UCLA Samueli School of Engineering

Structural Simulation of Jellyfish Propulsion

Jacob Kaminsky and Jake Kremer
Professor M. Khalid Jawed, UCLA Department of Mechanical and Aerospace Engineering

Project Overview

Jellyfish achieve **efficient swimming** by **contracting their bell** to expel water, followed by a **passive relaxation** phase.

Goal: Simulate this **contraction-relaxation cycle** of a jellyfish swimming through water using Python

- 2D model with connected nodes

Implementation

Basic Overview: Simulates jellyfish motion by solving time-dependent forces and updating node positions.

- Key components:
    1. **Node angles** (getK) for contraction/relaxation cycles.
    2. **Velocity direction** (velocityDirection) for drag alignment.
    3. **Newton's method** (objfun) to update positions via force equations.

High Reynolds Number Regime:

- Jellyfish operate in **high Reynolds Number regime** (inertia-dominated flow)
- **Drag Force**:

$$F_D = -C_D A \cdot \frac{1}{2} \rho v^2 \cdot \hat{u}$$

Projected Area:

$$A = \Delta L \cdot r_0$$

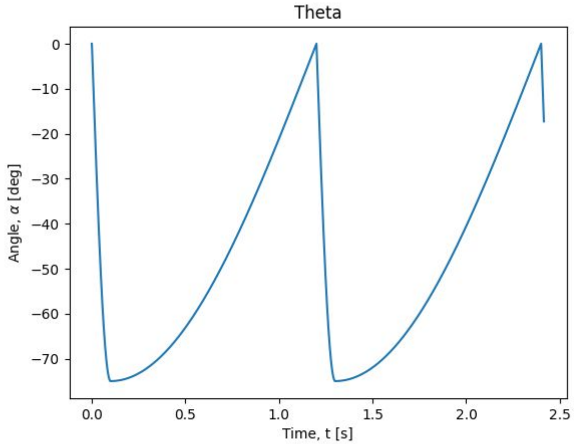
Why It Matters:

- Accurately replicates **jellyfish propulsion** for insights into efficient underwater movement.
- Applications in **biomimetic robotics** and fluid dynamics research.

UCLA Samueli School of Engineering

Implementation - getK Function

Purpose: Simulates the movement of a jellyfish by calculating the **adjustment angle** (**k**) for each node based on time (t) and its position relative to the midpoint



1. **Define Jellyfish Motion Cycle:**
   - **Period:** 1.2 seconds (complete cycle duration).
   - **Activation phase:** 0.1 seconds (controls contraction phase).
2. **Calculate Node Angle (k):**
   - If within **activation phase**, use a **sine function** to model rapid contraction
   - Outside activation, use a **cosine function** for gradual relaxation.
3. **Node-Specific Adjustment (k_Node):**
   - Distributes the angle across nodes, with:
     - **Midpoint node** contributing the maximum adjustment.
     - Nodes farther from the midpoint scaling down non-linearly

$$k = \begin{cases} \text{MaxAngle} \cdot \sin\left(\frac{\pi}{2} \cdot \frac{t_{\text{mod}}}{\text{Activation}}\right), & \text{if } 0 \leq t_{\text{mod}} < \text{Activation} \\ \text{MaxAngle} \cdot \cos\left(\frac{\pi}{(\text{Period} - \text{Activation}) \cdot \text{CosStretch}} \cdot (t_{\text{mod}} - \text{Activation})\right), & \text{if Activation} \leq t_{\text{mod}} < \text{Period} \end{cases}$$

UCLA Samueli School of Engineering

$$k_{\text{Node}} = \frac{k}{\frac{\text{MidIndex}}{2} + \frac{\text{MidIndex} - (\text{MidIndex} \cdot 0.1) \cdot |\text{Node} - \text{MidIndex}|^2}{\text{MidIndex}}}$$

Implementation - velocityDirection Function

Purpose: Computes the **unit vector of velocity** (v_hat) for each node at a given time step. Ensures the **drag force** is applied directly opposite to the node's movement.

