Gradient Free Design of Microfluidic Structures on a GPU Cluster

Austen Duffy - Florida State University

SIAM Conference on Computational Science and Engineering

March 2, 2011

Acknowledgements

This work is with

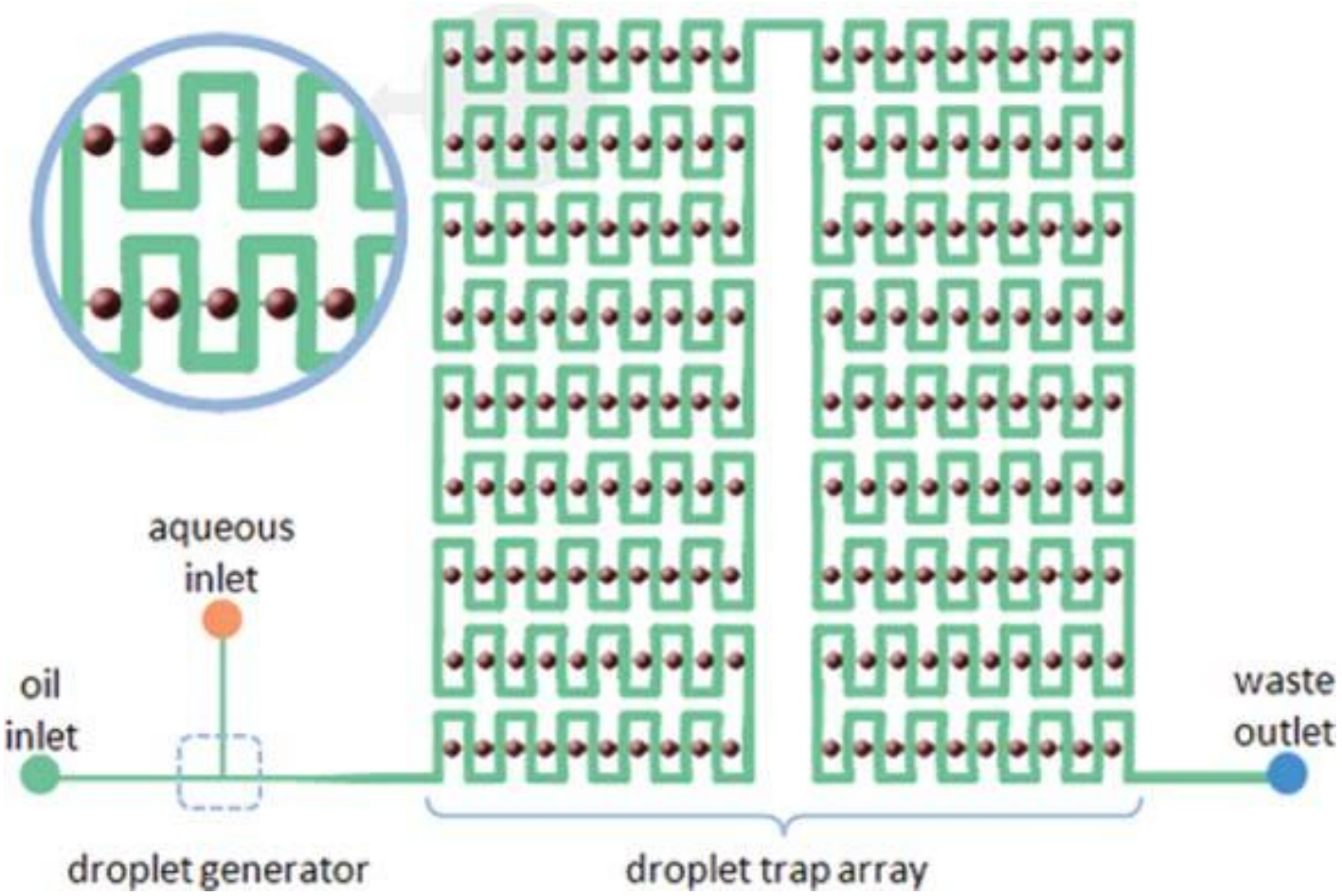
- Mark Sussman - FSU Mathematics
- Michael Roper - FSU Chemistry

