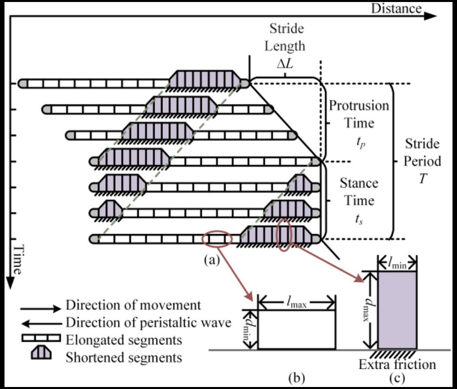
263F Final Presentation

NATHAN GE

Worm Simulation

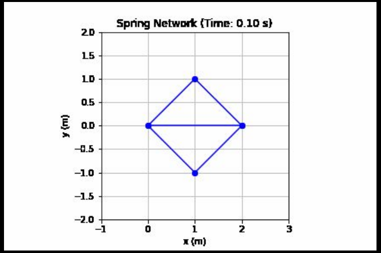
Objective: soft-body simulation of a worm-like robot

- Replicate biologically inspired locomotion patterns (peristaltic crawling, lateral undulation, and inching), to study forward locomotion.
- Simulate interactions with the environment: frictional contact with surroundings



https://iopscience.iop.org/article/10.1088/1748-3190/ab8af6

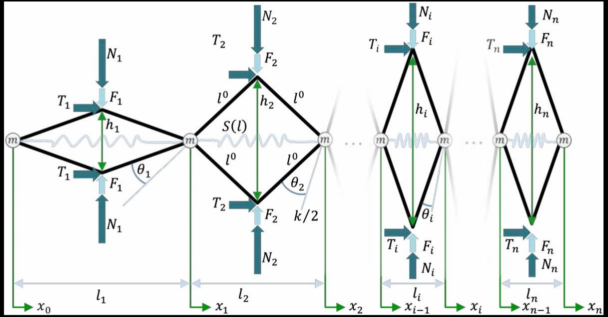
Worm Modeling



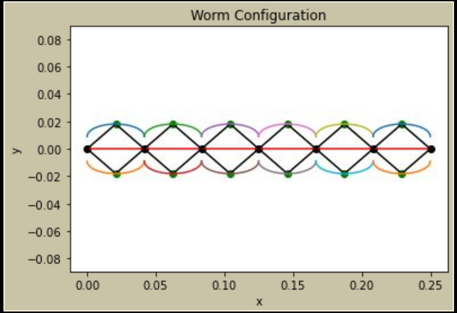
2D planar soft-worm model viewed from the side (discrete segment approach)

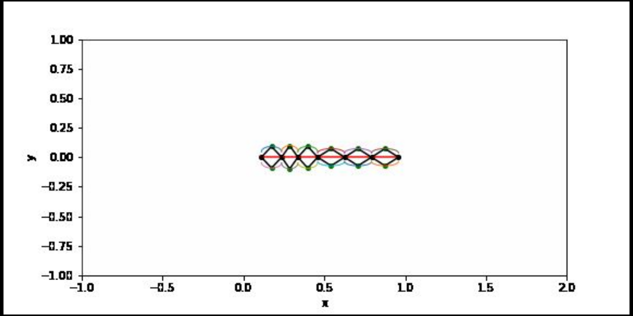
Spring Network

- Rhombic segment. coupled length and height change
- Horizontal spring to return worm to rest length
- Torsion spring at the nodes resists differences in height between adjacent segments