1. **Calculate Velocity Vector (v_dir):**
   - Represents the change in position between the current (q_new) and previous (q_old) time steps for the node in 2D space
2. **Normalize Velocity Vector (v_hat):**
   - If the velocity vector has a non-zero magnitude, normalize it to get the unit vector.
   - Otherwise, set the unit vector to zero

$$\mathbf{v}_{\text{dir}} = \begin{bmatrix} q_{\text{new}}[2 \cdot \text{node}] - q_{\text{old}}[2 \cdot \text{node}] \\ q_{\text{new}}[2 \cdot \text{node} + 1] - q_{\text{old}}[2 \cdot \text{node} + 1] \\ 0 \end{bmatrix}$$

$$\mathbf{v}_{\text{hat}} = \begin{cases} \dfrac{\mathbf{v}_{\text{dir}}}{\|\mathbf{v}_{\text{dir}}\|}, & \text{if } \|\mathbf{v}_{\text{dir}}\| \neq 0 \\ \mathbf{0}, & \text{if } \|\mathbf{v}_{\text{dir}}\| = 0 \end{cases}$$

UCLA Samueli School of Engineering

Implementation - TimeLoop

Purpose: Governs the **time-stepped simulation** of the jellyfish's motion. Iteratively calculates node positions, angles, and velocities at each time step to mimic jellyfish movement.

1. **Run objfun:**
   - **Calculates bending and stretching energy.**
   - **Determines drag force** acting on the jellyfish.
   - **Newton's method update** to find the new configuration (q) and error convergence.
2. **Update Node Positions:**
   - For each node:
     - Use getK to calculate the **angle** (theta_Node) based on time.
     - Calculate **displacements** in x and y (x, y) relative to the midnode using trigonometric relationships.
     - Adjust positions **left or right** of the midnode to reflect the jellyfish's shape.

$$x = |(\text{midNode} - 1) - i| \cdot \Delta L \cdot \cos\left(|\theta_{\text{Node}}| \cdot \frac{\pi}{180}\right) \qquad y = |(\text{midNode} - 1) - i| \cdot \Delta L \cdot \sin\left(|\theta_{\text{Node}}| \cdot \frac{\pi}{180}\right)$$

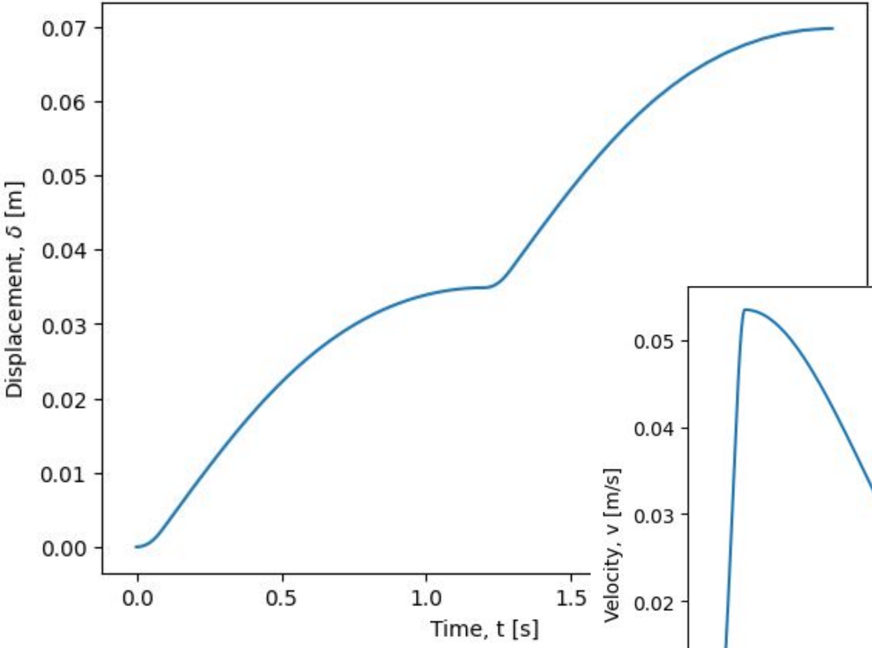
For nodes **left** of the midnode:

