Simulation of Large Deformations in Elastic Solids via an Eulerian Approach

Shayan Hoshyari[1] Department of Computer Science, University of British Columbia

Abstract

A considerable challenge in simulating large deformations in solids lies in keeping track of the material motion. Lagrangian and Eulerian frameworks are among the main approaches for dealing with such problems. The former explicitly keeps track of the solid during the simulation, whereas the latter uses a fixed spatial grid. The goal of this project is to implement a large deformation elasticity solver based on an Eulerian framework. The solver can simulate the deformation of a 1-D medium and its collision with different fixed barriers. In 2-D, the solver can find the steady state deformation of a solid subject to fixed boundary conditions. Code has also been written for simulating the translation and deformation of a solid with a free a surface. However, the implementation still suffers from bugs and is not fully functional. The simplest working underlying numerical methods are used, i.e., a first-order finite-difference spatial discretization, a first-order explicit Euler temporal discretization, and handling of collisions via non-penetrating linear constraints on velocity.

1. Introduction

Simulation of large deformations in solids is of interest in many fields such as mechanical engineering, computer graphics, and biomechanics. For example, muscle tissue is a soft material, and its correct modeling, either for the purpose of physics based animation, or biomechanics design, requires modeling of large deformations.

An important challenge in simulation of large deformations is keeping track of the solid and its motion. A Lagrangian formulation explicitly follows each material particle. Thus, tracking free surfaces between different material, and handling constitutive models that include displacement values are tasks well suited for Lagrangian methods. However, these methods may require complicated geometrical and topological operations, such as remeshing, mesh untangling, and collision detection. Eulerian frameworks, on the other hand, are formulated using a fixed spacial grid. Although they are mostly suitable for constitutive models that include rate of the displacement, they can still be used for free surface tracking, albeit by introduction of additional variables such as reference maps and/or level sets. Hybrid approaches are also possible, such as Arbitrary Eulerian Lagrangian (ALE) [1] and Eulerian on Lagrangian (EoL) methods [2].

Many industrially relevant problems include more than just a single solid with specified boundaries. Rather, they include interactions between different materials, such as collisions between solids, or fluid-solid interactions. Solution methods to these problems can be very different and heavily depend on the quantities of interest. For example, solutions in the computer graphics community usually tend to be fast and visually plausible at the cost of accuracy, whereas engineering communities might be more interested in accuracy at the cost of solution speed.

Levin et al. [3] introduced a robust yet not so complicated Eulerian framework for the simulation of transient frictionless solid contact in computer graphics. This Eulerian framework was based on the earlier

[1]M.Sc. student, email: `hoshyari@cs.ubc.ca`, personal website: `http://h00shi.github.io`

work of Kamrin et al. [4], and was later extended by Teng et al. [5] to include fluid-solid interactions. The goal of this work is the development of a C++/MATLAB (for 2-D/1-D) partial implementation of the work of Levin et al. [3] and Kamrin et al. [4]. Examples of the following problems have been solved for a single solid:

1. Steady-state problems with clamped boundaries in 2-D.
2. Transient problems with free boundaries and collision with fixed barriers in 1-D.
3. Uniform translation of a single solid in 2-D.

The biggest challenge of this work has been rediscovering many subtle points that are crucial to the convergence of the solver, but are not covered in the references. Examples include the distance cut-off for distinguishing ghost and internal vertices, boundary conditions, and the effect of ghost vertex extrapolation on collision constraints. Getting these procedures to work has required extensive testing, deep study of the failure cases, initial implementations in 1-D, and starting off by solving the equations with several terms deactivated. Therefore, the solver is not able to solve more complicated problems given the time limit of a course project.

The numerical method will be described in section 2, followed by the results in section 3. Finally, the project is concluded section 4.

2. Method

2.1. Problem Statement

The deformation of a solid in Eulerian coordinates is prescribed by the equations

$$\begin{aligned} &\rho(v_t + v\cdot\nabla v) = \nabla\cdot T + \rho b, && (a)\\ &\rho = \rho_0 \det(\nabla\xi), && (b)\\ &\xi_t + v\cdot\nabla\xi = 0, && (c)\end{aligned} \qquad (1)$$

where ξ is the reference map (or material coordinate), T is the Cauchy stress tensor, x is the physical coordinate, v is the velocity, t is time, and b is the vector of body forces. Also, ρ_0 and ρ represent the density in the initial reference conditions and the physical space, respectively.