https://iopscience.iop.org/article/10.1088/1748-3182/8/3/035003





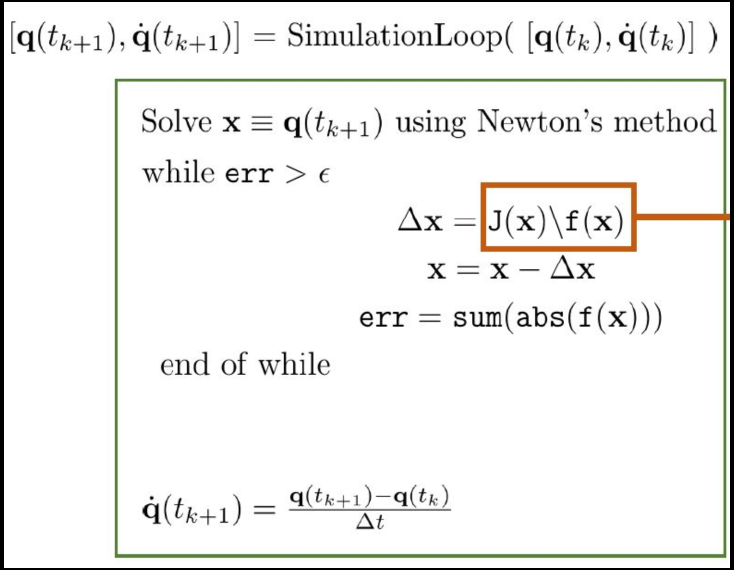
Numerical Solver

Implicit Euler's with Newton Raphson iteration

At each time step (compute force residuals and jacobians):

$$\underbrace{\mathbf{M}\,\ddot{\mathbf{q}}}_{F_{\text{inertia}}} - \underbrace{\frac{\partial E^{\text{elastic}}}{\partial \mathbf{q}}}_{F_{\text{elastic}}} - \underbrace{\mu \mathbf{N}}_{F_{\text{friction}}} - \mathbf{F}_{\text{contract}} = 0$$

$$Inertia : M\frac{1}{\Delta t}\left(\frac{q(t_{k+1}) - q(t_k)}{\Delta t} - \dot{q}(t_k)\right)$$



$[\mathbf{q}(t_{k+1}), \dot{\mathbf{q}}(t_{k+1})] = \text{SimulationLoop}(\ [\mathbf{q}(t_k), \dot{\mathbf{q}}(t_k)]\ )$

```
Solve $\mathbf{x} \equiv \mathbf{q}(t_{k+1})$ using Newton's method
while $\texttt{err} > \epsilon$
        $\Delta\mathbf{x} = \boxed{\texttt{J}(\mathbf{x})\backslash\texttt{f}(\mathbf{x})}$
        $\mathbf{x} = \mathbf{x} - \Delta\mathbf{x}$
        $\texttt{err} = \texttt{sum}(\texttt{abs}(\texttt{f}(\mathbf{x})))$
end of while

$\dot{\mathbf{q}}(t_{k+1}) = \frac{\mathbf{q}(t_{k+1}) - \mathbf{q}(t_k)}{\Delta t}$
```

Elastic energy

$$\underbrace{\mathbf{M}\,\ddot{\mathbf{q}}}_{F_{\text{inertia}}} - \underbrace{\frac{\partial E^{\text{elastic}}}{\partial \mathbf{q}}}_{F_{\text{elastic}}} - \underbrace{\mu \mathbf{N}}_{F_{\text{friction}}} - \mathbf{F}_{\mathbf{contract}} = 0$$

Stretching

$$E^s = \frac{k l_0}{2}\left(1 - \frac{l}{l_0}\right)^2$$

$$\textbf{where}: l = \sqrt{(x_{k+1} - x_k)^2 + (y_{k+1} - y_k)^2}$$

Bending

$$E^b = \frac{k_{\text{torsion}}}{2}(\Delta\text{cross})^2$$

$$\textbf{where}:$$

$$\begin{aligned}
\Delta\text{cross} &= \text{cross} - \text{cross}_0, \\
\text{cross} &= \mathbf{t}_1 \times \mathbf{t}_2 = t_{1x}t_{2y} - t_{1y}t_{2x}, \\
\mathbf{t}_1 &= \frac{\mathbf{e}_1}{\|\mathbf{e}_1\|}, \qquad \mathbf{t}_2 = \frac{\mathbf{e}_2}{\|\mathbf{e}_2\|}, \\
\mathbf{e}_1 &= \mathbf{p}_b - \mathbf{p}_a, \qquad \mathbf{e}_2 = \mathbf{p}_c - \mathbf{p}_b.
\end{aligned}$$