And supported by the National Science Foundation through grant DMS-1016381

Motivation

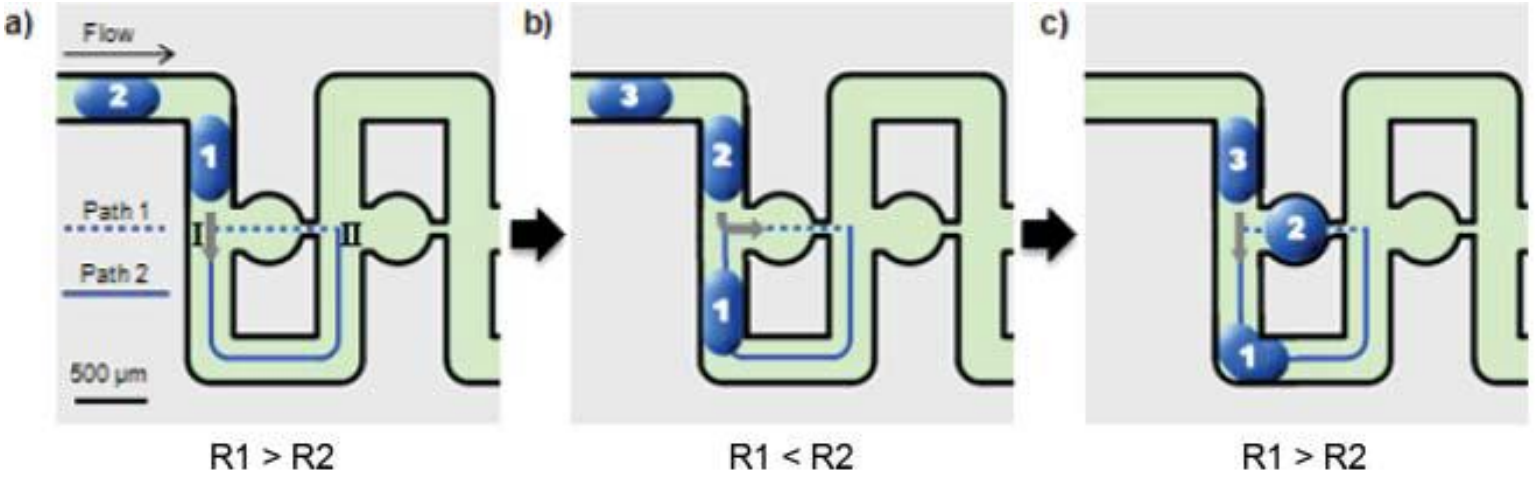
- Microfluidic Devices are leading to rapid advances in various areas including biotechnology where the 'lab-on-a-chip' has become very popular.
- Lab-on-a-chip devices allow for high throughput drug assays with small amounts of reactants which may be very expensive to produce.
- Precise control over droplet formation is a key factor for succesful experiments, and is dependent on channel geometry and surfactant properties.

Example Lab on a Chip Design



W. Shi, J. Qin, N. Ye, B. Lin. Droplet-based microfluidic system for individual *Caenorhabditis elegans* assay. *Lab on a Chip*, 2008.

