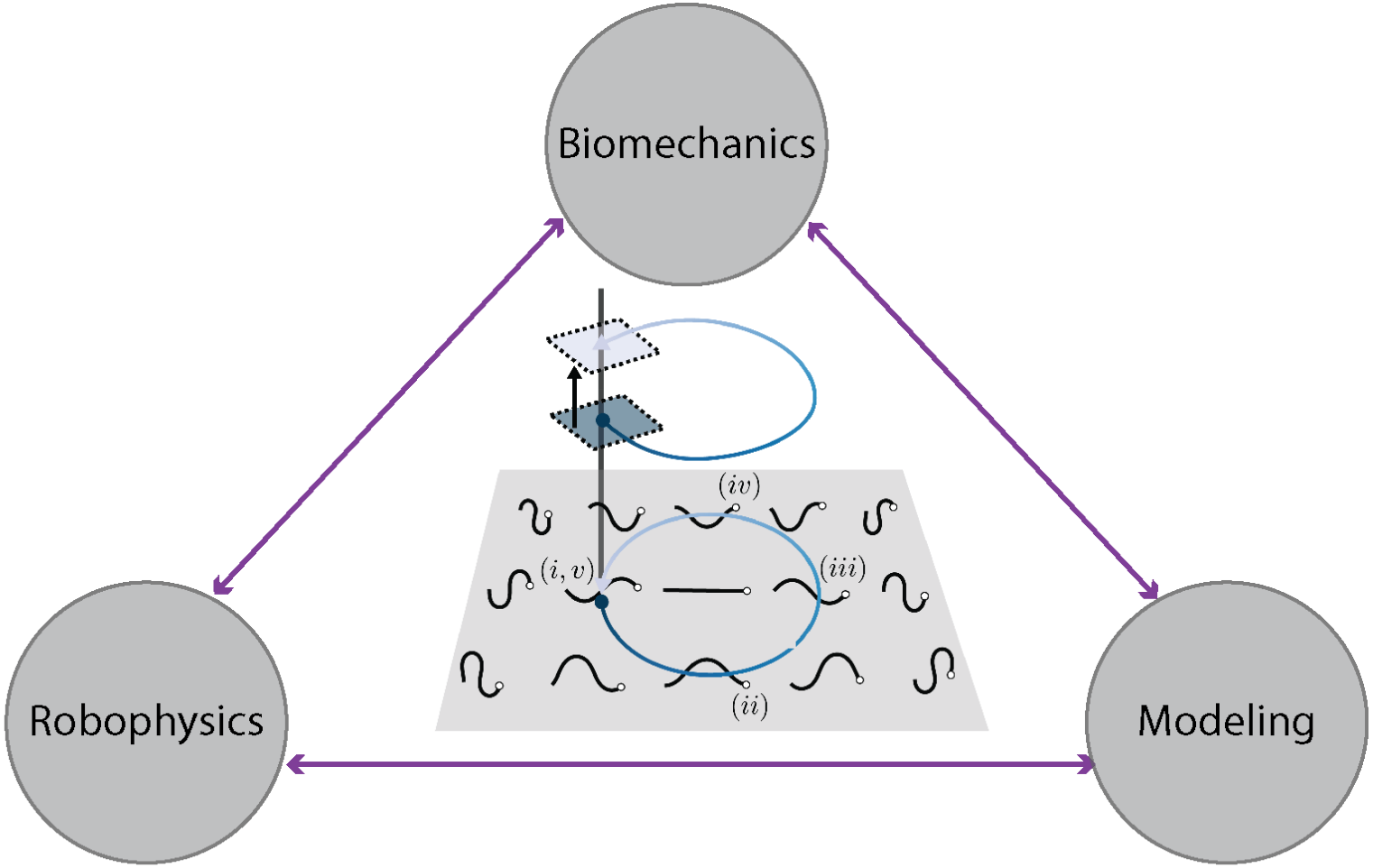
Geometry of locomotion: biomechanics, modeling, and robophysics