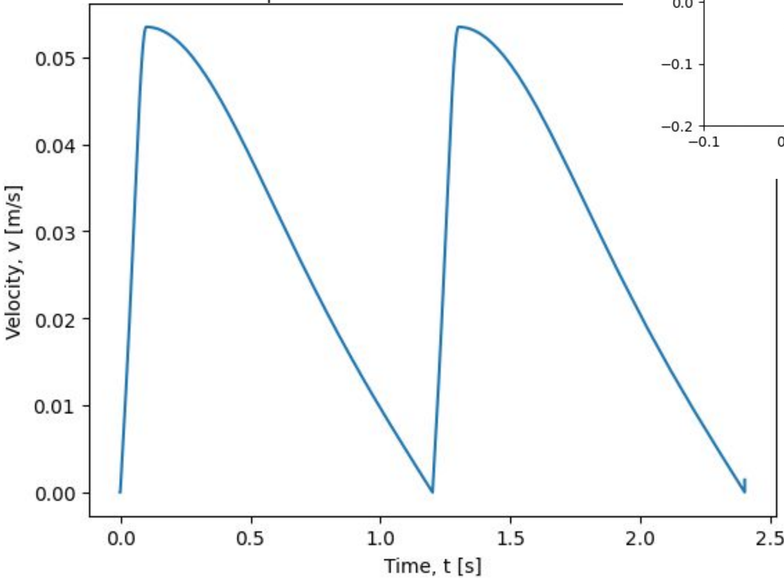
$$q[2i] = q[2 \cdot \text{midNode} - 2] - x$$
$$q[2i+1] = q[2 \cdot \text{midNode} - 1] - y$$

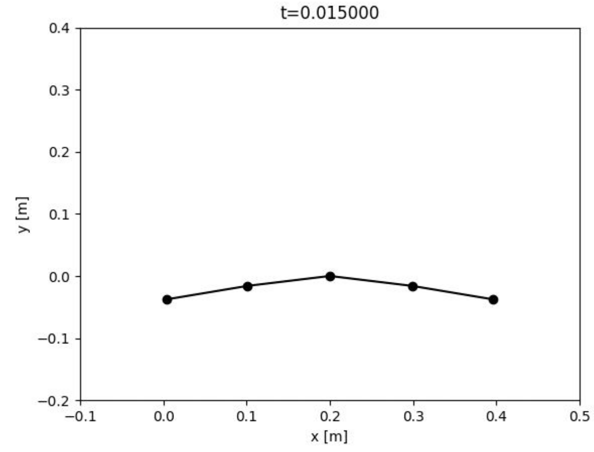
For nodes **right** of the midnode:

$$q[2i] = q[2 \cdot \text{midNode} - 2] + x$$
$$q[2i+1] = q[2 \cdot \text{midNode} - 1] - y$$

