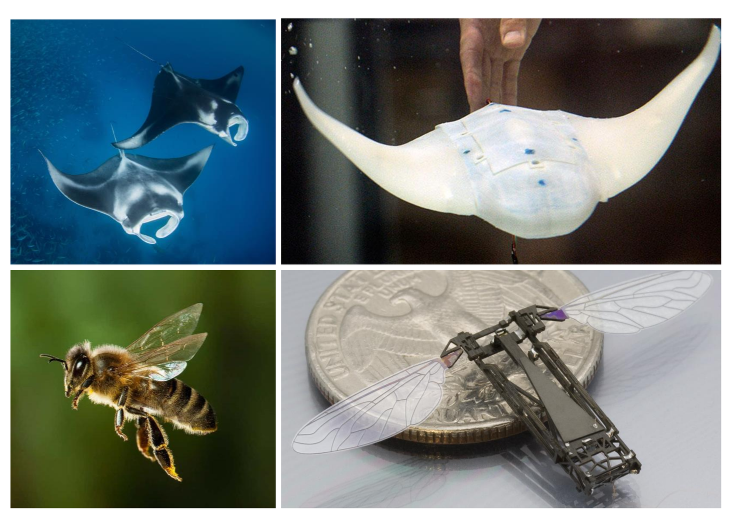
The Immersed Interface Method accurately simulates 2D fluid flows in complex moving domains

James Gabbard · MIT Van Rees Lab

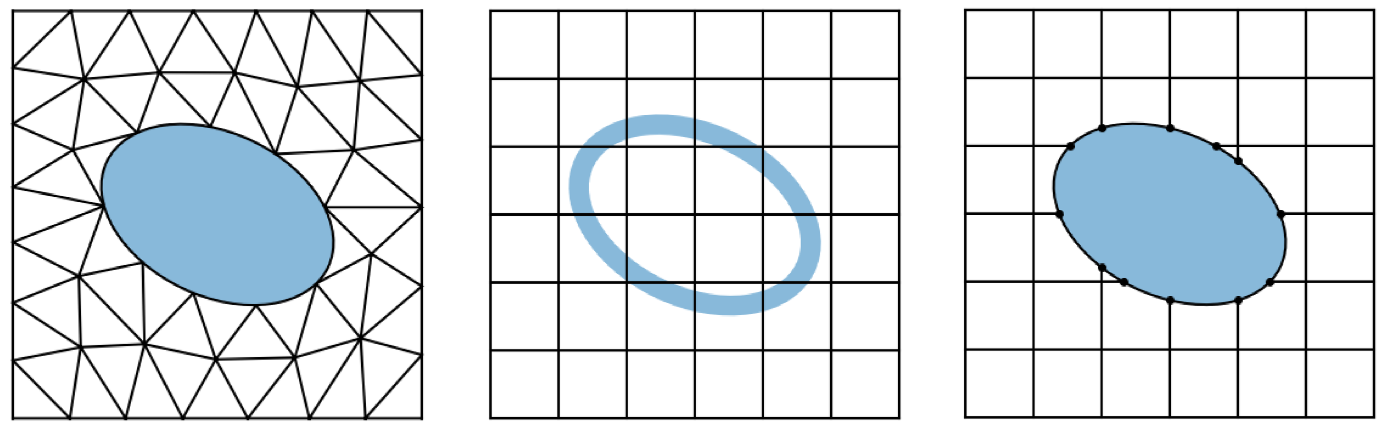
Background



The mechanics of **flying and swimming creatures** are inspiring a new generation of **underwater and aerial robots**.

These systems create **unsteady flows** that involve **moving bodies**, which are a challenge for many existing CFD packages.

There are three popular approaches to simulating these types of flows:



Body-Fitted Grid · Immersed Boundary · Immersed Interface

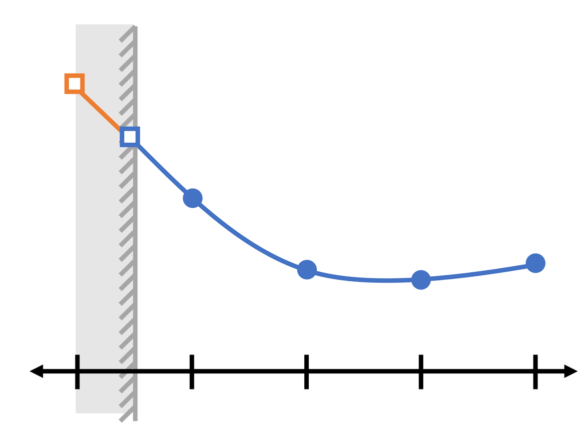
- **Body-Fitted Grids** achieve high order accuracy, but require expensive grid adaptation and re-meshing.
- **Immersed Boundary Methods** use an inexpensive Cartesian grid, but lose accuracy due to 'smearing' at solid boundaries
- **Immersed Interface Methods (IIM)** keep the speed of Cartesian grids, without sacrificing accuracy at solid boundaries.