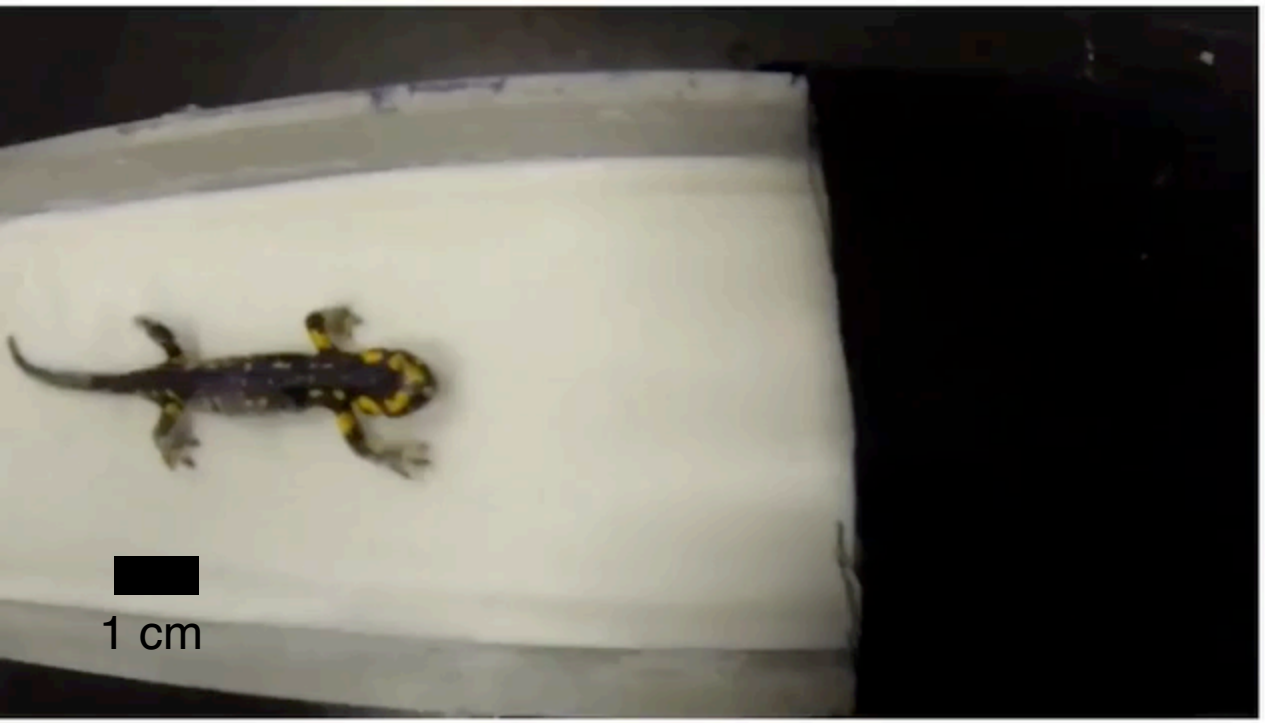


Georgia Tech

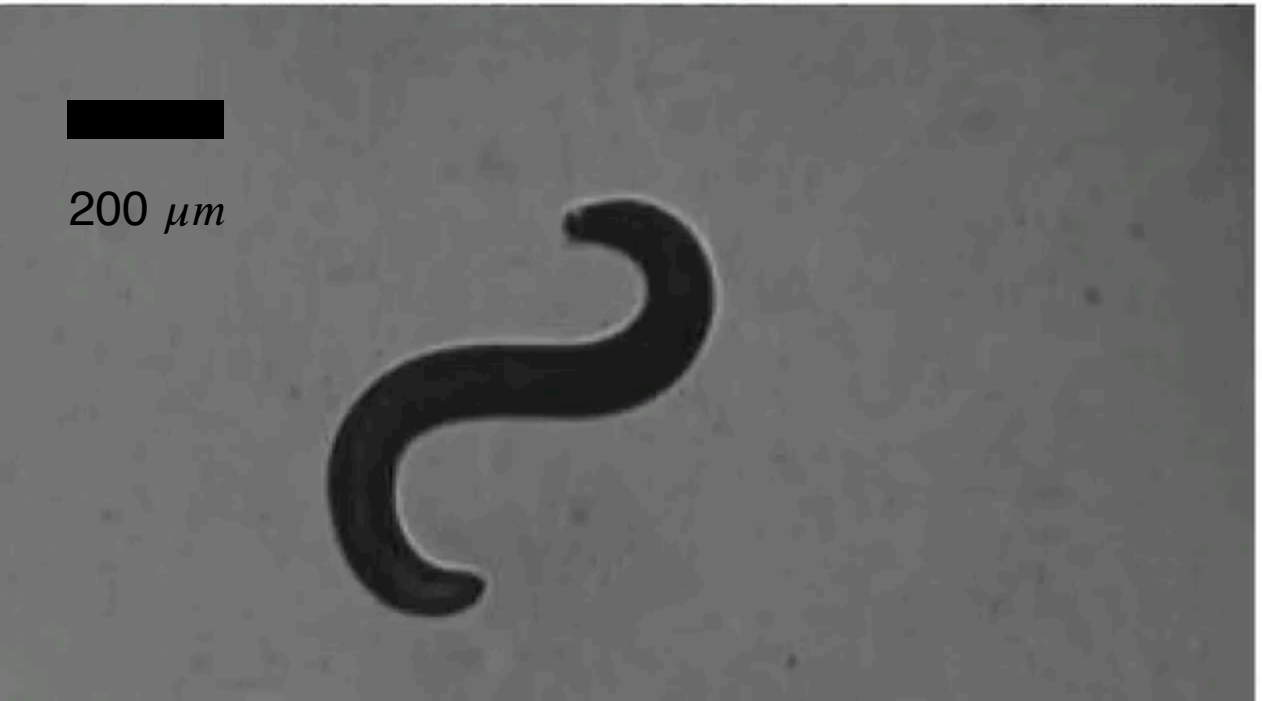
Dr. Baxi Chong

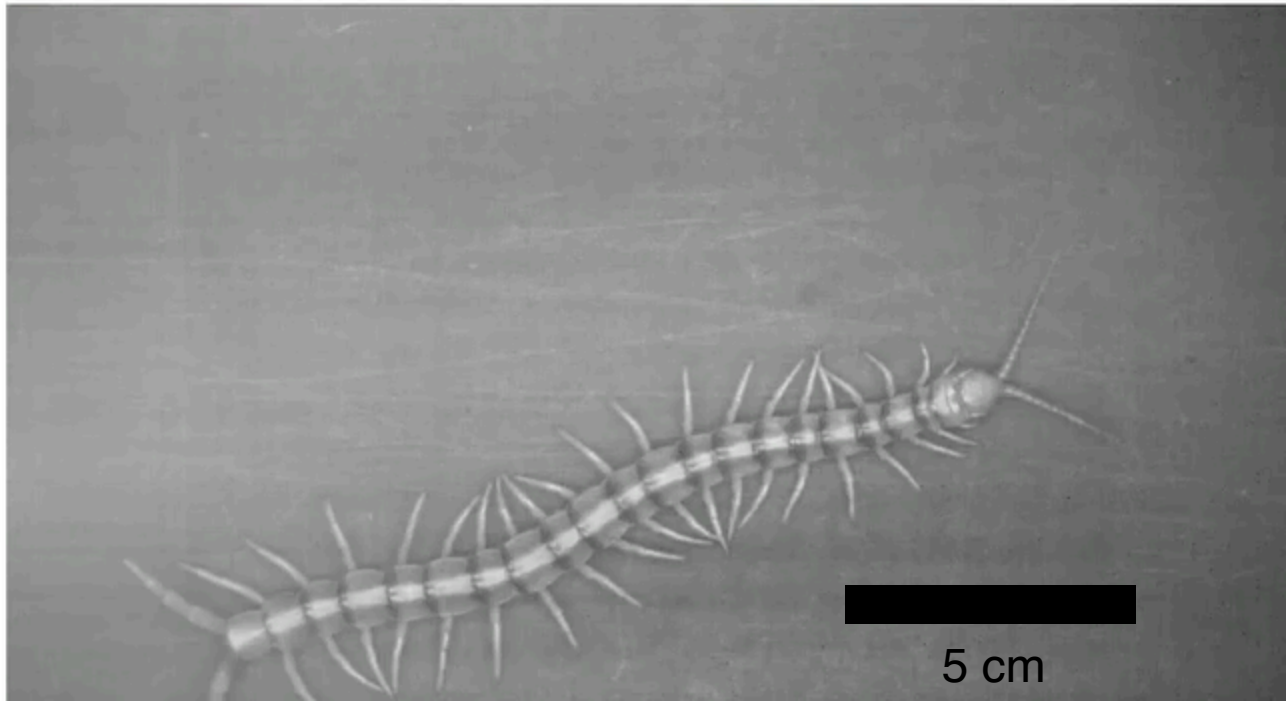
Georgia Tech Robophysics Boot Camp

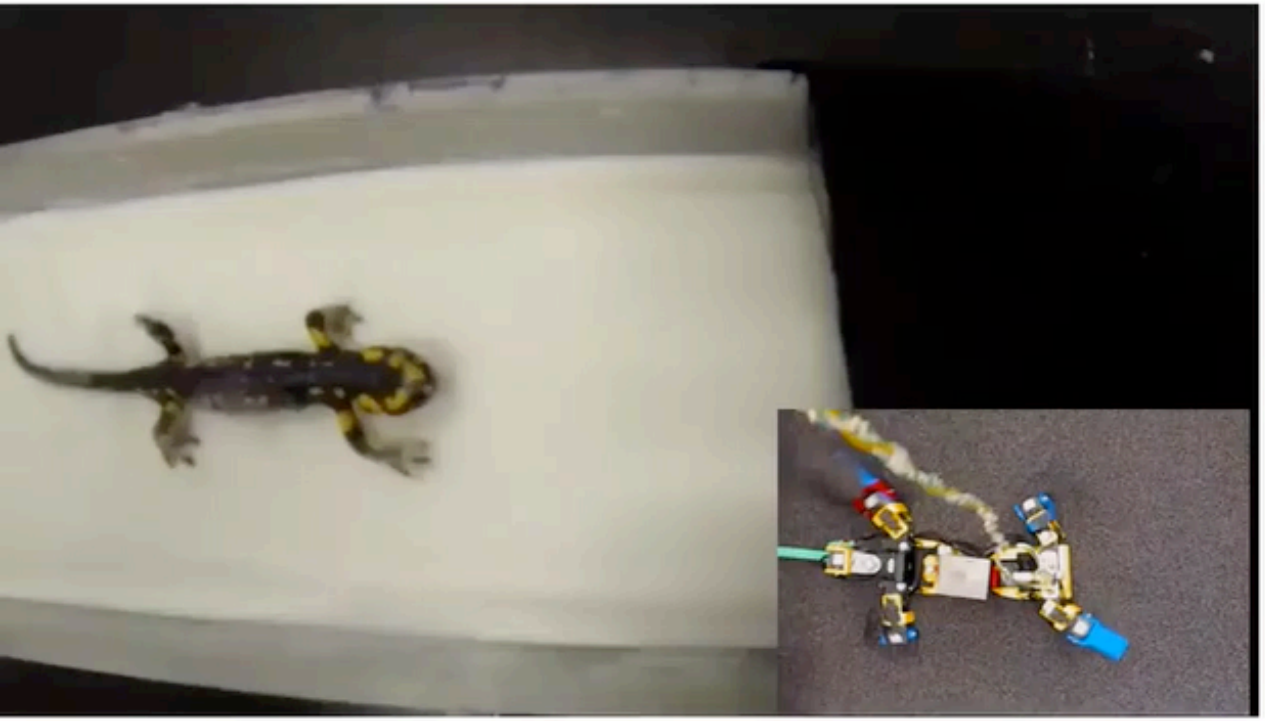
Howey Physics Building, Rooms N201 and N202
837 State Street, Atlanta 30332
July 14—16, 2025