Given a solid as shown in figure 1-a, this project considers two cases. In the first case, the location of the solid Ω and its boundary $\partial\Omega$ in the physical domain are prescribed, along with fixed boundary conditions

$$v(x,t) = 0; \qquad \xi(x,0) = \xi_0(x) \qquad x \in \partial\Omega.$$

The goal is then to find the steady state deformation of the solid. Therefore, a damping term $\mu\nabla^2 v$ is added to the right hand side of equation (1-a), and the solution is advanced in time until its rate of change becomes negligible. Since the steady-state solution of the velocity would be zero, this term will not introduce additional errors. Section 3 further shows the effect of this damping term.

In the second case, the solid has a boundary which can move freely, so that the domain that the solid occupies Ω would be changing over time. Assuming that the boundary of the solid $\partial\Omega$ is identified as the iso-value of an implicit function ϕ, the free surface boundary conditions can be written as:

$$T(x,t)n = 0; \qquad \phi_t + v\cdot\nabla\phi = 0 \qquad x \in \partial\Omega(t),$$

where n is the unit normal on the solid boundary. For fluid simulations, the signed distance function is usually used as the implicit function ϕ, and is solved for as one of the variables. In this work, I follow

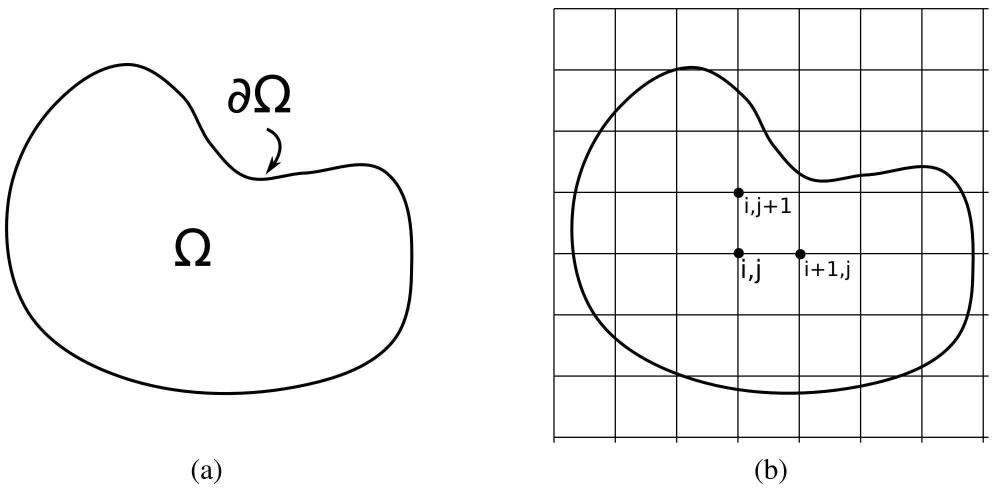


Figure 1: Schematic of (a) the domain that the solid occupies in the physical space (b) the computational grid

Levin et al. [3], and use the reference map ξ itself as the surface tracking implicit function. Kamrin et al. [4], on the other hand, follow the former approach and explicitly solve for the signed distance function. Admittedly, I am still not very comfortable with free surfaces, and don't have an intuition on the benefits of using one surface tracking variable over the other. Collision of the solid with fixed barriers can also be modeled by constraining the velocity to be non-penetrating on the barrier boundary [3].

To discretize the problem, the physical domain will be placed on a structured grid, as shown in figure 1-c, where a control volume Ω_{ij} is associated with every vertex. Then, the velocity and reference map values are solved for at every grid point and time level. This process is addressed is three main steps: discretization of the spacial terms, boundary conditions, and time advance.

2.2. *Discretization of the Spacial Terms*

Finite-difference approximations are mainly used in this work to evaluate a discrete approximation of the differential operators in equation (1). To deal with the discontinuous density field due to the free boundary, however, some ideas have to be borrowed from the finite-volume method: equation (1-a) is integrated inside a control volume Ω_{ij}, and the divergence theorem is used to yield

$$\left(\int_{\Omega_{ij}} \rho d\Omega\right)(v_t + v \cdot \nabla v - b) - \int_{\partial\Omega_{ij}} T n dA = 0, \tag{2}$$

where the term $m_{ij} = \int_{\Omega_{ij}} \rho d\Omega$ is evaluated using quadrature with a subgrid resolution. The conventional finite-difference approximations can then be applied to discretize the equations (1-b), (1-c), and (2).