Friction

$$\underbrace{\mathbf{M}\,\ddot{\mathbf{q}}}_{F_{\text{inertia}}} - \underbrace{\frac{\partial E^{\text{elastic}}}{\partial \mathbf{q}}}_{F_{\text{elastic}}} - \underbrace{\mu \mathbf{N}}_{F_{\text{friction}}} - \mathbf{F}_{\text{contract}} = 0$$

Predictor Corrector
Simulate ground contact. Normal forces are computed indirectly from fixed DoF force residuals

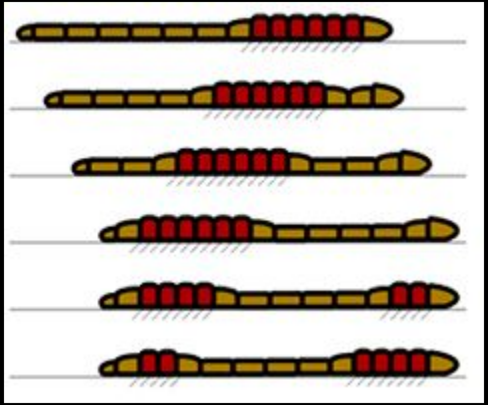
If velocity threshold is exceeded,
Low Friction when segment moves right
High friction when segment moves left

```
if N > 0:
        if v_x > 1e-6:  # Moving RIGHT (forward) - low friction
            F_friction[x_dof] = -mu_forward * N * np.sign(v_x)
        elif v_x < -1e-6:  # Moving LEFT (backward) - high friction
            F_friction[x_dof] = -mu_backward * N * np.sign(v_x)
```

Peristalsis

$$\underbrace{\mathbf{M}\,\ddot{\mathbf{q}}}_{F_{\text{inertia}}} - \underbrace{\frac{\partial E^{\text{elastic}}}{\partial \mathbf{q}}}_{F_{\text{elastic}}} - \underbrace{\mu \mathbf{N}}_{F_{\text{friction}}} - \mathbf{F}_{\textbf{contract}} = 0$$

Traveling contraction pattern paired with anisotropic friction produces forward motion



Contraction

State arrays

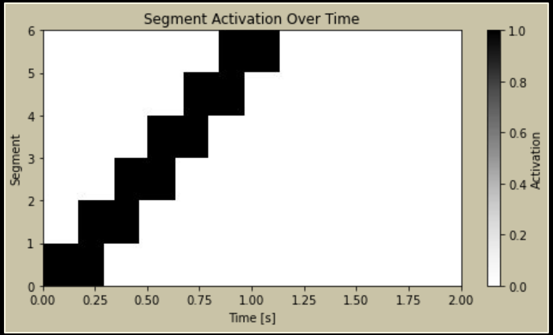
```
self.phase = np.zeros(n_segments, dtype=float)
self.active = np.zeros(n_segments, dtype=bool)
self.activated_this_wave = np.zeros(n_segments, dtype=bool)
```

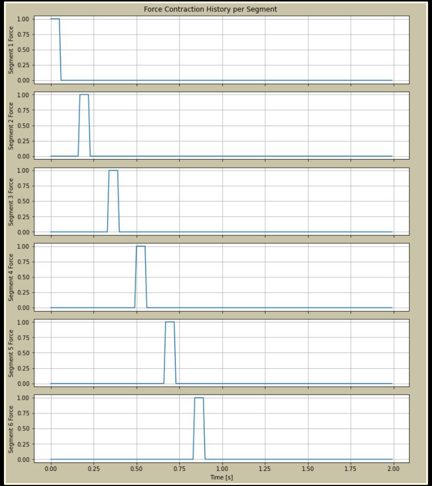
Step 1: activate segments based on wave

Step 2: increment phase and deactivate over-threshold segments

Step 3: pulse generator

Step 4: compute forces for all segments





Sine-wave controller

Real peristalsis requires *all segments simultaneously having contraction levels determined by a traveling wave function*.

