substrate as a result of front-surface particle contamination was found to be dependent on the magnitude of the pressure pulse applied to the particle and the initial area of contact between the particle and glass substrate. The pressures generated from a back-surface particle being blown off the surface provided sufficient loading to severely damage (crack) the glass substrate. A sub-surface cerium oxide particle showed a strong surface interaction that influenced the formation and direction of the cracking that ultimately resulted in the blow-out of the damaged material, leaving a relatively clean crater in the glass.

Introduction

The Beamlet spatial filter failure caused significant damage to the laser and a month of downtime. Experimental evidence indicates that particulate contamination on the lens surface or embedded below the surface may have been the precursor to failure.

A failure scenario is the following: a particle on the back surface is irradiated by laser light, creating a hot expanding plasma which then imparts a severe pressure loading on the glass surface, and creates micro-cracks and unacceptable damage. For the National Ignition Facility (NIF) at Lawrence Livermore National Laboratory (LLNL) to set cleanliness and surface finish requirements, we must determine the particle contamination size that results in critical damage morphologies after laser irradiation.

Particulate contamination on optics can act as an obscuration in the beam, or as a local initiation site for laser damage. An on-going experimental program, initiated in FY-96 and funded by NIF, is looking at damage morphologies on optical surfaces contaminated with particles of well-controlled size and composition.

To improve our understanding of damage initiation and growth, a modeling effort, complementing the experimental work, was initiated in FY-96. The modeling group is a multidisciplinary LLNL team with representatives from NIF, Physics, the Laser Program, and Mechanical Engineering.

The Physics staff is responsible for formulating theoretical models of the problem with emphasis on plasma physics, laser/matter interaction, and radiation transport. The Laser Program is using LASNEX to model the laser interaction with surface particulate and plasma formation, and to calculate the magnitude of the surface pressure pulse on the optic. Mechanical Engineering staff are using DYNA2D[1] to investigate the optic response to the surface pressures calculated from LASNEX and to investigate front-surface particulate contamination and sub-surface particulate contamination.

Progress

DYNA2D Brittle Damage Models

Four brittle damage models were incorporated into DYNA2D for modeling dynamic impact events with brittle materials, such as ceramics and glasses.

Three of the brittle damage models implemented use a scalar damage parameter to track cumulative damage, while the fourth incorporates a tensor damage via the explicit tracking of 3-D cracks within an element.

For the three scalar damage models, the constitutive response of undamaged material is treated with either an isotropic-elastic-plastic or a Steinberg-Guinan high-rate elastic-plastic flow rule. Failed material is treated with a piece-wise linear definition of yield stress vs pressure. The transition between undamaged and fully damaged material is accomplished through the evolution of the scalar damage parameter, *D*.

The three scalar damage models differ in their evolution equations for *D*. The shear modulus, yield strength, and pressure are transitioned between the undamaged values and fully-damaged values as D goes from 0 to 1. The three scalar damage models are the modified Tuler-Butcher[2] damage model, the modified Cagnoux-Glenn brittle damage model,[3,4] and the Steinberg-Tipton brittle damage model.[5]

The tensor damage model incorporated into DYNA2D to calculate the brittle damage response of glass is a modified Rubin-Attia 3-D crack damage model.[6] Fracture is modeled in a continuum sense by introducing a symmetric tensor, e^v, which denotes void strain. The void strain is determined by restricting the traction vector that is applied on a fracture surface.

The model allows the formation of up to three orthogonal fracture surfaces, which are normal to the right-handed orthonormal set of unit vectors characterizing the fracture triad. Compressive yielding and tensile failure are each treated separately. Compressive failure involves the evolution of a scalar damage variable, and tensile failure involves the formation of three fracture surfaces and the evolution of a void strain.

The 3-D crack damage model is used for problems where the directionality of the damage is more critical in matching experimental results. A hardened 440 stainless steel sphere with 0.05 cm diameter impacted against an optical quality fused silica glass surface at 305 m/s. The experimental damage morphology was a hemispherical sub-surface fracture region surrounded by a smaller, heavily fractured inner "plastic" region, and, in some cases, with a few Herzian cone cracks extending somewhat further into the target. The experimentally measured outer

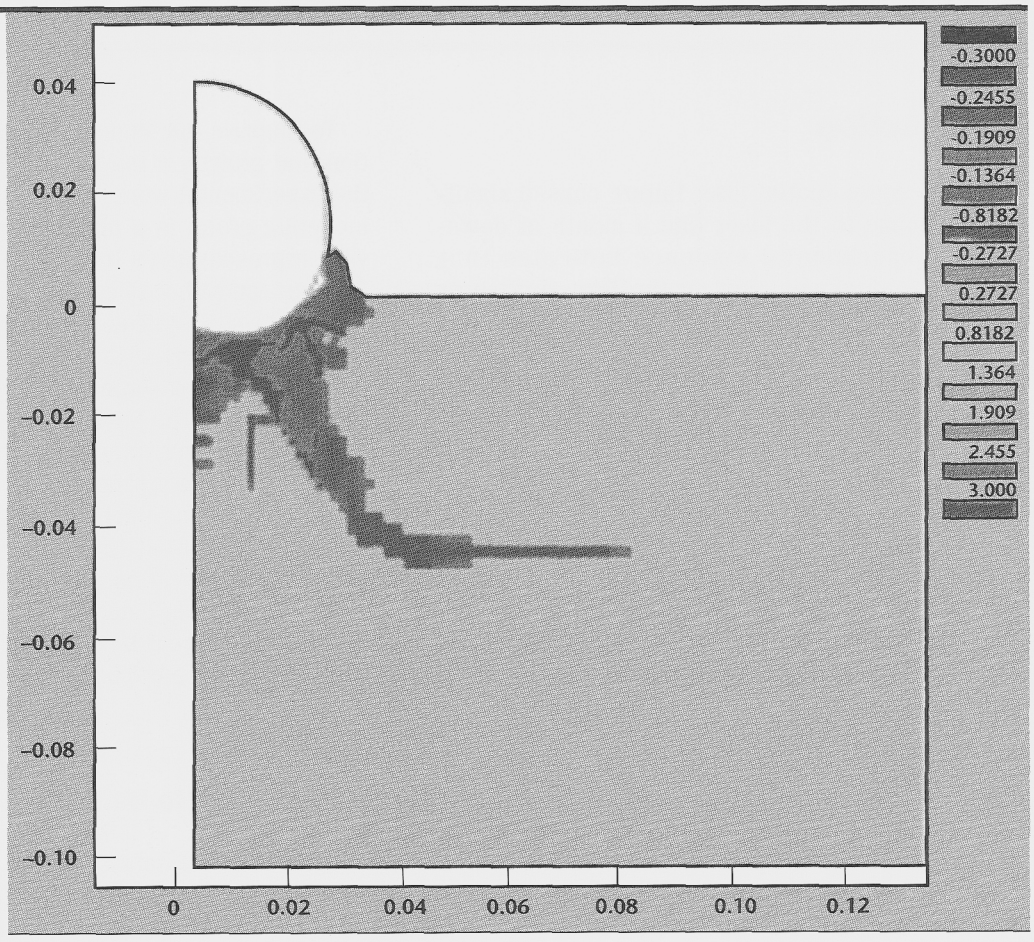


Figure 1. Steel ball impacts fused silica at 305 m/s. The fringe of damage is at 80 μs.

fracture diameter was 0.165 cm; the inner, more fractured region, had a diameter of 0.08 cm; and the depth of the fractured region was 0.10 cm.

Figure 1 shows the steel ball impact analysis 80 µs after impact, displaying fringes of element damage states. Strong evidence of Herzian cone cracks is observed on the plot, as is the compressive "plastic" fracture region beneath the steel ball. The conical crack grows at a semi-apex angle of 30°, then moves horizontally. The maximum fracture diameter is 0.162 cm, compared to the experimental value of 0.165 cm. The inner "plastic" fracture region is 0.07 cm in diameter, compared to the experimental value of 0.08 cm. The depth of the damage is 0.05 cm, indicating a less than hemispherical damage region.

A complete description of these models and their applications to ceramic and glass damage can be found in an earlier report.[7]

Front-Side Particle Contamination

If a metal particle is on the front side of an optic and is hit with a laser, a pressure pulse due to the laser light absorption on the surface of the particle will be transmitted through the particle and into the glass substrate. An aluminum spherical particle, 100 µm in diameter, was placed in contact with a silicon oxide substrate. Initial contact was assumed to be a 40° apex angle. A 150-kbar, 4-ns full-width half-maximum (FWHM) Gaussian pressure pulse was applied to the surface of the sphere with a cosine squared pressure distribution. The Steinberg-Guinan constitutive law, Mie-Gruneisen equation of state, and Tuler-Butcher failure criteria for spall were used to model the behavior of the aluminum particle. The silicon oxide substrate was modeled with the Modified Rubin-Attia 3-D crack damage model and a linear polynomial equation of state.