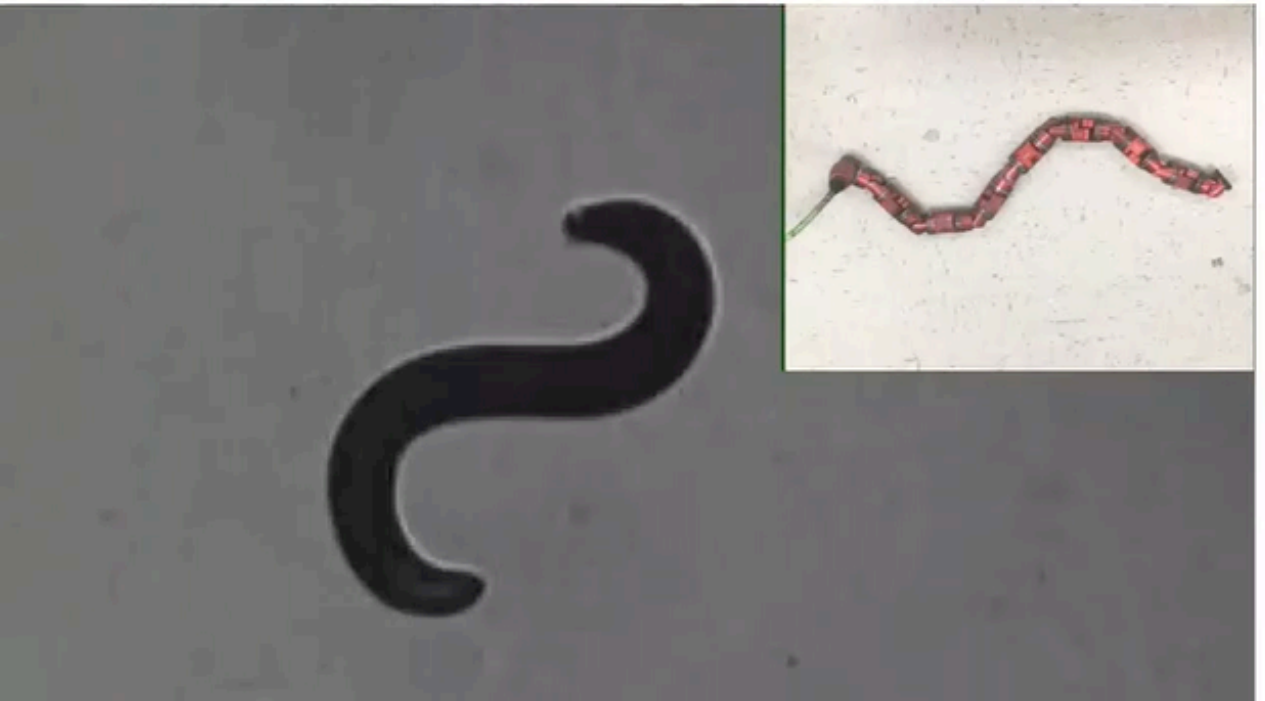

1 cm

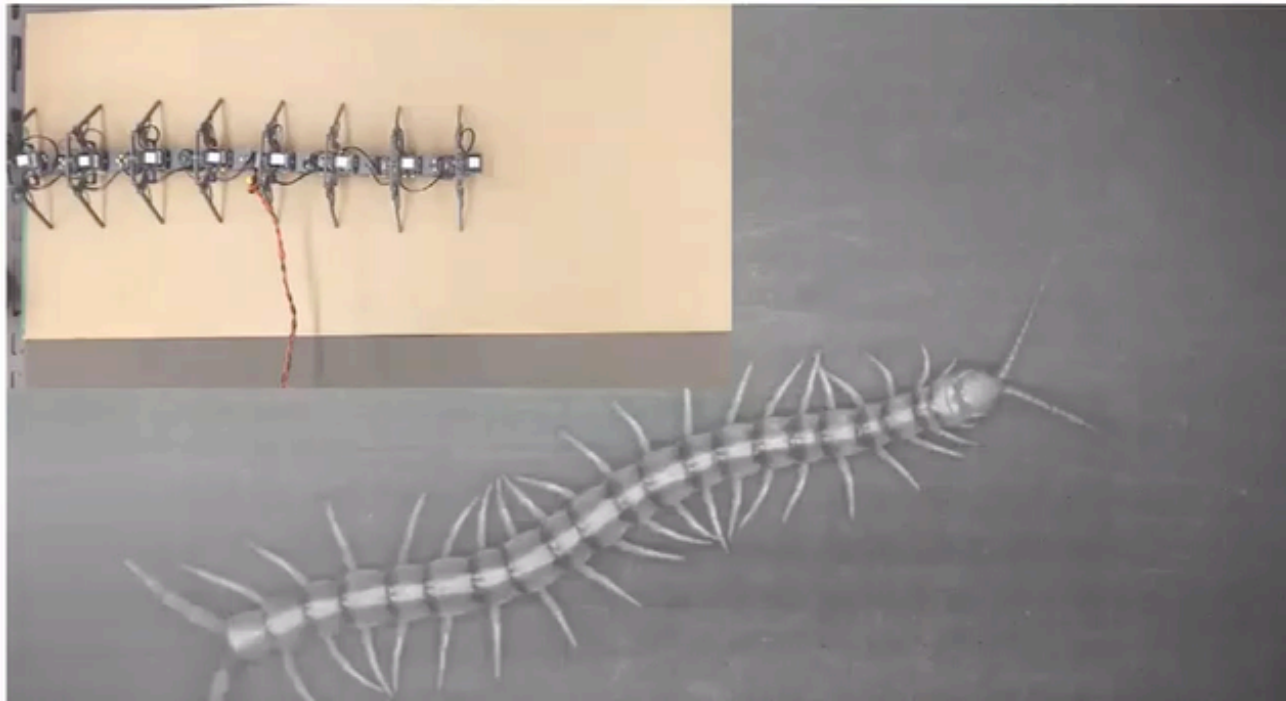

1 cm


200 μm


5 cm



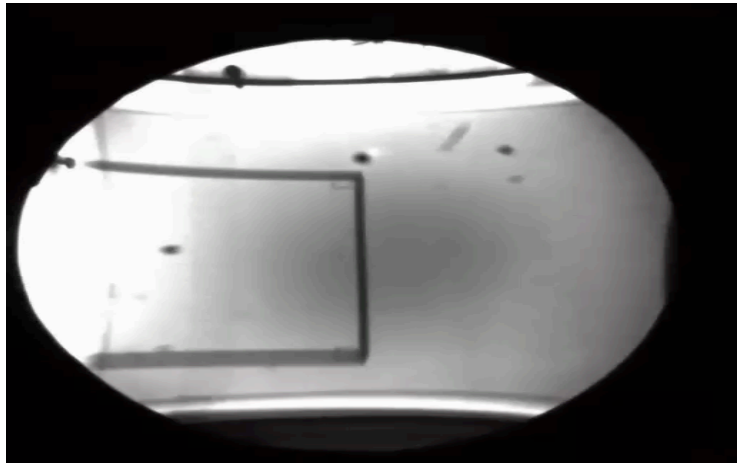




Undulating living systems across scales

Scincus scincus

Inertia << frictional resistance forces

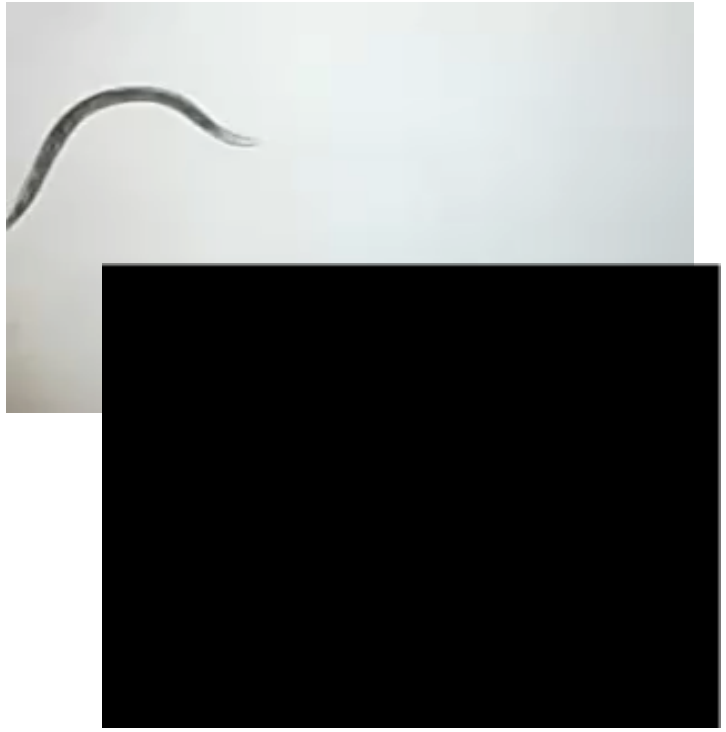


Data from: Sharpe et al, *JEB* 2015

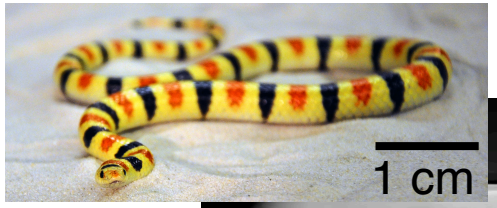
Inertia << viscous resistance forces

Nematode *C. elegans*

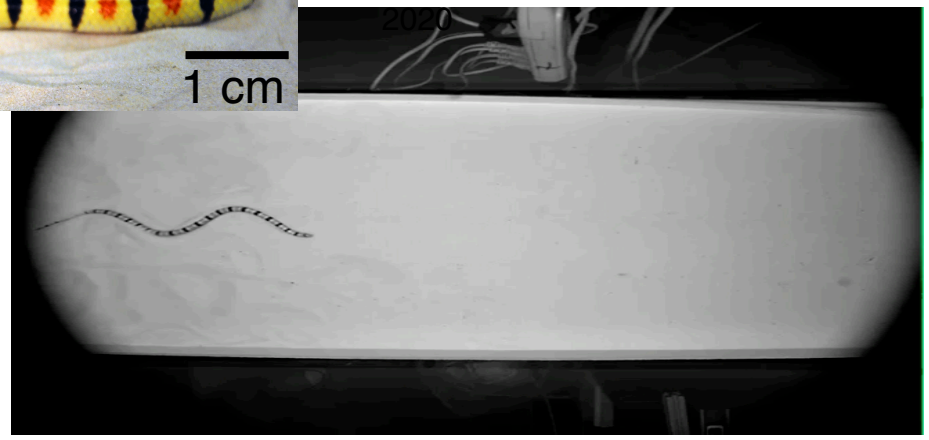
Inertia > fluid resistance forces



Chionactis occipitalis



Data from: Schiebel, *et al*, *eLife,* 2020



Viscous Swimming: coasting time

Viscous drag: $m\dot{v} = -\gamma v$

A non-dimensional number, the "Coasting Number", how long locomotor coasts relative to time between cycles

$$\mathscr{C} = \frac{\tau_{coast}}{\tau_{cycle}}$$

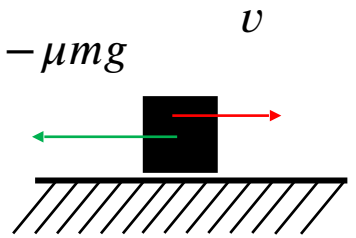
Related to Re -> $\mathscr{C} \sim Re\frac{\rho_b}{\rho}$

	Coasting time	Temporal period	Coasting number
C. elegans			0.01

Frictional Swimming: coasting number

Data from: Schiebel, *et al*, 2020

A non-dimensional number, the "Coasting Number", how long locomotor coasts relative to time between cycles



$$\mathscr{C} = \frac{\tau_{coast}}{\tau_{cycle}}$$

	Coasting time	Temporal period	Coasting numbe
C. elegans			0.01


Hamidreza Marvi[1], Chaohui Gong[2], Nick Gravish[1],