This goal of this research project was to **expand the capabilities** of existing Immersed Interface Methods, and create a tool that **efficiently simulates** a wide variety of **unsteady 2D flows** with **moving boundaries**.

Methods

How does it work?

The IIM uses data from a regular grid (●) and a boundary condition (□) to extrapolate a function from the domain boundary (////) to the next regular point grid point (□). Afterwards, a standard finite difference can applied to extrapolated function.



This strategy is used to discretize the **vorticity form** of the Navier-Stokes equations, which has three major components:

$$-\nabla^2\psi = \omega;\ \boldsymbol{u} = \nabla \times \psi$$ Kinematics (Poisson Equation)

$$\frac{\partial \omega}{\partial t} + \nabla \cdot (\boldsymbol{u}\omega) = \nu \nabla^2 \omega$$ Dynamics (Transport Equation)

$$\frac{\mathrm{d}\Gamma_C}{\mathrm{d}t} = -\nu \oint_C \partial_n \omega \, ds$$ Topology (Lamb's Flux Condition)

What is new and exciting about this strategy?

Our approach has unique advantages over existing IIM vorticity codes.

Using Lamb's Flux Cond. allows:

- Multiple bodies
- Outflow boundaries
- Conservative differencing

New IIM techniques allow:

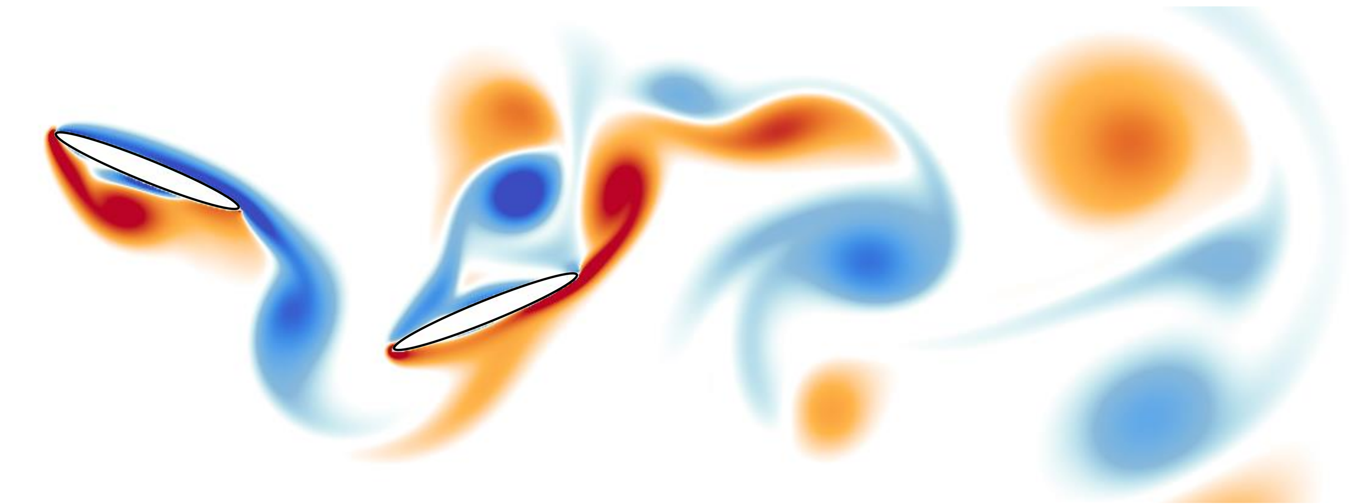
- Moving bodies
- Concave bodies
- Reconstructing the pressure field

What did we actually build?

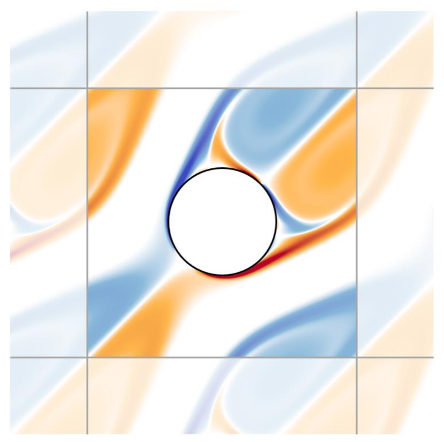
The final product of this project is a **2D Flow Solver written in C++.** It's built on a framework for **Block-Based, Shared Memory Parallelism,** to achieve high performance on workstations and individual nodes within a compute cluster.

Results

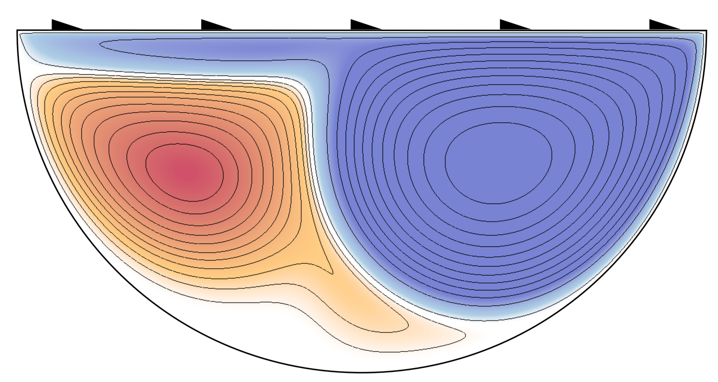
What types of flows can we simulate?



