Chrono::FSI
(Support for Fluid-Solid Interaction)

Arman Pazouki
Simulation Based Engineering Laboratory
University of Wisconsin-Madison
Flow cytometry using microfluidic techniques

- Fluorescence and laser-beam cell sorting
  - Limited particle size, $a < 50 \mu m$
  - Unknown effect of external field on cell viability

- Purification of 3D micro-tissues and cell aggregates
  - Finite size particles, $a \approx 25..500 \mu m$
Simulation of dense suspensions

“Dense” suspension
- Finite size particles (rigid bodies) interaction
  - Drafting, Kissing, and Tumbling (DKT)
- Short range interactions
  - Lubrication and collision

Flow characteristics
- Particle Reynolds number ≤ 1.0
- Channel Reynolds number: 66
  - Channel Dimension: (1.1, 1.0, 1.0) m
  - Volumetric concentration: 40%

Computational aspects
- 23,000 rigid ellipsoids: (1.5, 1.5, 2.0) cm
- 2,000,000 SPH markers.
- Simulation performed on a single GPU, NVIDIA GTX 480
  - 3.2 seconds of dynamics
  - 72 hrs to complete.

Animation shows the channel mid-section
Flow in porous media

- **Example applications**
  - Oil Recovery
  - Biology
    - Diffusion of macro-molecules within tissues
    - Blood flow through muscles
Interacting rigid and flexible objects in channel flow

Fluid:
\[ \rho = 1000 \text{ kg/m}^3 \]
\[ \mu = 1 \text{ N s/m}^2 \]
\[ (l_x, l_y, l_z) = (1.4, 1, 1) \text{ m} \]
\[ Re = 45 \]

Ellipsoids:
\[ \rho_s = 1000 \text{ kg/m}^3 \]
\[ (a_1, a_2, a_3) = (2.25, 2.25, 3) \text{ cm} \]
\[ N_r = 2000 \]
\[ Re_p = 2 \]

Beams:
\[ \rho_s = 1000 \text{ kg/m}^3 \]
\[ E = 0.2 \text{ MPa} \]
\[ a = 1.5 \text{ cm} \]
\[ l = 64 \text{ cm} \]
\[ N_f = 40 \]
\[ n_e = 4 \]
The Lagrangian-Lagrangian angle

- **Fluid**: Smoothed Particle Hydrodynamics (SPH)
- **Solid**
  - 3D rigid body dynamics (CM position, rigid rotation)
  - Absolute Nodal Coordinate Formulation (ANCF) for flexible bodies (nodes location and slope)

- Lagrangian-Lagrangian approach attractive since:
  - Consistent with Lagrangian tracking of discrete solid components
  - Straightforward simulation of free surface flows prevalent in target applications
  - Maps well to the GPU parallel computing model
Meshless Discretization using Smoothed Particle Hydrodynamics

- **Continuity:**
  \[
  \frac{d\rho_a}{dt} = \rho_a \sum_b \frac{m_b}{\rho_b} (v_a - v_b) \cdot \nabla a W_{ab}
  \]

- **Momentum (Navier-Stokes):**
  \[
  \frac{dv_a}{dt} = -\sum_b m_b \left[ \left( \frac{v_a}{\rho_a^2} + \frac{v_b}{\rho_b^2} \right) \nabla a W_{ab} - \frac{(\mu_a + \mu_b)x_{ab} \cdot \nabla a W_{ab}}{\rho_{ab}^2(x_{ab}^2 + \xi h_{ab}^2)} v_{ab} \right] + f_a
  \]

- **Lagrangian Kinematics:**
  \[
  \frac{dx_a}{dt} = v_a
  \]

- **Weakly Compressible model**
  \[
  p = \frac{cs^2 p_0}{\gamma} \left\{ \left( \frac{\rho}{\rho_0} \right)^\gamma - 1 \right\}
  \]

- **XSPH**
  \[
  v_x = v_a + \Delta v_a, \Delta v_a = \zeta \sum_b m_b (v_b - v_a) W_{ab}
  \]

- **Shepard Filtering**
  \[
  \rho_a = \sum_b m_b W_{ab}
  \]

GPU-based programming: How is it relevant?

