











Advanced Modeling & Simulation (AMS) Seminar Series
NASA Ames Research Center, July 16th, 2026

# A Multi-GPU-Accelerated Sharp-Interface Immersed-Boundary Solver for Extreme-Scale Flow Simulation with Moving Boundaries

Sushrut Kumar

Department of Mechanical Engineering, The Johns Hopkins University

# Team



Dr. Joshua Romero (NVIDIA)



Dr. Massimiliano Fatica (NVIDIA)



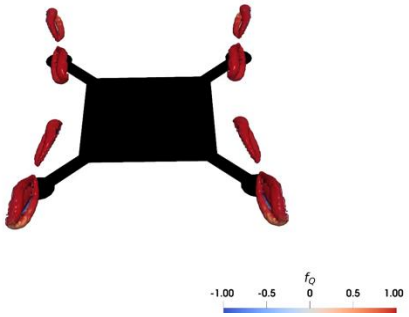
Prof. Jung-Hee Seo (JHU)



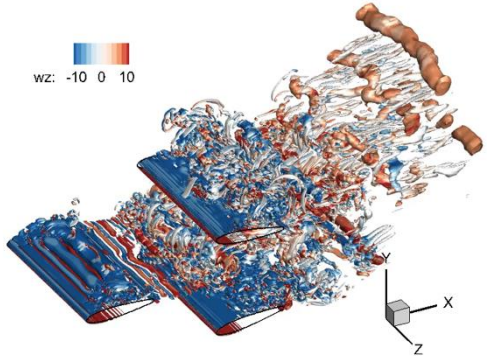
Prof. Rajat Mittal (JHU)

# Aero/Hydro Dynamics

Drone Aerodynamics



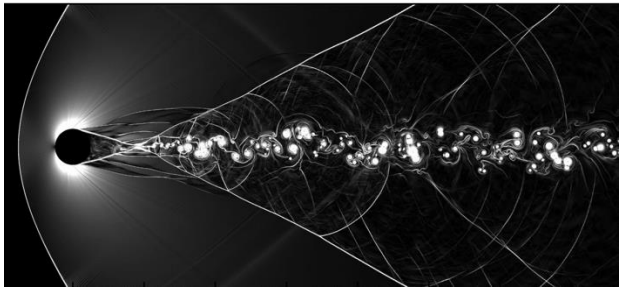

Multi-Foil Flapping Propulsor




Passive Control of Flow Separation



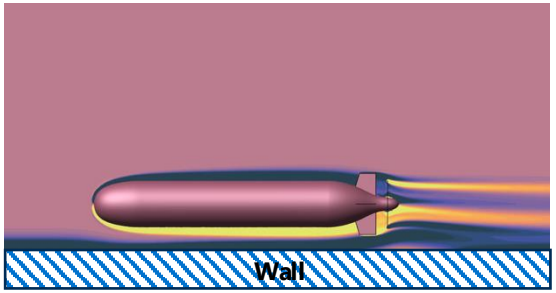
Cylinder Wake $M_\infty = 1.7,\ Re_\infty = 200{,}000$



Wing over waves

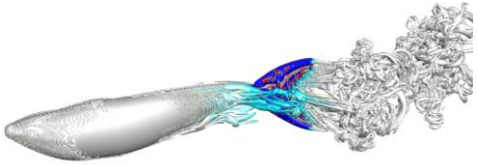


Dynamic Stability of AUV



# Biolocomotion

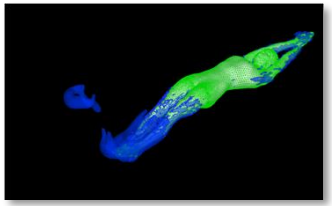
Carangiform Swimming



FSI in Caudal Fin



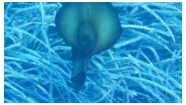
Human Swimming



Cetacean

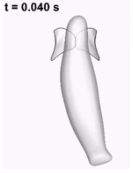


Unconventional Swimmers - Aplysia










Sea Butterflies

1 mm

1 cm

10 cm

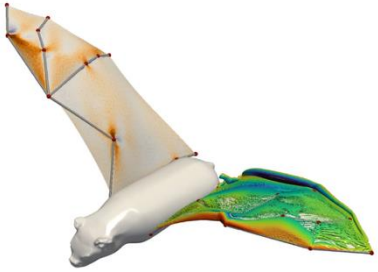
1 m

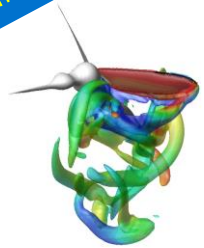
10 m

FSI in Bats

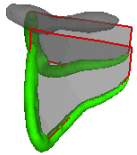
Cynopterus Brachyotis ~30 g

Pteropus Pumilus ~ 200g

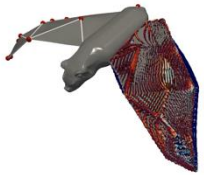




Mosquito



Butterflies



# Characteristics of the Computational Problems

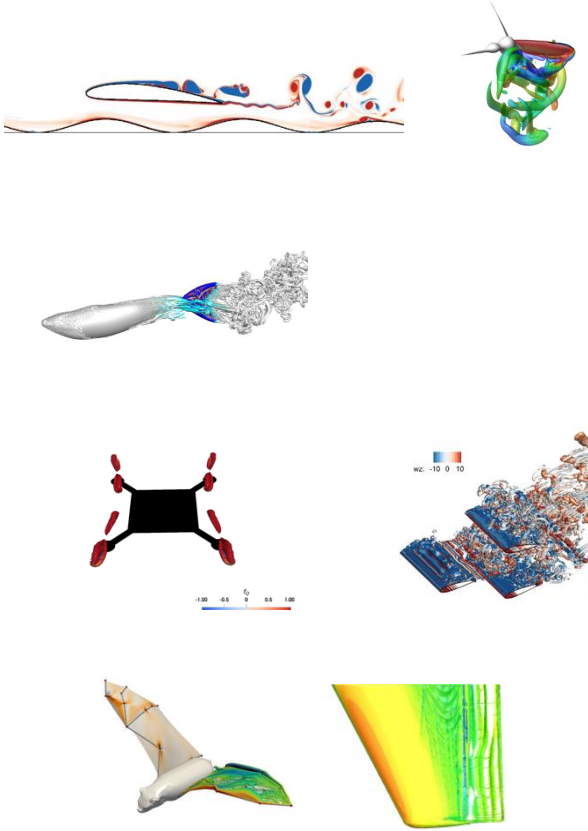
⇒ Thick solid bodies and zero-thickness membranes

⇒ Moving and deforming bodies

⇒ Multiple moving bodies complex shaped bodies within a same domain

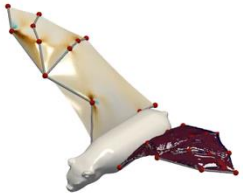
⇒ Perform direct numerical simulation at scale for unsteady flow problem

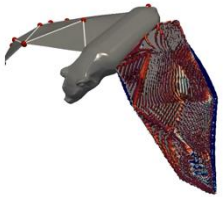
Rising Complexity



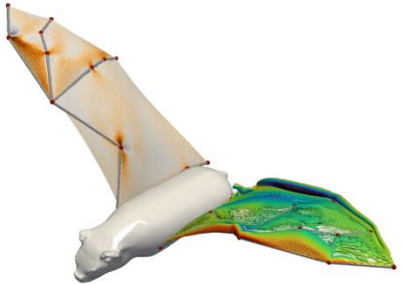
# Evolution of FSI Capabilities

Re = 1000
384 CPU Cores



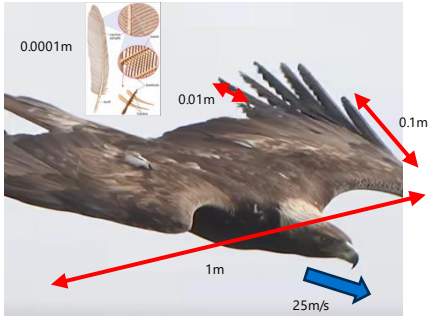


Re = 10,000
7680 CPU Cores



10X in Re
20X in CPUs
25X in wall-time

Re = 2,500,000




Re = 2.5 million
Grid > 20 Billion
Moving boundary + FSI

250X in Re
??X in CPUs
??X in wall-time

Rockfish - 45,072 CPU cores across 841 nodes

# Numerical Methods

Start with 3D unsteady Navier-Stokes equation for viscous incompressible flow

$$\frac{\partial u_i}{\partial x_i} = 0$$

$$\frac{\partial u_i}{\partial t} + \frac{\partial (u_i u_j)}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \nu \frac{\partial u_i}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j}\right)$$

→ Collocated Grid

→ Discrete Conversation

→ Low Numerical Dissipation

→ 2nd+ Order Numerics

**Advection-Diffusion Equation for Intermediate Velocity $u_i^*$**

$$\frac{u_i^* - u_i^n}{\Delta t} + \frac{1}{2}\left[3N_i^n - N_i^{n-1}\right] = -\frac{1}{\rho}\frac{\delta p^n}{\delta x_i} + \frac{1}{2}\left(D_i^* + D_i^n\right)$$

**Pressure Poisson Equation (enforcing $u_i^{n+1}$)**

$$\frac{1}{\rho}\frac{\delta}{\delta x_i}\left(\frac{\delta p^n}{\delta x_i}\right) = \frac{1}{\Delta t}\frac{\delta U_i^*}{\delta x_i}$$

Convection Term

$$N_i = \frac{\delta (U_j u_i)}{\delta x_j}$$

Diffusive Term

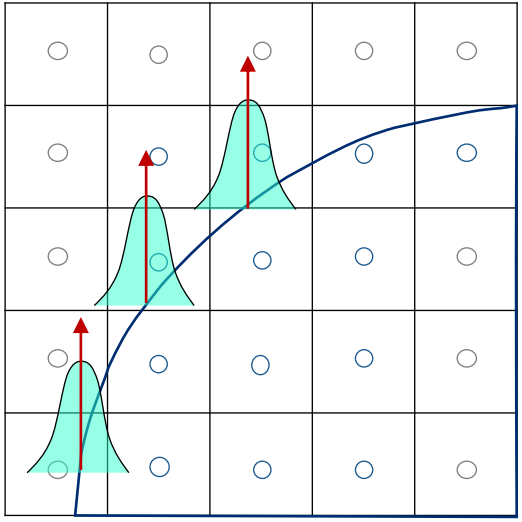
$$D_i = \nu \frac{\delta}{\delta x_j}\left(\frac{\delta u_i}{\delta x_j}\right)$$

$\frac{\delta}{\delta x}$ – Second order central difference operator

# Immersed Boundary Methods

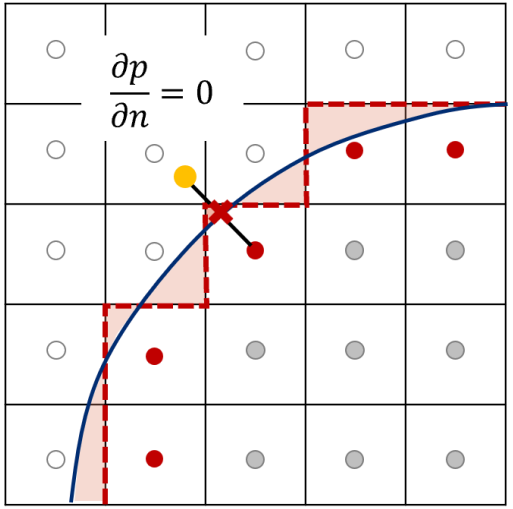
## Diffuse Interface Methods

Originally Idea by Peskin (1972)



## Sharp Interface Methods

Exact application of boundary condition on internal immersed body

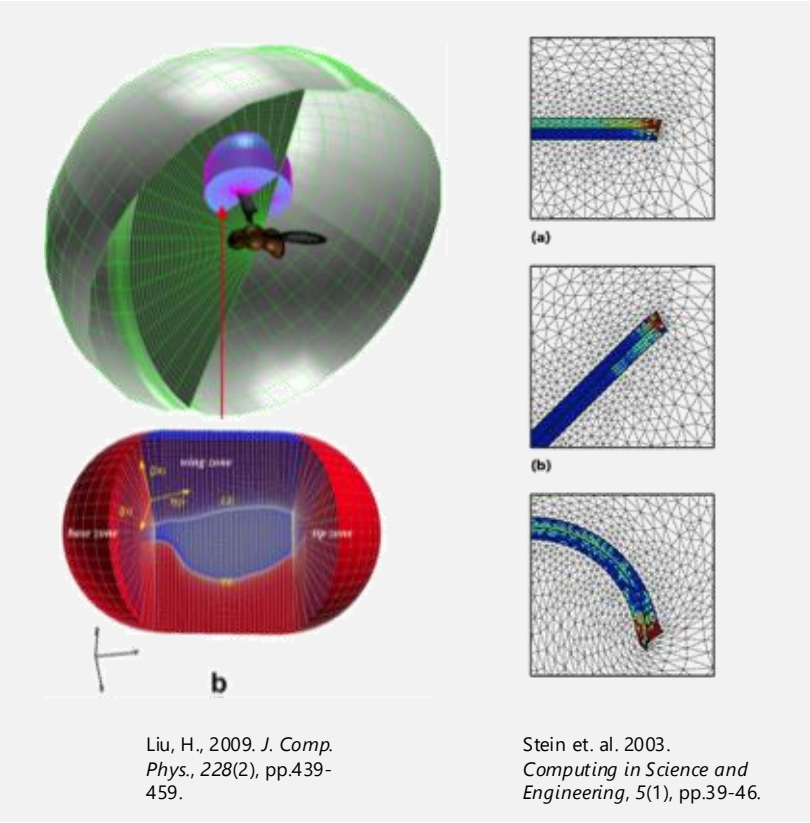


# Body Fitted Grids

- **Structured grid methods** (single block, multi-block)
- **Unstructured grid methods**
  (finite-volume, finite-element etc.)
- **Advantages**
  - Application of boundary conditions is relatively simple
  - Can generate optimal meshes for a given configuration
  - Easily provide enhanced resolution in localized region (boundary layers, interfaces, …)
- **Disadvantages**
  - Mesh quality and complexity issues – **non-linearity of fluid dynamics** amplifies mesh induced errors
  - Problem amplified for time-evolving geometries
  - GPU Implementation can be divergent.
  - **Mesh generation can be time & resource consuming**

***…mesh generation phase constitutes the dominant cost in terms of human intervention.* – NASA CFD Vision 2030**

***70% of overall time is spent in geometry handling and meshing.* – Sandia Study 2007.**



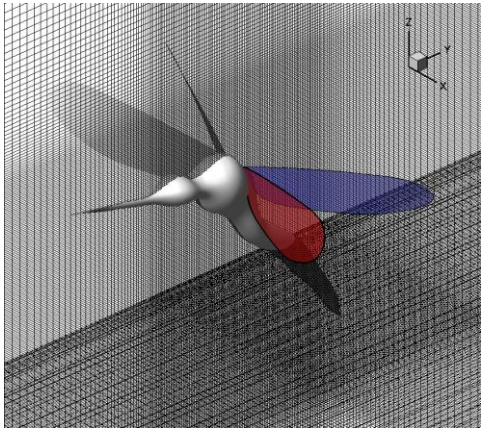

Liu, H., 2009. *J. Comp. Phys.*, *228*(2), pp.439-459.