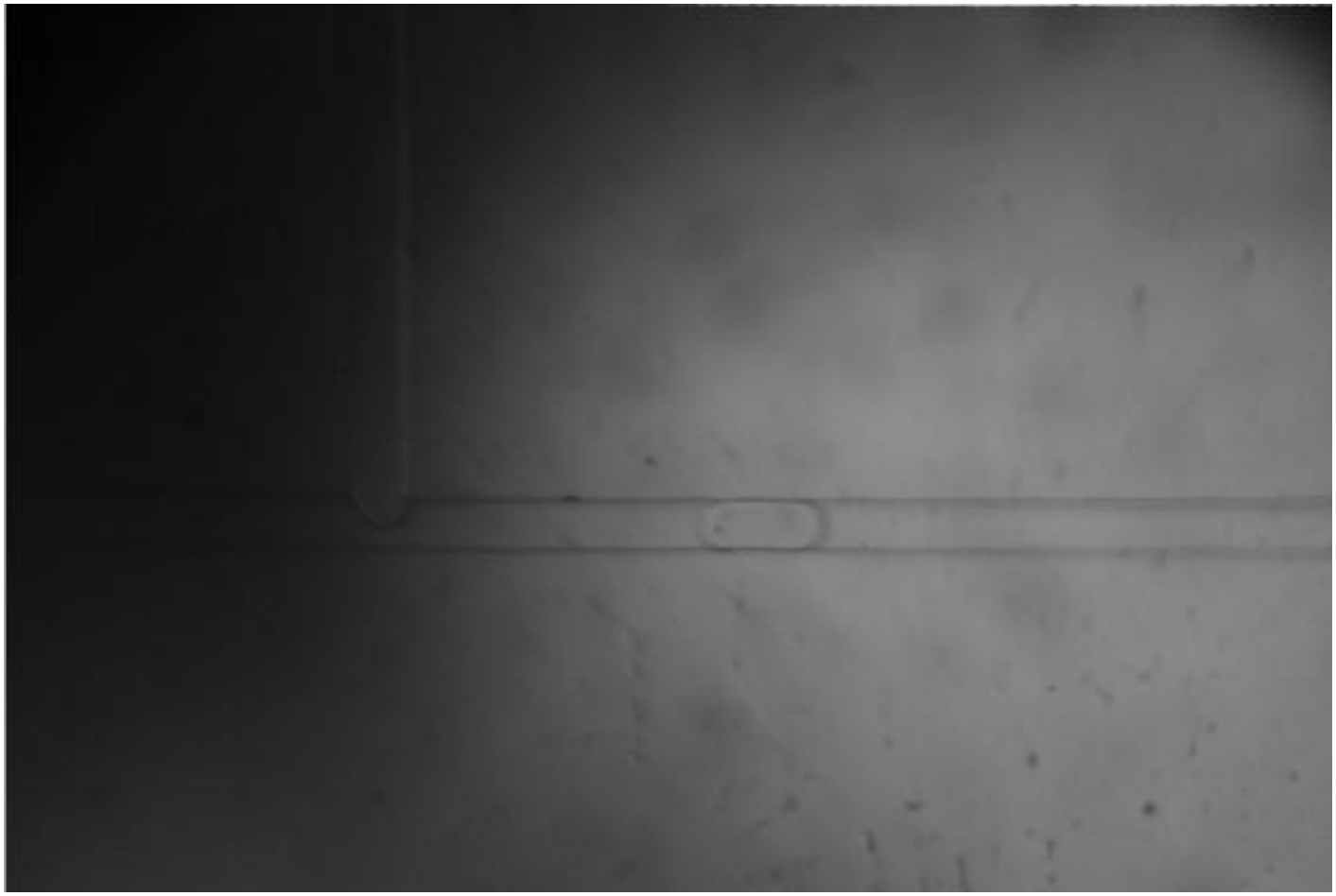
Droplet Trapping Device



Droplet sizes must be precise for proper trapping behavior.

W. Shi, J. Qin, N. Ye, B. Lin. Droplet-based microfluidic system for individual *Caenorhabditis elegans* assay. *Lab on a Chip*, 2008.

Microfluidic T-Junction



Experiment from Roper's Lab

Project Goals

With experimental backing from Roper's Lab, the two main computational goals of the project are to

- Improve the microfluidic simulation capabilities of our current multiphase flow code CLSVOF (coupled level set and volume of fluid code - structured, incompressible Navier-Stokes, adaptive mesh refinement)
- Develop an optimization code to couple with the CLSVOF code that will allow for numerical design of structures subject to two phase flow at the micro level