- Steady increase in FLOPs rate and number of processors
- Memory hierarchy, capacity, and access patterns
  - Bandwidth: 192 GB/s global memory, 2TB/s Shared memory

How can it be best leveraged?
- Single Instruction Multiple Data (SIMD)
- Memory access consideration

What does it require?
- HPC model, logistics and syntax
- Tradeoff between memory and process
SIMD and spatial subdivision

SIMD model

- Loops $\rightarrow$ parallel threads

Proximity calculation

- Increased process, decreased memory
- Interactions are processed per marker
Parallelization

- **thrust::reduce_by_key** to reduce surface reaction forces and torques on to nodal values

- Custom kernels to update solid objects

- Fine grain parallelization
  - Position
  - Rotation
  - Velocity
  - Angular velocity
  - ...

Chrono Fluid-Solid Interaction
Scaling analysis (all together, table)

<table>
<thead>
<tr>
<th>$N_m \times 10^6$</th>
<th>0.08</th>
<th>0.16</th>
<th>0.29</th>
<th>0.63</th>
<th>0.95</th>
<th>1.54</th>
<th>2.50</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N_r \times 10^3$</td>
<td>0.17</td>
<td>0.52</td>
<td>1.12</td>
<td>4.48</td>
<td>7.84</td>
<td>14.56</td>
<td>24.64</td>
</tr>
<tr>
<td>$N_f \times 10^3$</td>
<td>0.16</td>
<td>0.42</td>
<td>0.84</td>
<td>2.10</td>
<td>3.36</td>
<td>5.88</td>
<td>9.66</td>
</tr>
<tr>
<td>$t (\text{ms})$</td>
<td>45</td>
<td>74</td>
<td>120</td>
<td>230</td>
<td>343</td>
<td>522</td>
<td>820</td>
</tr>
</tbody>
</table>

$N_m \approx 3.0 \times 10^6$, $N_f = 0$

<table>
<thead>
<tr>
<th>$N_r$</th>
<th>0</th>
<th>36</th>
<th>120</th>
<th>480</th>
<th>1800</th>
<th>8400</th>
<th>33600</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t (\text{ms})$</td>
<td>906</td>
<td>919</td>
<td>923</td>
<td>925</td>
<td>926</td>
<td>926</td>
<td>921</td>
</tr>
</tbody>
</table>

$N_m \approx 3.0 \times 10^6$, $N_r = 0$

<table>
<thead>
<tr>
<th>$N_f$</th>
<th>0</th>
<th>45</th>
<th>140</th>
<th>440</th>
<th>1152</th>
<th>2100</th>
<th>4704</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\tau = 10$</td>
<td>$t (\text{ms})$</td>
<td>906</td>
<td>923</td>
<td>928</td>
<td>916</td>
<td>960</td>
<td>950</td>
</tr>
<tr>
<td>$\tau = 50$</td>
<td>$t (\text{ms})$</td>
<td>906</td>
<td>973</td>
<td>978</td>
<td>965</td>
<td>1066</td>
<td>1060</td>
</tr>
</tbody>
</table>
Constrained rigid body dynamics

- DVI Approach
- Bilateral Constraints
- Enhanced solver
- Robust MBD simulation
- Validations
- ....
chrono-vehicle
Integration Scheme using a Heterogeneous Computing Approach

<table>
<thead>
<tr>
<th>Fluid (GPU)</th>
<th>Chrono-Parallel (CPU)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fluid (t)</td>
<td>MBD (t)</td>
</tr>
<tr>
<td>Force SPH</td>
<td>Step Chrono</td>
</tr>
<tr>
<td>Add Forces</td>
<td>t - t + 0.5 * dt</td>
</tr>
<tr>
<td>Step Fluid</td>
<td>Copy Host To Device</td>
</tr>
<tr>
<td>t - t + 0.5 * dt</td>
<td>Forces</td>
</tr>
<tr>
<td>Apply Fluid BC</td>
<td>Update Position and</td>
</tr>
<tr>
<td></td>
<td>Velocity at the</td>
</tr>
<tr>
<td></td>
<td>interface</td>
</tr>
<tr>
<td>Force SPH</td>
<td>Step Chrono</td>
</tr>
<tr>
<td>Add Forces</td>
<td>t - t - dt</td>
</tr>
<tr>
<td>Step Fluid</td>
<td>Copy D2H</td>
</tr>
<tr>
<td>t - t - dt</td>
<td>Position/Velocity</td>
</tr>
<tr>
<td></td>
<td>Update Position and</td>
</tr>
<tr>
<td></td>
<td>Velocity at the</td>
</tr>
<tr>
<td></td>
<td>interface</td>
</tr>
<tr>
<td>Fluid (t - dt)</td>
<td>MBD (t + dt)</td>
</tr>
</tbody>
</table>