1. Spatial modulation: Wave number k controls how many waves fit along the body
2. Temporal modulation: Angular frequency ω controls contraction speed
3. Wave direction: Sign in phase determines propagation direction
4. Force rectification: max(0, A_i) ensures unidirectional contraction
5. Symmetric application: Equal and opposite forces on top/bottom connectors create squeezing motion

$$\text{Phase: } \phi_i(t) = k \cdot x_{\text{norm},i} + \omega t$$

$$\text{Amplitude: } A_i(t) = \sin(\phi_i(t)) = \sin(k \cdot x_{\text{norm},i} + \omega t)$$

$$\text{At fixed time } t_0\text{,the spatial pattern is: } A(x, t_0) = \sin(kx - \omega t_0)$$

$$\text{At fixed position } x_0\text{,the temporal pattern is: } A(x_0, t) = \sin(kx_0 - \omega t) = \sin(\phi_0 - \omega t)$$

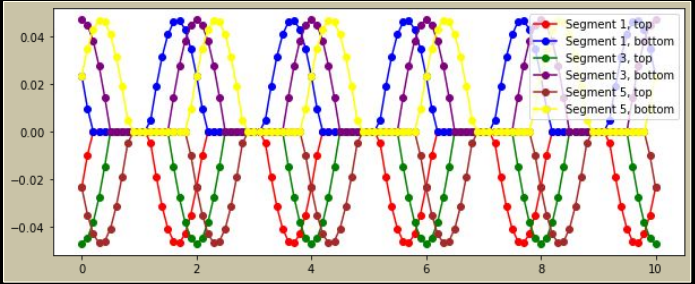
Sine-wave controller

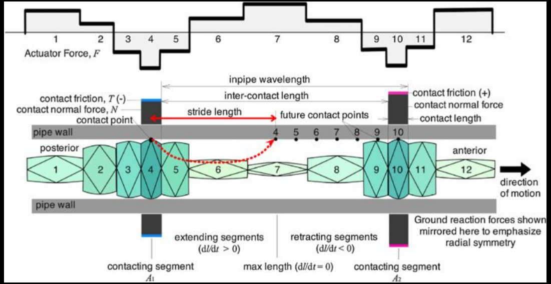
Total contractile force: $\mathbf{F}\mathrm{contract}(t) = \sum i = 0^{n_{\mathrm{seg}}-1} \mathbf{F}\mathrm{contract}, i(t)$

where $(\mathbf{F}\mathrm{contract}, i(t))$ is the force contribution from segment i :

$$\mathbf{F}\mathrm{contract}, i(t) = \begin{cases} -F_{\mathrm{seg},i}(t)\mathbf{e}y & \text{at top connector DOF} \\ +F_{\mathrm{seg},i}(t)\mathbf{e}y & \text{at bottom connector DOF} \\ \mathbf{0} & \text{elsewhere} \end{cases}$$

$$F_{\mathrm{seg},i}(t) = F_{\max} \cdot \max(0, A_i(t))$$





Performance measuring framework

Cost of Transport

- Energy Expenditure
  - work_step = F_contract.dot(q_new_flat - q_old_flat)
  - cot_step = work_step / (delta_com)
- Energy (frictional slippage losses)
  - work_step = F_friction.dot(q_new_flat - q_old_flat)
  - cot_step = work_step / (delta_com)

Workflow

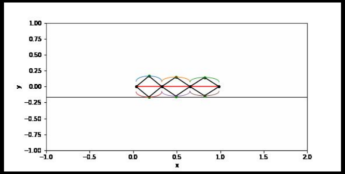
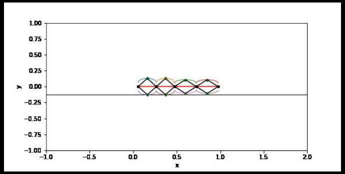
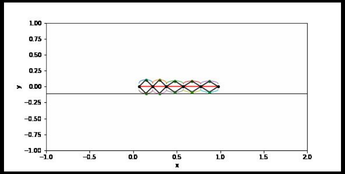
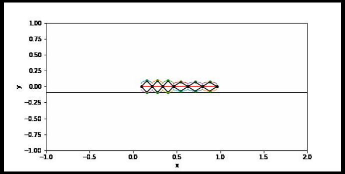
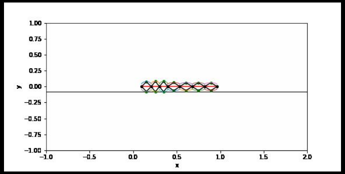
Parametrization: code structure accommodates various testing conditions with a simple user interface