Challenges

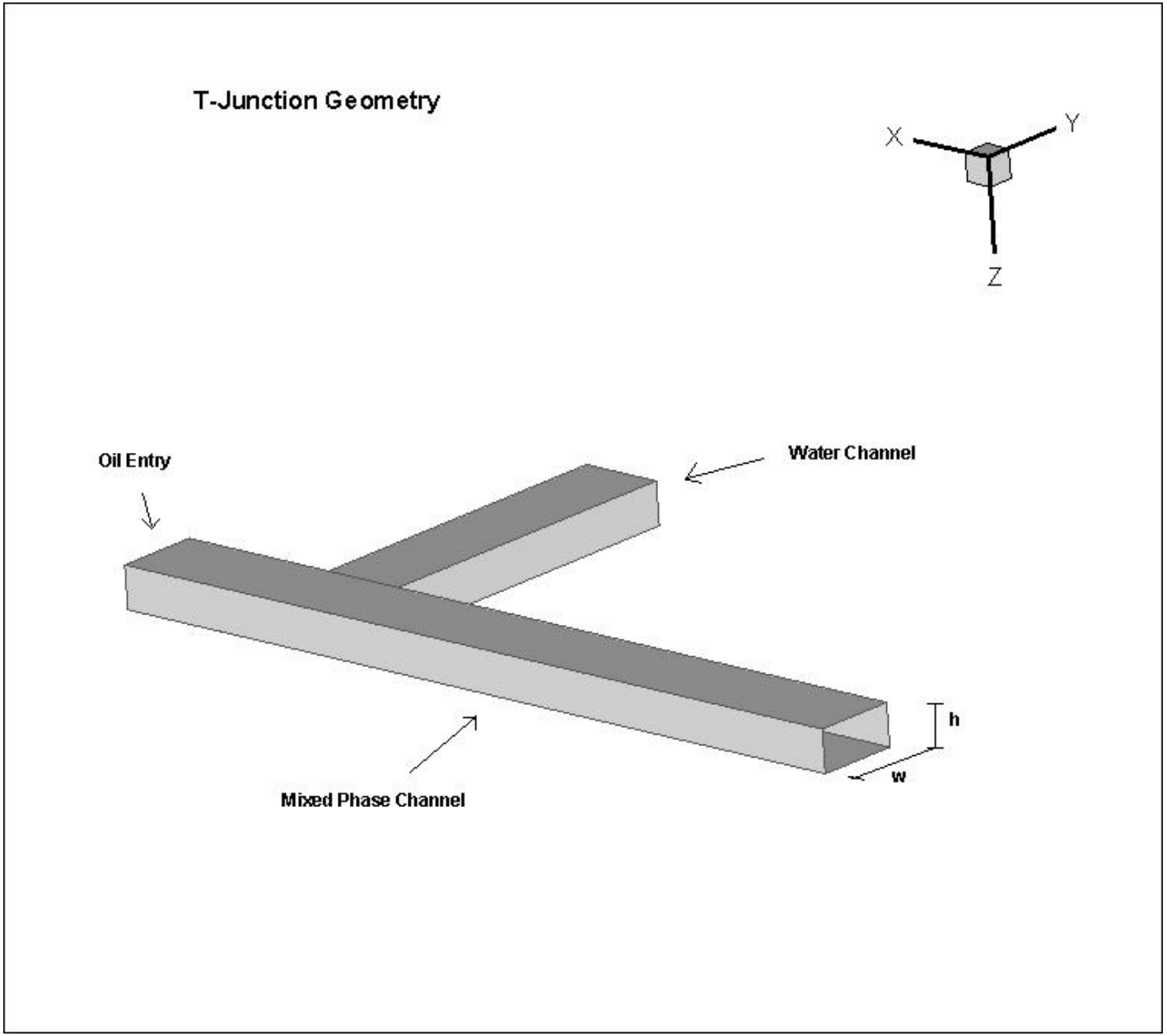
- Optimization in two phase flows can be very difficult, gradient computation may not be feasible.
- In microfluidic simulations, surface tension can impose stringent stability constraints leading to lengthy flow solves.

The Problem

We would like to develop a procedure for computationally designing microfluidic T-Junctions so that they can consistently produce precise droplets of a desired size.

We will be seeking to discover things such as the ideal geometry, fluid inflow rates, and surfactant properties that a T-Junction should incorporate in order to operate in an optimal manner for its desired purpose.

Computational T-Junction Geometry



Mathematical Formulation

The design problem can be formulated as an optimization problem

$$min \quad J(u, \psi)$$

$$subject \ \ to \quad N(u, \psi)$$

Where J is the cost function (typically in integral form) and N is a governing PDE system with state variable u and design variable ψ.

Cost Function

The cost function details haven't been completely worked out yet, but it should depend heavily on the droplet volume and resemble

$$J = \int_V Droplet_{ideal} - \int_V Droplet_{actual}$$

Design Variables

The design variables will depend on

- T-Junction Geometry (can be described by as few as 2 control points for rectangular channels)
- flow rates of the continuous and dispersed fluids
- surfactant properties
- Others?