Stein et. al. 2003. *Computing in Science and Engineering*, *5*(1), pp.39-46.

# Implementation of SI-IBM

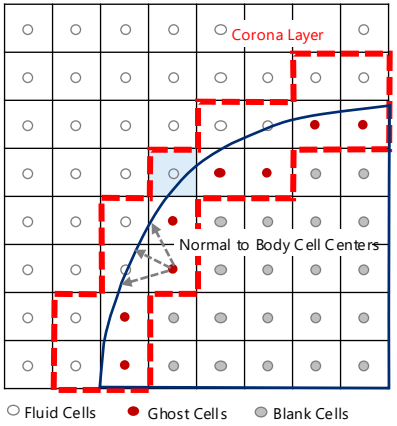
## Identification of Ghost and Fresh Cells

Background Cartesian grid

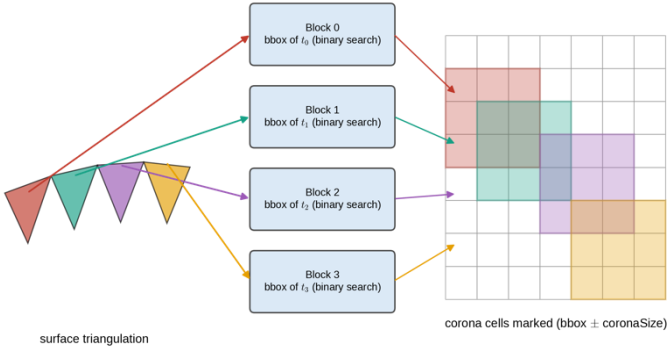


Immersed body represented using triangulated mesh

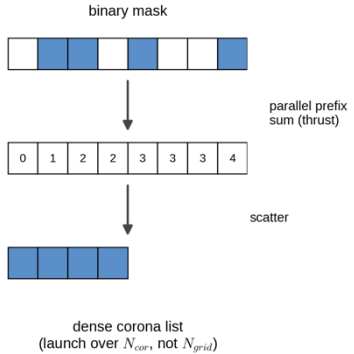
Reducing Computational Complexity
$O(NN_{tri})$ to $O(N_C N_{tri})$




Threads within a block cooperate to mark bbox




Compacting a 3D grid to a COO



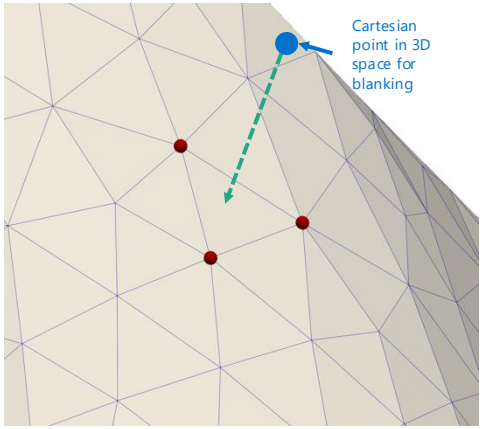



Mittal, R., et al. (2008). *Journal of computational physics*, *227*(10), 4825-4852.

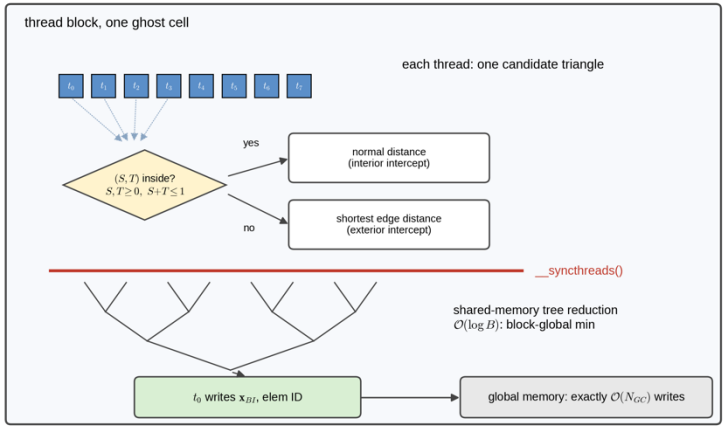
# Implementation of SI-IBM

## Identification of Ghost and Fresh Cells

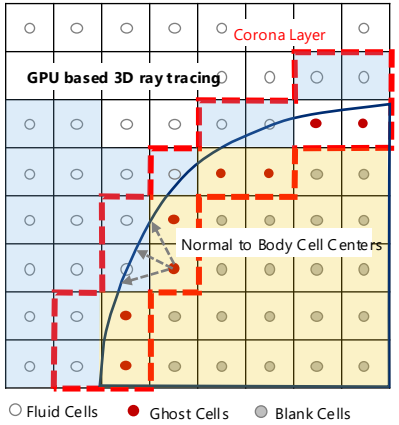
Identification of Top-K nearest points




Identification of nearest triangle with valid intercept



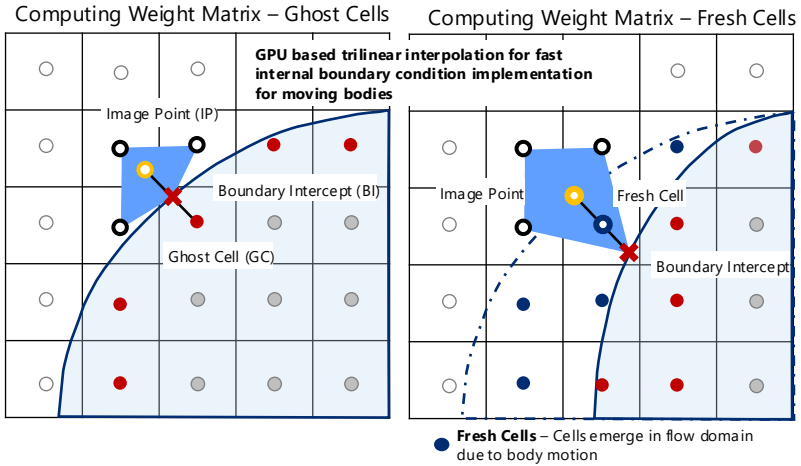

Blanking Cells - Ray Tracing

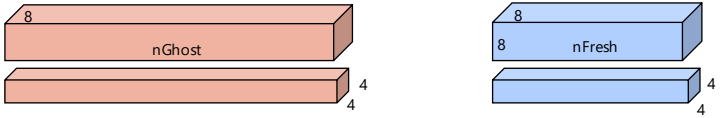



Mittal, R., et al. (2008). *Journal of computational physics*, *227*(10), 4825-4852.

# Implementation of SI-IBM

## Identification of Ghost and Fresh Cells




**Ghost Cell Value Calculation**

$\{C\} = [V]^{-1}\{\phi\}, \quad \beta = f(C, \vec{x}_{IP})$

$\phi_{IP} = \sum_{i=1}^{4/8} \beta_i \phi_i$

$V \in \mathbb{R}^{8\times 8 \times nG} \quad C \in \mathbb{R}^{8\times nG} \quad \phi \in \mathbb{R}^{8\times nG}$

Dirichlet BC

$\phi_{GC} - \sum \beta_i \phi_i = 2\phi_{BI}$

Neumann BC

$\phi_{GC} - \sum \beta_i \phi_i = \Delta l_p \left(\frac{\delta \phi}{\delta n}\right)_{BI}$

**Fresh Cell Value Calculation**

$\frac{X_i^{n+1} - X_i^n}{\Delta t} = V_i^{n+\frac{1}{2}}$

$\{C\} = [V]^{-1}\phi, \quad \beta = f(C, \vec{x}_{IP})$

$\phi_F = \sum_{i=1}^{4/8} \beta_i \phi_i$

$\phi_F - \sum \beta_i \phi_i = 0$

$V \in \mathbb{R}^{8\times 8 \times nF} \quad C \in \mathbb{R}^{8\times nF} \quad \phi \in \mathbb{R}^{8\times nF}$

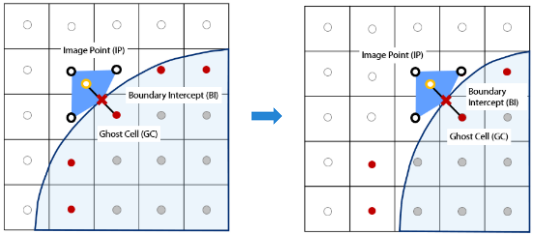
Mittal, R., et al. (2008). *Journal of computational physics*, *227*(10), 4825-4852.

# Efficient Sparse Linear Iterative Solver

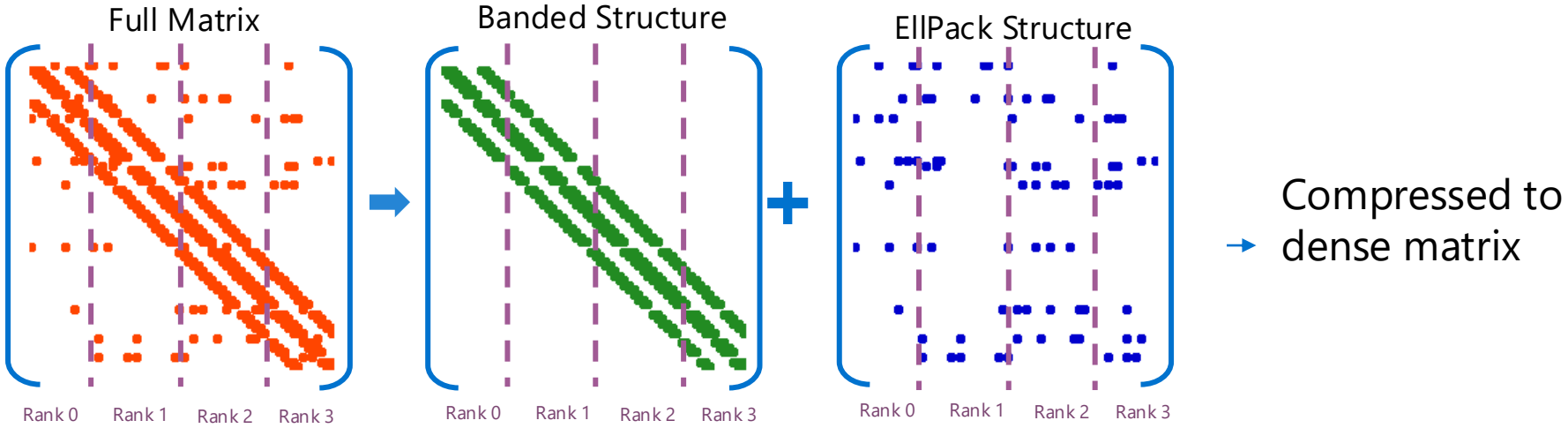
## Choice of iterative solver

1. Advection Diffusion – Scheduled Relaxed Jacobi (SRJ), $O(10)$speedup compared to LSOR
2. Pressure Poisson Solver - BiCGStab with SRJ Preconditioner

Yang, X. I., & Mittal, R. (2014). *Journal of Computational Physics*, *274*, 695-708.