The damage in the glass begins at the outer contact points where shear stresses develop first. The damage grows into the substrate, then connects to the subsurface damage (12 µm below the surface) where the shear stresses reach their maximum. Damage then continues to expand in a semi-circular

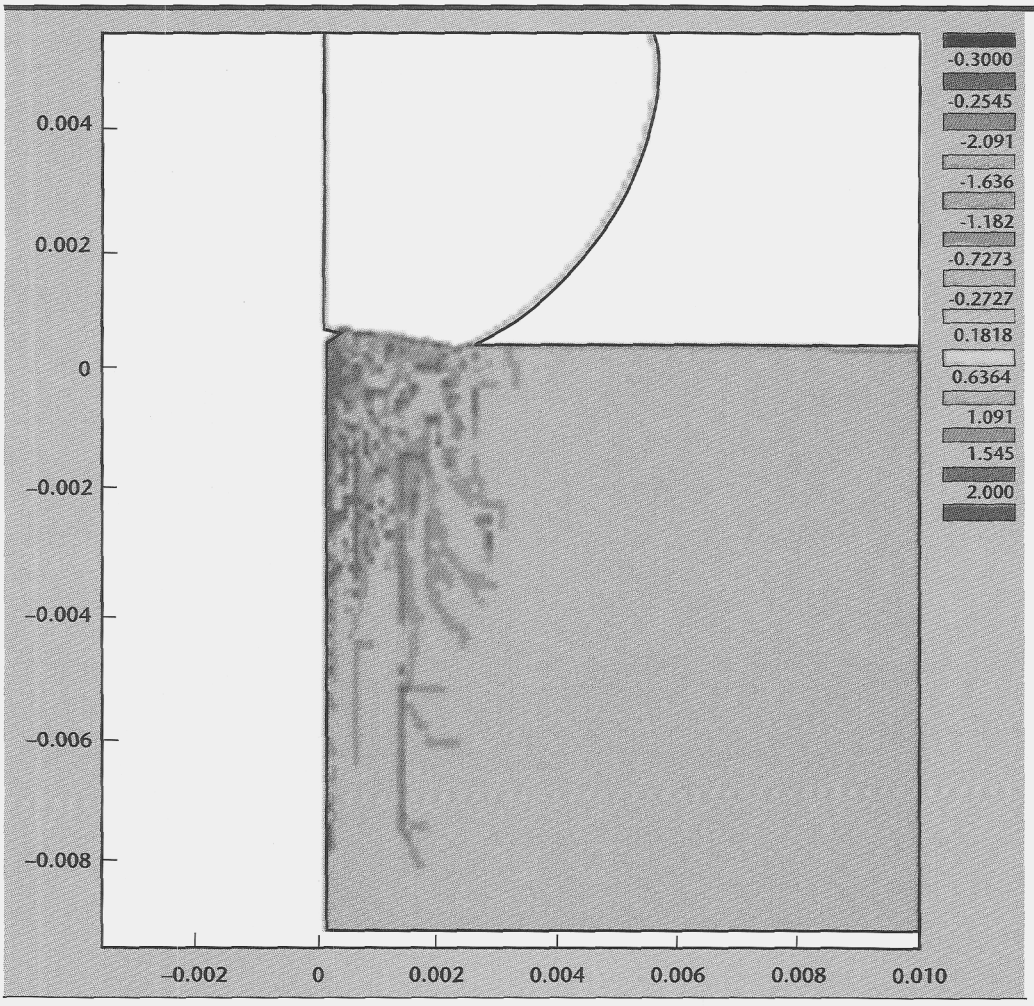


Figure 2. Aluminum sphere 100 µm diameter, 40° contact angle, 150 kbars, 4-ns FWHM Gaussian Pulse; time = 50 ns. The fringe of damage is in glass substrate.

fashion with visible fingers of damage emanating from the main sub-surface damage region (**Fig. 2**). The aluminum particle separates from the substrate and the gap grows until the damaged glass expands and fills the gap. Significant damage of the glass occurs to a depth of 40 µm, with fingers of damage growing to 80 µm deep. The diameter of damage is 70 µm.

Back-Side Particle Contamination

Back-surface particle contamination of optics components involves a particle on the back surface being irradiated by laser light, creating a hot expanding plasma, which then imparts a severe pressure loading on the glass surface and creates microcracks and unacceptable damage. To determine the pressure loading on the glass surface, LASNEX was used to model the formation of the plasma and its subsequent pressure loading of the glass. An aluminum disc 100 µm in diameter and 5 µm thick was placed on the backside of a silicon oxide substrate. A 18-J/cm^3 Gaussian pulse, 3-ns FWHM, was applied to the optic. Pressure as a function of time and radius at the solid/liquid interface (0.1 eV) was obtained from the LASNEX calculation.

To facilitate the transfer of pressure data from LASNEX to DYNA2D, a table look-up feature was incorporated into DYNA2D to apply this time- and spatially-varying pressure history to a virgin glass substrate. A DYNA2D model, 400 µm in diameter and 300 µm deep, was generated to apply the pressure histories. Non-reflecting boundary conditions were used to simulate a semi-infinite block. A pressure boundary surface from a radius 0 to 80 µm was specified on the semi-infinite block and the table look-up feature was used to apply the pressures as a function of radius and time. The silicon oxide substrate was modeled with the modified Rubin-Attia 3-D crack damage model.

The pressure pulse starts out in a rectangular pattern, but dispersion causes the pulse to spread out spherically. Damage behind the main pressure pulse causes pressure concentrations to occur. Damage begins at the outside radius of the disc where shear stresses are maximum. Damage follows the pressure pulse, then begins to move out in finger-like cracks at ~45° (**Fig. 3**). Damage is measured to a depth of 120 µm, with a diameter of 170 µm.

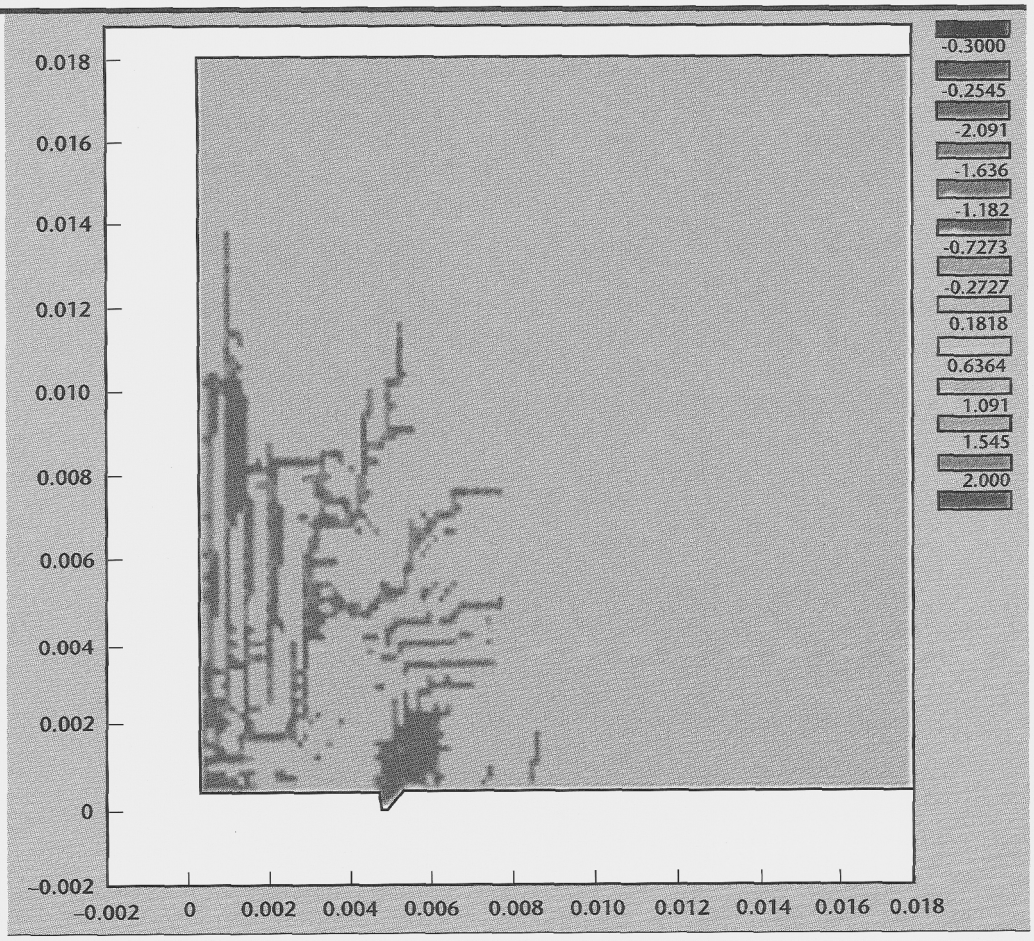


Figure 3. Aluminum disc 100 µm diameter, 5 µm thick on silicon oxide back surface; 18 J/cm^3 Gaussian Pulse, 3-ns FWHM. The fringe of damage is at time = 60 ns.

Sub-Surface Particle Contamination

Sub-surface particle contamination of the optics has been shown to cause unacceptable damage after irradiation. It has been postulated that cerium oxide particles, left over from the polishing process, are absorbing energy, expanding and causing small craters on the glass surface. DYNA2D was used to investigate this phenomenon by explicitly modeling the cerium oxide particle and the glass substrate. Energy was deposited into the particle allowing it to expand and damage the glass substrate. It was assumed that the glass around the cerium oxide particle does not absorb light and heat up.

The cerium oxide particle modeled was a 100-nm-diameter particle, 300 nm deep in a silicon oxide substrate. A linear polynomial equation of state with energy dependent bulk modulus and Gruneisen gamma was used for the cerium oxide particle. The silicon oxide substrate was modeled with the modified Rubin-Attia 3-D crack damage model.