The advective terms $v \cdot \nabla v$ and $v \cdot \nabla \xi$ are evaluated using a first-order upwind difference scheme

$$\begin{aligned}(v \cdot \nabla \chi)_{ij} = \max(v_x, 0)\frac{\chi_{i,j} - \chi_{i-1,j}}{\Delta x} &+ \min(v_x, 0)\frac{\chi_{i+1,j} - \chi_{i,j}}{\Delta x} \\ &+ \max(v_y, 0)\frac{\chi_{i,j} - \chi_{i,j-1}}{\Delta y} + \min(v_y, 0)\frac{\chi_{i,j+1} - \chi_{i,j}}{\Delta y} \quad \chi = v_x, v_y, \xi.\end{aligned} \tag{3}$$

The gradient of the reference map is evaluated at the cell centers using central difference formulas

$$(\nabla \xi)_{i+\frac{1}{2},j+\frac{1}{2}} = \begin{bmatrix} \frac{1}{2\Delta x}(\xi_{i+1,j} + \xi_{i+1,j+1} - \xi_{i,j} - \xi_{i,j+1})^T \\ \frac{1}{2\Delta y}(\xi_{i,j+1} + \xi_{i+1,j+1} - \xi_{i,j} - \xi_{i+1,j})^T \end{bmatrix}^T, \tag{4}$$

which can be used to find the density and the Cauchy stress tensor at control volume boundaries to evaluate the surface integral term in equation (2).

Combining the mentioned numerical approximations by introducing the differential operators R_1 and R_2, and neglecting the boundary conditions, yield the following spacial discretization of the governing equations

$$\begin{aligned} \frac{dv_h}{dt} &= R_1(v_h, \xi_h) \\ \frac{d\xi_h}{dt} &= R_2(v_h, \xi_h), \end{aligned} \tag{5}$$

where v_h and ξ_h are the vector of the nodal velocity and reference map values, respectively.

2.3. *Boundary Conditions*

Applying the boundary conditions are quite different depending on the condition type, and are as follows.

2.3.1. *Clamped Boundaries*

Assuming that the grid domain is large enough that the nodes on its boundaries are more than a cell away from Ω, every vertex can be denoted either as active, ghost, or inactive. Vertices inside Ω which are sufficiently far from $\partial\Omega$, will be denoted as active. Here, I will use a threshold of $\min(\Delta x/2, \Delta y/2)$ as a metric to determine being sufficiently far. Then, any vertex which shares a cell with an active vertex, will be denoted as a ghost. All the other vertices are considered as inactive. The idea is to only solve for the unknowns at the active vertices, and then extrapolate the solution from the active vertices and the boundary conditions to find the solution values at the ghost vertices. To achieve this, the ray from every ghost vertex to the nearest boundary point is followed until it intersects the first edge of the mesh whose vertices are both active. The values of the solution at the nearest boundary point, and the vertices of the mentioned edge are then used for extrapolation. For an annular domain, figure 2 shows the ghost vertices, the rays cast from them to the closest boundary point, and the first internal edge that they intersect.

2.3.2. *Free Boundaries*

Handling problems with free boundaries has proved to be more challenging. For these problems, the velocity v is only solved for at vertices with a non-zero mass m_{ij}, and is then extrapolated to a larger vicinity of the solid or the whole domain. The extrapolation procedure is taken from Bridson [6] where the velocity at the closest active vertex (with respect to the Manhattan norm) is used as the extrapolated value. Also, the stress terms Tn have to be set to zero for the control volume boundaries that lie outside the solid. Unlike the velocity, the reference map ξ is solved for throughout the whole domain.

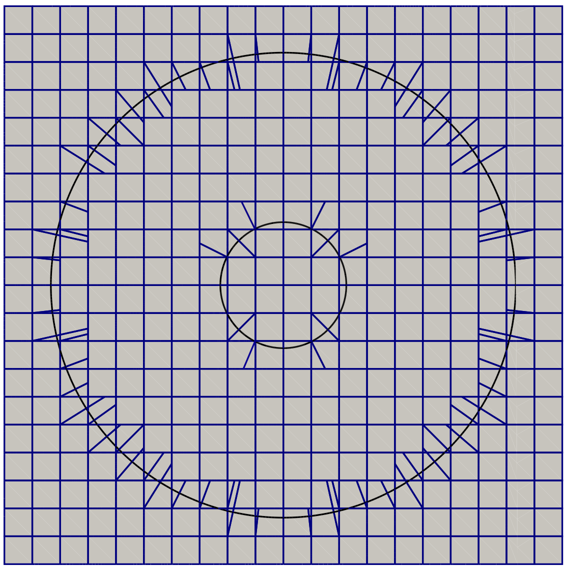
2.3.3. *Time Advance*

Time advance is finding the vector of nodal values at time level $n + 1$, given the vector of nodal values at time level n. The method used in this work is that of the explicit Euler:

1. Update the velocity at vertices with non-zero mass: $v_h^{n+1} = v_h^n + \Delta t R_1(v_h^n, \xi_h^n)$.
2. Extrapolate the velocity.
3. Update the reference map: $\xi_h^{n+1} = \xi_h^n + \Delta t R_2(v_h^{n+1}, \xi_h^n)$.

where the superscript $(.)^n$ is used to denote the corresponding time level of a solution vector.