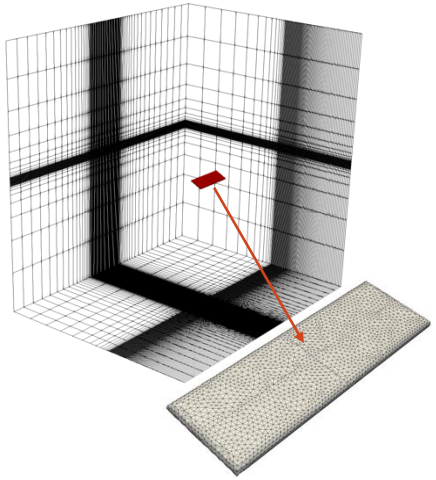



Moving stencil changes the sparsity every temp
→ Split matrix-vector multiplication
1. Matrix free static structure and GCM based dynamic structure.
2. Communication dependent and independent segments

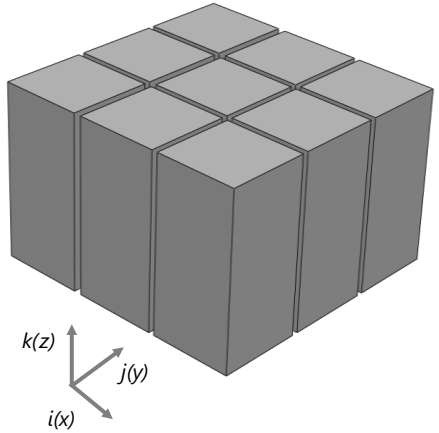


# Scaling to Large Grids

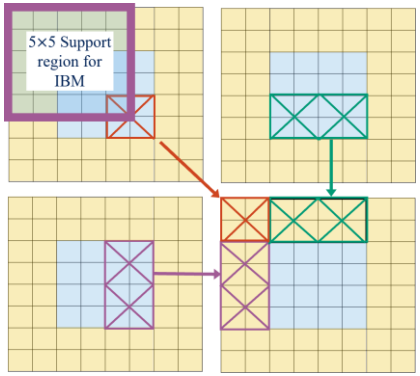
**Cartesian Grid**



**2D Domain Decomposition**

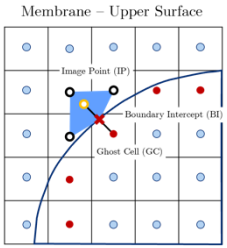


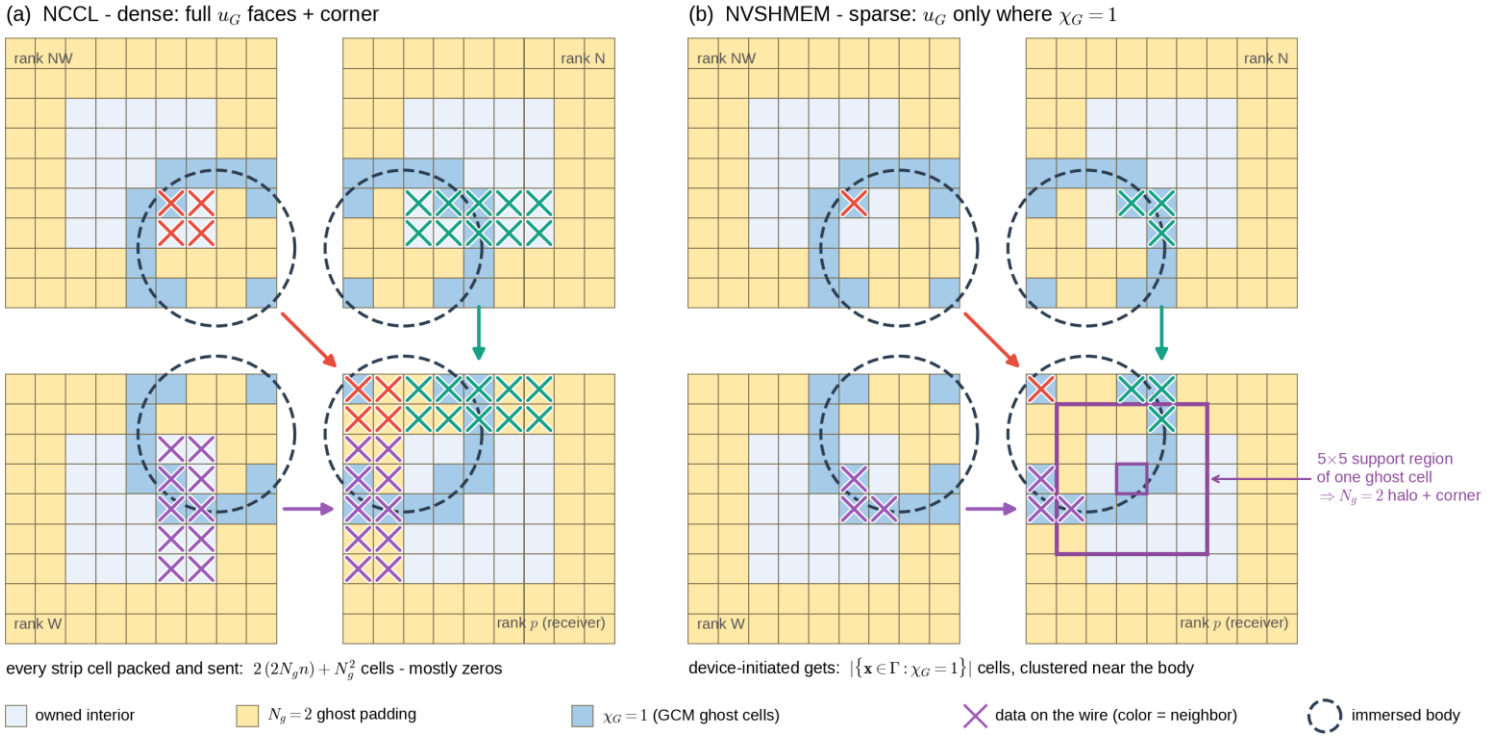

**Halo Exchange**



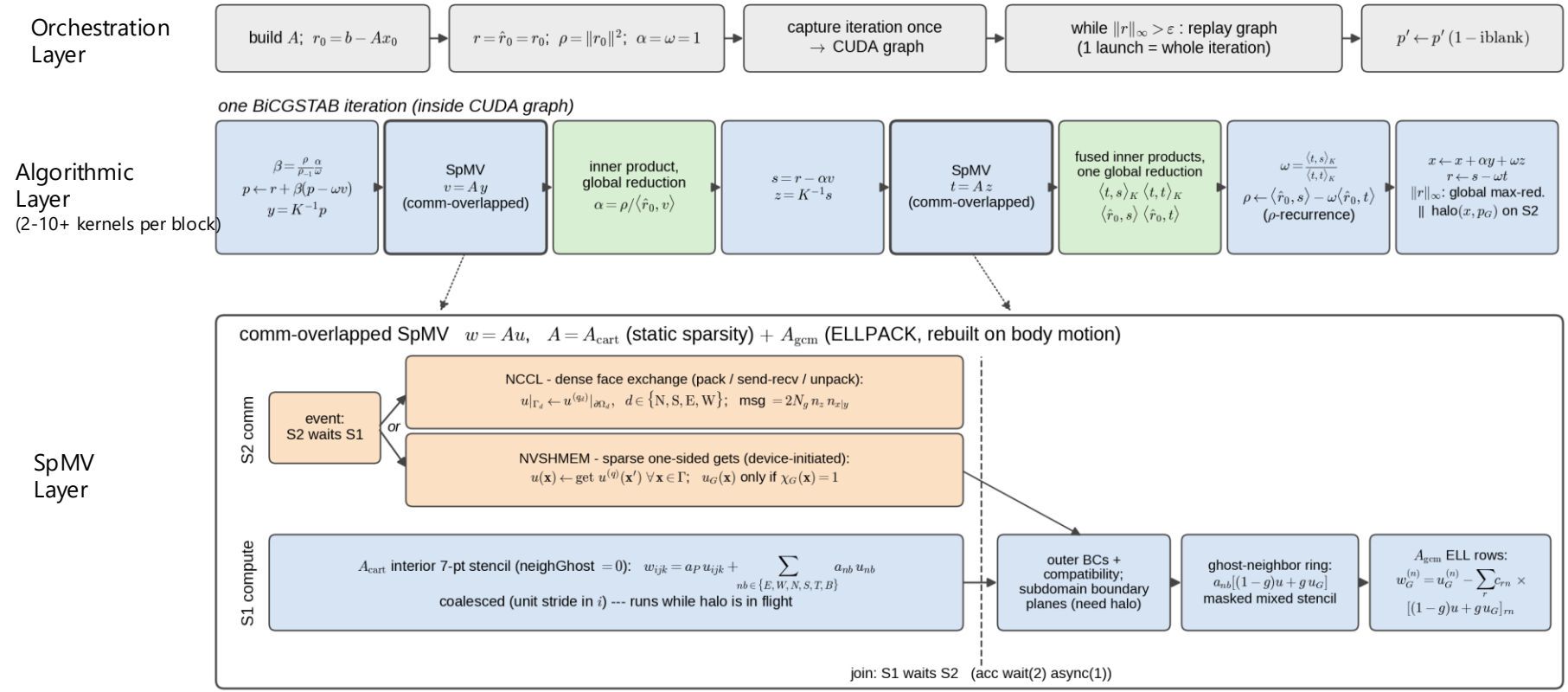
# Communication Backends for Ghost Cells

Efficient treatment of membrane/shells requires a dual-field approach

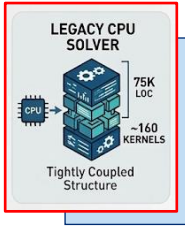


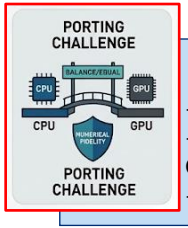
# Dual-Stream Comm-Compute Overlap



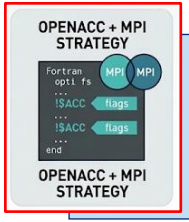
# GPU Programming Model



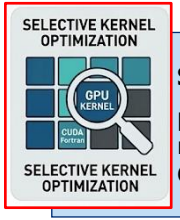

**Legacy CPU solver:**
- 75K LOC ;
- 160 compute kernels;
- tightly coupled algorithmic structure.



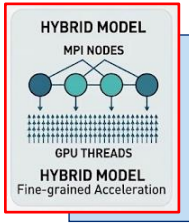

**Porting challenge**:
- preserve numerical fidelity
- restructuring for heterogeneous (CPU–GPU) execution
- future hardware portability.




**Core strategy:**
directive-based acceleration using **OpenACC + MPI (Fortran)** to minimize code intrusion and retain maintainability.



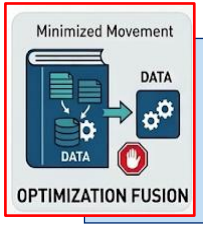

**Selective kernel optimization:**
performance-critical routines reimplemented using **CUDA Fortran**, with OpenACC–CUDA interoperability.



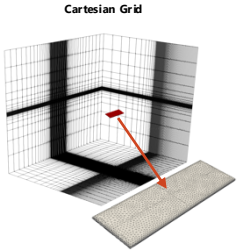

**Hybrid model:**
- coarse-grained parallelism via MPI,
- fine-grained acceleration on GPUs.

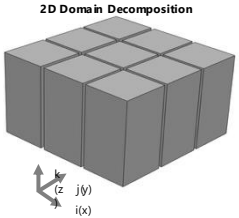



**Leveraging vendor libraries**
**cuBLAS, Thrust** for dense linear algebra, and array compaction to avoid reimplementation.