Henry Astley[1], Matt Travers[2], Ross Hatton[3], Joe Mendelson[1,4],
Howie Choset[2], David Hu[1], and Daniel Goldman[1]

Georgia Institute of Technology[1],
Carnegie Mellon University[2],
Oregon State University[3],
Zoo Atlanta[4]

C. cerastes sidewinding,
Marvi et al., *Science* 2014

The geometry of motion

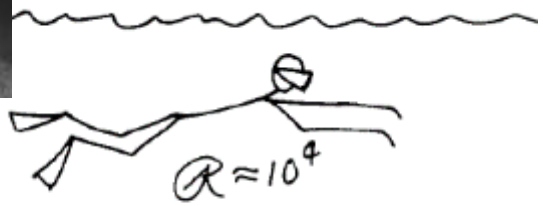
depends on sequence of shapes...not time...

Life at low Reynolds number

E. M. Purcell
Lyman Laboratory, Harvard University, Cambridge, Massachusetts 02138
(Received 12 June 1976)

Editor's note: This is a reprint (slightly edited) of a paper of the same title that appeared in the book *Physics and Our World: A Symposium in Honor of Victor F. Weisskopf*, published by the American Institute of Physics (1976). The personal tone of the original talk has been preserved in the paper, which was itself a slightly edited transcript of a tape. The figures reproduce transparencies used in the talk. The demonstration involved a tall rectangular transparent vessel of corn syrup, projected by an overhead projector turned on its side. Some essential hand waving could not be reproduced.

EM Purcell (1912-1997)



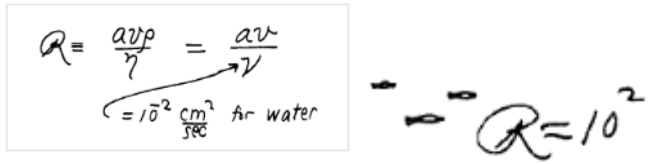
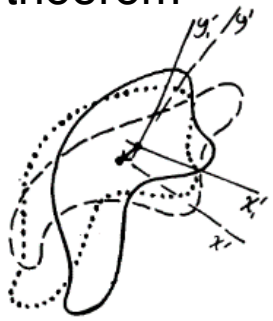
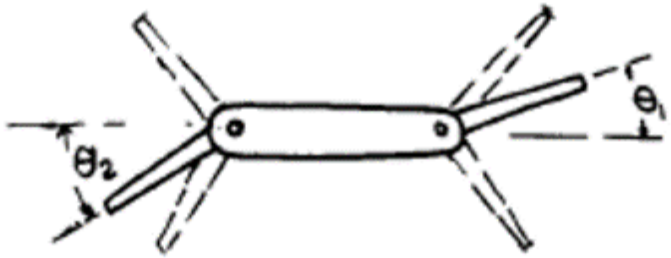


Figure 3.