2/8/2016
Chrono Fluid-Solid Interaction
Vehicle mobility in fording
Point cloud approach

Advantage:

• Resolution independent

• Most of the data stays on the GPU
  • Less data transfer

• Experimentally and numerically validated for particle suspension. Resolution independence is demonstrated.
• Chrono Representation

• Fluid system Representation
Improving wall boundary condition

- No slip condition is only marginally improved
- The main advantage is in improving the no-penetration condition.
  \[ \mathbf{v}_a = 2\mathbf{v}_a^p - \tilde{\mathbf{v}}_a, \]
  \[ \tilde{\mathbf{v}}_a = \frac{\sum_{b \in F} \mathbf{v}_b W_{ab}}{\sum_{b \in F} W_{ab}}. \]
  \[ p_a = \frac{\sum_{b \in F} p_b W_{ab} + (\mathbf{g} - \mathbf{a}_a) \cdot \sum_{b \in F} \rho_b r_{ab} W_{ab}}{\sum_{b \in F} W_{ab}}, \]
Direction of the future works

- Merge feature/fsi with develop branch after cleaning up and adopting a similar Chrono language in the fsi module.

- Moving beyond shared-memory, heterogeneous computing.
  - Adopting Charm++
Charm++, Why and How?

- Over-decomposition
  - **Chare**: A message driven object (simple C++ object with entry method)
  - **Chare Array**: Collection of similar chares
  - **Chare Section**: For instance, 27 cells, including self

- Asynchronous nature

- Blocking: SDAG programming model (*when* (cond.))
  - Return the control to the runtime system. Wait to regain it as soon as the conditions are met.

- Charm++ Runtime System
  - Load balancing, fault tolerance
    - Distribute the compute load based on availability of the physical cores, at the system and node levels.
Charm++, Why and How?

Serial {
    Block1
}
when ( data arrives ) {
    serial {
        Block2
    }
}
Serial {
    Block3;
}

The Block inside the serial will be executed without interruption/preemption. The thread is reserved for the chare until the serial is done and the control is sent back to the scheduler.

Using SDAG, i.e. “when” statement, is important. With that, the control is sent back to scheduler until the data arrives. This is another level of over-decomposition.

PE 1, 2, 3 may/may not run on the same physical core. But for sure, for a given chare, they are run in the sequence shown,
Starting Point

• LeanMD –
  • Charm++ mini-applications effort - [http://charmplusplus.org/miniApps/](http://charmplusplus.org/miniApps/)
  • Mimics short-range force computation in NAMD

• Similarities with SPH
  • Binning used for neighbor search.
  • A similar periodic boundary concept (no wall boundary condition)

• Execution flow:
  1. Calculate forces
  2. Update Properties
  3. Migrate particles

• 20,000 particles, 0.5 sec. per step on 2 processors (out of Intel Core i5, 2.6 GHz)
Charm-SPH current state

- Many-core three-dimensional SPH
  - Distributed memory model
- Boundary conditions
  - Periodic
  - Wall boundary condition
- Explicit Euler
  - Runge-Kutta midpoint method will be implemented soon
- No Extended SPH and Filtering.
  - Will be added if needs be
Parallelism Model

- Bin-Chare association
- Compute Chare array
- Particle migration relying on 3D domain represented by Chares removes the need for sorting

Model:
- Send data
- Compute
- Return
- Reduce (contribute method)
Thank You!