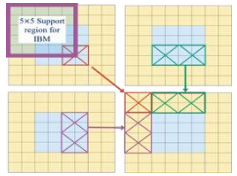


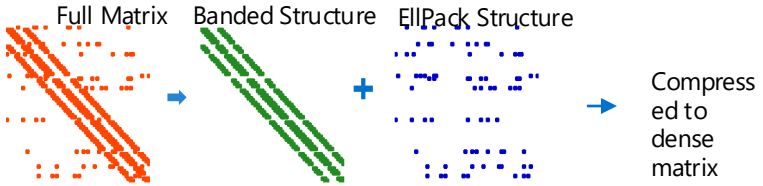

**Emphasize**
- memory locality
- data movement minimization
- kernel fusion – Mega Kernel




**Asynchronous Halo Exchange**
(MPI/NCCL/NVSHMEM)

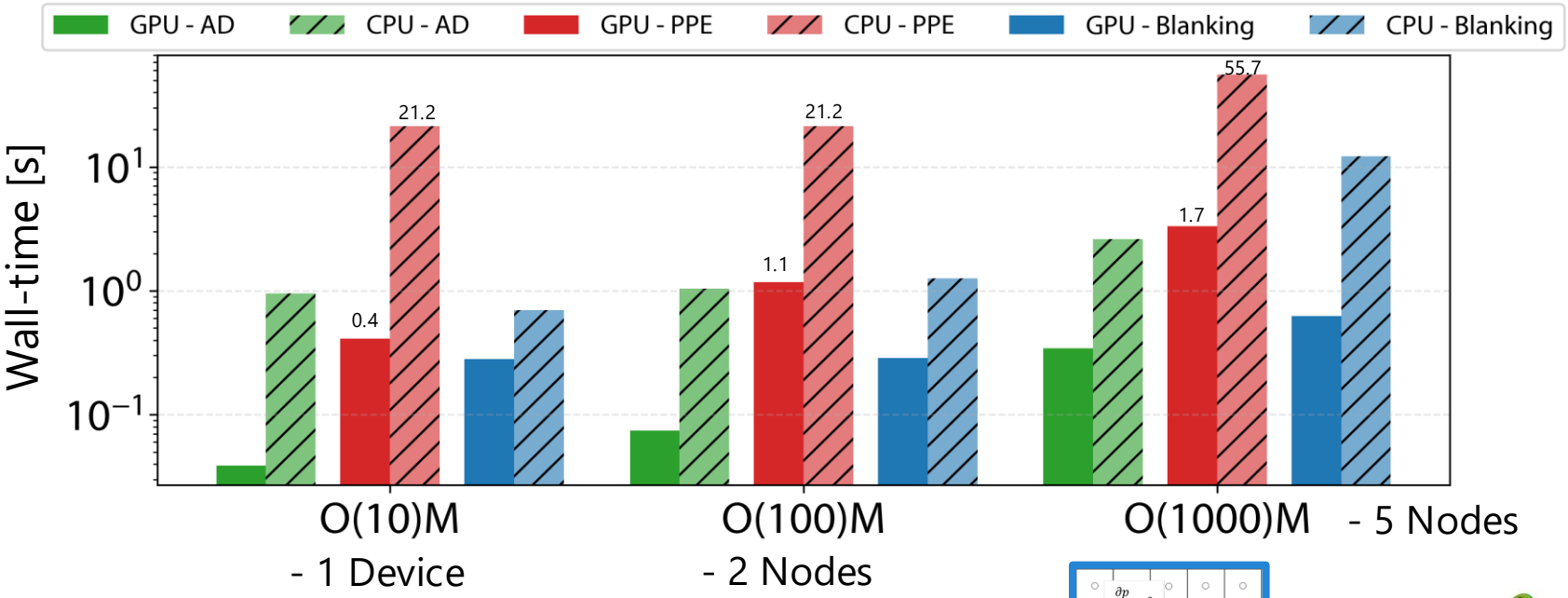


# >20X Improvements with GPU-Acceleration

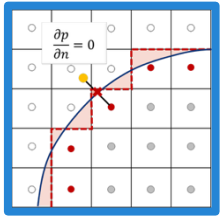
CPU – Intel Xeon Gold Cascade Lake 6248

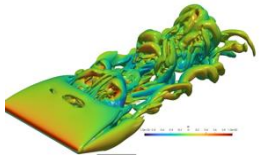
GPU – NVIDIA A100



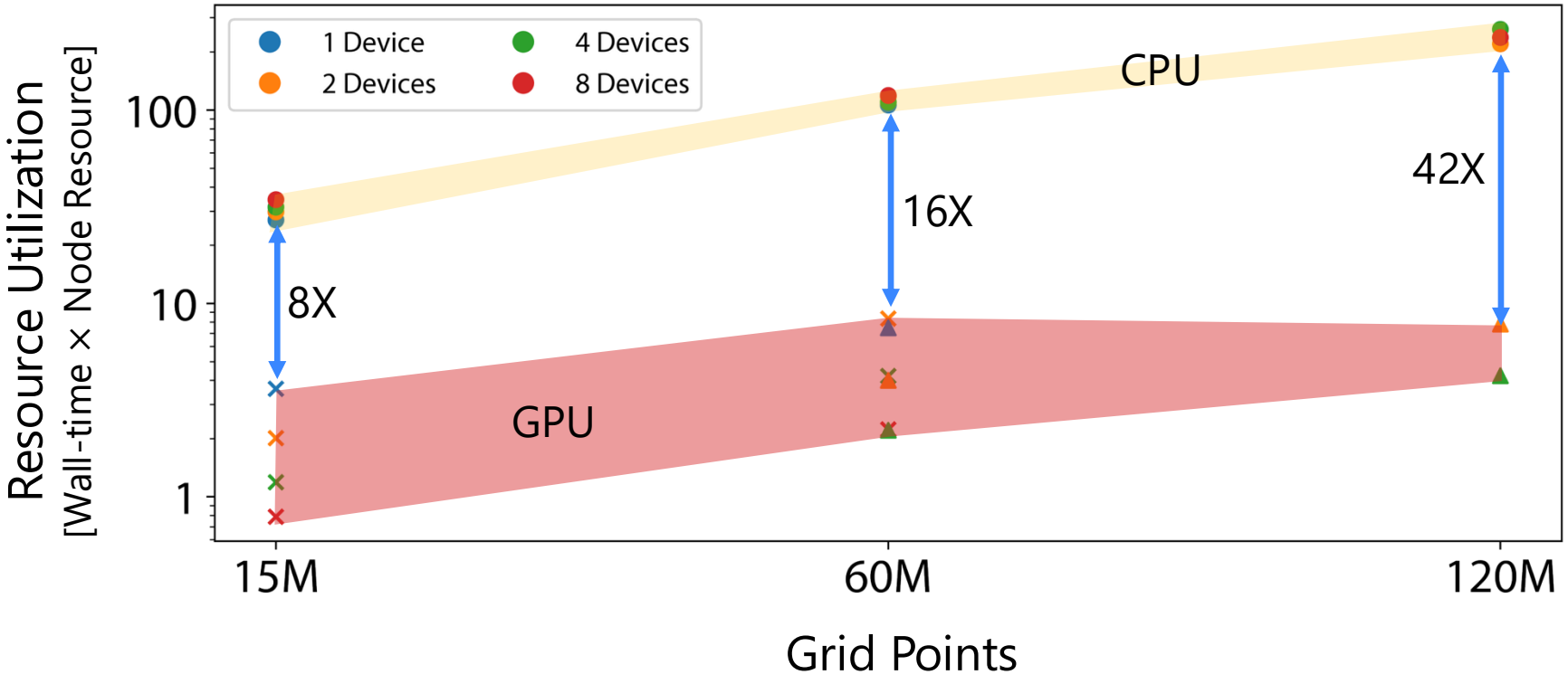

$$\frac{u_i^* - u_i^n}{\Delta t} + \frac{1}{2}\left[3N_i^n - N_i^{n-1}\right] = -\frac{1}{\rho}\frac{\delta p^n}{\delta x_i} + \frac{1}{2}\left(D_i^* + D_i^n\right)$$

$$\frac{1}{\rho}\frac{\delta}{\delta x_i}\left(\frac{\delta p^n}{\delta x_i}\right) = \frac{1}{\Delta t}\frac{\delta U_i^*}{\delta x_i}$$



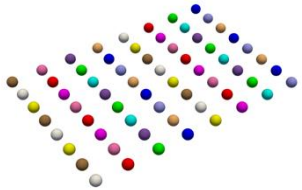



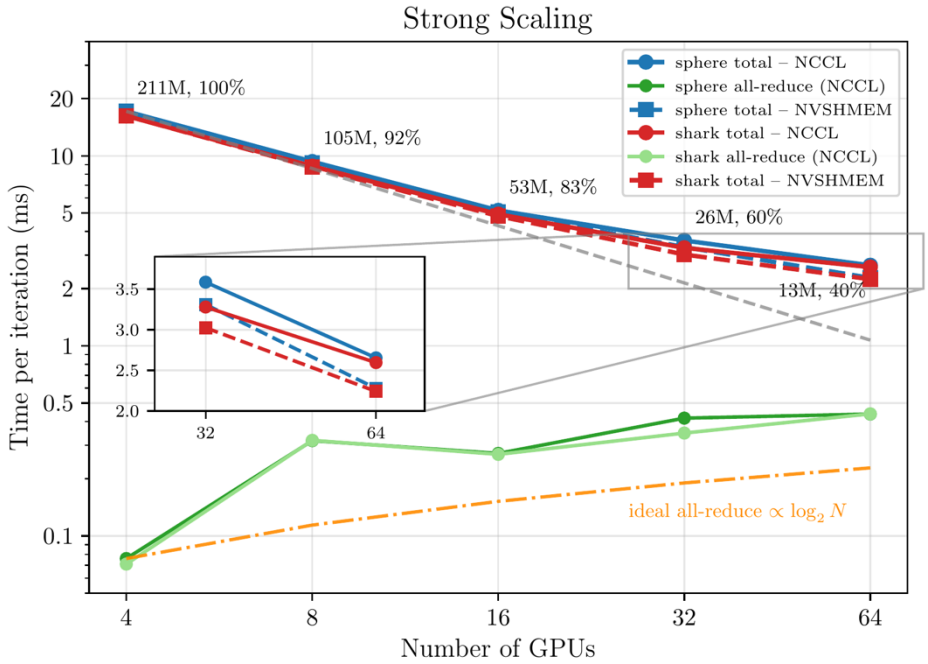
# 40X Resource Savings (A100 GPUs)

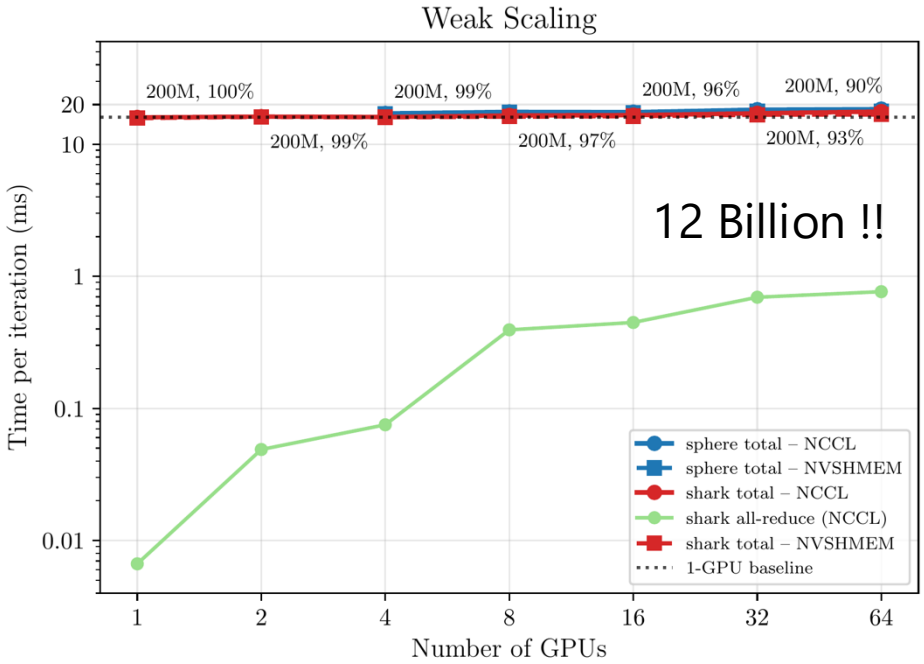


# Scaling Results





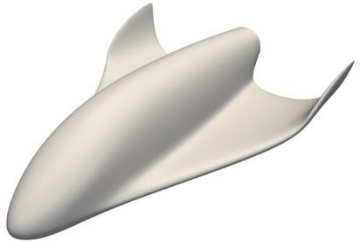


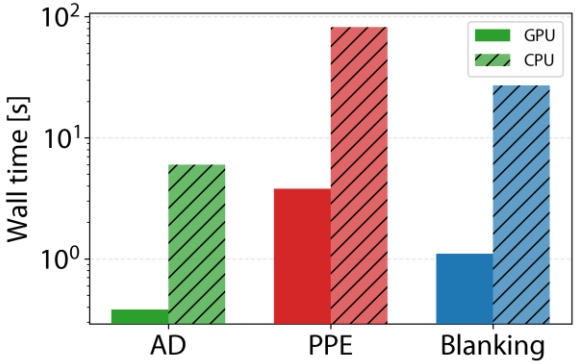



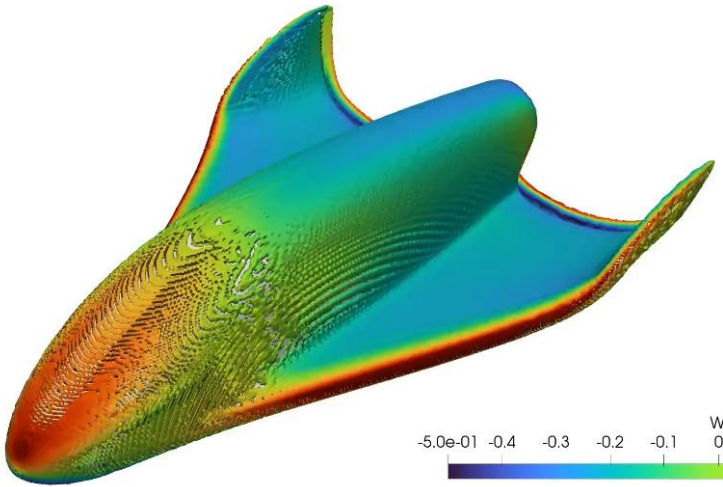

System – GB200 NVL72

# Flow over a Conceptual Flying Vehicle at $Re = 25,000$

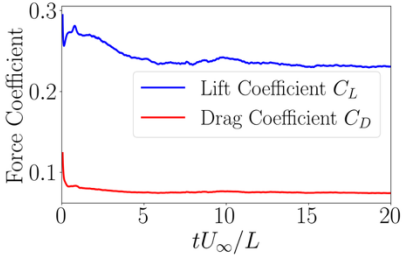
- Geometry – fetchcfd.com
- Grid – 800 X 500 X 484 ~ 194M
- Surface Mesh – 148457 Points
- 1 A100 Node – 4 GPUs PCIe Connected





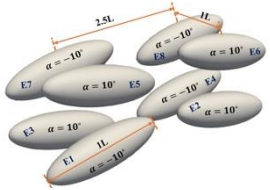



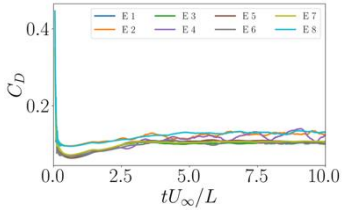


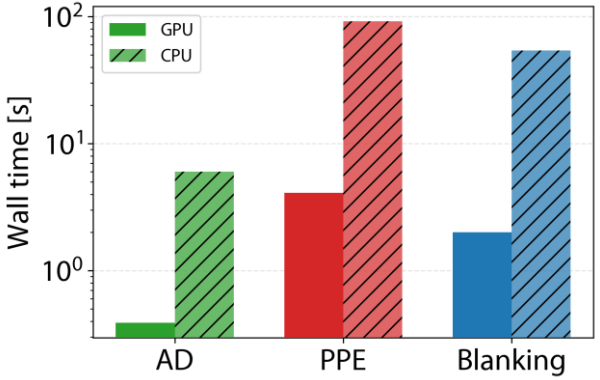



# Flow over an array of ellipsoidal particles at $Re$ = 10, 000

- Prolate Ellipsoid – 1:0.3
- Grid – 1300 X 400 X 400 ~ 208M
- Surface Mesh – 70000 Points
- 1 A100 Node – 4 GPUs PCIe Connected



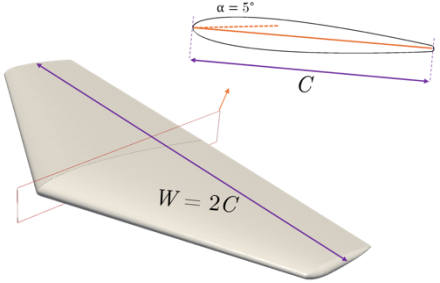



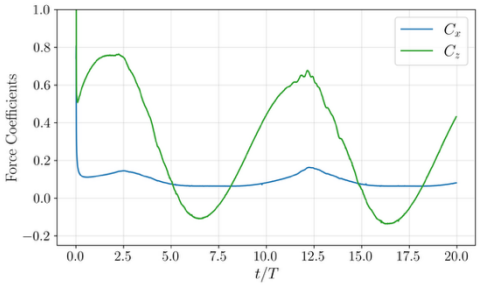


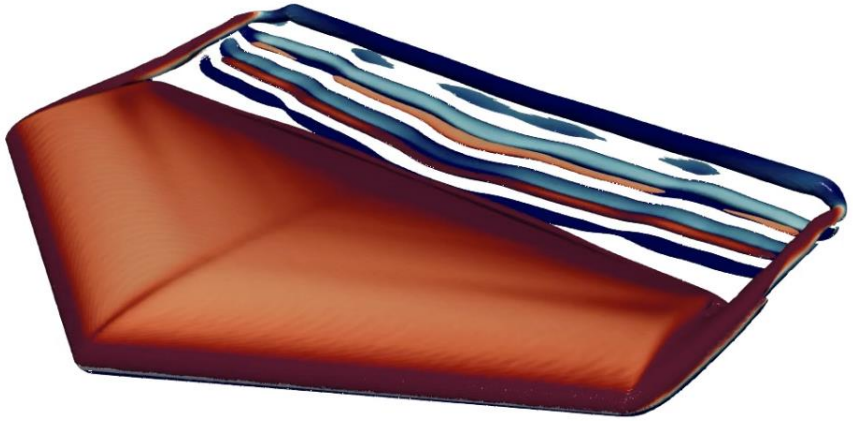

# Flow over a pitching wing at Re = 10000

- Grid – 1300 X 400 X 400 ~ 208M
- Surface Mesh – 70000 Points
- 4 RTX Blackwell 6000 GPU - PCIe
- $\Delta t = 0.0005$
- CFL < 0.3