Energy was deposited into the cerium oxide particle at a constant rate for 3 ns. Pressure in the cerium oxide particle builds up to approximately 36 kbars at 1.8 ns, then the glass begins to damage and the cerium oxide particle expands and relieves pressure. At 1.0 ns, the initial compressive wave has reached the upper surface and reflected back as a tensile wave which initiates damage at the 90° location. At 1.9 ns, tensile waves reflected off the surface have created a spall plane near the surface and the damage is growing down toward the particle. At 2.0 ns, the glass around the particle is undergoing compressive failure as the crack is growing out and toward the surface (**Fig. 4**). The surface damage grows down and converges with the particle. At 2.9 ns, the conical crack emanating from the 90° location on the particle has reached the surface, and one would expect the material within the damaged region to be blown off (**Fig. 5**).

Future Work

Different particle shapes were not addressed for the front-surface contamination study, but would also play an important role in understanding the transfer of pressure into the glass substrate, resulting in its damage. Irregular-shaped particles would absorb laser light differently, and subsequently load the glass substrate in a non-uniform manner.

The back-surface contamination study should be enhanced to look at different particle shapes and

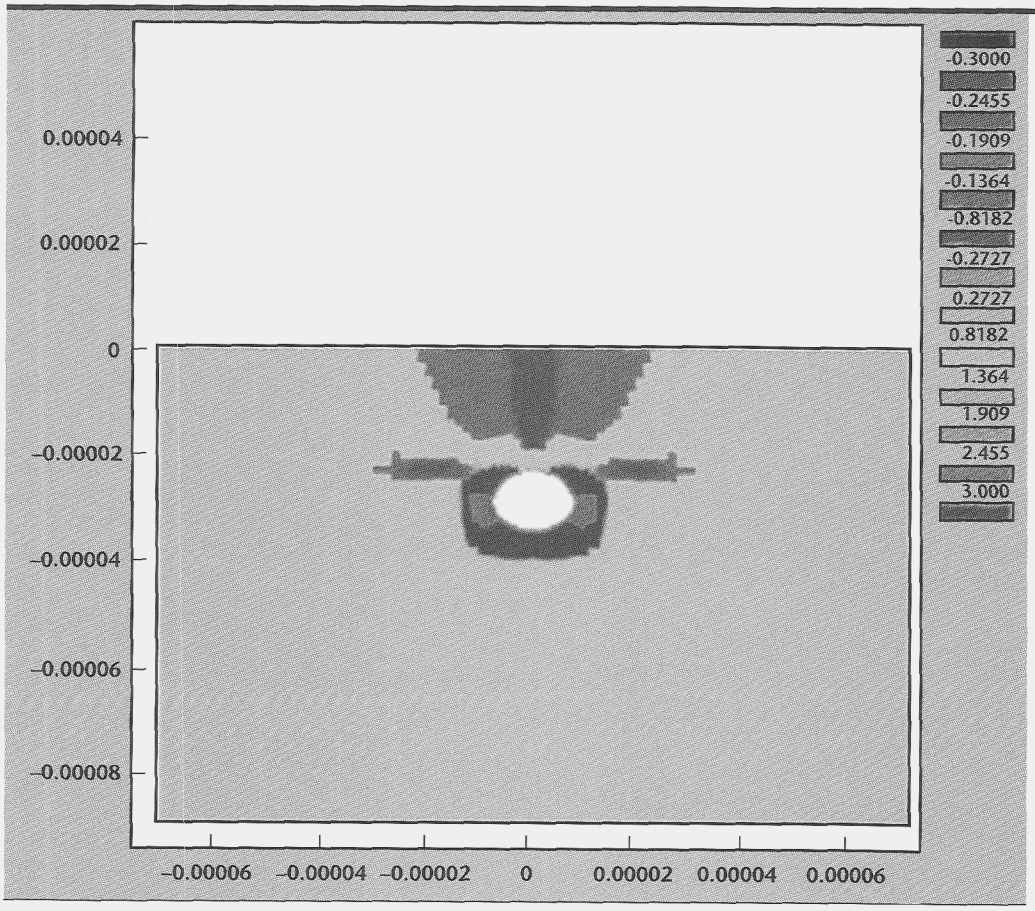


Figure 4. Cerium oxide particle 100 nm diameter, 300 nm deep in silicon oxide substrate. The fringe of damage is at time = 2.0 ns.

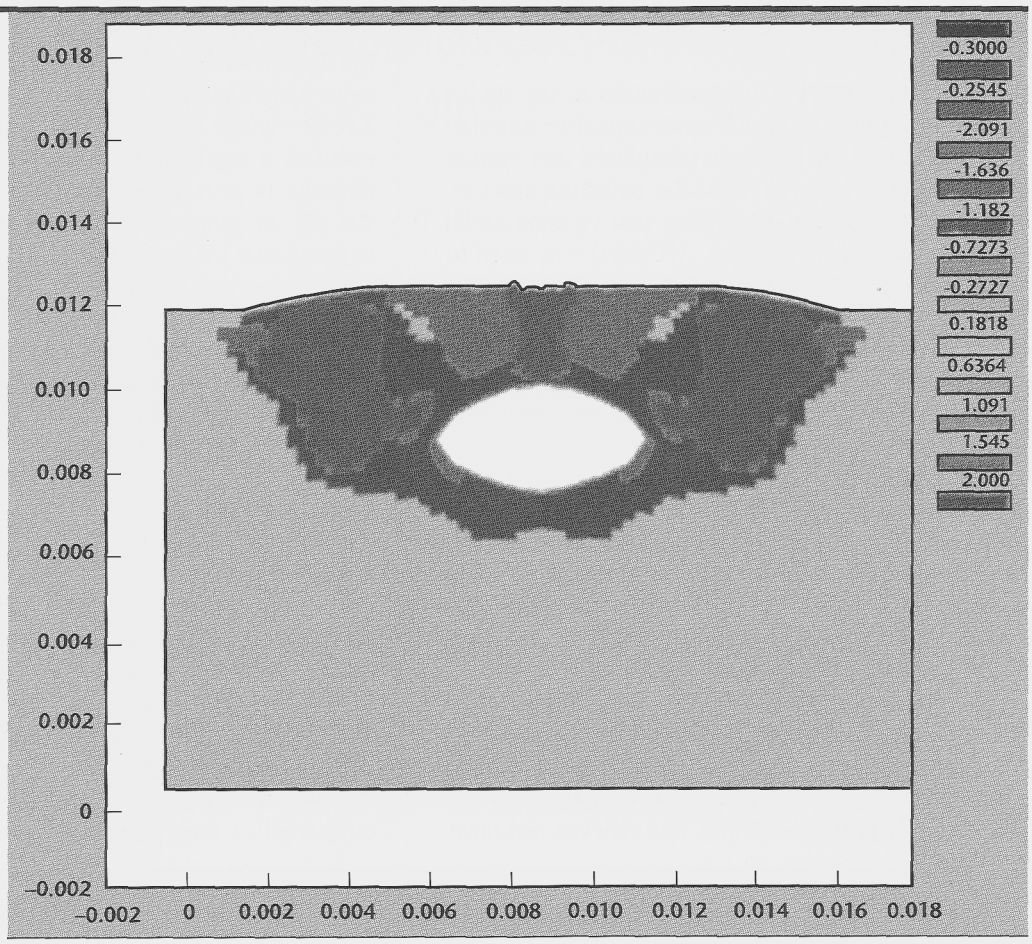


Figure 5. Cerium oxide particle 100 nm diameter, 300 nm deep in silicon oxide substrate. The fringe of damage is at time = 2.9 ns.

materials. The particle shape and material would significantly affect the formation of plasma and subsequent loading on the glass substrate.

Sub-surface particle size variations were not addressed here, but would also play an important role in understanding the formation of craters in the surface of the glass. For the cases investigated here, the crater was at a depth such that surface interaction influenced the formation and direction of the damage (cracking). Additional depths should be looked at, to find out where this surface interaction diminishes. In those cases, a more accurate definition of the equation of state of the cerium oxide particle is needed to handle the solid/liquid/vapor transition. Also a more accurate energy deposition scheme would be needed.

References

1. Whirley, R. G., B. E. Englemann, and J. O. Hallquest (1992), *DYNA2D A Nonlinear, Explicit, Two-Dimensional Finite Element Code For Solid Mechanics User Manual*, Lawrence Livermore National Laboratory, Livermore, CA, April, (UCRL-MA-110630).

2. Tuler, F. R., and B. M. Butcher (1968), "A Criterion for the Time Dependence of Dynamic Fracture," *Int. J. Fracture Mechanics*, December, Vol. **4**, (4).

3. Cagnoux, J. (1984), "Phenomenological Model of Pyrex Glass Fracture," *J. De Physique*, Col C5, suppl au no. 8, Tome 46.

4. Glenn, L. A., B. Moran, and A. S. Kusubov (1992), *Modeling Jet Penetration in Glass*, Lawrence Livermore National Laboratory, Livermore, CA, April, (UCRL-JC-103512).

5. Steinberg, D. J., and R. E. Tipton (1994), *A New Fracture Model for Ceramics*, Lawrence Livermore National Laboratory, Livermore, CA, March, (UCRL-JC-116953).

6. Rubin, M. B., and A. V. Attia (1990), *A Continuum Tensile Failure Model With Friction*, Lawrence Livermore National Laboratory, Livermore, CA, August, (UCRL-ID-104759).