Flapping Foils · External Flows with Multiple Moving Bodies



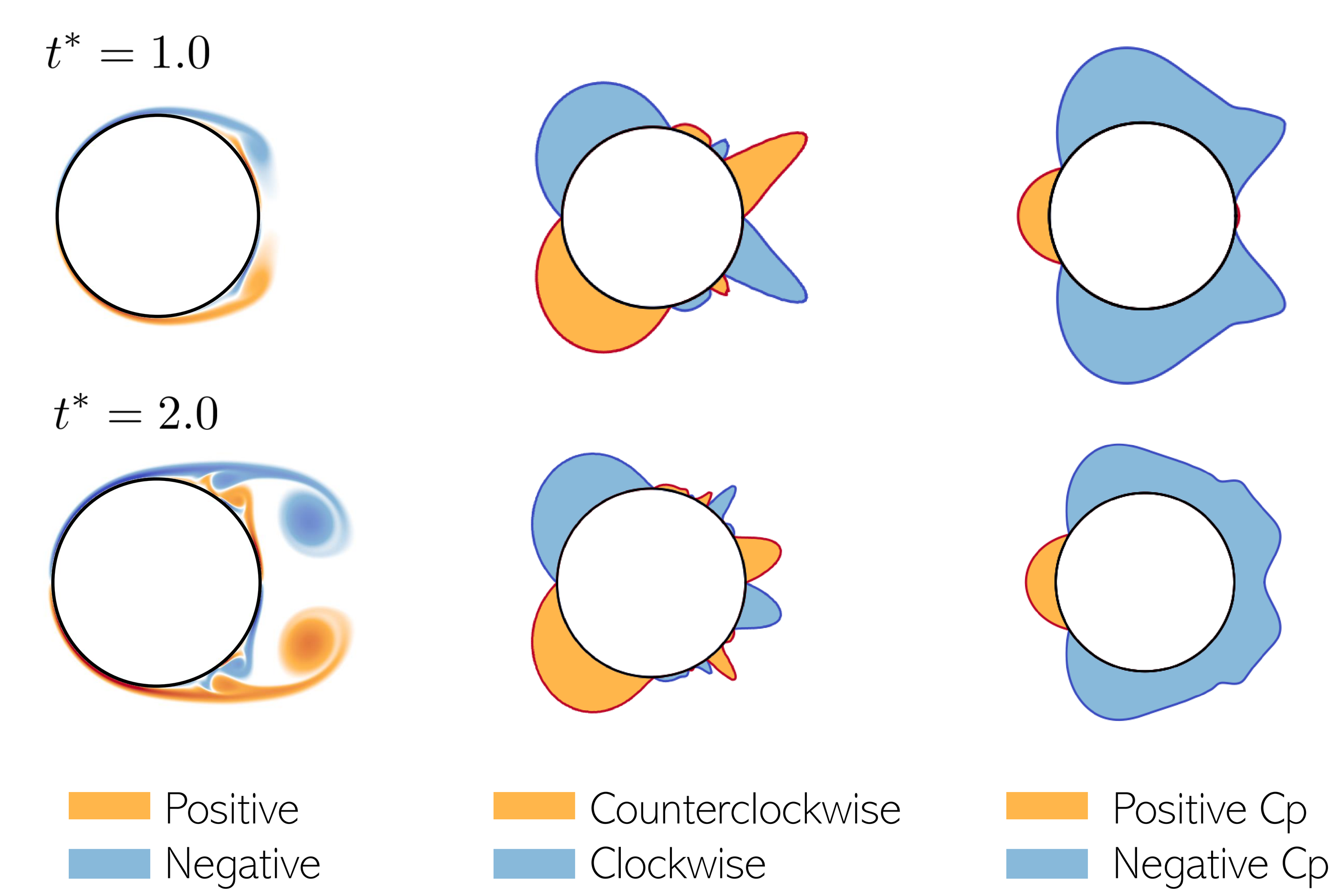
Cylinder Arrays · Periodic Flows



Lid-Driven Cavities · Internal Flows

What do we learn from these simulations?

Time-dependent pressure and shear-stress distributions on immersed surfaces. Here we show a cylinder of diameter D moving at speed U_∞, at two different times $t^* = U_\infty t / D$. The Reynolds number is 3000.



What's next for this simulation strategy?

Design Applications – uncertainty quantification · surrogate optimization

3D Simulations – turbulence models · distributed memory parallelism

Fluid Structure Interaction – nonlinear elasticity · coupling methods

Image Credits (Clockwise from top left): nationalgeographic.com · sciencenewsforstudents.org · robots.ieee.org/robots/robobee · newscience.com