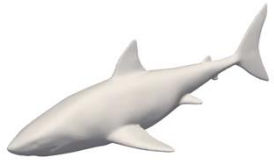

$$\alpha(t) = \alpha_m \sin\left(\frac{\pi t}{T}\right)$$



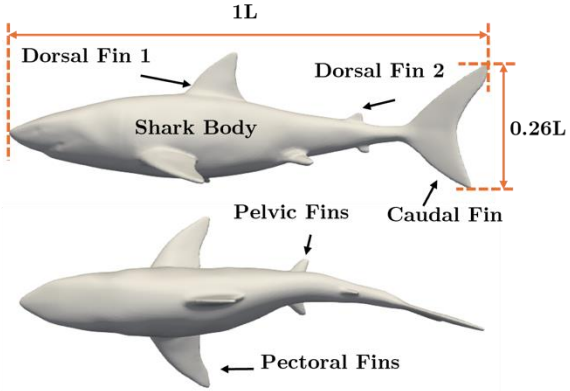



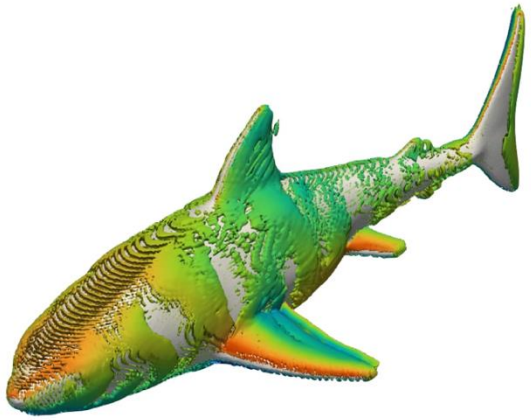
# Flow over a swimming shark at $Re = 30000$

- Grid – 1300 X 400 X 400 ~ 208M
- Surface Mesh – 70000 Points
- 4 RTX Blackwell 6000 GPU - PCIe
- $\Delta t = 0.0005$
- CFL < 0.3



$$v(x,t) = -kcA(x)\cos(k(x-ct)+\phi)$$



Seo and Mittal, 2002, Bioinsp. & Biomim.

# A GPU-Accelerated Sharp Interface Immersed Boundary Solver for Large Scale Flow Simulations

Sushrut Kumar, Joshua Romero, Jung-Hee Seo, Massimiliano Fatica and Rajat Mittal

**AIAA 2026-0705**
**Session: CFD on Large-Scale Meshes and Development of Testing Techniques**

Published Online: 8 Jan 2026 • https://doi.org/10.2514/6.2026-0705

Read Now Tools Share

**Abstract:**

Immersed boundary methods (IBMs) facilitate the simulation of flows around stationary, moving, and deforming bodies on Cartesian grids. However, extending these simulations to the large grid sizes required for realistic flow problems remains a significant computational challenge. In this work, we present the implementation and acceleration of ViCar3D, a sharp-interface immersed boundary solver, on graphical processing units (GPUs). We utilize OpenACC, CUDA Fortran and MPI to reprogram \emph{ViCar3D}, a sharp-interface immersed boundary solver, on multi-GPU architectures. Verification and scalability studies are performed for two benchmark cases: two-dimensional flow past a circular cylinder and direct numerical simulation (DNS) of flow past a finite rectangular wing. For the latter, we observe an approximately 20X speedup (node-to-node comparison) relative to the CPU-based implementation. The GPU-accelerated solver is capable of simulating complex 3D flows with up to 200 million mesh points on a single node equipped with four GPUs. Strong and weak scaling tests demonstrate maximum scaling efficiencies of 92\% and 93\%, respectively, on multi-GPU systems. We further test the code to simulate fluid flow past complex-shaped single-body and multi-body cases.

Previous Chapter Next Chapter

# GPU-Accelerated Simulations of Moving Boundary Problems and Fluid-Structure Interaction at Extreme Scales

Sushrut Kumar, Joshua Romero, Jung-Hee Seo, Massimiliano Fatica and Rajat Mittal

**AIAA 2026-4766**
**Session: CFD Methods for Applied Aerodynamics**

Published Online: 4 Jun 2026 • https://doi.org/10.2514/6.2026-4766

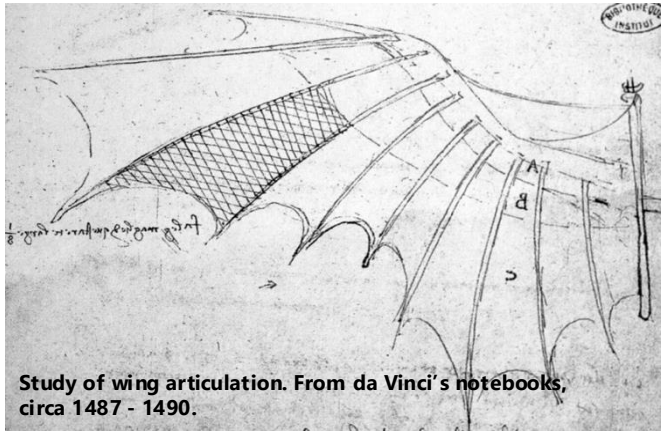
Read Now Tools Share

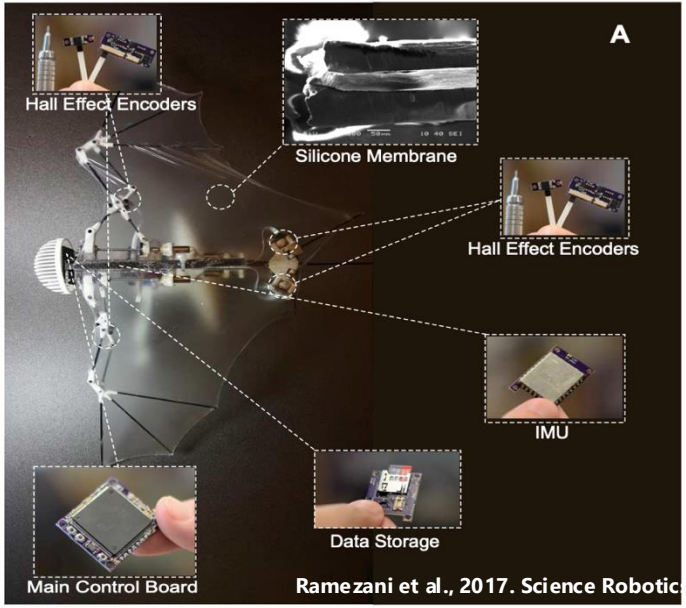
**Abstract:**

Computational fluid dynamics and fluid-structure interaction simulations involving moving and deforming bodies is extremely hard. In this work, we present a graphical processing unit (GPU) optimized implementation of the sharp-interface immersed boundary method. The method allows performing simulation around complex stationary as well as moving bodies on a Cartesian grid. We base our implementation on the ViCar3D framework and make use of OpenACC, CUDA, NCCL and MPI. We test the implementation across grid sizes ranging from O(10million) to O(1billion) points and achieved a 20X speedup compared to existing CPU implementation. We next present our multi-GPU implementation by utilizing CUDA streams and NCCL communicators. This enables us to obtain a >90% strong and weak scaling efficiencies. Next we demonstrate the capability of the developed software to simulate a turbulent fluid flow and coupled fluid-structure interaction in flapping bat wing in flight at Re=5000.

# Application of Methodology to Fluid-Structure Interaction Problems – Bat Membrane Wing

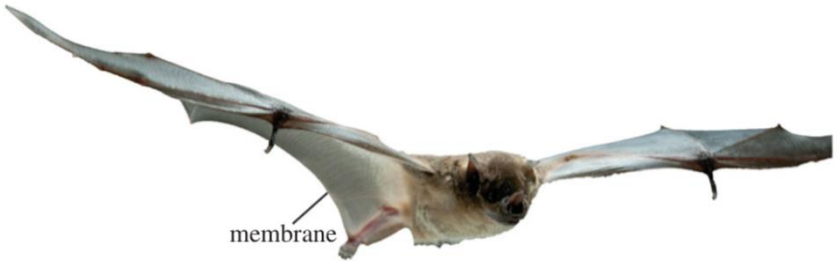


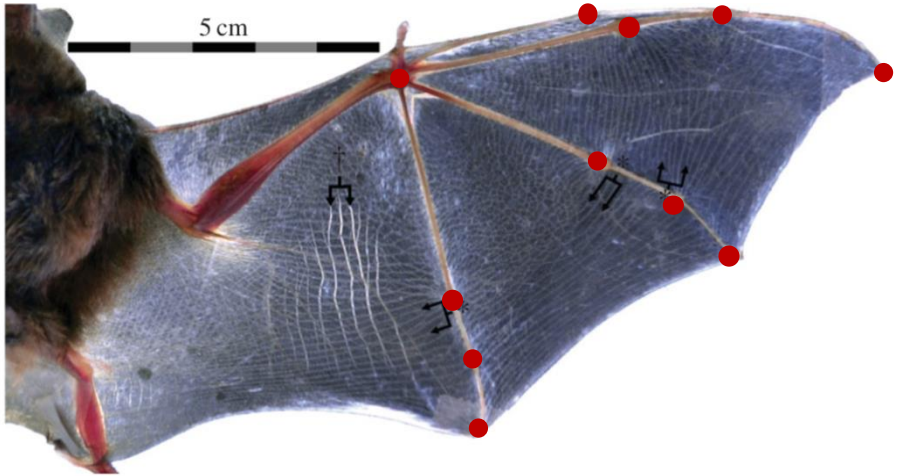


Study of wing articulation. From da Vinci's notebooks, circa 1487 - 1490.



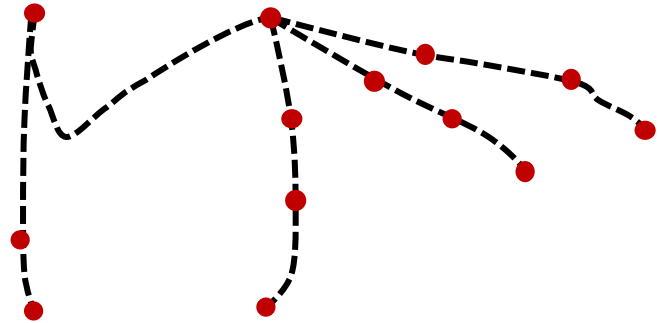
# Bat Wing

Bat flight very robust despite using a soft wing



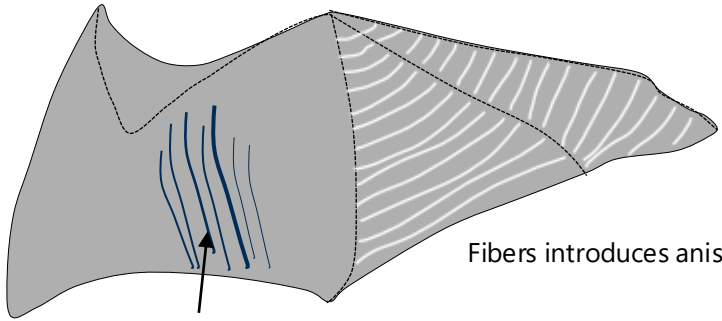




Skeleton



Joints along with muscle actuation provides localized control of wing twist and camber

Soft Membrane

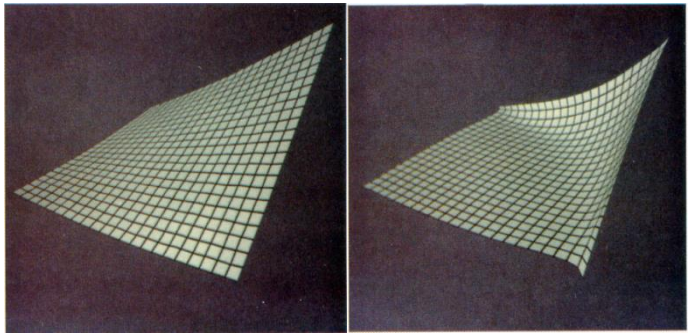


Fibers introduces anisotropy

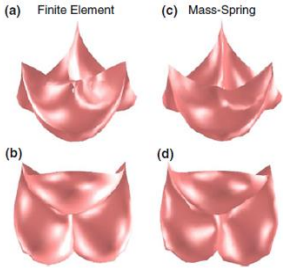
Muscles – Controls Wing Stiffness

Cheney et al., *Journal of the Royal Society Interface*(2015)

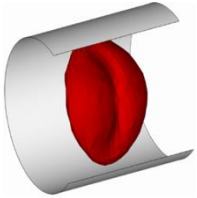
# Membrane Deformation Model



Terzopoulos, D., Platt, J., Barr, A. and Fleischer, K., 1987, August. In Proceedings of the 14th annual conference on Computer graphics and interactive techniques (pp. 205-214).



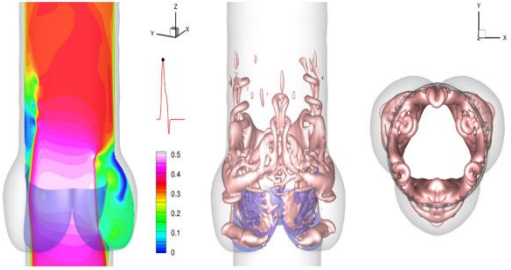

Hammer, P.E., Sacks, M.S., Del Nido, P.J. and Howe, R.D., 2011. Annals of biomedical engineering, 39(6), pp.1668-1679.



Fedosov, D.A., Caswell, B. and Karniadakis, G.E., 2010. Biophysical journal, 98(10), pp.2215-2225.



Bridson, R., Marino, S. and Fedkiw, R., 2005.. In ACM SIGGRAPH 2005 Courses (pp. 3-es).