7. Faux, D. R. (1997), *Brittle Damage Models in DYNA2D*, Lawrence Livermore National Laboratory, Livermore, CA, September, (UCRL-ID-128520).

Parallel Algorithm Research for Solid Mechanics Applications Using Finite Element Analysis

Carol G. Hoover and Daniel C. Badders
Defense Technologies Engineering Division
Mechanical Engineering

Anthony J. De Groot and Robert J. Sherwood
Defense Sciences Engineering Division
Electronics Engineering

Two important methods for treating parallel contact for explicit finite element methods are described. Using parallel computers, performance enhancements for small, medium, and large engineering models have resulted in tremendous turnaround improvements for design calculations.

Introduction

Significant advances in the development of new contact algorithms and partitioning techniques have led to scaleable production applications using ParaDyn, the parallel version of DYNA3D. This has resulted in several benefits to our engineering design programs at Lawrence Livermore National Laboratory (LLNL).

First, calculations are now performed in a day or less for problems which previously ran over several weeks. Next, new models are being generated for mesh sizes between one million and ten million elements. This is an order of magnitude larger than the largest models possible in the past. Finally, longer time simulations (problems running for a few million steps) for small- to moderate-sized problems are now possible on both distributed and shared memory systems using the message-passing paradigm implemented in ParaDyn.

A successful strategy for parallel implementation of the explicit finite element method used in DYNA3D (ParaDyn) was described in an earlier report.[1] The method is based on dividing the finite element mesh among the processors, and executing ParaDyn on the individual subdomains. This is a message-passing parallel model, with nodal point data identified as the shared data between processors.

The algorithms for calculating internal deformations involve only one communication step, which occurs when the element stress gradients are scattered to the nodal points that are shared. A more difficult problem is to calculate the contact forces needed to prevent penetration at a material interface and include those in the communication step.

Contact occurring in applications can be characterized by whether the problem involves predictable motion of surfaces or whether large deformation and motion on the grid results in arbitrary interactions among the surfaces in the problem.

In the first case, contact is treated by defining pairs of surfaces that are expected to interact with each other, and the search for material interpenetration can be limited to neighborhoods on these surfaces. We refer to this as "local" contact. The more general case occurs for problems in which the motion of the surfaces is arbitrary. In this case a localization technique for detecting contact, referred to as "bucketing," is used. Bucketing is simply the construction of an Eulerian grid of boxes (in one, two, or three dimensions) and the assignment of the surface nodes to the buckets.

An example of arbitrary contact is a ball rolling on a plane. A novel aspect of our original implementation of a parallel algorithm for arbitrary contact was the notion of using more than one partitioning for the problem, one for the finite-element deformation calculations and a second one for the contact calculations. This idea of using two partitions for the same problem is needed also for local contact algorithms.

Conceptually, there are several useful ideas for implementing parallel algorithms for local contact. One idea is to allocate each surface pair entirely to one processor, and to distribute the set of all surface

pairs among as many processors as possible. There is no additional communication among processors (due to contact) using this method, because the contact is assumed to occur only between points on surfaces that were defined as a pair.

A second approach is to extend the bucketing methods used in arbitrary contact to the local contact algorithms. Then the parallel implementation would involve allocating the buckets to processors. In this case, communication occurs for contact calculations in buckets that are on the boundary of a processor.

Yet a third method is to allocate surface pairs to processors, and, for the contact calculation, choose a coordinate system with one surface fixed and the other moving. Again, this algorithm requires additional communication for contact as the moving surface crosses the boundary of a processor. Each of these methods involves the development of techniques for partitioning objects, and data structures for connecting the contact calculations with the full mesh partitioning. We describe the implementation of the first method for local contact in ParaDyn in the "Progress" section of this report.

The automatic contact algorithm in DYNA3D was developed to implement arbitrary contact and to eliminate the need for the time-consuming task of surface definition for large models. The original algorithm has been improved and made much more robust in the last few years.[2]

The current algorithm is designed also to more efficiently search for contact with new options. For example, boxes defined on the grid can be used to limit the domain for the automatic contact application. Boxes are further augmented by the implementation of multiple automatic contact definitions, giving more flexibility to the analyst modeling complex problems. Finally, materials can be selected for inclusion or exclusion from a specific automatic contact definition.[3]

The steps in the automatic contact algorithm include 1) an algorithm for generating the faces on every surface in the regions defined for the contact; 2) bucketing methods to localize the search for contact onto a coarse grid; 3) a local search to find contact nodes and penetrated facets; and 4) a contact force calculation.

Our first implementation of parallel contact[1] was developed in the automatic contact algorithm because it treated the general case for arbitrary contact and it incorporated many contact types into a single algorithm. Bucketing methods make the algorithm particularly compatible with parallel computing methods. We report on progress in developing the latest version of the algorithm in the section below.

Progress

This section emphasizes advances in our parallel algorithm research, followed by a discussion of some of the parallel production calculations simulated with the ParaDyn program. It concludes with progress on other projects that are part of the broader activities covered in our parallel algorithm development efforts.

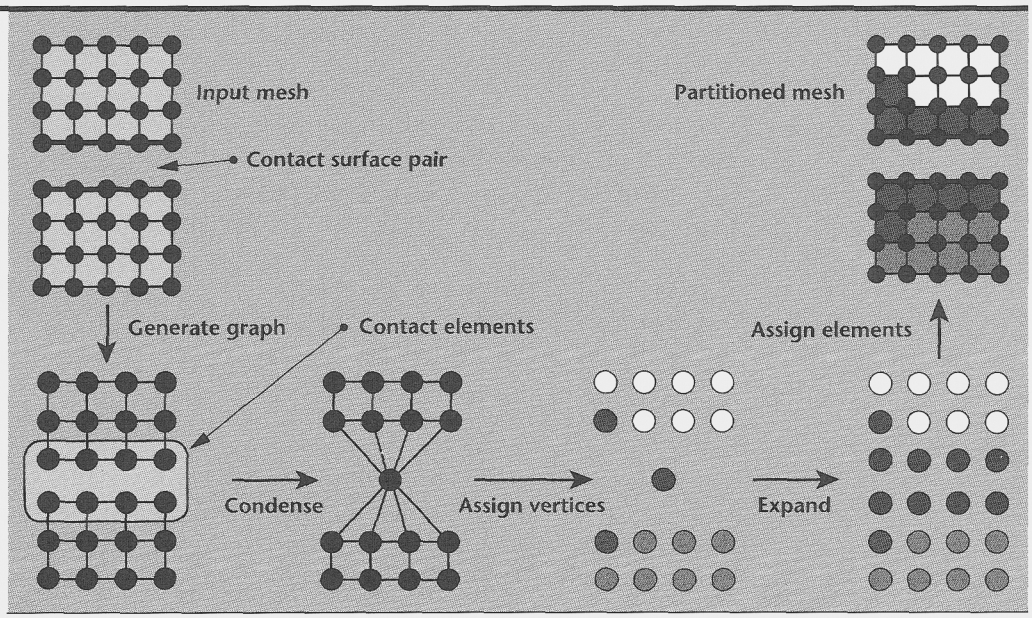


Figure 1. Illustration of the five logical steps used to allocate a contact surface pair to a processor. We condense the vertices associated with contact before giving the graph to METIS, to ensure that all elements involved in contact are in the same processor.

Combined Mesh/Boundary Partitioning Method for Local Contact

There are two conceptual and developmental steps in the implementation of our new algorithm for local contact. The first step is a technique for ensuring that a contact surface pair is allocated fully to a processor, and at the same time, distributing all pairs of surfaces among the processors for efficient load balancing. This partitioning step is implemented in preprocessing software, which provides many tools for mesh and boundary partitioning.[4]

The second step is to implement changes in ParaDyn so that the method applies to penalty contact methods, Lagrange contact, and other special purpose algorithms for nodes in contact with surfaces.

We use the METIS software[5] to partition our meshes. The METIS algorithms use graph-based methods in which a graph has vertices and interconnecting edges. For finite element meshes, we identify the vertices with the elements in the problem and the edges with the nodes shared between elements.

Once the graph is generated (in this case the graph is the element-to-element connectivity matrix), the METIS algorithms find an optimal division of the graph corresponding to the number of processors in the problem. Since two partitions are needed for the problem, the contact surfaces must be identified, and a connection made to the partition for the full mesh.

The new idea needed to allocate the surface pairs to processors is based on the notion of condensing a contact surface pair into a "super element," containing all of the elements having a facet on either of the surfaces. Each element has a vertex weight determined by its element type. Each super element has a vertex weight equal to the number of elements condensed, and an edge weight corresponding to the total connectivity of the nodes on the surface. This idea is represented in **Fig. 1**.

Next, the graph for the mesh is replaced by a graph with the super elements and the remaining elements in the mesh. The result is partitioned with METIS. This is a very advantageous approach because it automatically finds an optimal load balance among the partitions, including the super elements. Another advantage is that all of the required shared node communication for both the contact forces and the deformation forces is known in advance at this preprocessing step.

Preprocessing software which automatically partitions boundary conditions and allocates the surface pairs identified for the parallel local contact has been a significant software development effort in the last year. Six new preprocessing tools have been developed to fully implement the needed capability. In addition, we have made extensive modifications to the program for generating the partition file, and this has increased the speed by over an order of magnitude for some problems.