For 1-D problems, the time advance scheme can be further modified to handle collisions with fixed barriers. Before updating the velocity, a check is made to see whether the solid penetrates a barrier (shown


(a)

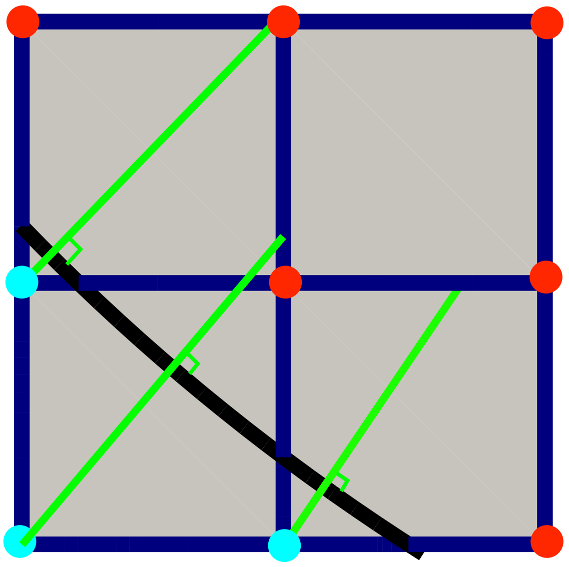

(b)

Figure 2: The rays that are used for extrapolating the solution values at ghost vertices for clamped boundaries: (a) all the rays (b) a closeup

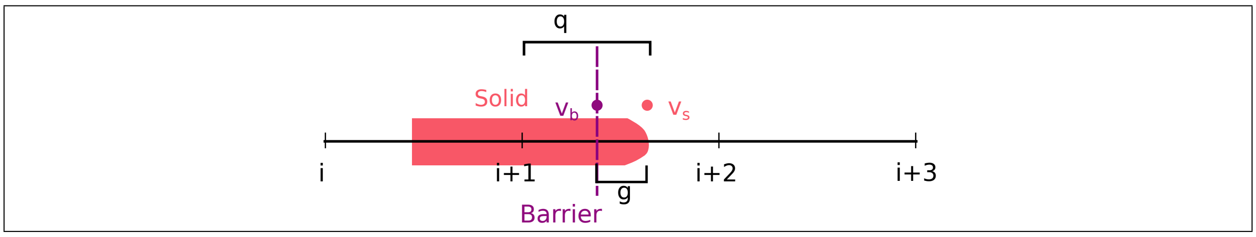


Figure 3: Schematic of 1-D collision

in figure 3). Subsequently, the rate of the change of the penetration volume, $\dot{g}$, is enforced to be negative during the time advance. This rate of change can be written as

$$\dot{g} = v_s - v_b = \underbrace{v_{i+1}}_{\text{active}}(1-q) + \underbrace{v_{i+2}}_{\text{ghost}}(q) - \underbrace{v_b}_{\text{barrier}} = v_{i-1} - v_b,$$

where v_s is the velocity of the solid at its endpoint, and v_b is the velocity of the barrier. Other quantities are depicted in figure 3. Since the constraint is linear, it can further be written in the compact form $Jv_h < c$. It can then be incorporated in the time advance scheme by solving the quadratic program

$$v_h^{n+1} = \arg\min_{v_h}(\frac{1}{2}{v_h}^T M v_h + R_{1*}^T v_h) \text{ subject to } Jv_h \leq c,$$

where M is a diagonal matrix containing the m_i values, and $R_{1*} = M{v_h}^n + \Delta t R_1({v_h}^n, {\xi_h}^n)$.

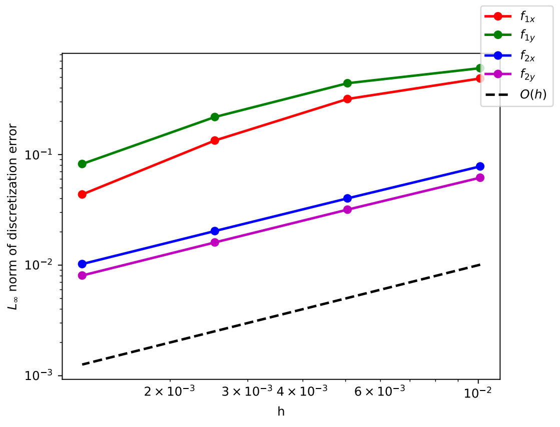


Figure 4: Truncation error of the finite-difference operators with mesh refinement