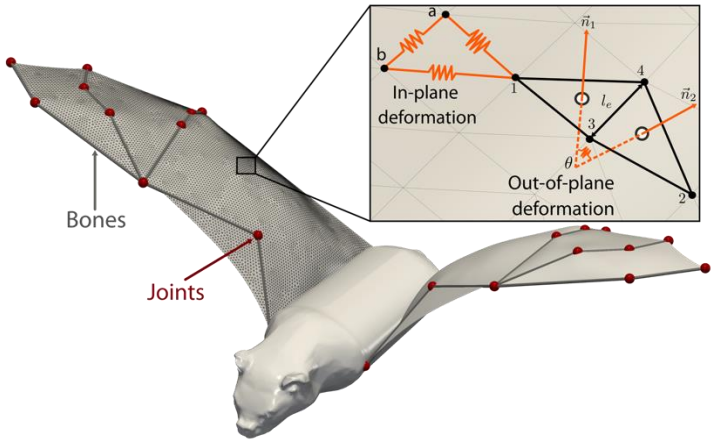


de Tullio, M.D. and Pascazio, G., 2016.. Journal of Computational Physics, 325, pp.201-225.



Spandan, V., et al., 2017. Journal of computational physics, 348, pp.567-590.

# Membrane Dynamics Solver




## Modeling membrane dynamics

$$M\frac{d^2\vec{X}_N}{d^2t} = \vec{F}_{ext} + \vec{F}_{int}$$

$$\vec{F}_{ext} = \frac{1}{3}(\Delta p\vec{n}A^e + \tau.\vec{s}A^e),$$

$$\vec{F}_{int} = \vec{F}_{elastic} + \vec{F}_{bending} + \vec{F}_{damp} \qquad \vec{F}_{damp} = C\frac{d\vec{X}_N}{dt}$$

- $\vec{X}_N$ - Node($N$) Position vector
- $M$ - Mass matrix
- $C$ - Damping matrix, viscoelastic damping can be incorporated using this.

## In-plane deformation (elastic forces)

$$\vec{F}^a_{elas} = -k_e(l - l_o)\frac{\vec{r}_{ab}}{l} \qquad \vec{F}^b_{elas} = -k_e(l - l_o)\frac{\vec{r}_{ba}}{l} \qquad k_e = \frac{Eh\Sigma_i A^e_i}{l^2}$$

## Out-of-plane deformation (bending forces)

$$\vec{F}^i_{bend} = k\frac{|\vec{l}_e|^2}{|\vec{n}_1| + |\vec{n}_2|}\left[\sin\left(\frac{\theta}{2}\right) - \sin\left(\frac{\theta_o}{2}\right)\right]m_i, \qquad i = 1,2,3\ \&\ 4$$

$$m_1 = |\vec{l}_e|\frac{\vec{n}_1}{|\vec{n}_1|^2} \qquad m_2 = |\vec{l}_e|\frac{\vec{n}_2}{|\vec{n}_2|^2} \qquad k = \frac{Eh^3}{12(1-\nu^2)}$$

$$m_3 = \frac{(\vec{x}_1 - \vec{x}_4)\cdot\vec{l}_e}{|\vec{l}_e|}\frac{\vec{n}_1}{|\vec{n}_1|^2} + \frac{(\vec{x}_2 - \vec{x}_4)\cdot\vec{l}_e}{|\vec{l}_e|}\frac{\vec{n}_2}{|\vec{n}_2|^2}$$

$$m_4 = -\frac{(\vec{x}_1 - \vec{x}_3)\cdot\vec{l}_e}{|\vec{l}_e|}\frac{\vec{n}_1}{|\vec{n}_1|^2} + \frac{(\vec{x}_2 - \vec{x}_3)\cdot\vec{l}_e}{|\vec{l}_e|}\frac{\vec{n}_2}{|\vec{n}_2|^2}$$

$\vec{n}_1, \vec{n}_2$ - surface normal vectors
$\vec{l}_e$ - shared edge


de Tullio, M.D. and Pascazio, G., 2016.. *Journal of Computational Physics*, *325*, pp.201-225.

Hammer, P.E., Sacks, M.S., Del Nido, P.J. and Howe, R.D., 2011. *Annals of biomedical engineering*, *39*(6), pp.1668-1679.

Bridson, R., *et al.*, 2005. In *ACM SIGGRAPH 2005 Courses* (pp. 3-es).

Gelder, A.V., 1998. *Journal of graphics tools*, *3*(2), pp.21-41.

# Coupling

- Weak Coupling – High Density Ratios
- Strong Coupling – Low Density Ratios

## Flow Solver (GPU)

(Eulerian Frame)

Initialize Fluid Solver and solve at 1st timestep

$t > t_{final}$ — Yes → End

$n \rightarrow n+1$ — No

Compute Hydrodynamic Forces

$\vec{F}_{ext}^{n}$

Move and Set Immersed Boundary (call structural solver)

$\vec{X}_N^n \rightarrow \vec{X}_N^{n+1}$

Solve Flow

$$\frac{\partial \vec{u}}{\partial t} + \nabla \cdot \vec{u}\vec{u} = -\nabla p + \frac{1}{Re}\nabla^2 \vec{u}$$

$$\nabla \cdot \vec{u} = 0$$

$t = t + 1$

$\vec{X}_N^{n+1}$

$\vec{F}_{ext}^{n}, \vec{X}_N^{n}$

## Structural Solver (CPU)

(Lagrangian Frame)

Obtain Hydrodynamic Forces

$\vec{F}_{ext}^{n}$

Predict Node($N$) Positions

$\vec{X}_N^{n+\frac{1}{2}} = \vec{X}_N^{n} + \frac{\Delta t}{2}\vec{U}_N^{n}$

Calculate Internal Forces

$\vec{F}_{int}^{n+\frac{1}{2}}\left(\vec{X}_N^{n+\frac{1}{2}}\right) = \vec{F}_{elastic}^{n+\frac{1}{2}} + \vec{F}_{bending}^{n+\frac{1}{2}}$

Obtain BCs

$|\vec{U}_N^{k+1} - \vec{U}_N^{k}| < \epsilon$ — Yes → $\vec{U}_N^{n+1} = \vec{U}_N^{k+1}$

$k \rightarrow k+1$ — No

Calculate Damping Force

$\vec{F}_D = -0.5C\vec{U}_N^{k} - 0.5C\vec{U}_N^{k+1}$

Solve

$M\vec{U}_N^{k+1} = R(\vec{U}_N^{n}, \vec{F}_{int}^{n+\frac{1}{2}}, \vec{F}_{ext}^{n} + \vec{F}_D)$

Correct Node Velocities

$\vec{X}_N^{n+1} = \vec{X}_N^{n} + \frac{\Delta t}{2}(\vec{U}_N^{n} + \vec{U}_N^{n+1})$

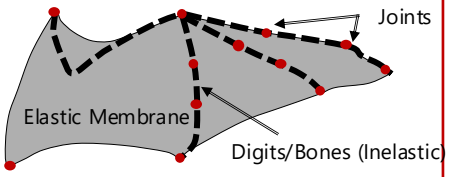
$k = k + 1$

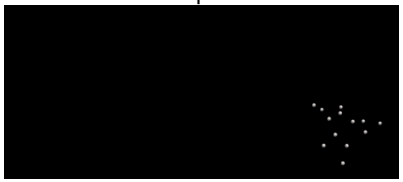
$BCs$

$t$

## Computing Boundary Conditions

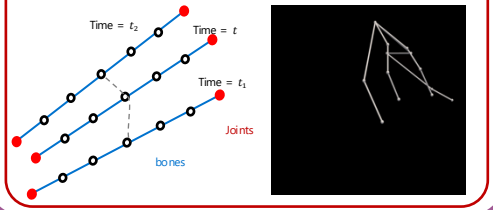




Courtesy of Dr. Dan Riskin and Swartz Lab, Brown University

Reconstruction using cubic spline interpolation



# Flapping Flag

Simulation Parameters

$L = 1$

$h = 0.01L$

$\frac{\rho_s h}{\rho_f L} = 1$

$\beta = \frac{Eh^3}{12(1-\nu^2)\rho_f U_\infty^2 L^3} = 0.0001$

$Re = 200$

$\frac{\Delta_{min}}{L} = 0.02$

$\frac{\Delta t U_\infty}{L} = 0.0001$

$L_x \times L_y \times L_z = 8L \times 8L \times 2L$

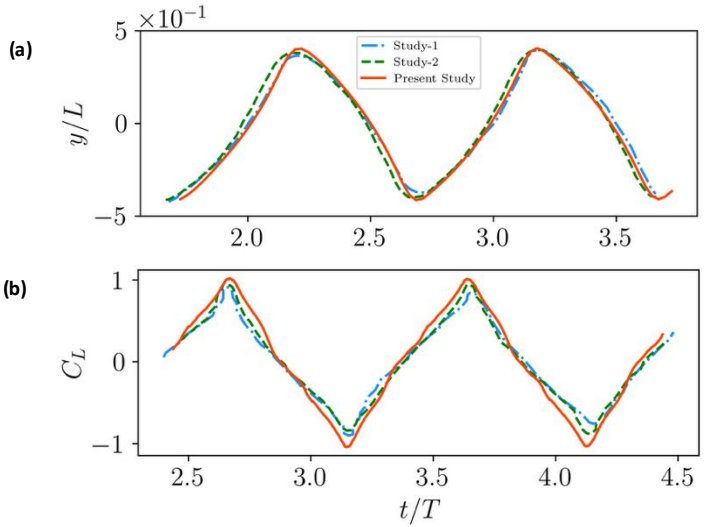
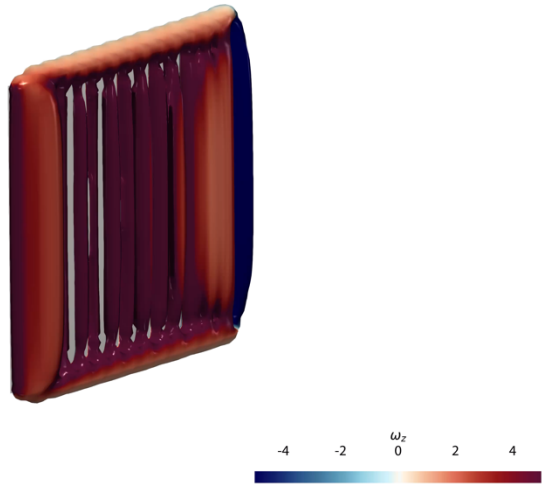



fig : (a) Displacement of midpoint at trailing edge of flag; (b) Time variation of $C_L$



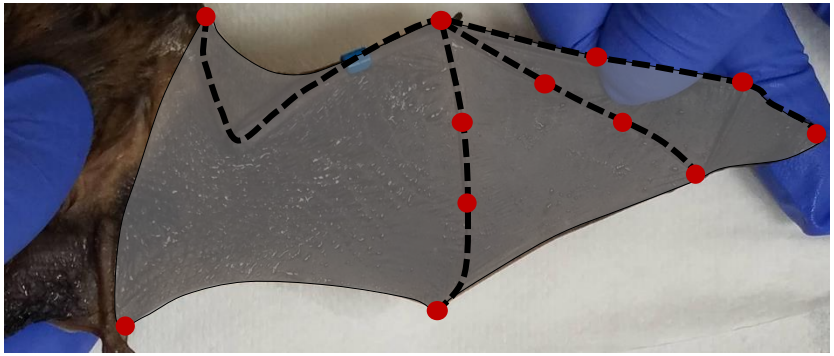

| | Present Study | Study 1 | Study 2 |
|---|---|---|---|
| $A/L$ | 0.819 | 0.812 | 0.795 |
| $St$ | 0.274 | 0.263 | 0.265 |

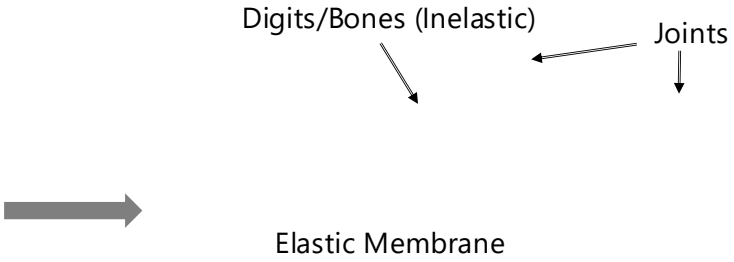
**Study 1** - Tian, F.B., *et al.*, 2014. *Journal of computational physics*, *258*, pp.451-469.
**Study 2** - de Tullio, M.D. and Pascazio, G., 2016.. *Journal of Computational Physics*, *325*, pp.201-225.

# Our Methodology

**Membrane** – Fluid-Structure Interaction

**Joints** – From experiment






**Bone joint kinematics from experiments**

Riskin, D.K., *et al.*, 2008. Quantifying the complexity of bat wing kinematics. *Journal of theoretical biology*, *254*(3), pp.604-615.