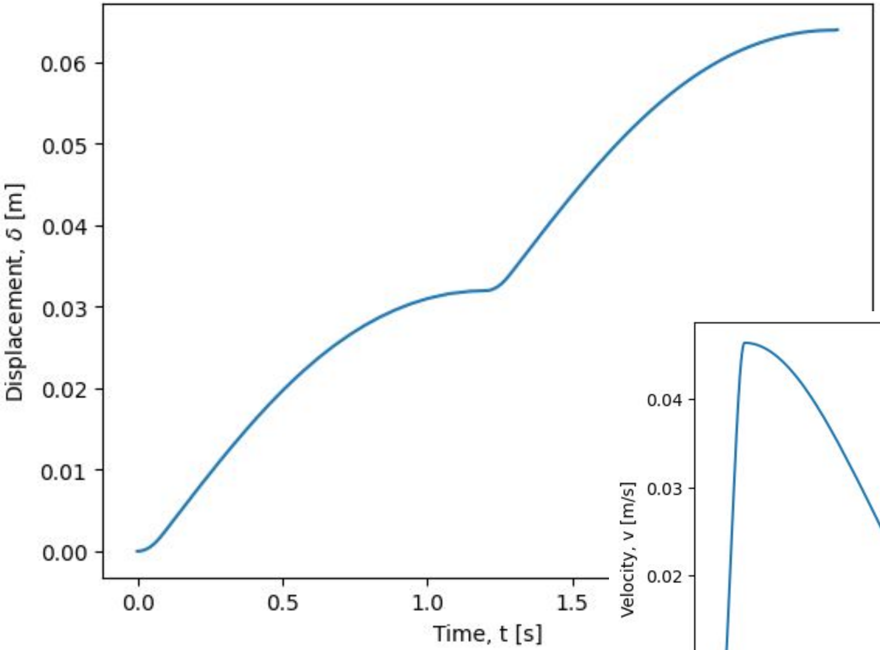
Results - 5 Nodes

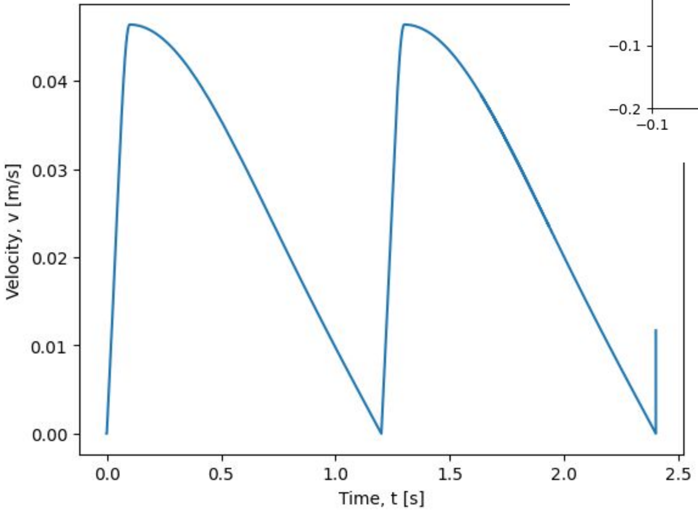


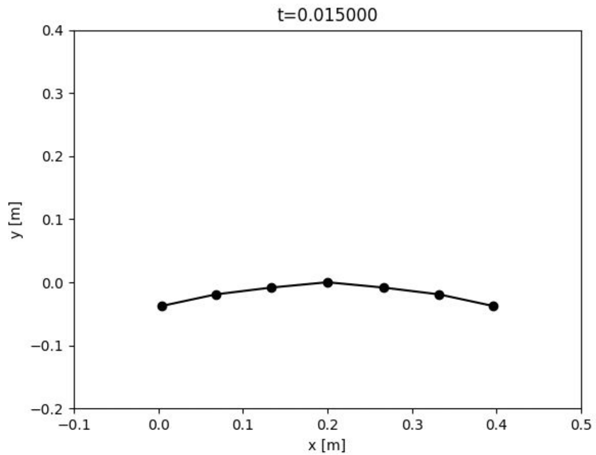




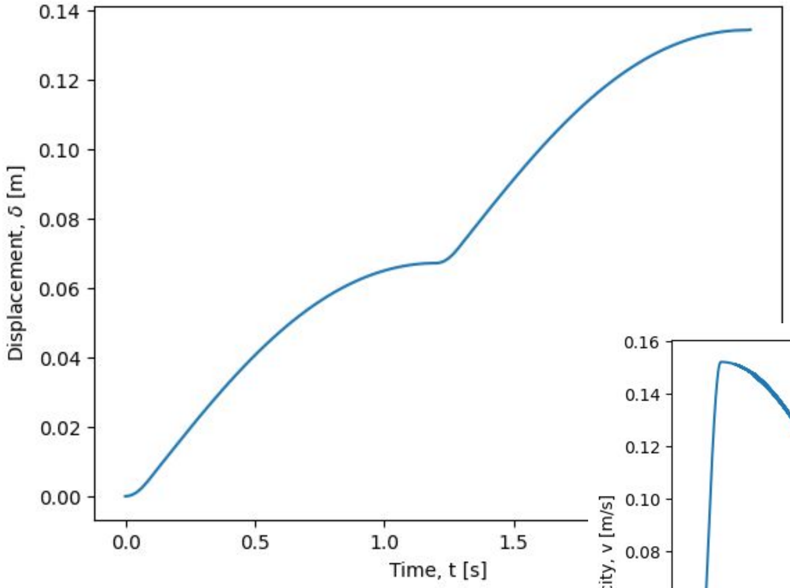
Results - 7 Nodes

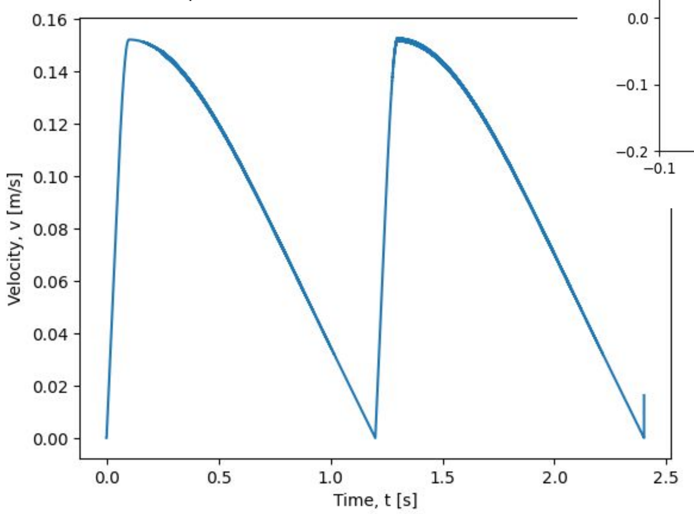


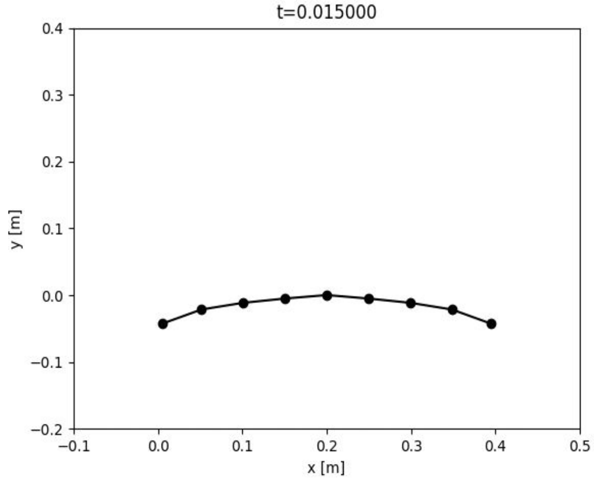




Results - 9 Nodes

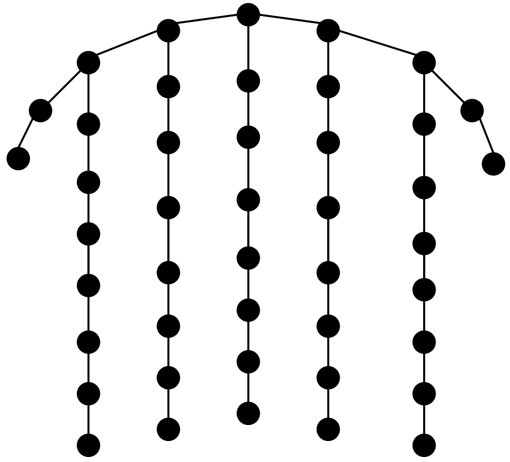






Conclusion and Future Improvements

- Successful in producing **motion that mimics real-life jellyfish**
- Results can be used to **understand jellyfish movement** and applied to the design of **biomimetic swimming robots**



Future Ideas

- **Refine velocity plots**, smaller sampling rate or interpolation method
- Explore jellyfish of **various bell widths**
- **Add tentacles** (additional passive nodes)
- **Make 3D simulation** (use plates and rods)

Works Cited

[1] Dabiri, S., & Gharib, M. "*Jellyfish as a Model for Underwater Propulsion.*" Annual Review of Marine Science, 2020.

[2] Alexander, R. M. "*Elastic Structures and Movement in Aquatic Organisms.*" Journal of Experimental Biology, 2010.