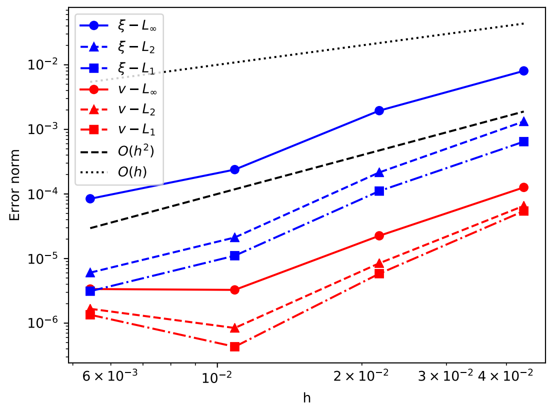


Figure 5: Discretization error in the solution of the annular washer problem with mesh refinement

3. Results

3.1. Testing the Finite Difference Operators

To test the implementation of the finite-difference operators, the manufactured solution,

$$v = \begin{bmatrix} \cos(x)\cos(y) \\ sin(2y)\sin(x) \end{bmatrix} \qquad \xi = \begin{bmatrix} \cos(2x)\cos(y) \\ \sin(2y)\cos(x) \end{bmatrix}, \tag{6}$$

is used to construct source terms f_1 and f_2,

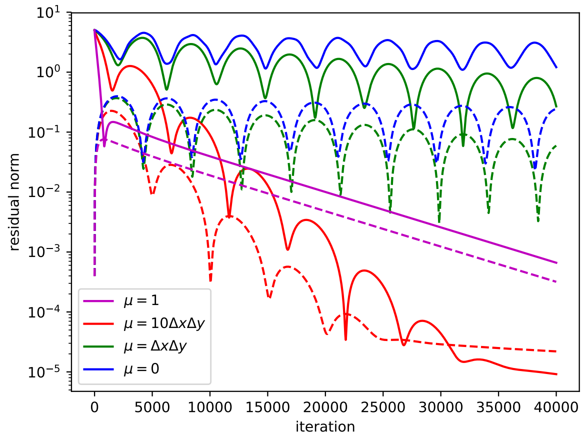
$$\begin{aligned} f_1 &= -v \cdot \nabla v + \frac{1}{\rho}\nabla \cdot T \\ f_2 &= -v \cdot \nabla \xi, \end{aligned} \tag{7}$$

which can be evaluated both analytically, e.g., Matlab's symbolic engine, and using the mentioned finite-difference operators. The norm of the difference of the values obtained using these two methods should approach zero with a rate of $O(h)$, where h is the mesh length scale. Figure 4 shows the L_∞ norm of the mentioned difference for a series of refined meshes starting from 11×9 on the domain $[0, 1] \times [0, 1]$. The desired order of accuracy is attained verifying the implementation of this part of the project. For simplicity, the source terms are not evaluated at the grid boundaries. Also, a very simple stress model $T = [1, 5; 5, 9](\nabla\xi)^{-1}$ has been used.

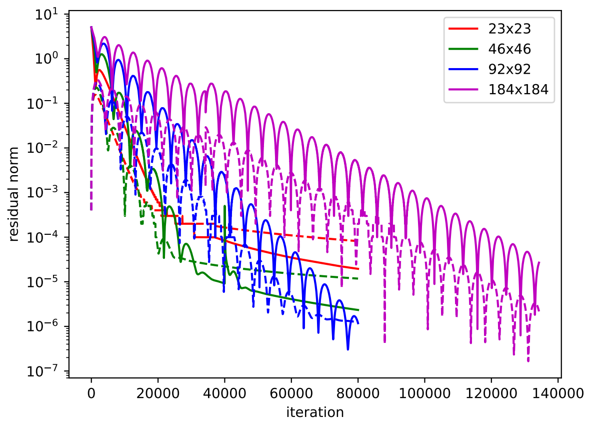
3.2. A 2-D Steady State Problem – Annular Washer Shear

In this problem which is taken from Kamrin et al. [4], the outer boundary of a annular shear is rotated while its inner boundary is held fixed. The solid and the mesh resemble those of figure 2-a. The reference and physical solid domains are both an annulus with inner radius of 0.1 and an outer radius of 0.4. The outer boundary is rotated $\frac{\pi}{6}$ radians counter-clockwise, yielding the boundary conditions

$$\begin{aligned} \xi_x &= r\cos(\theta) \qquad & \xi_y &= r\sin(\theta) \qquad & r &= 0.1 \\ \xi_x &= r\cos(\theta - \pi/6) \qquad & \xi_y &= r\sin(\theta - \pi/6) \qquad & r &= 0.4, \end{aligned}$$