"Scallop theorem"



"Purcell 3-link swimmer"



Gait in shape configuration space

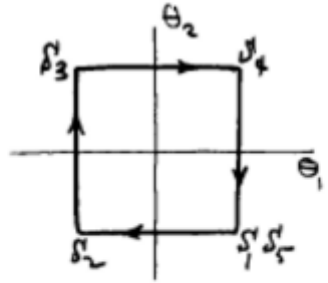
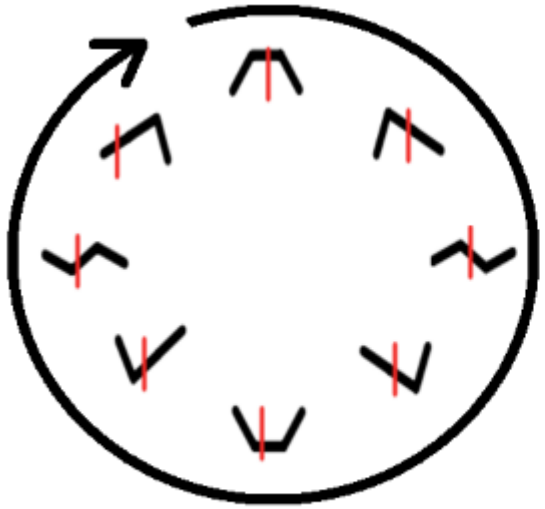
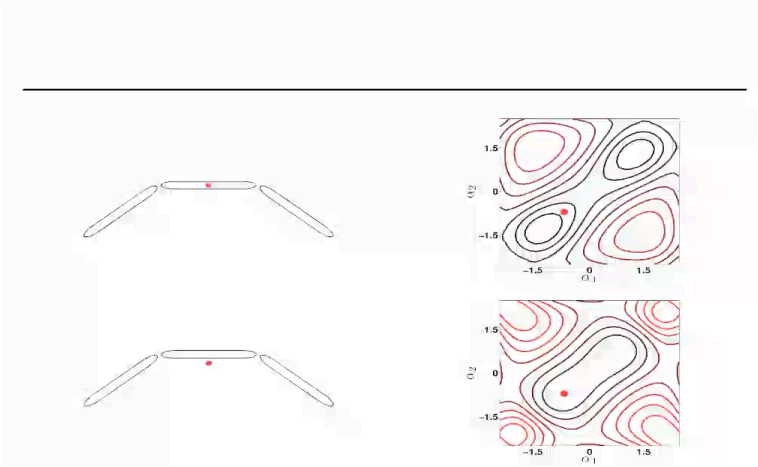


Figure 7.

Geometric Mechanics



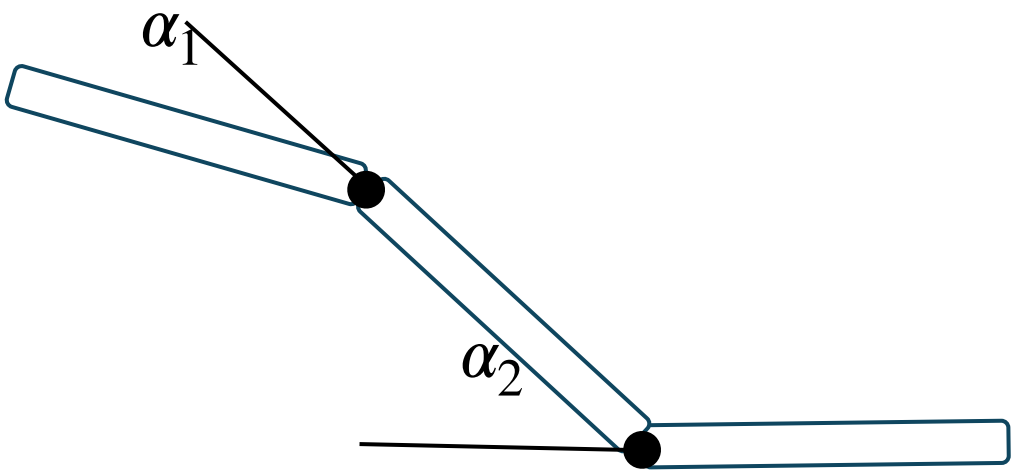
Interaction with environments



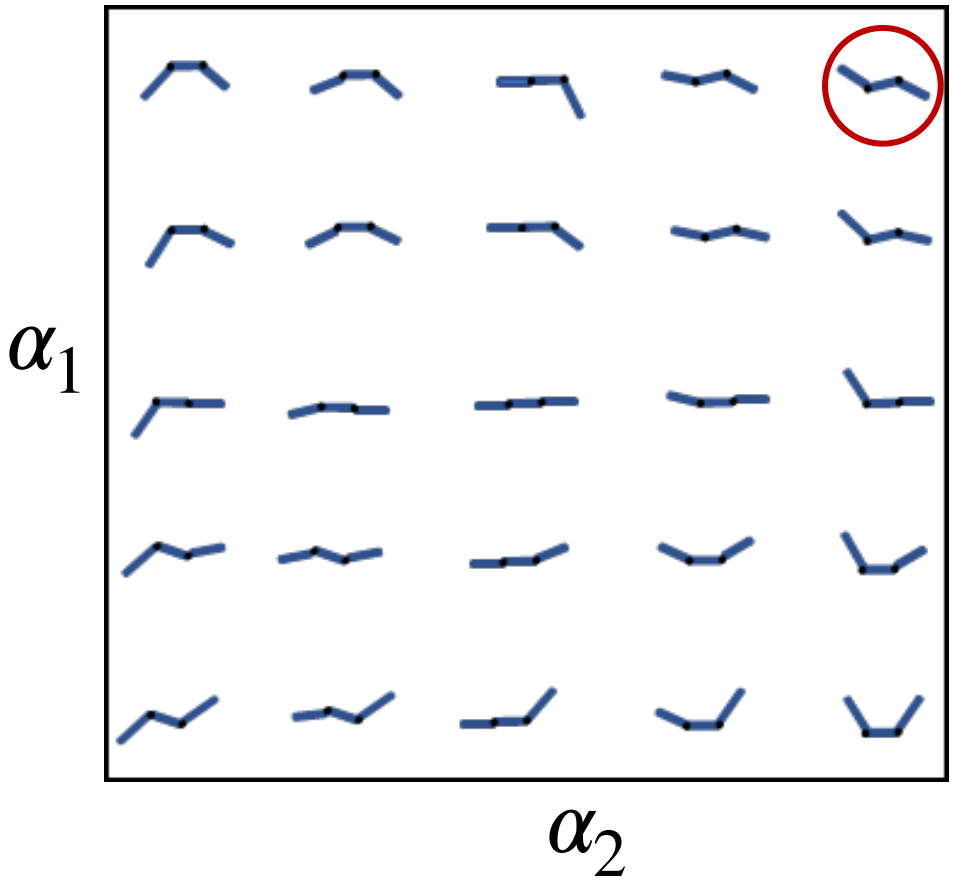
Purcell, *American journal of physics* 1977
Wilczek, et al. *World Scientific*, 1989.
Hatton, et al. *IJRR* 2011