[3] Kanso, E. "*Bio-inspired Flexible Structures in Fluid Dynamics.*" Journal of Biomechanics, 2016.

[4] Olsen, Z. J., & Kim, K. J. "*Design and Modeling of a New Biomimetic Soft Robotic Jellyfish Using IPMC-Based Electroactive Polymers.*" Frontiers in Robotics and AI, 2019 Nov 1;6:112. doi: 10.3389/frobt.2019.00112. PMID: 33501127; PMCID: PMC7805721.

[5] Gemmell, B. J., Costello, J. H., Colin, S. P., Stewart, C. J., Dabiri, J. O., Tafti, D., & Priya, S. "*Passive energy recapture in jellyfish contributes to propulsive advantage over other metazoans.*" Proceedings of the National Academy of Sciences, 110(44), 17904–17909, 2013. https://doi.org/10.1073/pnas.1306983110

[6] Patricia J. Yang, Matthew Lemons, David L. Hu, "*Rowing jellyfish contract to maintain neutral buoyancy.*" Theoretical and Applied Mechanics Letters, Volume 8, Issue 3, 2018, Pages 147-152. ISSN 2095-0349. [https://www.sciencedirect.com/science/article/pii/S2095034918301028]

[7] Mould, D. "*Aesthetic Flows in Fluid Simulation.*" Graphics Interface, 2005. [https://gigl.scs.carleton.ca/sites/default/files/david_mould/jelly.pdf]