Courtesy of Dr. Dan Riskin and Swartz Lab, Brown

# Simulation Results "Pteropus Pumilus"

## Kinematics

$$St = \frac{fA}{U_\infty} = 0.25$$

$$\overline{U} = 3.71\ m/s$$

$$\frac{dU}{dt} = -1.05\ m/s^2$$

## Structural Solver

$$\frac{\rho_s}{\rho_f} = 1000$$

$$\frac{E}{\rho_f \overline{U}^2} = 10^6$$

$$h = 0.01A$$

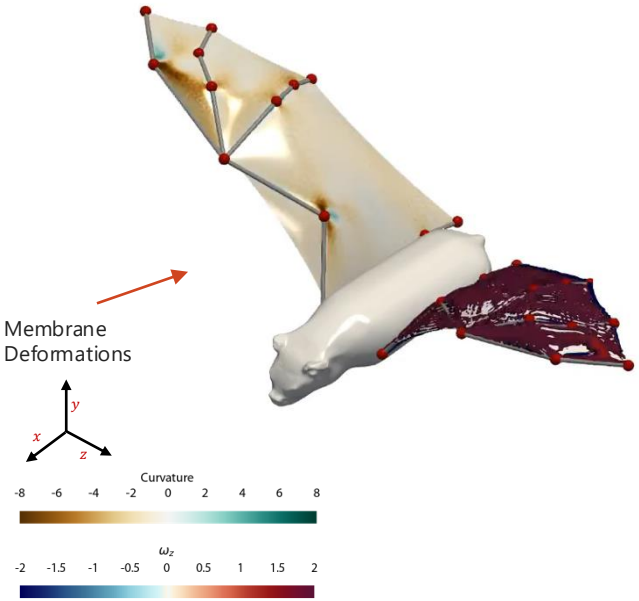
## Flow Solver

$$Re = \frac{\overline{U}A}{\nu} = 1000$$

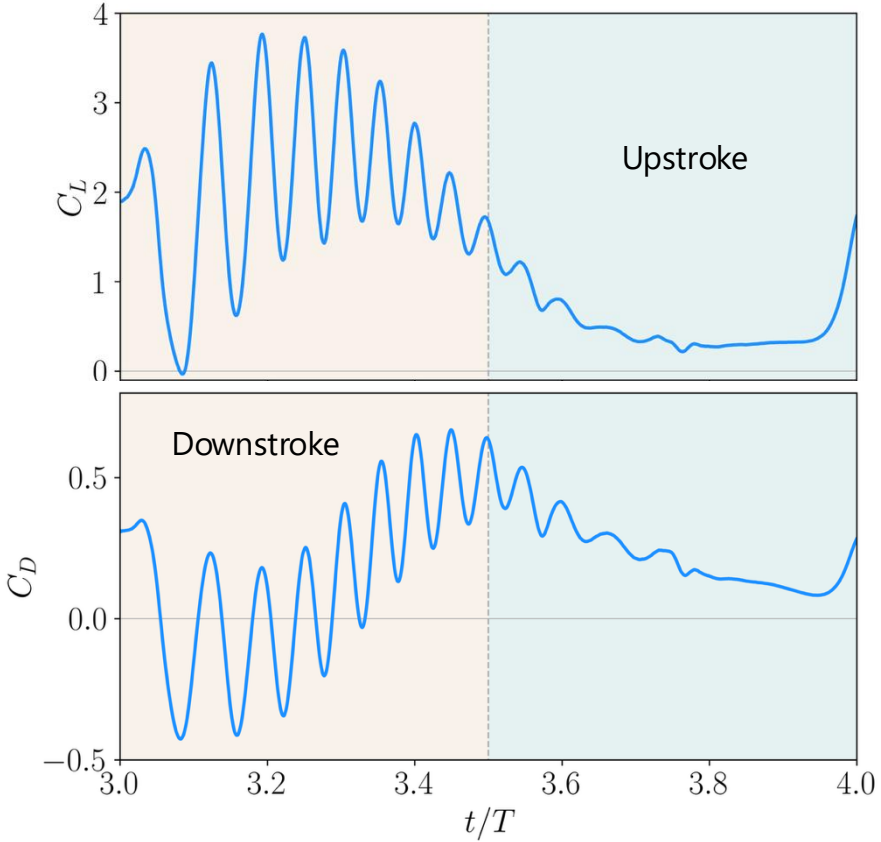
$$\frac{\Delta_{min}}{A} = 0.005$$

$$\frac{\Delta t U_\infty}{A} = 0.001$$

| | Downstroke | Upstroke | Total |
|---|---|---|---|
| $\overline{C_L}$ | 2.04 | 0.54 | 1.29 |
| $\overline{C_D}$ | 0.14 | 0.23 | 0.19 |

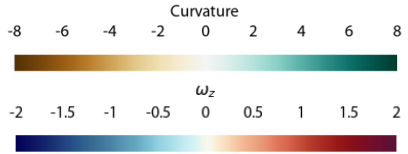


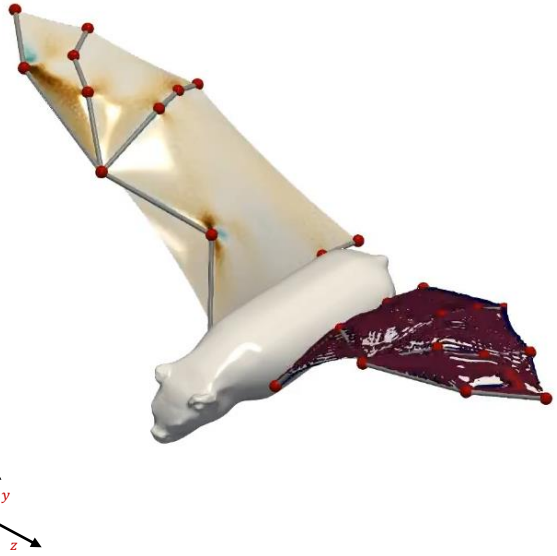

Isosurfaces of Q-Criterion



# What generates pressure forces on the bat wing??

Iso-surfaces of $Q$ −criterion colored by vorticity




$$\vec{\nabla} \cdot \vec{u} = 0$$

$$\vec{\nabla} p = -\rho \frac{\partial \vec{u}}{\partial t} - \rho \vec{u} \cdot \vec{\nabla} \vec{u} + \mu \nabla^2 \vec{u}$$

$$F_p = -\int_{Body} p\, \vec{n}\, dS$$

- Added Mass (self, freestream, other bodies)
- Viscous Diffusion
- Vortex Induced
  - Vortex 1
  - Vortex 2
  - ...

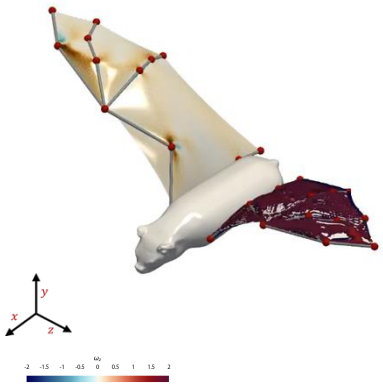
- Pressure is an elliptic variable
- Simultaneously affected by everything
- How to partition the effect of various features/mechanisms ?

# Application of FPM to Bat FSI Dataset

Zhang, C., et al., 2015. PloS one , Menon, K. et al., 2022 Physical Review Fluids

Flow field

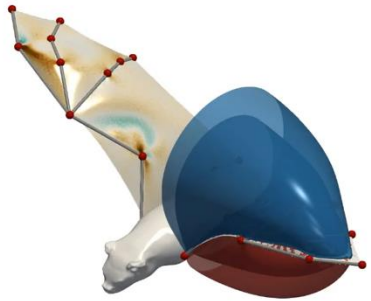
$$\frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} = -\nabla p + \frac{1}{Re} \nabla^2 \vec{u}$$



+

Influence field for lift force

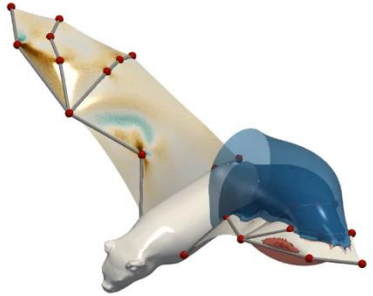
$$\nabla^2 \phi = 0 \qquad \hat{n}.\nabla\phi = \begin{cases} n_y & \text{on } B \\ 0 & \text{on } \Sigma \end{cases}$$



/

Influence field for Drag force

$$\nabla^2 \phi = 0 \qquad \hat{n}.\nabla\phi_i = \begin{cases} n_x & \text{on } B \\ 0 & \text{on } \Sigma \end{cases}$$



$$\int_v \left( \frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} = -\nabla p + \frac{1}{Re} \nabla^2 \vec{u} \right) . \nabla\phi \, dv \Rightarrow \int_S p\vec{n}_i dS = - \int_S \hat{n}.\left( \frac{d\vec{U}_B}{dt} \phi \right) dS - \int_V 2Q\phi dV + \frac{1}{Re} \int_V \nabla^2 \vec{u}.\nabla\phi \, dV - \int_\Sigma \hat{n}.\left( \frac{d\vec{U}_\infty}{dt} \phi \right) d\Sigma$$

Pressure Forces

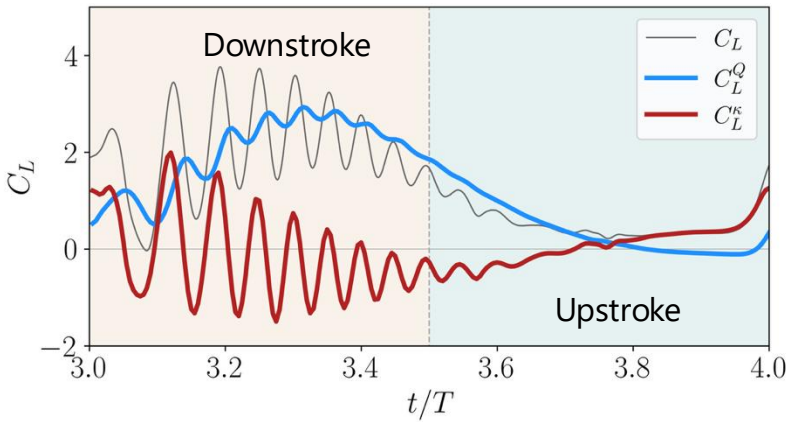
Added Mass Forces

Vortex Induced Forces

Viscous Diffusion Induced Forces

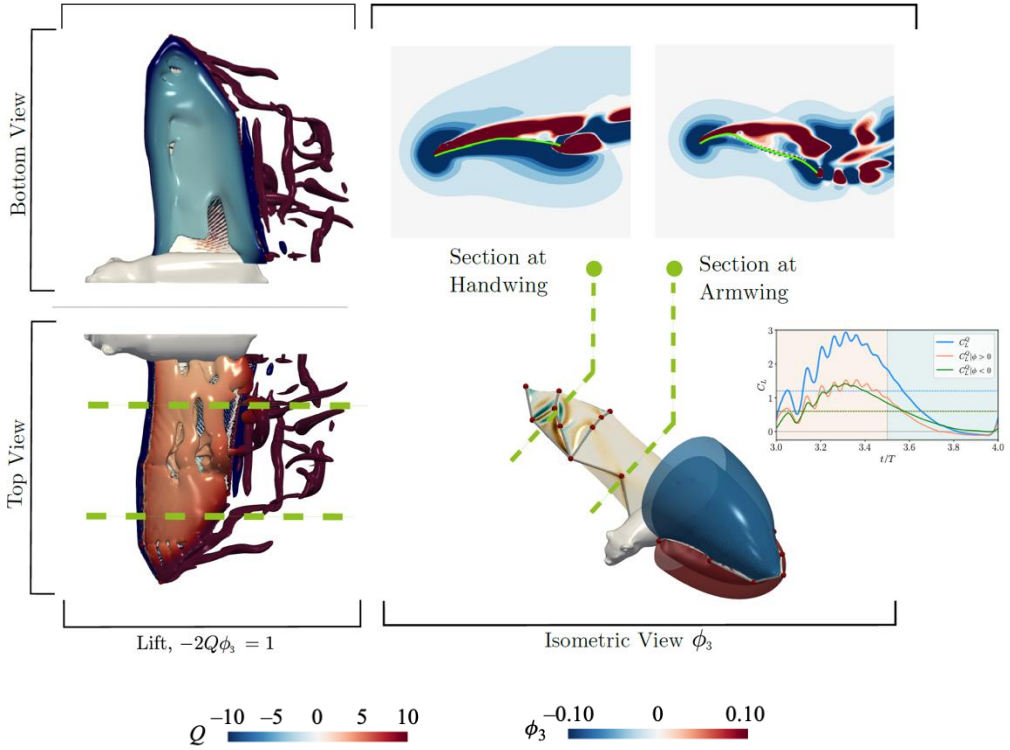
Freestream acceleration Induced Forces

# Decomposition of Lift



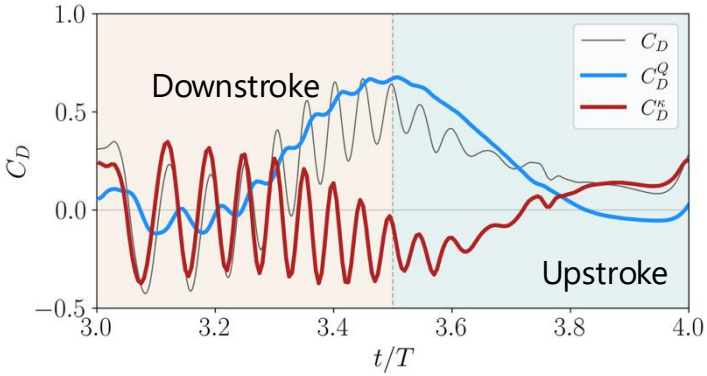

| | Downstroke | Upstroke | Total |
|---|---|---|---|
| $\overline{C_L}$ | 2.04 | 0.54 | 1.29 |
| $\overline{C_L^Q}$ | 1.97 | 0.42 | 1.20 |
| $\overline{C_L^\kappa}$ | -0.09 | 0.06 | -0.01 |

## Isosurfaces of Vortex-Induced Lift



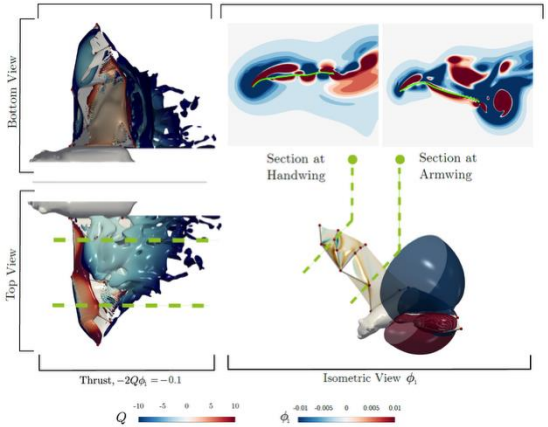

*Menon & Mittal, JFM (2021)*

# Decomposition of Drag

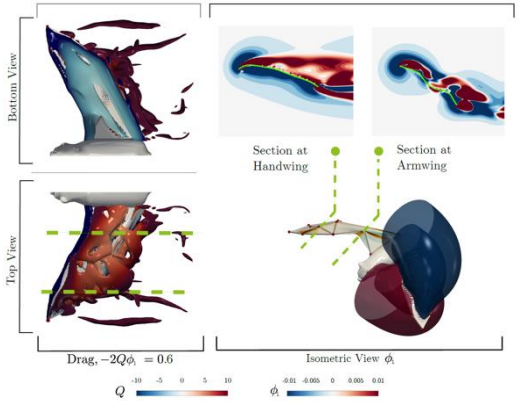


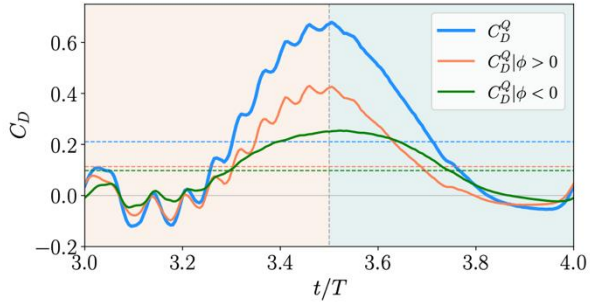

| | Downstroke | Upstroke | Total |
|---|---|---|---|
| $\overline{C_D}$ | 0.14 | 0.23 | 0.19 |
| $\overline{C_D^Q}$ | 0.21 | 0.21 | 0.21 |
| $\overline{C_D^\kappa}$ | -0.04 | -0.005 | -0.02 |