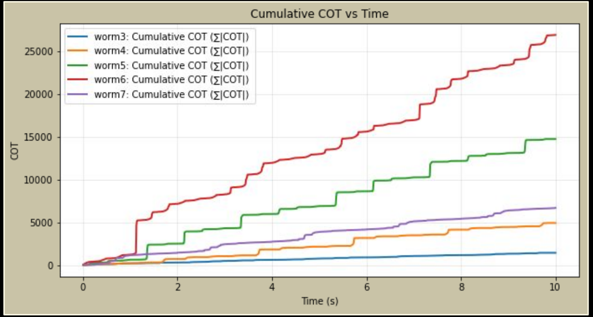
GENERATE MODEL	RUN SOLVER	ANIMATE	PLOT DATA
01	02	03	04
Define worm length, number of segments, and fixed DoFs. The y direction of each main node is clamped	Define the worm being simulated, run time, time step (dt), max iterations per solver iteration, and contraction parameters	Run animation generator	Plot metrics (Cost of Transport, center of mass travel)

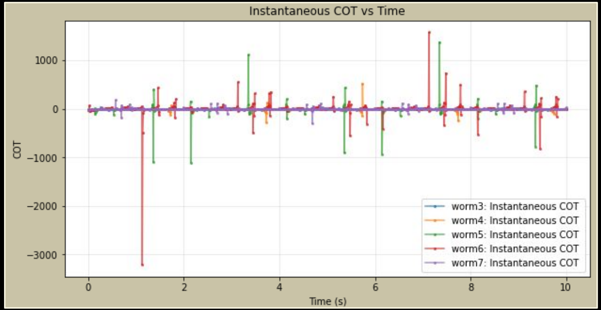
```
1. worm6 = WormModel(name ="worm6", length=param.length, n_segments=6, fixedDOFs=[1,7,13,19,25,31,37])

2. contract1 = { 'contraction_type': 'multiple_segments', 'wave_type': 'traveling', 'T_wave': 1.0, 'wavelength': 1.0}
frames1, times1, data1 = solve.solver_with_predictor_corrector(worm6, dt=param.dt, maxTime=10.0, tol=1e-6, maximum_iter=param.maximum_iter,
contraction_params=contract1)

3. ani1 = worm6.animate_worm(frames1, times1)

4. axes = solve.plot_sim_results(data1, frames1, times1,
                                 ax_first_node=ax_first, ax_last_node=ax_last, ax_com=ax_com,
                                 ax_cot_inst=ax_cot_inst, ax_cot_cum=ax_cot_cum, worm_name="worm6",
                                 label_suffix=" run A", main_node_indices=worm6.main_node_indices)
```

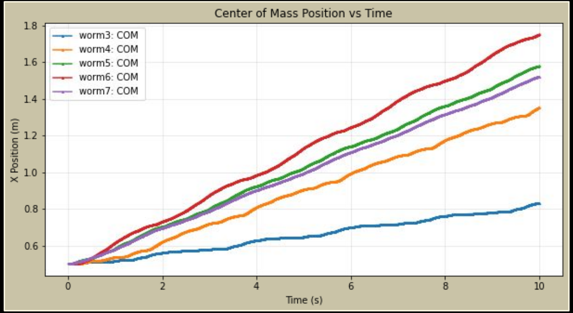
Results

Effect of segment count on performance

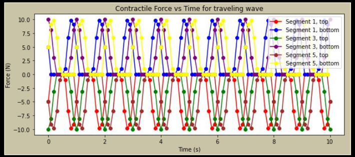

Cumulative COT vs Time
worm3: Cumulative COT (∑|COT|)
worm4: Cumulative COT (∑|COT|)
worm5: Cumulative COT (∑|COT|)
worm6: Cumulative COT (∑|COT|)
worm7: Cumulative COT (∑|COT|)
COT
Time (s)