Geometric Mechanics

- Shape variable

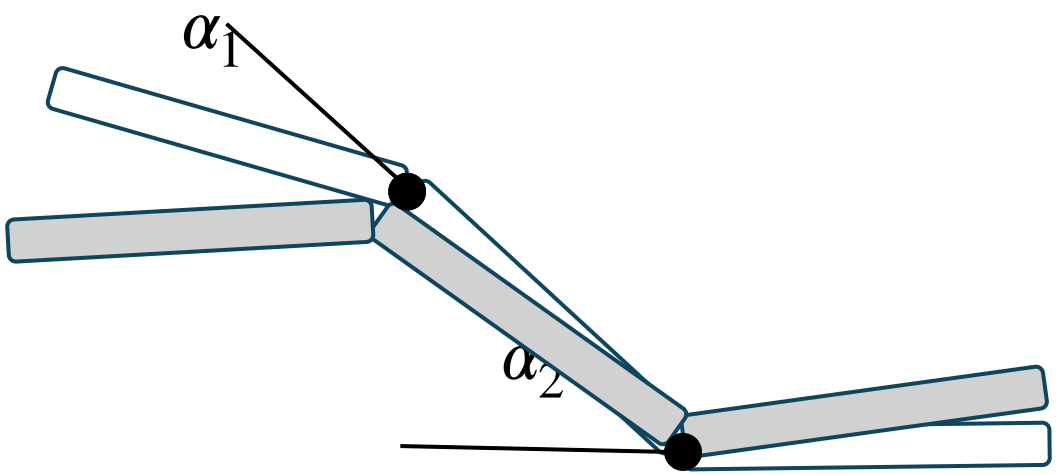


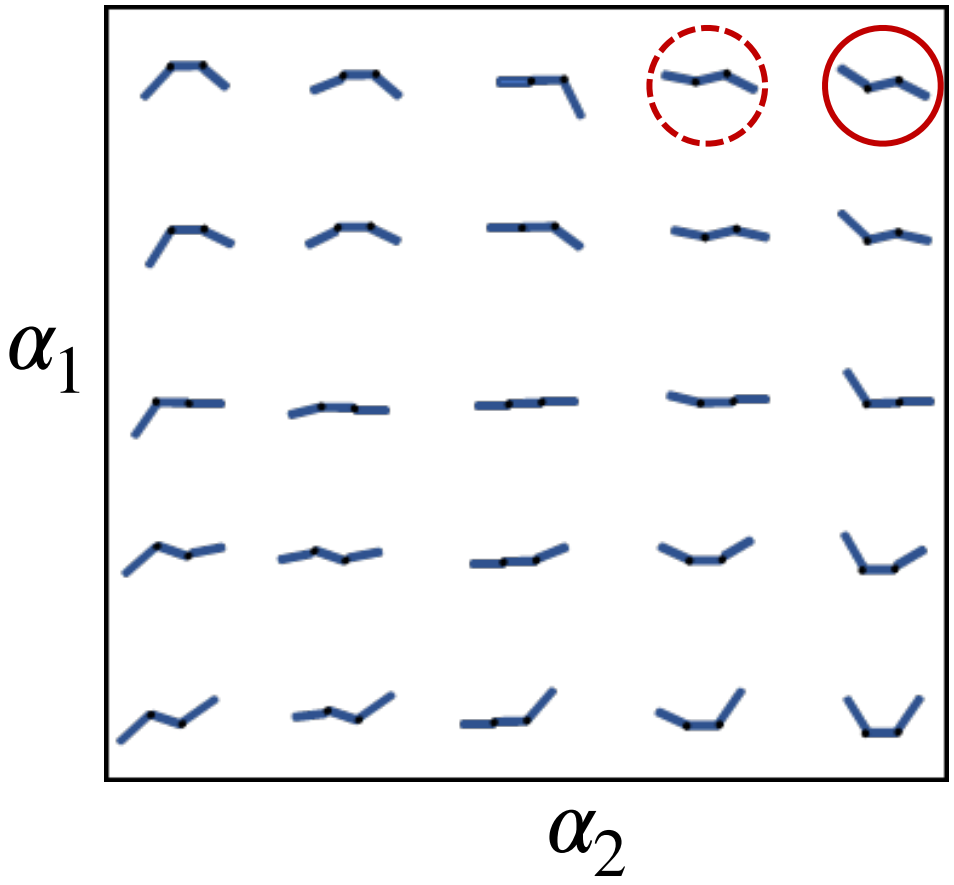
Shape space



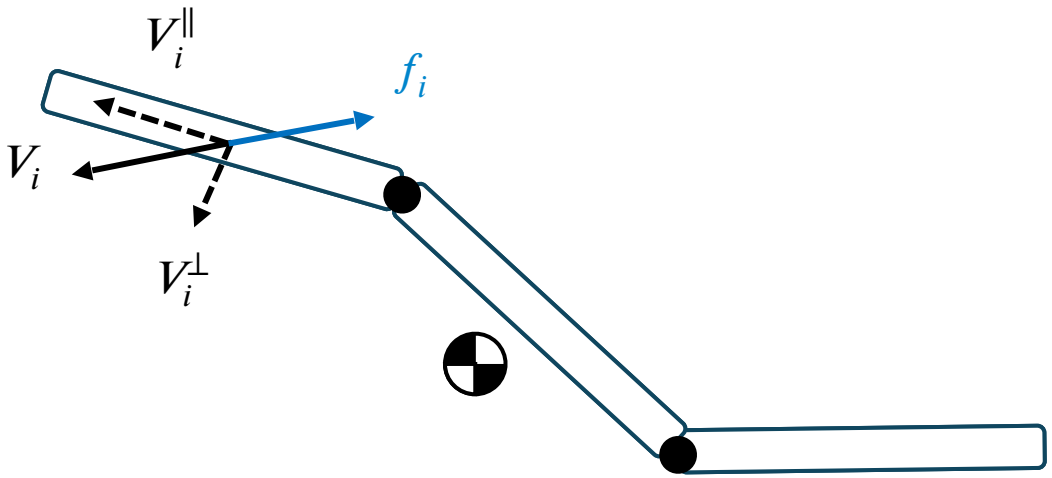
Geometric Mechanics

- Shape variable





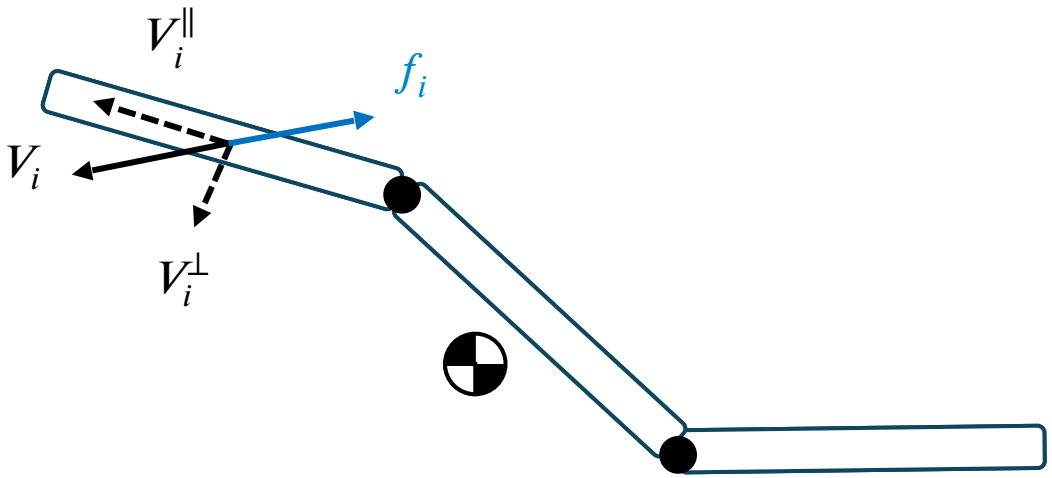
Viscous fluid



V_i: body velocity of link i

f_i: Reaction force on link i

Viscous fluid



V_i: body velocity of link i

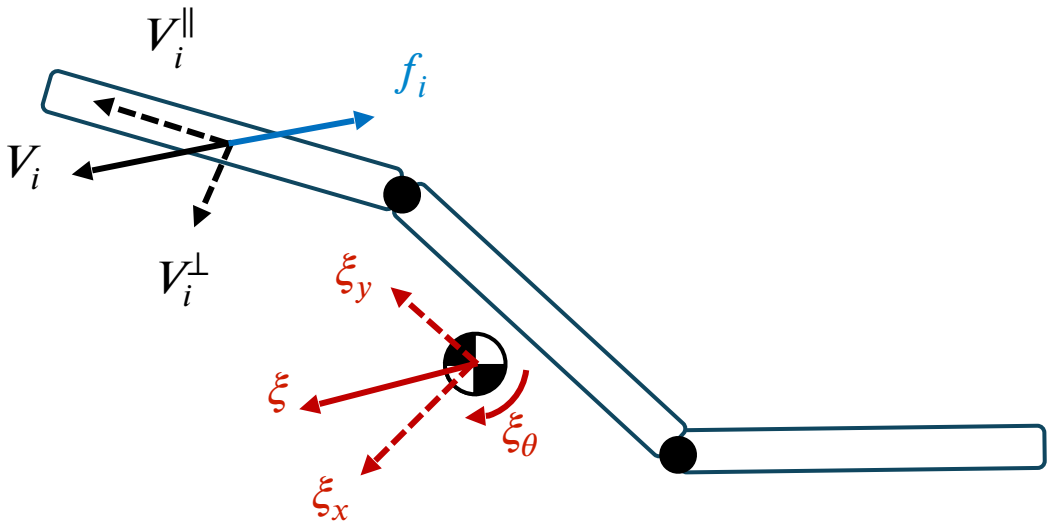
From geometry, V_i is a function of shape velocities and CoM velocities