Isosurfaces of Vortex-Induced Thrust

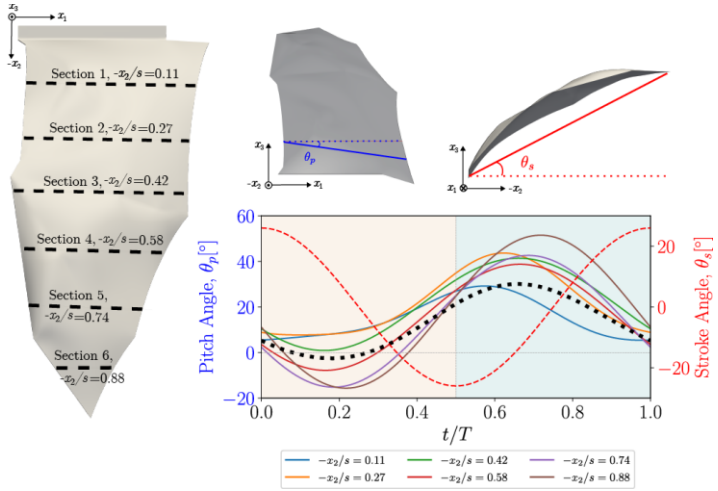


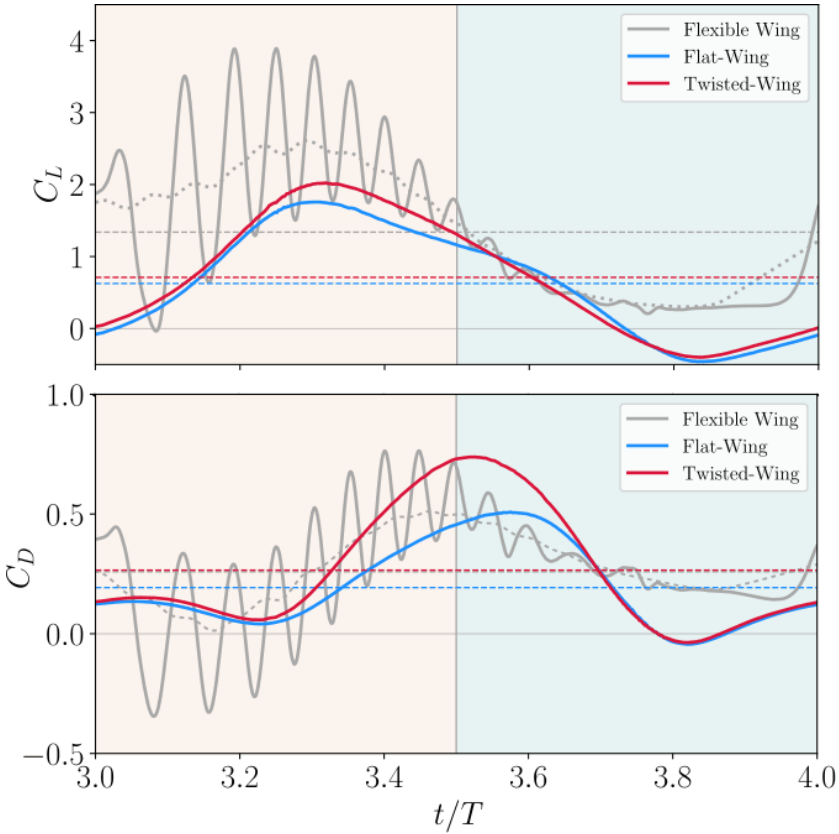

Isosurfaces of Vortex-Induced Drag



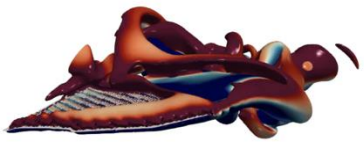



# Comparison to Equivalent Stiff Wings

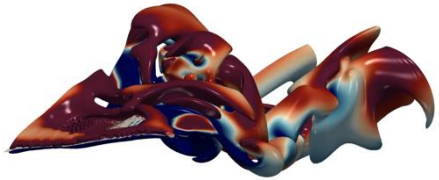


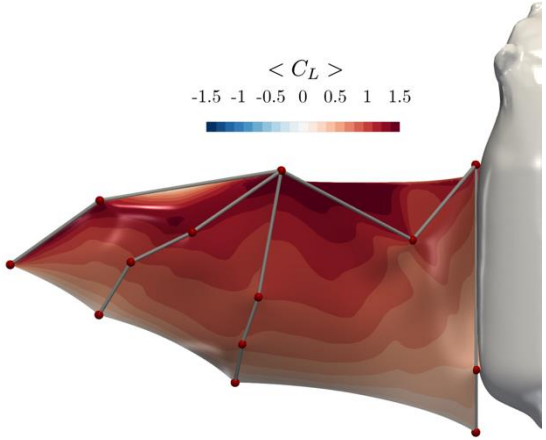

Flapping with fixed twist

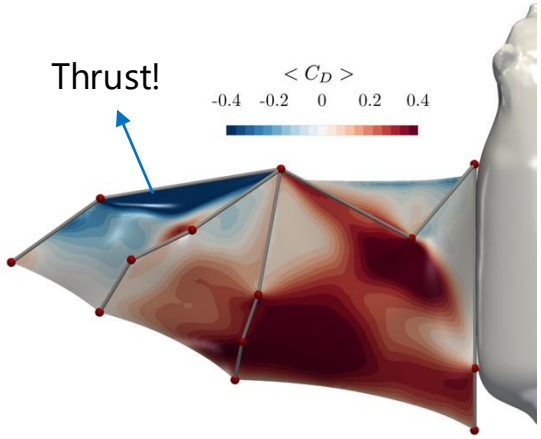


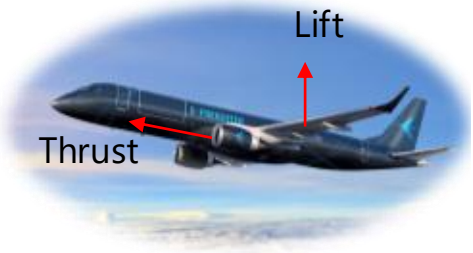
Flapping with varying twist



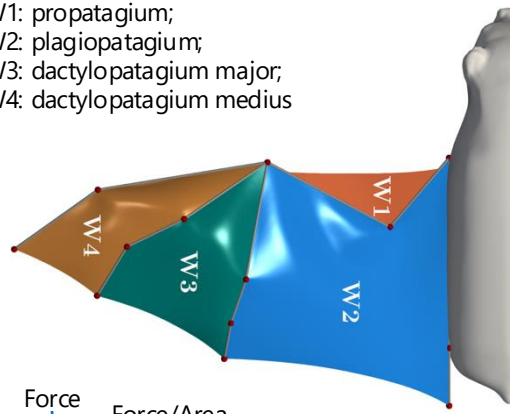
# Wing Function is Segregated

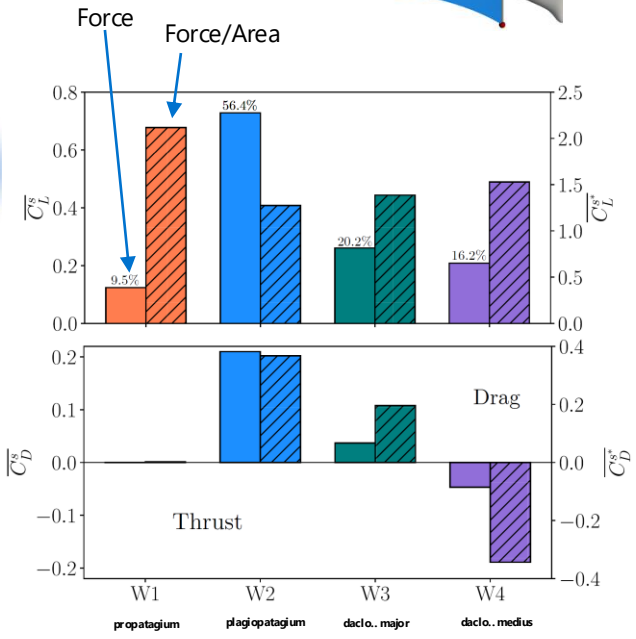



W1: propatagium;
W2: plagiopatagium;
W3: dactylopatagium major;
W4: dactylopatagium medius





Journal of Fluid Mechanics

## Article contents

Abstract
Introduction
Computational model
Results
Conclusions
Supplementary movie.
Funding
Declaration of interests
References

# Computational modelling and analysis of the coupled aero-structural dynamics in bat-inspired wings

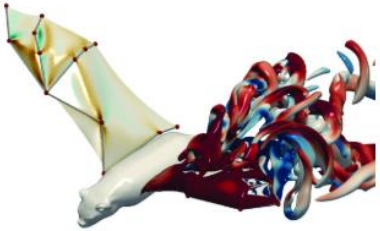
Published online by Cambridge University Press: **09 May 2025**

Sushrut Kumar, Jung-Hee Seo and Rajat Mittal

Show author details



## Abstract



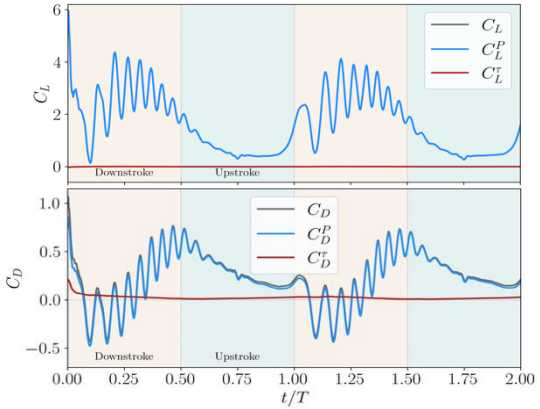
We employ a novel computational modelling framework to perform high-fidelity direct numerical simulations of aero-structural interactions in bat-inspired membrane wings. The wing of a bat consists of an elastic membrane supported by a highly articulated skeleton, enabling localised control over wing movement and deformation during flight. By modelling these complex deformations, along with realistic wing movements and interactions with the surrounding airflow, we expect to gain new insights into the performance of these unique wings. Our model achieves a high degree of realism by incorporating experimental measurements of the skeleton's joint movements to guide the fluid–structure interaction simulations. The simulations reveal that different segments of the wing undergo distinct aeroelastic deformations, impacting the flow dynamics and aerodynamic loads. Specifically, the simulations show significant variations in the effectiveness of the wing in generating lift, drag and thrust forces across different segments and regions of the wing. We employ a force partitioning method to analyse the causality of pressure loads over the wing, demonstrating that vortex-induced pressure forces are dominant while added-mass contributions to aerodynamic loads are minimal. This approach also elucidates the role of various flow structures in shaping pressure distributions. Finally, we compare the fully articulated, flexible bat wing with equivalent stiff wings derived from the same kinematics, demonstrating the critical impact of wing articulation and deformation on aerodynamic efficiency.

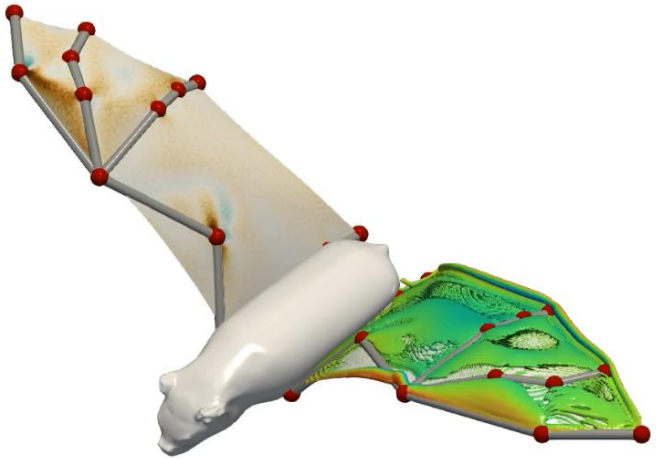
# Fluid-Structure Interaction in Bat Membrane Wing at Re=5000

| Kinematics | Structural Solver | Flow Solver |
|---|---|---|
| $St = \frac{fA}{U_\infty} = 0.25$ | $\frac{\rho_s}{\rho_f} = 1000$ | $Re = \frac{\overline{U}A}{\nu} = 5000$ |
| $\overline{U} = 3.71\ m/s$ | $\frac{E}{\rho_f \overline{U}^2} = 10^6$ | $\frac{\Delta_{min}}{A} = 0.0035$ |
| $\frac{dU}{dt} = -1.05\ m/s^2$ | $h = 0.01A$ | $\frac{\Delta t U_\infty}{A} = 0.0005$ |

**Flow over a pitching wing at Re = 5000**

- Grid – 452M
- 1 RTX Blackwell Max-Q Node (4 GPUs)



## Conclusion

- Presented the GPU-implementation of Sharp-Interface Immersed Boundary Method
- Demonstrated 20X speedup compared to 30-year-old optimized CPU implementation
- Demonstrated the capability to perform DNS involving stationary and moving bodies.
- Discussed the flow physics associated with flapping membrane wings.

## Funding


- National Science Foundation
- Office of Naval Research
- NVIDIA Academic Research Grant
- HPC Resources through NSF Access Discover, NVIDIA Academic Research Grant, DoD HPC Modernization program.