Governing PDE System

The level set equations for incompressible multi-phase flow

$$\nabla \cdot u = 0,$$

$$\frac{\partial u}{\partial t} + (u \cdot \nabla)u = \frac{1}{\rho}\nabla \cdot (-pI + 2\mu D) + g - \frac{\sigma\kappa\nabla H}{\rho}.$$

$$\frac{\partial \phi}{\partial t} + (u \cdot \nabla)\phi = 0.$$

$$\rho = \rho_D H + \rho_C(1 - H) \qquad \mu = \mu_D H + \mu_C(1 - H) \qquad H = \begin{cases} 1 & \phi \geq 0 \\ 0 & \phi < 0 \end{cases}$$

- ϕ, the level set function, is positive in the dispersed droplet phase and negative in the continuos phase.
- $u = u, v, w$ are the velocity components
- t is time, p is pressure
- I is the unit tensor, D is the rate of deformation tensor
- g is gravity, σ is the surface tension
- $\kappa = \nabla \cdot n$ is the interface curvature
- ρ is the density, μ is the viscosity
- $_D$ and $_C$ subscripts represent 'dispersed' and 'continuous' phases

The Optimization Procedure

Due to the nature of the problem, we have chosen a non-intrusive derivative free approach for the optimization procedure.

We are currently developing a hybrid optimization method which uses

- a genetic algorithm (GA) to obtain the global solution
- the multidirectional search (MDS) method to obtain a locally refined solution
- a Monte Carlo method for generating an initial population of solutions for the GA.

A Multigrid Strategy

Multigrid methods have been shown to reduce the number of iterations required for convergence of gradient based optimization methods.

S.G. Nash. A multigrid approach to discretized optimization problems. *Optimization Meth. & Soft.*, 2000.

R.M. Lewis, S.G. Nash. Model problems for the multigrid optimization of systems governed by differential equations. *SIAM J. Sci. Comp.*, 2005.

Multigriding strategies may also be beneficial to derivative free methods

Hybrid Optimization Algorithm

Given an initial design ϕ and a series of m meshes $H_1, H_2, ..., H_m$
! Use MC to generate initial population of P solutions.
Call generate_population(ϕ,P)
1. Begin GA
for i= 1 to m **do**
  Call GA(ϕ, H_i)
  if cost $<$ lowcost **then**
    store solution
    lowcost = cost
  end if
end for
if Global Solution Obtained **then**
  Keep best solution ϕ_0, Go To 2.
else
  Go Back To 1.
end if

Hybrid Optimization Algorithm (continued)

2. Begin Local Refinement
for i= 1 to m **do**
  Call MDS(ϕ_0, H_i)
end for
if Local Solution Obtained **then**
  Optimization Complete
else
  Go Back To 2.
end if

Remarks on the Optimization Algorithm

- Each call to GA or MDS requires multiple iterations.
- Each iteration requires O(N) flow solves, where N is the number of design variables.
- GA and MDS are both highly parallel, flow solves can be performed simultaneously.
- A low number of design variables make this problem ideal for a small cluster, each node can run a flow solve in parallel

Accelerating the Flow Solver

Our derivative free algorithm is still hindered by many iterations (and hence frequent calls to the flow solver), so any improvements which can speed it up are vital to the success of the code.

To make this solver fast, we intend to develop an improved explicit treatment of the surface tension which is trivially parallel and can be solved on a GPU.

Sussman and Ohta Surface Tension Treatment

Given a "distance function" ϕ, initialize a provisional level set function $d^0 = \phi$.

for k=1 to N **do**

a. Find the curvature κ near the zero level set of d^{k-1}.

b. For each separate interfacial segment Γ_i find the average curvature κ_i^{avg}

c. extend $\kappa - \kappa_i^{avg}$ into a small narrow band about the zero level set.

d. $d^k = d^{k-1} + \Delta\tau(\kappa - \kappa^{avg})$

e. reinitialize d^k.

end for

Replace κ with the quantity $\frac{d^N - d^0}{\Delta t}$.

M. Sussman, M. Ohta. A stable and efficient method for treating surface tension in incompressible two-phase flows. *SIAM J. Sci. Comp.*, 2009.

Problems

While Sussman and Ohtas algorithm was successful in easing the time step constraint associated with surface tension dominated flows involving droplet rupture/recoalescence, the advantages are limited to large density ratio flows and not applicable to problems applicable to Roper's Lab.

New surface tension algorithms must be developed!

GPU Acceleration

Exploitation of the computational power of emerging hybrid many-core architectures is key to the success of this project.

With recent advances in GPU technology, parallel CFD codes are able to be accelerated on distributed hybrid architectures with multiple cores sharing a single GPU.

A.C. Duffy, D.P. Hammond, E.J. Nielsen. Production level CFD code acceleration for hybrid many-core architectures. *submitted to Parallel Computing*, 2010.