(a)



(b)

Figure 6: Solver convergence for the annular washer problem: (a) effect of damping (b) effect of mesh size

where (r, θ) is the polar coordinates of a point on the boundary. The initial condition for v is zero everywhere, while ξ_0 is evaluated by linearly interpolating the boundary conditions. The Levinson-Burgess model is used as the stress constitutive relation

$$T = f_1(I_3)B + f_2(I_1, I_3)I. \tag{8}$$

Here, $B = (\nabla\xi)^{-1}(\nabla\xi)^{-T}$, I is the identity matrix, $I_1 = \mathrm{tr}(B)$ and $I_3 = \det(B)$ are the first and third invariants of B, respectively, and

$$f_1 = \left(3 + \frac{1}{I_3}\right)\left(4\sqrt{I_3}\right)^{-1} \qquad f_2 = \left(\sqrt{I_3}\left(\frac{5}{6} + \frac{1 - I_1}{4I_3^2}\right) - \frac{4}{3}\right).$$

The problem is solved on a series of nested meshes, and the error in the numerical solution is evaluated by comparison to its exact counterpart $v = 0$ and $\xi = (A - B/r^2)e_\theta$. As desired, figure 5 shows that the discretization error is reduced with mesh refinement.

The effect of mesh size and damping on the solver convergence are also studied. Figure 6-a shows the effect the damping term for a mesh size of 43×43 and $\Delta t = 0.1\Delta x$. It is observed that the convergence is considerably slow with no or small damping coefficients μ, while $\mu = 10\Delta x\Delta y$ seems to accelerate the convergence the most. Figure 6-b shows the solver convergence with $\Delta t = 0.1\Delta x$ and $\mu = 10\Delta x\Delta y$ over different mesh sizes. Not surprisingly, the finer the mesh, the more iterations are required.

3.3. A 1-D Transient Problem – Oscillating Solid

In this problem, a 1-D solid is compressed to 0.1 of its original volume and is then released with zero initial velocity, as shown in figure 7. The simple stress model $T = 1/\xi_x - 1$ is used, along with a mesh size of 200, and a time-step of $\Delta t = 0.1\Delta x$. The solution at various times is shown in figure 8. As expected, the solid keeps oscillating due to lack of damping.

3.4. A 1-D Transient Problem – Collisions

In this problem, a 1-D solid is placed between two fixed barriers (shown in figure 9). By setting the initial velocity to the uniform value of one, and using no initial deformation, i.e., $\xi(x, t = 0) = x$, the solid

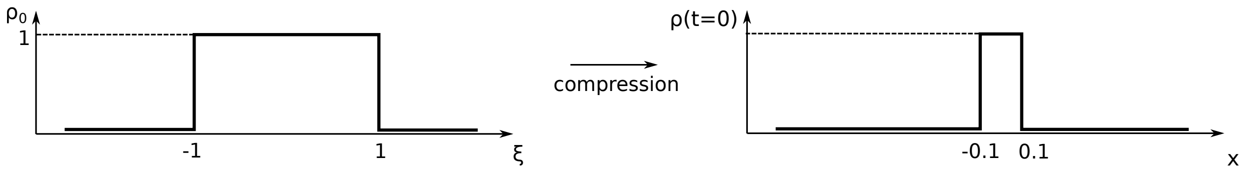


Figure 7: Schematic of the 1-D oscillating solid problem.