The memory requirements for our software have been sufficiently reduced so that all of the problems of current interest can run with considerably less than 2 GB of memory. METIS still requires more than 2 GB of memory for one problem of interest.

Finally, work is in progress to implement fully dynamic memory allocation for the dual generation program. The pay-off for this code development effort has been an automated software system that has effectively enabled a full range of production models for parallel calculations.

Additional work was necessary in ParaDyn to most efficiently implement this method for local

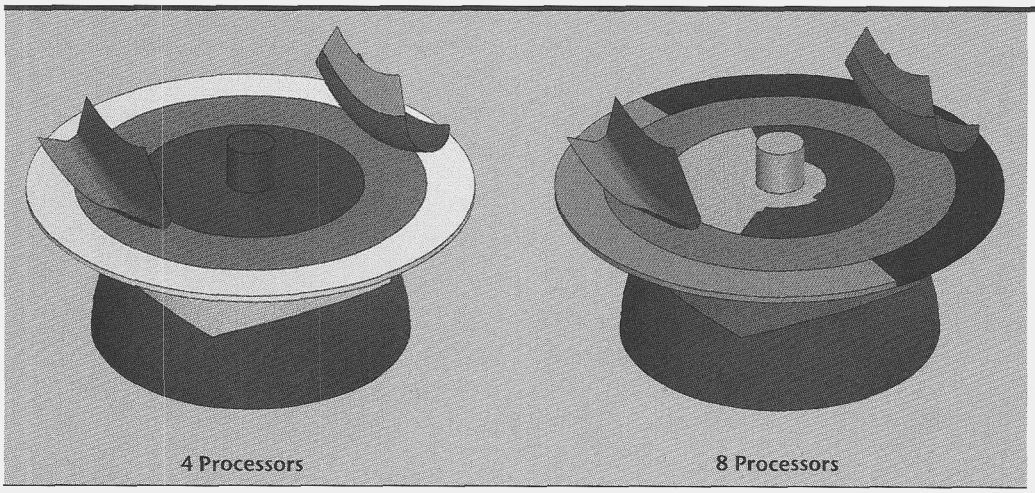


Figure 2. Illustration of metal-forming application in which a spinning plate is formed by two rollers of different size and shape. The shading shows the partitioning for four and eight processors, with the surface ring under the rollers fully contained in one processor.

contact. For example, a re-ordering of the algorithm so that the communication step followed the calculation of the Lagrange contact forces has resulted in one, rather than two communication steps. Results for parallel contact using both penalty and Lagrange methods have been checked for agreement with single processor results.

Partitioning with local contact surfaces for two different models is shown in **Figs. 2** and **3**. These examples demonstrate visually that this combined mesh/contact partitioning method results in a reasonable division for both models. The effectiveness of the partitioning can also be analyzed quantitatively.

METIS provides a balance statistic which takes on a value of 1.0 for a well-balanced calculation. In practice, we find that balance numbers between 1.0 and 1.1 provide a reasonable partition. Another check for the partitioning quality is to inspect the first few lines of the partition file. This gives the statistics for the allocation of the elements and nodes to the processors, as well as the number of shared nodes.

It is clear conceptually, and verified with the performance statistics for our production applications, that for a fixed problem size the maximum number of processors for an efficient calculation may be limited by the size of the largest contact surface. A variation of our algorithm to increase the maximum number of processors is to split large contact surfaces among one or more neighboring processors. We are currently investigating algorithms of this kind.

Parallel Automatic Contact Algorithm

Parallel arbitrary contact is treated in ParaDyn with a geometric partitioning of the contact surfaces into one-dimensional strips. The direction of the strips is determined dynamically and is the coordinate direction with the longest length which contains contact surface nodes. The strip widths are adjusted to equalize roughly the number of contact nodes allocated to a strip, and each strip is associated with a processor. Because the contact is arbitrary, new buckets are computed every ten time steps during the calculation. At these same times, the partitioning strips used in the parallel algorithms are recalculated and the contact buckets are redistributed among the processors. The use of these techniques localizes the contact search and load balances the parallel calculation.

The addition of new capabilities, such as boxes and multiple automatic contact, has necessitated a review and rewriting of the data structures and storage for the parallel algorithm. Although signif-

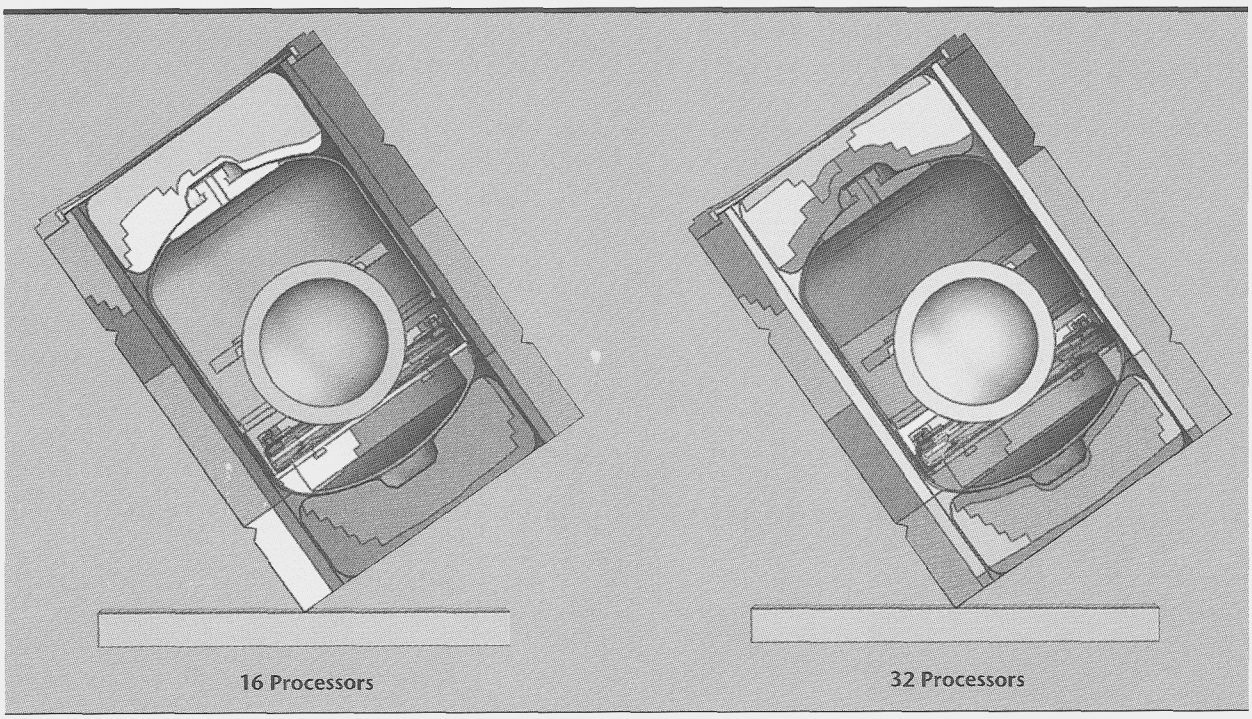


Figure 3. Illustration of the combined mesh/contact partitioning for a shipping container application. The problem included 27 contact surface pairs.

icant work is still in progress in this activity, recently we were successful in running the crash worthiness benchmark using the parallel automatic contact algorithm The calculation was noteworthy because the use of parallel computers made it possible to run the problem for a much longer simulation time than the previous simulation on a serial computer.[6] This progress is a tribute to both the robustness of our new algorithm modifications, and the performance enhancements provided by parallel computers.

Parallel Applications and Performance

ParaDyn is currently running on several parallel computers available at LLNL, and also on the parallel computers provided by the Department of Defense (DOD) High-Performance Computing and Modernization Program through the Major Shared Resource Center (MSRC) at the U.S. Army Engineer Waterways Experiment Station (WES) in Vicksburg, Mississippi. These include 256-processor IBM SP-2 systems; a 128-node (256-processor) Meiko CS-2 system; a 256-processor T3E and a 256-processor T3D; a 32-processor Origin 2000; 12- and 16-processor Power Challenge Arrays; and 10-processor DEC Alpha SMP workstation clusters.

The range of production applications that have been simulated on these systems varies from a very small (fewer than 5000 elements) spin-forming application run for more than two million time steps, to large-scale applications with more than a million elements. The following are highlights of some of the applications run with ParaDyn.

One of the first very large calculations with ParaDyn was an advanced design concept studying shock interactions with an underground target. This calculation modeled a shock at some distance from a target and required sufficient zoning in the region of the shock interface to represent the physics. This resolution resulted in a model with 2.5 million zones.

The performance on several parallel systems using up to 256 processors is shown, along with the partitioned mesh, in **Fig. 4**. This initial proof-of-concept application did not include contact in the model. Subsequent calculations with contact surfaces using either the local algorithms or the automatic contact algorithm are expected to scale as well as this original model.

Two mid-sized models were particularly beneficial to the W87 Life Extension Program by providing parameter studies in production runs which previously took too long to meet design schedules.