GPU Advancements

Architecture	Cores	L1 Cache	L2 Cache	Memory Access Speed
G80	112	16 KB [1]	128 KB [2]	57.6 GB/s GDDR3
GT200	240	24 KB [1]	256 KB [2]	102 GB/s GDDR3
Fermi	448	48/16 KB [3]	768 KB	144 GB/s GDDR5

GPU architecture evolution from G80, which approximately coincided with the release of Intel's quad core CPUs, to Fermi which coincided with the release of Intel's six core processors. GPU advancements over the last few years have noticeably outpaced those of CPUs. Representative GPUs are: G80-GeForce 8800 GT, GT200-Tesla C1060, Fermi-Tesla C2050.
[1] shared memory, [2] texture memory, [3] Configurable L1/shared memory

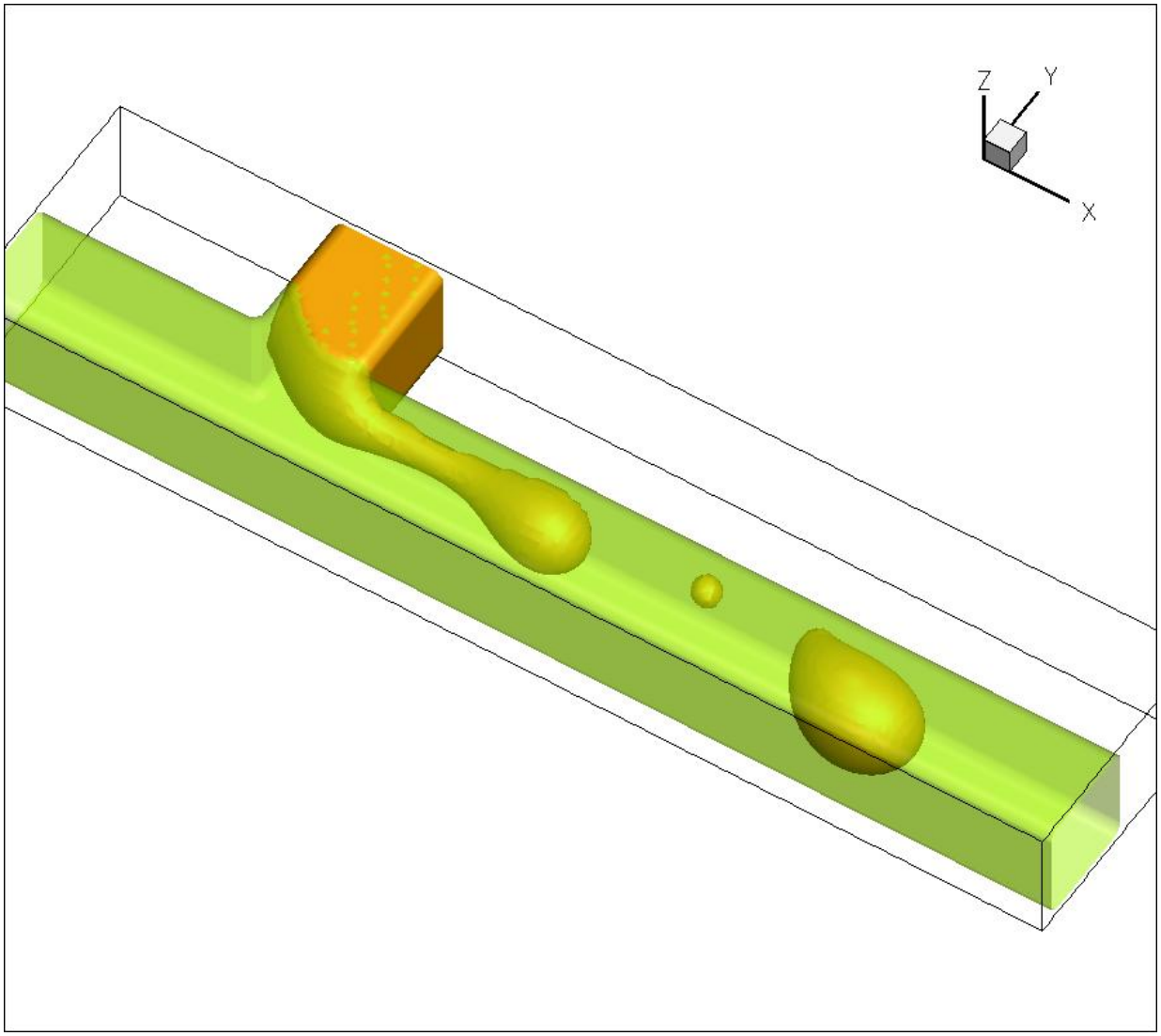
Project summary

The project goal is to develop a code for hybrid many-core architectures that couples LS-AMR-VOF to a novel MG/OPT code for the purpose of simulation based design of two-phase microfluidic devices.