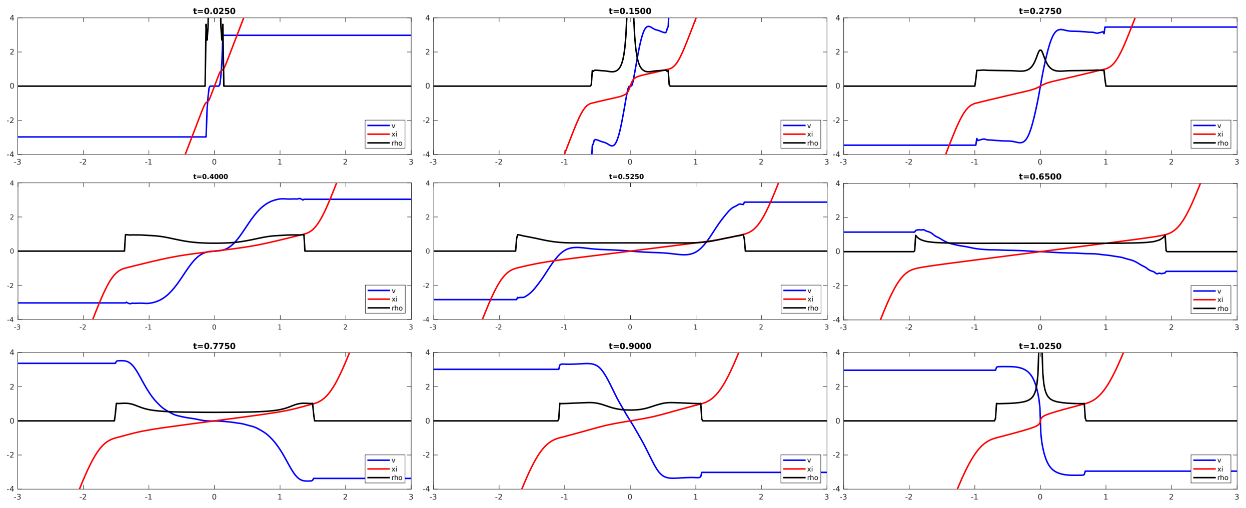


Figure 8: Solution of the 1-D oscillating solid problem at various times.

will keep colliding with and bouncing off the barriers. This is consistent with the obtained solution shown in figure 10.

In this problem, the effect of mesh size on the solution is also studied by solving the same problem on mesh sizes 200, 400, and 800. Figure 11 shows the density profile at various times when the solid is bouncing off the right barrier on the finest mesh. Interestingly, the shape of the density profile seems to be the same between the different solutions. On the other hand, the coarse mesh solutions accumulate considerable amounts of phase lag over time.

3.5. *A 2-D Transient Problem – Uniform Translation*

My attempts in dealing with 2-D transient problems have not been as successful in the time frame of the project. The only problem I have been able to touch is that of a circular solid with a uniform initial velocity, no gravity, and no initial deformation. As one would expect, the exact solution of this problem would only include uniform translation according to the initial velocity, independent of the stress model. A circular solid defined with the density field, $\rho_0(\xi) = 1$ for $||\xi||_2^2 \leq 1$ and 0 otherwise, is placed in the domain $[-2, 6] \times [-1.25, 1.25]$ with a uniform initial velocity $v_0 = 1$, and an initial reference map distribution of $\xi = x$. To prevent additional hidden bugs affecting the results, the stress term in the equations has been dropped, i.e., $T = 0$.

Figure 12 shows the solutions obtained from a grid size of 160×50 and a time-step of $\Delta t = 0.01$. The velocity extrapolation was only performed for vertices located within a topological distance of 10 from the object. As expected, the velocity field stays constant within the extrapolation range. The reference map, however, does not seem to preserve its linear profile within the solid, and builds up an increasing slope at the front of the object. This in turn results in a non-constant density profile with a growing kink at the front

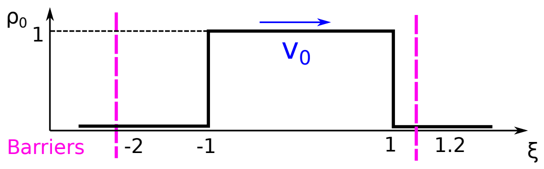


Figure 9: Schematic of the 1-D collision problem.

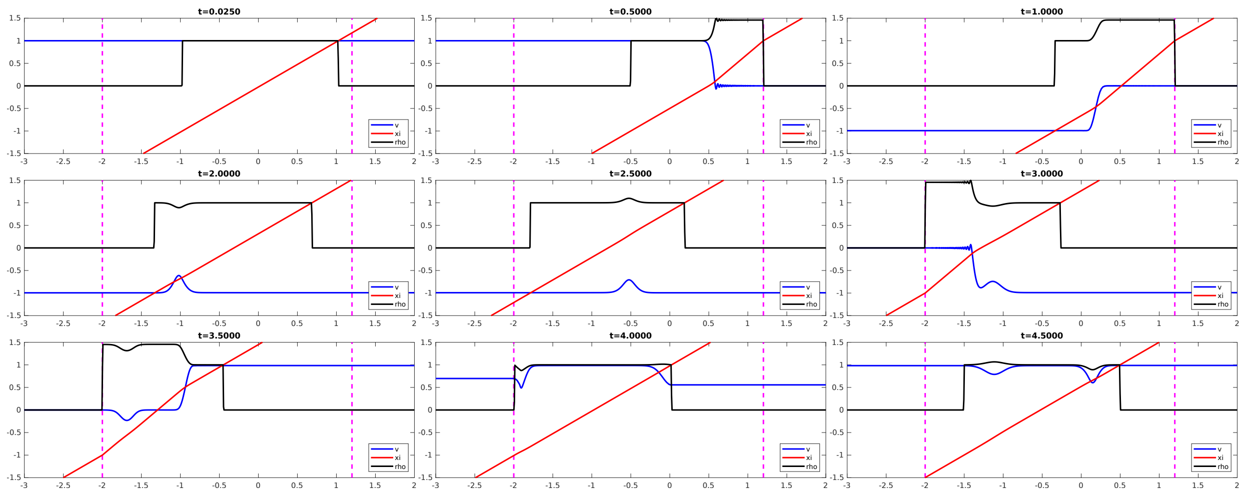


Figure 10: Solution of the 1-D collision problem at various times.