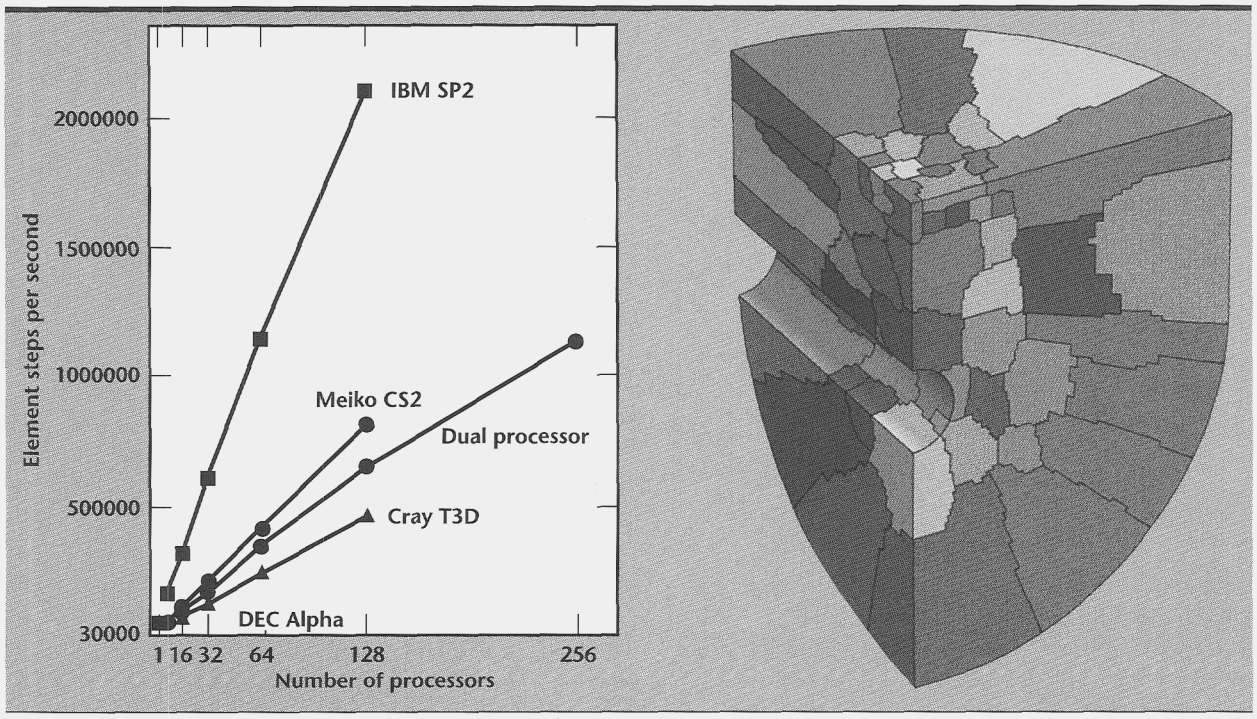


Figure 4. ParaDyn performance (a) for a million-zone problem. Problems in the multi-million element class will benefit from computers with larger numbers of processors and faster processor speeds. Illustrated in (b) are the automatic mesh partitioning results for this million-zone problem.

Figure 5 shows the performance and model input for these applications. In the first case (**Fig. 5a**), each ParaDyn production calculation was processed in half a day, where previously the same calculation would have required 27 days. Prior to the availability of parallel computers, this model would have been at the limit of the capabilities of the serial computers.

The model in **Fig. 5b** was the benchmark driving the ideas designed into our new local contact algorithm. The collaboration with the analyst resulted in a re-definition of the sliding surface interfaces at the mesh generation step to better fit the parallel algorithm. The optimal number of processors for an efficient parallel calculation was improved by a factor of four after the parallel local contact algorithm was available.

The adoption of MPI, a message-passing interface standard, plus our highly optimized domain-partitioning techniques, now makes it possible to run ParaDyn very efficiently on a symmetric multiprocessor (SMP) workstation, such as the DEC/8400 450-MHz Alpha system. Because the processors on an SMP are generally the fastest technology available, these systems are very promising parallel computing platforms for an entirely different class of problems, the small- to medium-sized problems with requirements for longer time simulations.

One application illustrating this is a spin-forming model with only 4800 elements. The partitioned mesh is shown in **Fig. 2**. The model was run for 1.1 s in 2.3 million time steps. The problem ran in 34 h on four DEC processors, compared to 96 h on a single DEC processor. Finally, this use of optimized partitioning methods with message-passing most likely results in better performance on an SMP than other types of parallelism, such as compiler directives.

Our collaborative DOD project with WES has broadened the range of applications that have been simulated with ParaDyn. **Figure 6** shows an example of their activities. Their largest model has been an application with 3.5 million elements.

Graphics and I/O

Several tasks in graphics and I/O software development needed for a parallel visualization tool are ongoing. The file-combining capability for parallel runs has been incorporated into the GRIZ visualization tool. This, plus the installation of the single processor version of GRIZ on the parallel computers, was a very valuable capability enhancement for the analysts. The new mesh I/O library (Mili) software is being added to DYNA3D/ParaDyn, and modifications have been added to GRIZ to provide the read capability for Mili files.

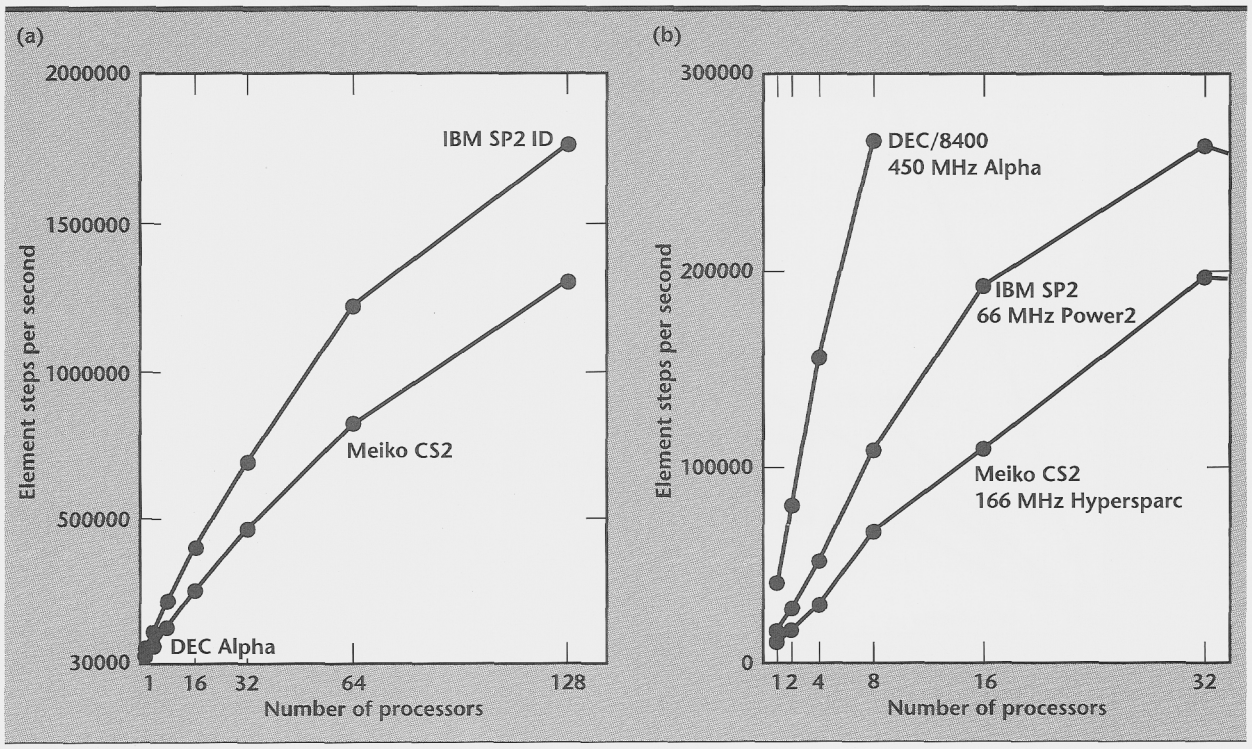


Figure 5. Performance data for two moderate-sized weapons applications. In (a) the mesh included 100,000 elements with 30 slide surfaces, containing a total of 20,000 nodes; (b) shows the performance for the shipping container application of Figure 3. The problem input specified 50,000 elements, 27 contact surfaces, 28,000 contact-related elements, and 3,200 elements in the largest contact surface pair.

