<table>
<thead>
<tr>
<th><strong>Title</strong></th>
<th>Development of a Finite Volume Based Structural Solver for Large Rotation of Non-Orthogonal Meshes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Authors(s)</strong></td>
<td>Cardiff, Philip; Ivankovic, Alojz; Karac, Aleksandar; et al.</td>
</tr>
<tr>
<td><strong>Publication date</strong></td>
<td>2012</td>
</tr>
<tr>
<td><strong>Conference details</strong></td>
<td>7th OpenFOAM Workshop, Technische Universität Darmstadt Germany 25-28 June 2012 Darmstadt</td>
</tr>
<tr>
<td><strong>Publisher</strong></td>
<td>OpenFOAM Workshop</td>
</tr>
<tr>
<td><strong>Item record/more information</strong></td>
<td><a href="http://hdl.handle.net/10197/4768">http://hdl.handle.net/10197/4768</a></td>
</tr>
</tbody>
</table>
Development of a Finite Volume Based Structural Solver for Large Rotation of Non-Orthogonal Meshes

Philip Cardiff

philip.cardiff@ucd.ie
philip.cardiff@gmail.com

Alojz Ivanković
Aleksandar Karač
Željko Tuković
Presentation Outline

Background & Motivation
Mathematical Model
Implementation in OpenFOAM
Numerical Test Cases
Mesh Movement
Summary & Conclusions
Background & Motivation
Stance with Active Hill Muscle
*Gluteus Medius*
Half Gait cycle  

heel-strike to push-off

Ground Reaction (N)
949.4014
800
600
400
200
65.912

Time: 0.000 s
Mathematical Model
Implementation in OpenFOAM
Finite Volume Stress Analysis

Governing Equation

\[
\frac{\delta}{\delta t} \int_{\Omega} \rho \dot{v} \, d\Omega = \int_{\Gamma} \mathbf{\sigma} \cdot d\Gamma + \int_{\Omega} \rho \mathbf{b} \, d\Omega
\]

Constitutive Relation

\[
\mathbf{S} = 2\mu \mathbf{E} + \lambda \text{tr}(\mathbf{E}) \mathbf{I}
\]

Strain Measure

\[
\mathbf{E} = \frac{1}{2} \left( \nabla \mathbf{u} + \nabla \mathbf{u}^T + \nabla \mathbf{u}^T \cdot \nabla \mathbf{u} \right)
\]

2nd Piola-Kirchhoff Stress Tensor

St. Venant-Kirchhoff Hyperelastic Relation

Lagrangian Green Strain Tensor

Conservation of Linear Momentum
The presented mathematical models can be arranged in the general form:

**1.5.2 Equations**

1.1, with the computational node overlap and fill the space completely. A typical control volume cell is shown in Figure.

In the solution domain, the discretisation comprises the discretisation of space. The total specified simulation time is divided into a finite number of time increments, \( t = 1, 2, \ldots, N \).

Discretisation of the solution domain comprises the discretisation of time and the discretisation of the governing equations. The following sections outline the adopted finite volume discretisation procedure.

The mathematical models of the governing equations presented in the preceding section are now discretised using the cell-centred finite volume method. It is important to note that the discretisation process provides a discrete approximate version of the previously presented exact integral relations.

The mathematical models of the governing equations are

\[
\frac{\delta}{\delta t} \int_{\Omega_u} \rho_u \frac{d\mathbf{u}}{dt} \, d\Omega_u = \frac{1}{\delta t} \int_{\Gamma_u} [2\mu \delta \mathbf{E}_u + \lambda tr(\delta \mathbf{E}_u) \mathbf{I}] \cdot d\Gamma_u + \\
\frac{1}{\delta t} \int_{\Gamma_u} \left[ \mathbf{S}_u \cdot \delta \mathbf{F}_u^T + \delta \mathbf{S}_u \cdot \delta \mathbf{F}_u^T \right] \cdot d\Gamma_u + \int_{\Omega_u} \rho_u \frac{\delta \mathbf{b}}{\delta t} \, d\Omega_u
\]