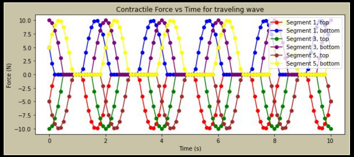


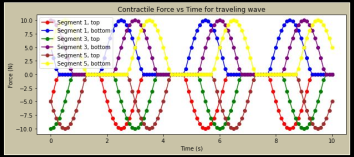


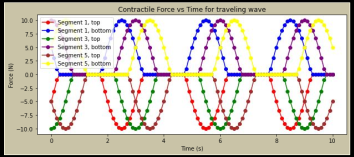
Results

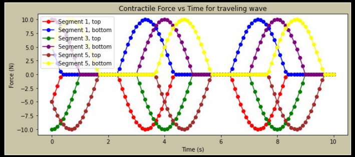
Effect of contraction pattern on performance
6 segment worm with different period length of contraction pattern

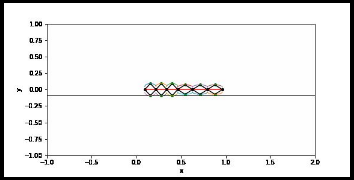


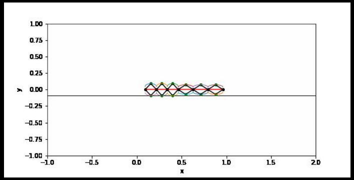


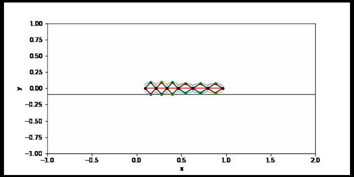


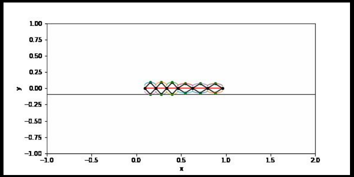


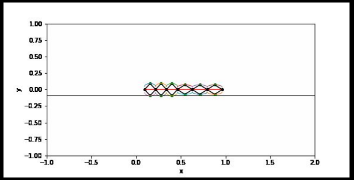


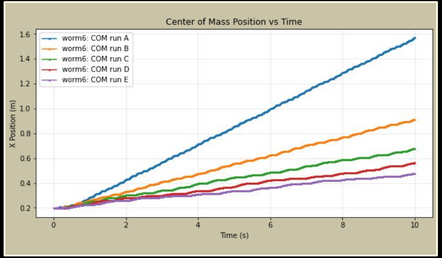


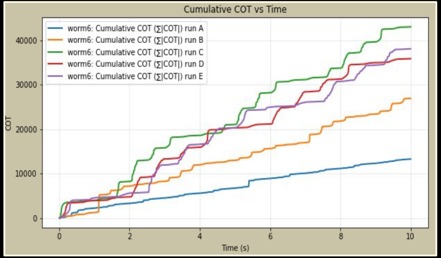












Problems

Numerical instabilities

- Mass on nodes (high interior forces)
- Torsional spring
- Weight

Clamping y-axis of central nodes

Next Steps

Solver

- Explore other numerical methods
  - Hybrid pipeline
    - fast real-time controller in PD
    - offline FEM validation
    - learned residual corrector (real-time simulation with FEM accuracy, small neural network or regression model to predict the residual for any new PD state)
- More realistic frictional forces (more complex predictor corrector implementation, include frictional anchor forces instead of velocity based approach)

Contraction

- Non-rectified contraction, other variations of the current contraction approach

Controls

- Real time velocity control
- Stable Heteroclinic Channels (often used in rhythmic pattern generation (CPGs – central pattern generators) for locomotion)
- Neural intent (CPG trained on real larva behavior producing contraction timing signals)

Next Steps

Environmental Factors

- Uneven/sloped terrain?
- Pipe scenario

Model Improvements

- Resolve numerical instabilities to simulate more realistic scenarios
- Variable torsion and linear spring stiffnesses
- Pathing with directed contraction patterns