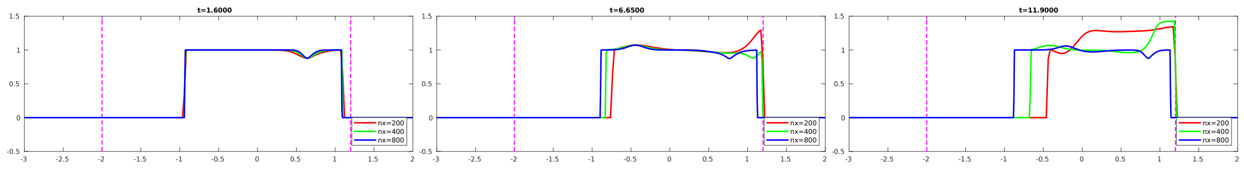


Figure 11: Effect of mesh size on the solution to the 1-D collision problem. The density is shown at various times when the object on the finest mesh has collided with the right barrier and is being bounced back. As more time passes, the solution on the coarse meshes accumulate larger amounts of phase lag.

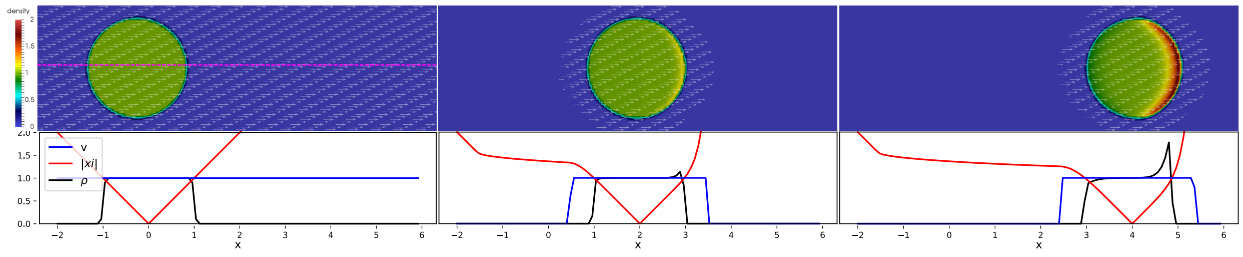


Figure 12: Snapshots of the solution for the uniform translation problem of a 2-D circular solid. The figures from left to right show $t = 0, 0.2$, and 0.4, respectively. The upper figures show the density and velocity on the whole domain, while the lower ones are the plots of the quantities on the $x = 0$ line.

of the object. As a possible remedy, the velocity can be extrapolated throughout the whole domain, rather than just the vicinity of the object.

In addition to the previous bug, I have not been able to obtain any results for a non-zero stress model. In such a case, the velocity becomes oscillatory after some time, and the solution blows up. The solver cannot be further developed until the culprit of these errors are found and eliminated.

4. Conclusions

In this work, a solution method was implemented for the simulation of large elastic deformations based on an Eulerian formulation. Various problems were solved in one and two dimensions, and the effects of various parameters were studied.

References

[1] J. Donea, S. Giuliani, J.-P. Halleux, An arbitrary lagrangian-eulerian finite element method for transient dynamic fluid-structure interactions, Computer methods in applied mechanics and engineering 33 (1-3) (1982) 689–723.
[2] Y. Fan, J. Litven, D. I. Levin, D. K. Pai, Eulerian-on-lagrangian simulation, ACM Transactions on Graphics (TOG) 32 (3) (2013) 22.
[3] D. I. Levin, J. Litven, G. L. Jones, S. Sueda, D. K. Pai, Eulerian solid simulation with contact, in: ACM Transactions on Graphics (TOG), Vol. 30, ACM, 2011, p. 36.
[4] K. Kamrin, C. H. Rycroft, J.-C. Nave, Reference map technique for finite-strain elasticity and fluid–solid interaction, Journal of the Mechanics and Physics of Solids 60 (11) (2012) 1952–1969.
[5] Y. Teng, D. I. Levin, T. Kim, Eulerian solid-fluid coupling, ACM Transactions on Graphics (TOG) 35 (6) (2016) 200.
[6] R. Bridson, Fluid simulation for computer graphics, CRC Press, 2015.