f_i: Reaction force on link i

Viscous fluid

CoM velocities

$$\begin{bmatrix} \xi_x \\ \xi_y \\ \xi_\theta \end{bmatrix} \begin{matrix} \rightarrow \text{Forward} \\ \rightarrow \text{Lateral} \\ \rightarrow \text{Rotational} \end{matrix}$$

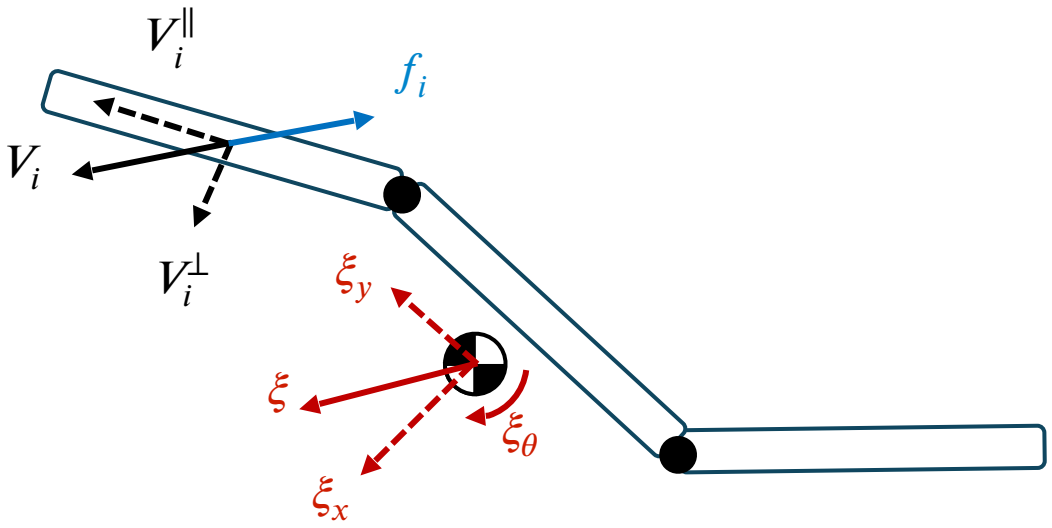


V_i: body velocity of link i

From geometry, V_i is a function of shape velocities and CoM velocities

f_i: Reaction force on link i

Viscous fluid



CoM velocities

$$\begin{bmatrix} \xi_x \\ \xi_y \\ \xi_\theta \end{bmatrix} \begin{matrix} \rightarrow \text{Forward} \\ \rightarrow \text{Lateral} \\ \rightarrow \text{Rotational} \end{matrix}$$

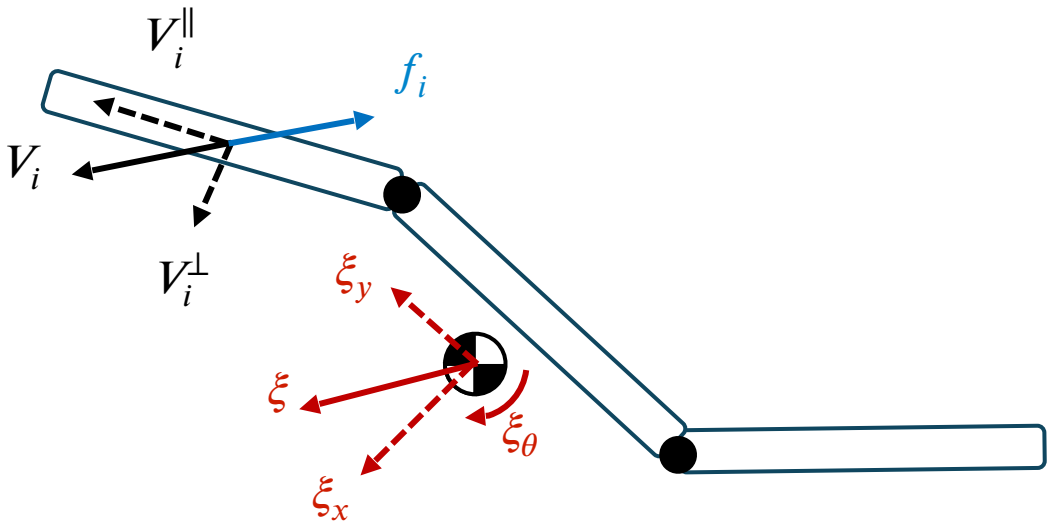
V_i: body velocity of link i

From geometry, V_i is a function of shape velocities and CoM velocities

f_i: Reaction force on link i

$$f_i = -CV_i$$

Viscous fluid



CoM velocities

$$\begin{bmatrix} \xi_x \\ \xi_y \\ \xi_\theta \end{bmatrix} \begin{matrix} \rightarrow \text{Forward} \\ \rightarrow \text{Lateral} \\ \rightarrow \text{Rotational} \end{matrix}$$

V_i: body velocity of link i

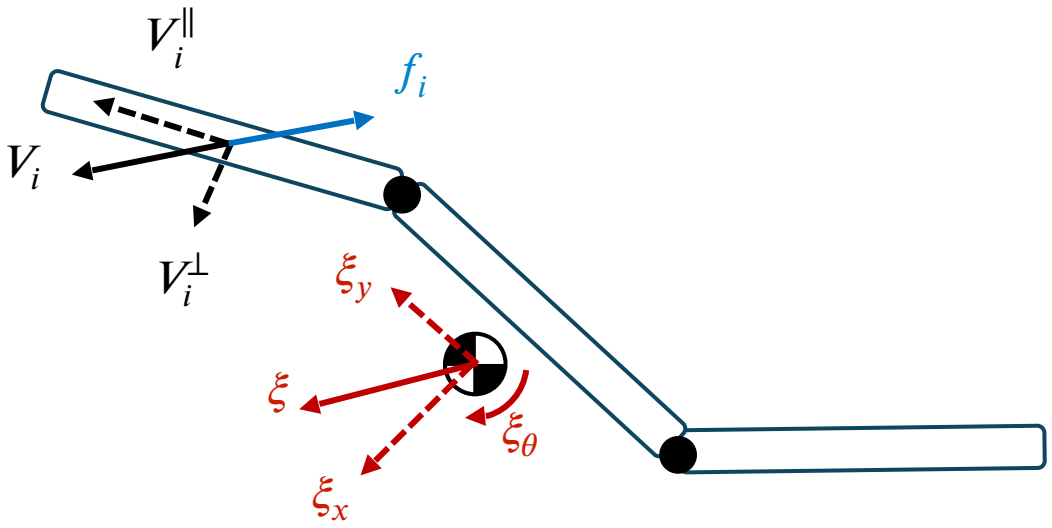
From geometry, V_i is a function of shape velocities and CoM velocities

f_i: Reaction force on link i

$$f_i = -\boldsymbol{C} V_i \quad \sum_i f_i = 0$$

Quasi-static assumption

Viscous fluid



$$\begin{bmatrix} \xi_x \\ \xi_y \\ \xi_\theta \end{bmatrix} = \boldsymbol{F}\left(\boldsymbol{\alpha_1}, \boldsymbol{\alpha_2}, \dot{\boldsymbol{\alpha}}_1, \dot{\boldsymbol{\alpha}}_2\right)$$