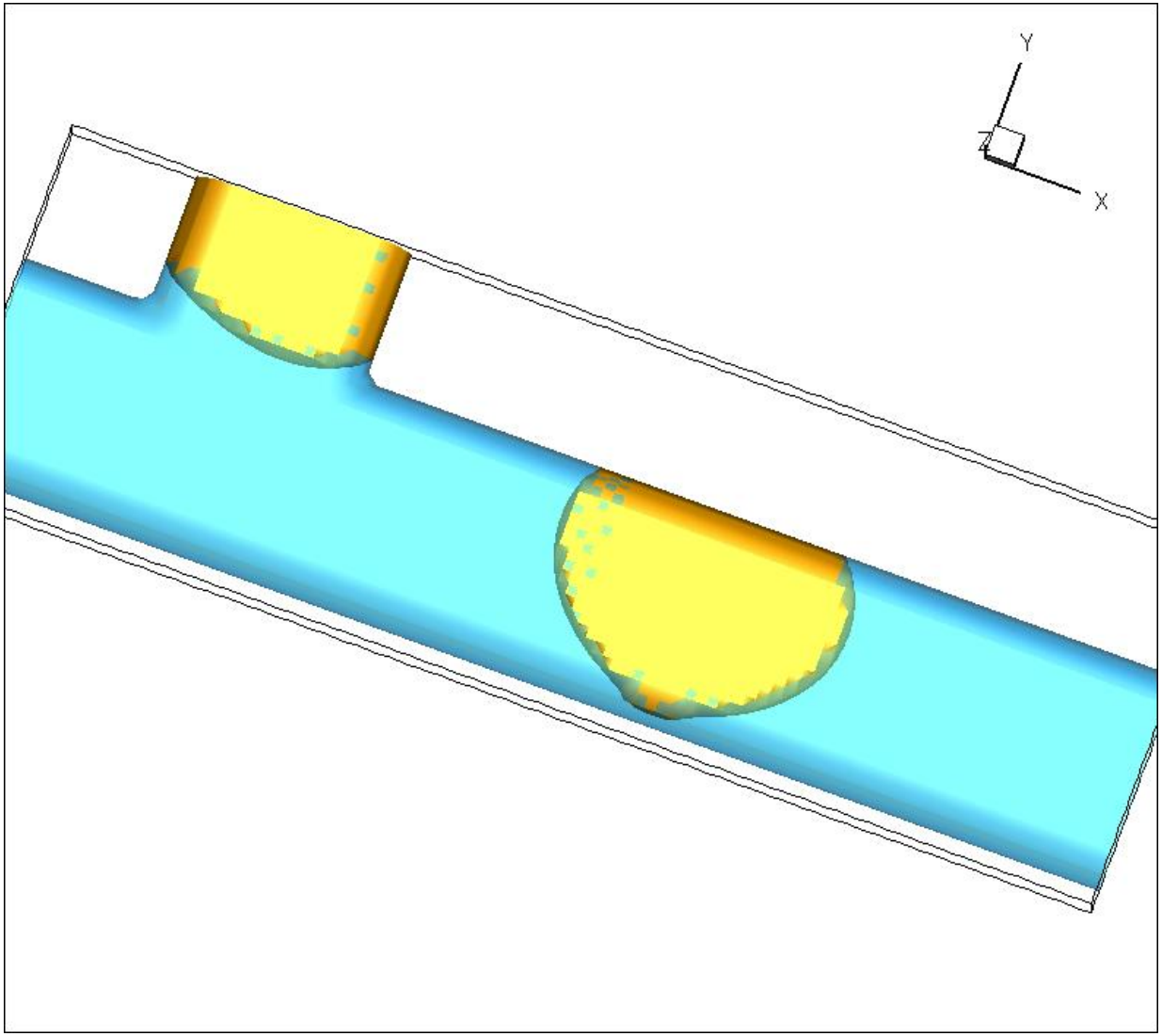
Progress to date

- LS-AMR-VOF is currently capable of microfluidic simulations through T-Junction channels, but requires improvement in the surface tension treatment
- A multigrid MDS code has already been developed, but current results are mixed with limited test cases
- GPU acceleration of the LS-AMR-VOF code is well underway, but some bugs need to be worked out