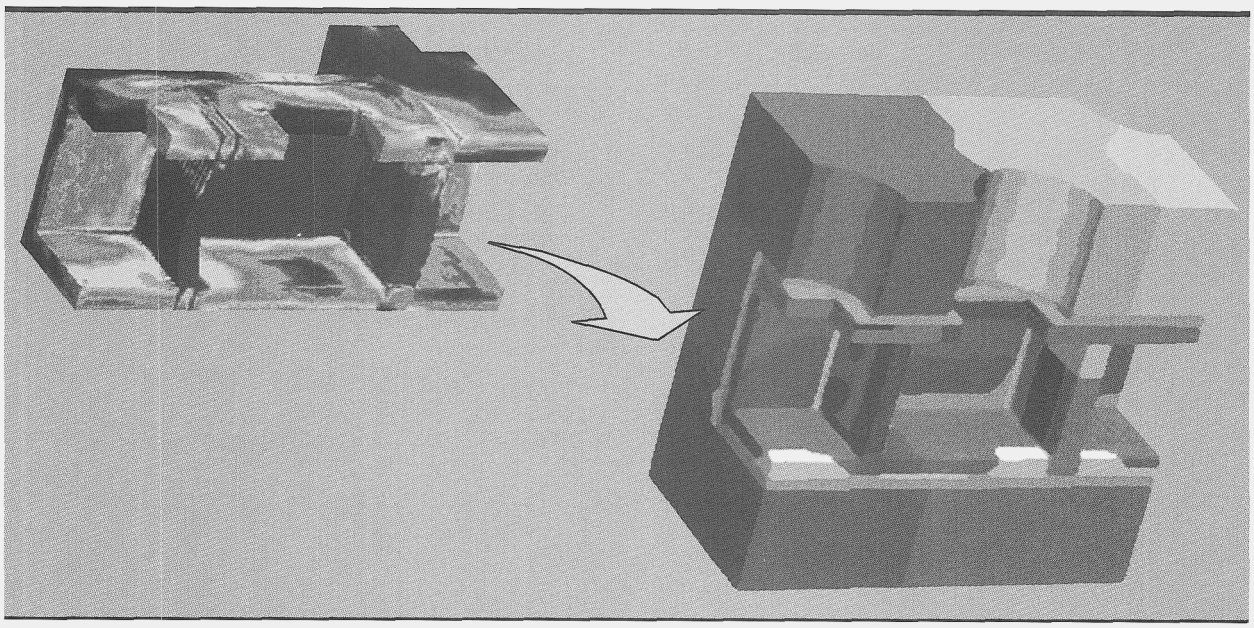


Figure 6. Model representing the loading of a structure by air blast and fragments. The simulation corresponds to experiments conducted at the U.S. Army Engineer Waterways Experiment Station, in collaboration with the Defense Special Weapons Agency. The results (a) show extensive rebar and concrete damage. The mesh (b) is partitioned for 32 processors.

Parallel Implicit Algorithms

We conducted a summer project to evaluate linear solvers for symmetric sparse matrices, to determine their potential use in a parallel version of the NIKE3D program. Our findings indicate that new solver strategies, iterative as well as direct, will be needed. Furthermore, most of the more important NIKE3D applications will require new solver techniques (for example, multi-grid methods), and will involve very challenging mathematical research.

Future Work

The code development in ParaDyn will continue to focus on optimizing the parallel algorithms which we currently have developed. We anticipate adding one more local contact algorithm either to fill in the capability which is not treated in the automatic contact, or to provide an alternate method for optimizing the performance for local contact.

Parallel algorithm development for NIKE3D will steadily increase, particularly as mathematical research moves forward in the area of multi-grid methods, and as we pursue the sub-structuring methods developed by Charbel Farhat from the University of Colorado.

Acknowledgments

Research grants from the LDRD Program and the Weapons Program at LLNL have provided the opportunity for this parallel project to mature to a production capability. The multi-million element problems run at LLNL are possible because of computers and grants provided by the Accelerated Strategic Computing Initiative (ASCI). Additional support for the ParaDyn Project has been provided by the DOD High Performance Computing and Modernization Program through a Common HPC Support Software Initiative (CHSSI) project, and a coordinated grant from the Defense Special Weapons Agency (DSWA). We gratefully acknowledge this support and the excellent collaborations with the DOD that have resulted from it.

Figure 6 was provided by the Waterways U. S. Army Engineer Experiment Station in Vicksburg, Mississippi.

References

1. Hoover, C. G., A. J. De Groot, J. D. Maltby, and R. J. Procassini (1995), *ParaDyn-DYNA3D for Massively Parallel Computers*, Lawrence Livermore National Laboratory, Livermore, CA, (UCRL-53868-94).

2. Zywicz, E. (1997), private communication, Applied Mechanics Group, Lawrence Livermore National Laboratory, Livermore, CA.

3. Whirley, R. G. and B. E. Engelmann (1993), *DYNA3D: A Nonlinear, Explicit, Dimensional Finite Element Code for Solid and Structural Mechanics—User Manual*, Lawrence Livermore National Laboratory, Livermore, CA, (UCRL-MA-107254 Rev. 1).

4. Hoover, C. G., D. C. Badders, A. J. De Groot, and R. J. Sherwood (1997), *Parallel Algorithm Research for Solid Mechanics Applications Using Finite Element Analysis*, Lawrence Livermore National Laboratory, Livermore, CA, (UCRL-ID-125471).

5. Karypis, G. and V. Kumar (1995), *METIS: Unstructured Graph Partitioning and Sparse Matrix Ordering System*, University of Minnesota, Department of Computer Science, Minneapolis, MN, (http://www.cs.umn.edu/~karpis).

6. Schauer, D. A., C. G. Hoover, G. J. Kay, A. S. Lee, and A. J. De Groot (1996), *Crashworthiness Simulations with DYNA3D*, Transportation Research Board, (Paper No. 961249).

An Accurate One-Step Elasto-Plasticity Algorithm for Shell Elements in DYNA3D

Jerry I. Lin
Defense Technologies Engineering Division
Mechanical Engineering

In the explicit finite element code DYNA3D, shell elements are treated as assemblages of plane stress laminae, hence the algorithms for the plane stress constitutive calculations play a significant role in the robustness and efficiency of the elements. A one-step algorithm is presented to find the analytical solution for the stress state of a von Mises lamina.

Compared to the existing iterative algorithms in DYNA3D, the proposed method is more efficient and easier to vectorize since no iterations are required. It also offers the same accuracy as the existing methods, and is far superior to other non-iterative schemes, such as the Stress Scaling Algorithm. Both isotropic and kinematic hardening laws can be accommodated, and rate-dependent material characteristics can be easily implemented.

Introduction

Advances in modern computers and developments in computational technologies in recent years have made detailed dynamic analyses of complex structures possible. Thin-walled or shell-type segments often exist in these structures, and usually are modeled by shell or membrane elements. Under impulsive loadings, such as those generated by explosive blast, ballistic impact, vehicle crash, or mechanical forming of metals, many of the shell structures exhibit strong non-linearities. To account for the material non-linearities, elasto-plastic constitutive models must be implemented in the analysis programs.

In explicit finite element codes, unlike their implicit or static counterpart, constitutive evaluation at the element level makes up a major part of the computational cost. To make feasible the simulation of a non-linear transient event, which often includes tens to hundreds of thousands of elements, and requires tens to hundreds of thousands of time steps in many applications, speedy constitutive evaluation is of critical importance. In addition to the efficiency issue, the possible existence of highly non-linear material behavior and severe element deformation necessitates reliable stress calculations, to avoid error accumulation through excessive time steps. Because of these two considerations, the constitutive algorithms adopted in modern explicit finite element codes must be both fast and accurate to be of any practical use.

Most of the elasto-plasticity constitutive algorithms can be categorized in two classes: iterative and one-step. Iterative methods solve non-linear equations through iterations, and one-step methods use a procedure that first predicts a trial stress state, then returns it to the yield surface, approximately.

As expected, iterative methods are, in general, more accurate, but time-consuming, whereas one-step methods are relatively fast, but less accurate.

DYNA3D[1,2] includes both an iterative method[3,4] and an one-step method[5] in its constitutive library. The one-step algorithm is an adoption of a radial scale-back algorithm.

Because of the inefficiency of fully iterative algorithms in a vectorized code environment, a compromised fixed three-iteration version of the iterative algorithm was also added to DYNA3D. The fixed three-iteration algorithm simply executes the iterative procedure three times and stops, disregarding convergence status. This allows the iterative loop to be unwrapped, which leads to vectorization.

Note that the most widely used shell element[6] in DYNA3D is based on the Mindlin Theory. Hence, the plane stress state means that only $\sigma_{ZZ} = 0$, and non-zero transverse shear stresses, that is, σ_{XZ} and σ_{YZ}, are considered in constitutive calculation.

In this work, an analytic solution to the constitutive equations for a von Mises material with infinitesimal strain increments is presented. This procedure is directly applicable to DYNA3D because of the inherent small time step size in explicit finite

element codes. Strain increments in a DYNA3D run for applications such as ballistic impact, penetration simulation, vehicle crash, and metal forming range from 10^{-7} to 10^{-4}. Our study shows that the strain increments of this size are small enough to be considered infinitesimal for stress increment evaluation purposes.

Progress

The Elasto-Plastic Algorithm for Shell Elements

Given the strain increments, the current stresses, the current strains, and the material properties, a constitutive algorithm in an explicit time-marching scheme must find the corresponding stress increment. Let us consider a material that is isotropic within elastic limit, and exhibits isotropic/kinematic hardening behavior beyond yielding. Furthermore, the material follows the von Mises yield criterion and the associated flow rule. At a given material point, that is, an element integration point in finite element analysis, **Eqs. 1** to **4** must be satisfied.