**Decompose into Implicit & Explicit Terms**

\[
\frac{d}{dt} \int_{\Omega_\phi} \rho_\phi \frac{d\phi}{dt} \, d\Omega_\phi = \int_{\Gamma_\phi} (2\mu + \lambda) \nabla \phi \cdot d\Gamma_\phi + \int_{\Gamma_\phi} Q_\Gamma \cdot d\Gamma_\phi + \int_{\Omega_\phi} Q_\Omega \, d\Omega_\phi
\]
Implementation in OpenFOAM-1.6-ext

elasticNonLinSolidFoam

fvVectorMatrix DUEqn

( 
  fvm::d2dt2(rho,DU)
  ==
  fvm::laplacian(2*mu + lambda, DU)
  + fvc::div
      ( 
        mu*gradDU.T() + lambda*(I*tr(gradDU))
        - (mu + lambda)*gradDU
        + mu*(gradDU & gradDU.T())
        + 0.5*lambda*(gradDU && gradDU)*I
        + ((sigma + DSigma) & DF.T())
      )
);
Discretisation & Non-Orthogonal Correction

\[
\int_{\Gamma_{\phi}} D (\nabla \phi) \cdot d\Gamma_{\phi} = \sum_{f=1}^{F} \int_{\Gamma_{f}} D_{f} (\nabla \phi)_{f} \cdot d\Gamma_{f}
\]

\[
\approx \sum_{f=1}^{F} D_{f} \left[ |\Delta_{f}| \frac{\phi_{N} - \phi_{P}}{|d_{f}|} + \left\{ \text{Correction} \right\} \right] |\Gamma_{f}|
\]

Implementation of Custom Boundary Conditions with Non-Orthogonal Correction

- **fixedDisplacement**
- **solidTraction**
- **solidSymmetryPlane**
- **solidDirectionMixed**
Numerical Test Cases
Steady-State Rotation of a Sphere
Rotation of a Sphere

During quasi-static rotation, a body experiences zero stress. To examine that the current updated Lagrangian procedure correctly predicts zero stress, a sphere, radius = 0.1 m, subjected to a quasi-static rotation is considered. The mesh of the sphere, created in commercial meshing software ANSYS ICEM CFD, contains 1,664 hexahedral cells, some of which are moderately non-orthogonal, as shown in Figure 1.6. A Young’s modulus of 200 GPa, and a Poisson’s ratio of 0.3 has been employed.

A rotational displacement of 90° about the $z$-axis is applied to the boundary surface of the sphere, in increments of 1°. The displacement increment for boundary face $f$ is given by:

$$
\delta u_f = \begin{pmatrix}
\cos \theta & -\sin \theta & 0 \\
\sin \theta & \cos \theta & 0 \\
0 & 0 & 1
\end{pmatrix} \cdot C_f - C_f
$$

where $\theta$ is the increment of rotation, and $C_f$ is the positional vector of the boundary face centre.

As the small strain approach neglects the higher order strain terms, unphysical stresses are predicted when there are large rotations. To check this, the current rotation problem has additionally been simulated using the developed small strain total Lagrangian solver. For 1° of rotation, the small strain solver predicts the uniform stress.
Rotation of a Sphere

Initial Attempt with fixedValue Boundary (no boundary non-orthogonal correction)
Rotation of a Sphere

After 1° Rotation

fixedValue

fixedDisplacement
Tight solution tolerance required to ensure correct zero stresses.
Mesh movement becomes skewed

Rotation of a Sphere

Standard inverse-distance interpolation

Time: 0 s
Mesh Movement
Interpolation from cell centres to vertices
Interpolation

Determine value at P from values at N

Figure 1.4: Internal Vertex, P, Surrounded by Cell Centres, N

Consider an internal vertex, \( P \), in an unstructured mesh, as shown in Figure 1.4. The value of variable \( \phi \) is known at the \( N \) neighbour cell centres, \( N_i \). It is required to calculate a good approximation of \( \phi \) at the vertex \( P \).

We want to find the value of \( \phi \) at point \( P \). We know the value of \( \phi \) at all neighbours \( N \). \( P \) is at a vertex, \( N \) are at cell centres. The cells are of arbitrary shape and orientation, and \( P \) is connected to an arbitrary number of cells but at least one. We fit a plane to the values of \( \phi \) at the neighbours \( N \) surrounding \( P \). A plane has the general equation:

\[
LS(x, y, z) = ax + by + cz + d
\]

We use the least squares method to determine the best fit values of \( a, b, c \) and \( d \).