Microfluidic T-Junction

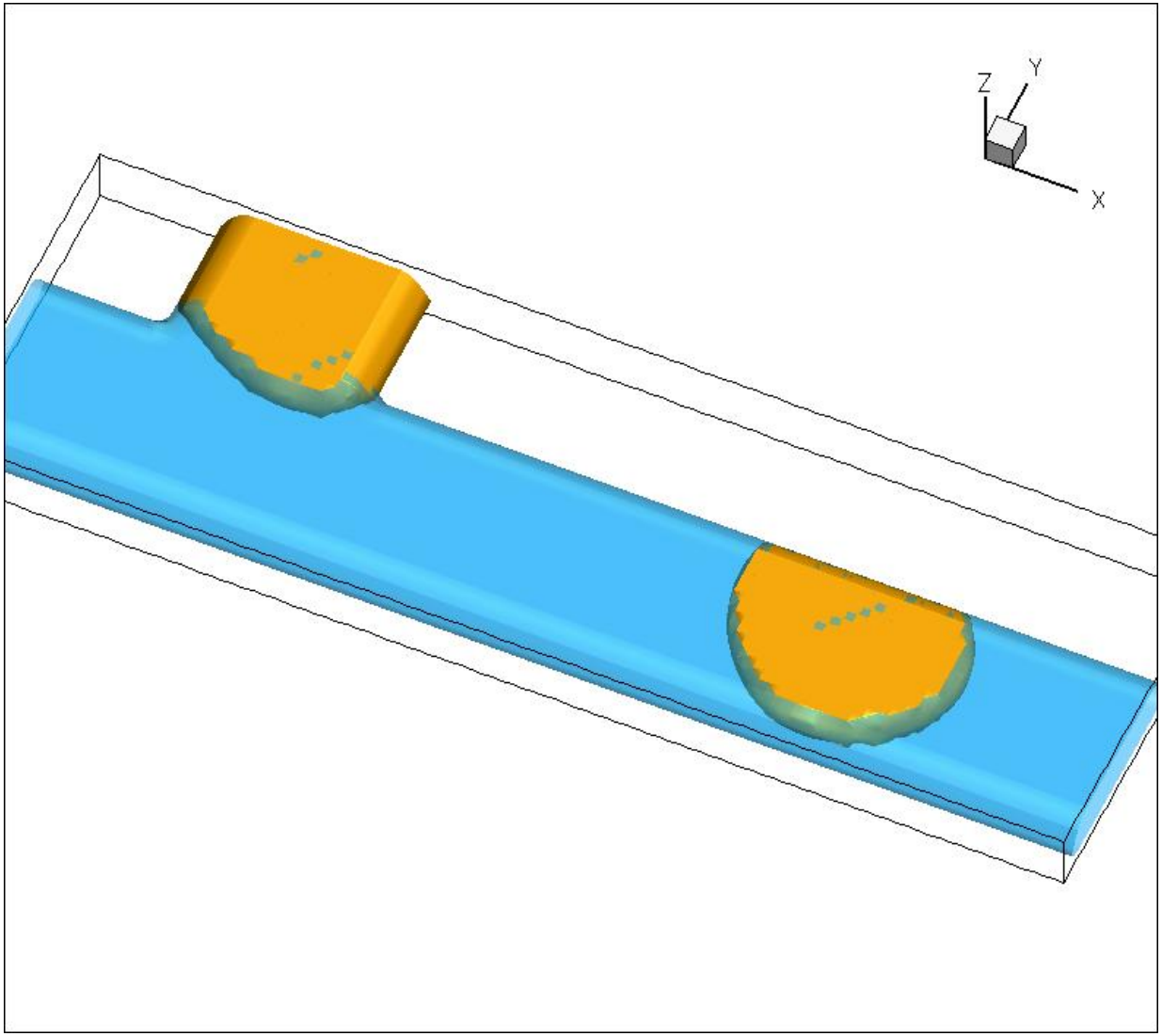


Microfluidic T-Junction



Simulation using data from Roper's Lab

Microfluidic T-Junction



Simulation using data from Roper's Lab

QUESTIONS?