Hooke's Law:

$$d\underline{\sigma} = \underline{C}\left(d\underline{\varepsilon} - d\underline{\varepsilon}^{p}\right) \tag{1}$$

The Flow Rule:

$$d\underline{\varepsilon}^{p} = \dot{\lambda}\frac{\underline{a}}{\|\underline{a}\|} \tag{2}$$

The Hardening Law:

$$d\bar{\sigma} = \beta H d\bar{\varepsilon}^{p} \tag{3a}$$

$$d\underline{\alpha} = 2(1-\beta)H\frac{\dot{\lambda}}{\|\underline{a}\|}\left(\underline{\sigma} - \underline{\alpha}\right) \tag{3b}$$

The Yield Condition:

$$f\left[\left(\underline{\sigma} + d\underline{\sigma}\right) - \left(\underline{\alpha} + d\underline{\alpha}\right)\right] = \left(\bar{\sigma} + d\bar{\sigma}\right)^{2} \tag{4}$$

where

f	is the yield function,
$\underline{C}$	the elastic constitutive matrix,
H	the plastic modulus,
$d\underline{\sigma}$	the stress increment,
$d\underline{\varepsilon}$	the strain increment,
$d\underline{\varepsilon}^{p}$	the plastic strain increment,
$d\underline{\alpha}$	the back stress increment,
$d\bar{\sigma}$	the effective stress increment,
$d\bar{\varepsilon}^{p}$	the effective plastic strain increment,
$\underline{\sigma}$	the current stress,
$\underline{\alpha}$	the current back stress,

and

$$\underline{a} = \frac{\partial f}{\partial \underline{\sigma}} \quad \text{the plastic flow vector.} \tag{5}$$

Underlined entities represent vectors or tensors, and $\|\underline{a}\|$ stands for the magnitude of vector $\underline{a}$.

Eqs. 1 to 4 are directly applicable to infinitesimal strain increments. Because of the inherent small time steps in an explicit calculation, our study shows that these equations can be applied effectively to the finite strain increments in a time step. Combining **Eqs. 1 to 4** results in a quadratic algebraic equation for the proportional coefficient $\dot{\lambda}$. Solving $\dot{\lambda}$ and substituting it into these equations yields the desired quantities, such as $d\underline{\sigma}$ and $d\underline{\alpha}$.

This algorithm has been implemented in DYNA3D, and the sample problems in the next section demonstrate its effectiveness.

Examples and Comparison

Two examples are used to compare the proposed and existing algorithms. The first example is a rigid blunt projectile impacting a frame-supported plate; the second a vehicle hitting a u-shaped rigid post. For the first example, only a quarter of the model was analyzed, because of the symmetry about two axes. These problems are chosen because they are shell-dominated models and exhibit high material non-linearities. The results produced by the fixed three-iteration scheme, which is designated as the default algorithm in DYNA3D, is used as a benchmark for accuracy as well as efficiency comparison in the validation process. **Figures 1 and 2** show the final deformation.

Two runs were made for each example, one by the proposed algorithm, the other by the DYNA3D default scheme. Only results by the proposed method are presented in the figures since there is no appreciable difference in the deformation shapes between the two runs.

To monitor the comparison more closely, key quantities such as nodal velocities and displacements and element effective stress at critical or sensitive locations were also examined for both problems. Sampling spots for the first example include the center point of the target plate and a

Table 1. Efficiency comparison for projectile impacting a plate.

	Overall CPU (s)	Constitutive evaluation CPU (s)
Proposed algorithm	947	120
Fixed 3-iteration algorithm	1093	256
CPU saving	13%	53%

Table 2. Efficiency comparison for vehicle crashing into a u-post.

	Overall CPU (s)	Constitutive evaluation CPU (s)
Proposed algorithm	8337	1070
Fixed 3-iteration algorithm	9316	2046
CPU saving	11%	48%

point on the diagonal line just outside the direct impact zone. Sampling spots for the second example are points on the fenders, where local shell buckling took place, and spots on the front bumper, where the direct impact occurred. Again, no significant differences were observed, hence no specific graphs are presented here.

Since both algorithms offer almost identical results, the superior computational efficiency of the proposed algorithm surely makes it more appealing than the default algorithm. **Tables 1 and 2** summarize the CPU time comparison on a DEC-Alpha workstation.

A rate increase of about 50% was achieved for both problems on constitutive evaluation. With other time-consuming features, such as contact slidelines and various other constitutive models, taken into account, the overall CPU saving is still at a significant level of more than 10%.

Future Work

Although intended for shell elements, the proposed algorithm can be extended to general 3-D continuum elements. It can also be modified to accommodate other similar materials, such as rate-dependent elasto-plastic materials and thermal elasto-plastic materials. These are the areas to explore in future years.

Reference

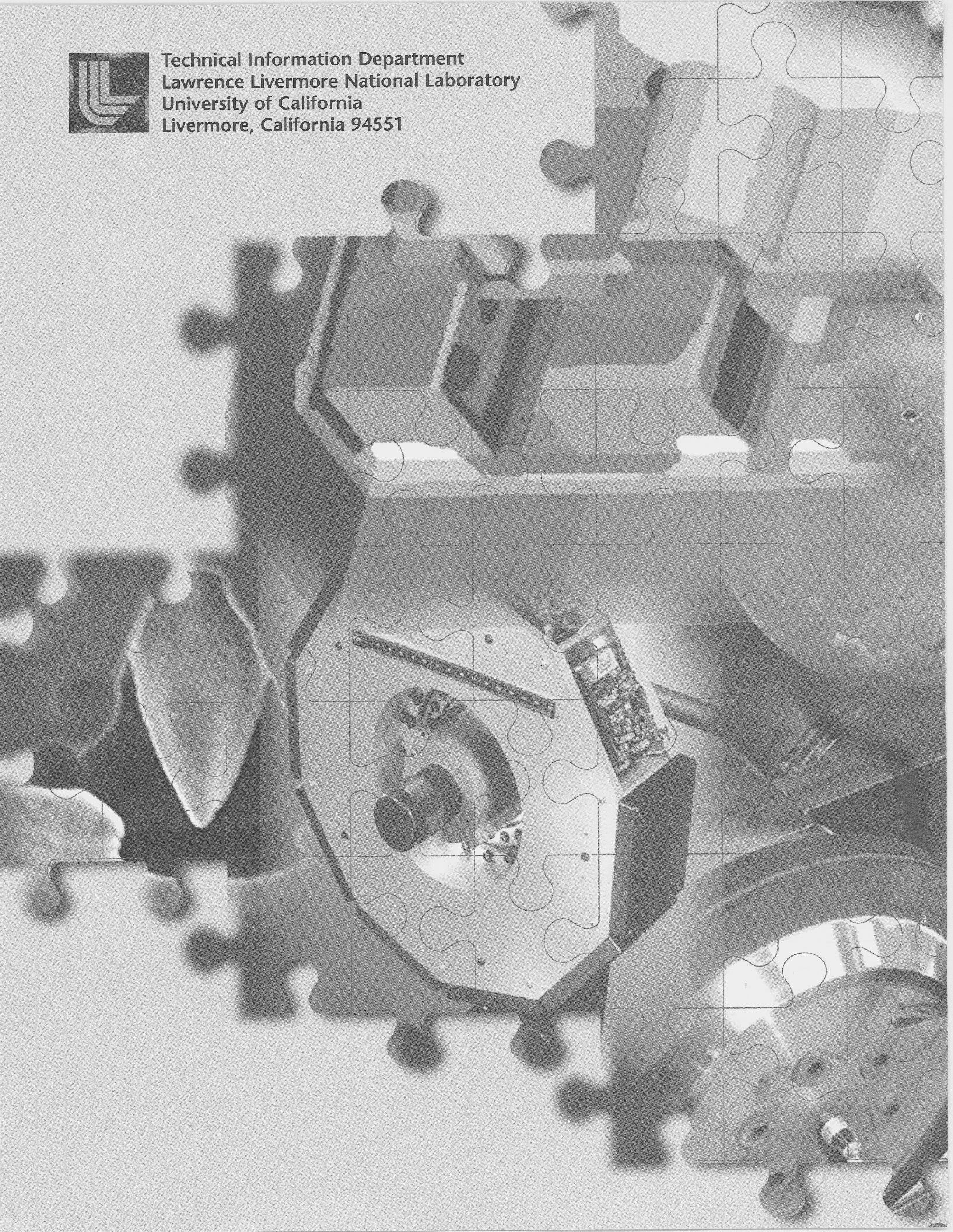
1. Whirley, R. G., and B. E. Engelmann (1993), *DYNA3D: A Nonlinear, Explicit, Three-Dimensional Finite Element Code for Solid and Structural Mechanics - User Manual,* University of California, Lawrence Livermore National Laboratory, Livermore, CA, (UCRL-MA-107254).

2. Whirley, R. G., J. O. Hallquist, and G. L. Goudreau (1988), *An Assessment of Numerical Algorithms for Plane Stress and Shell Elastoplasticity on Supercomputers,* University of California, Lawrence Livermore National Laboratory, Livermore, CA, (UCRL-99690).

3. Krieg, R. D., and S. W. Key (1976), "Implementation of a Time Dependent Plasticity Theory into Structural Computer Programs," *Constitutive Equations in Viscoplasticity: Computational and Engineering Aspects,* ASME, New York, NY, Vol. **20**, p. 125.

4. Hughes, T. J. R., and E. Hinton, Eds. (1986), "Finite Element Methods for Plate and Shell Structures," *Element Technology,* Pineridge Press International, Swansea, United Kingdom, Vol. **1**, p. 394.

5. Key, S. W., Z. E. Beisinger, and R. D. Krieg (1978), *HONDO II: A Finite Element Computer Program for the Large Deformation Dynamic Response of Axisymmetric Solids,* Sandia National Laboratory, Livermore, CA, (SAND78-0422).

6. Belytschko, T., J. I. Lin, and C. S. Tsay (1984), "Explicit Algorithms for the Nonlinear Dynamics of Shells," *Computer Methods in Applied Mechanics and Engineering,* **42**, pp. 225–251.

Technical Information Department
Lawrence Livermore National Laboratory
University of California
Livermore, California 94551