The error at \( N \), \( e_N \), between \( N \) and the plane \( LS \) is:

\[
e_N = LS(x, y, z) - (ax_N + by_N + cz_N + d)
\]

Where \( x_N, y_N \) and \( z_N \) are the 3D spatial coordinates of the point \( N \).
Inverse Distance Interpolation

The inverse distance interpolation method essentially calculates the value of a function at a point by finding a weighted average of the function values at surrounding cell centres. The weighting factor is given by:

\[ \omega_{PN_i} = \frac{1}{|\mathbf{r}_P - \mathbf{r}_{N_i}|} \]

where \( \mathbf{r}_P \) is the positional vector of the point at which the function is to be evaluated, and \( \mathbf{r}_{N_i} \) is the positional vector of the \( i \)th surrounding cell centre. The value of \( \omega_{PN_i} \) ensures that the interpolation weights are inversely proportional to the distance from the point to each cell centre.

The formula for the inverse distance interpolation is:

\[ \phi_P = \frac{\sum_{i=1}^{N} \omega_{PN_i} \phi_{N_i}}{\sum_{i=1}^{N} \omega_{PN_i}} \]

where \( \phi_P \) is the interpolated function value at the point, \( \phi_{N_i} \) is the function value at the \( i \)th surrounding cell centre, and \( \omega_{PN_i} \) is the weighting factor calculated as above. The summation over all surrounding cell centres \( N \) ensures that the interpolated value is a weighted average of the function values at the cell centres, with the weights proportional to the inverse of the distance to the point.
Least Squares Interpolation

leastSquaresVolPointInterpolation class

\[ \phi(x, y, z) = ax + by + cz + d \]

\[ e_{Ni} = \phi(x_{Ni}, y_{Ni}, z_{Ni}) - \phi_{Ni} \]

\[ = ax_{Ni} + by_{Ni} + cz_{Ni} + d - \phi_{Ni} \]

\[ J = \sum_{i=1}^{N} \left[ ax_{Ni} + by_{Ni} + cz_{Ni} + d - \phi_{Ni} \right]^{2} \]
Least Squares Interpolation

leastSquaresVolPointInterpolation class

\[
\begin{bmatrix}
\sum_{i=1}^{N} x_{Ni}^2 & \sum_{i=1}^{N} x_{Ni} y_{Ni} & \sum_{i=1}^{N} x_{Ni} z_{Ni} & \sum_{i=1}^{N} x_{Ni} \\
\sum_{i=1}^{N} x_{Ni} y_{Ni} & \sum_{i=1}^{N} y_{Ni}^2 & \sum_{i=1}^{N} y_{Ni} z_{Ni} & \sum_{i=1}^{N} y_{Ni} \\
\sum_{i=1}^{N} x_{Ni} z_{Ni} & \sum_{i=1}^{N} y_{Ni} z_{Ni} & \sum_{i=1}^{N} z_{Ni}^2 & \sum_{i=1}^{N} z_{Ni} \\
\sum_{i=1}^{N} x_{Ni} & \sum_{i=1}^{N} y_{Ni} & \sum_{i=1}^{N} z_{Ni} & N
\end{bmatrix}
\begin{bmatrix}
a \\
b \\
c \\
d
\end{bmatrix}
= 
\begin{bmatrix}
\sum_{i=1}^{N} \phi_{Ni} x_{Ni} \\
\sum_{i=1}^{N} \phi_{Ni} y_{Ni} \\
\sum_{i=1}^{N} \phi_{Ni} z_{Ni} \\
\sum_{i=1}^{N} \phi_{Ni}
\end{bmatrix}
\]

Solve this linear system using Gaussian elimination implemented in simpleMatrix class
Special treatment of boundaries

Coupled boundaries uses neighbour cell centre values, and empty patch values are included so A is not singular
Comparison of Inverse Distance & Least Squares Methods
Comparison of Inverse Distance & Least Squares Methods

Inverse Distance

Least Squares
Comparison of Inverse-Distance & Least Squares Methods

Inverse Distance

Least Squares
Comparison of Inverse-Distance & Least Squares Methods

Rotation: 0

Inverse Distance

Least Squares
An updated Lagrangian finite volume structural solver suitable for large rotations has been developed.

Boundary non-orthogonal correction is imperative.

Standard inverse distance interpolation shows poor performance on non-orthogonal grids.

Therefore, a least squares approach has been developed.
Development of a Finite Volume Based Structural Solver for Large Rotation of Non-Orthogonal Meshes

Philip Cardiff

Thanks

Alojz Ivanković
Aleksandar Karač
Željko Tuković

philip.cardiff@ucd.ie
philip.cardiff@gmail.com