CoM velocities

$\begin{bmatrix} \xi_x \\ \xi_y \\ \xi_\theta \end{bmatrix}$ → Forward, → Lateral, → Rotational

V_i: body velocity of link i

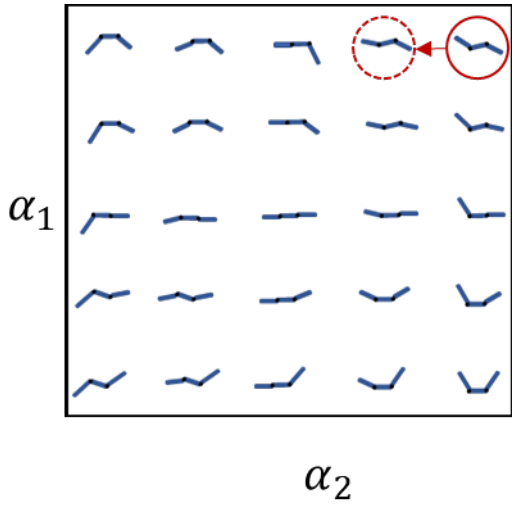
From geometry, V_i is a function of shape velocities and CoM velocities

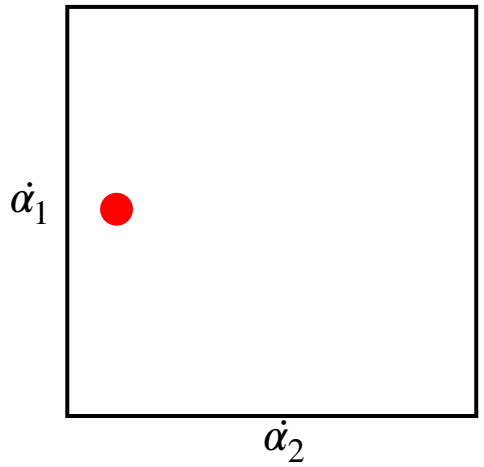
f_i: Reaction force on link i

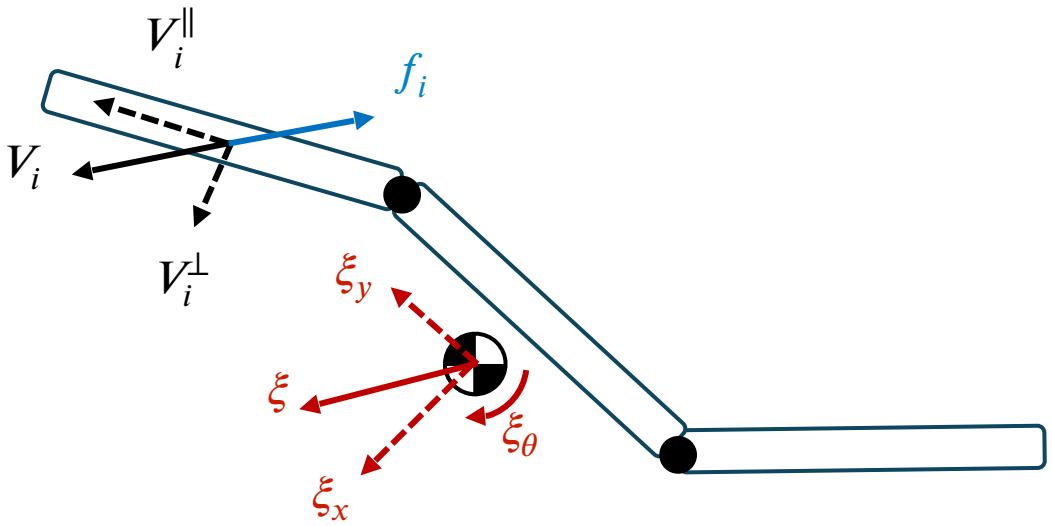
$$f_i = -\boldsymbol{C}V_i \qquad \sum_i f_i = 0$$

Quasi-static assumption

Viscous fluid



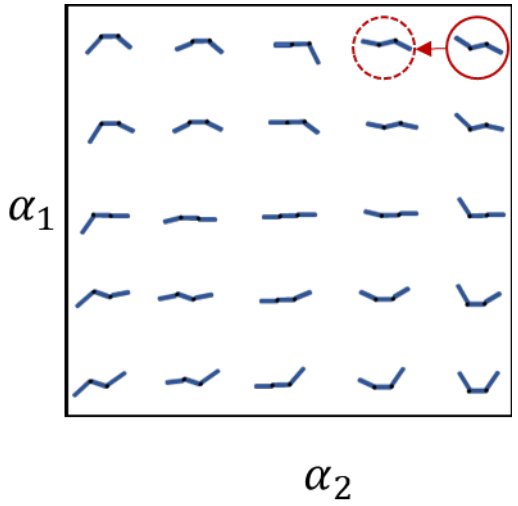


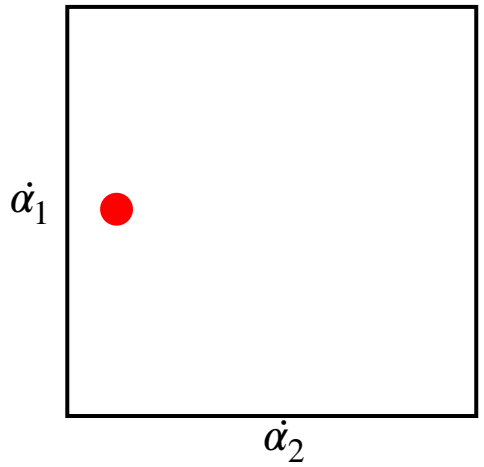


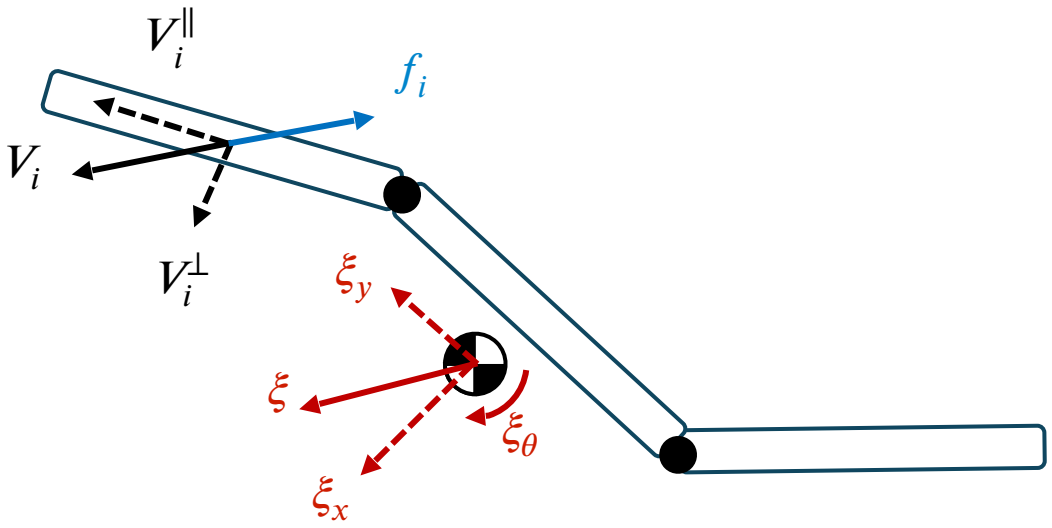
$$\begin{bmatrix} \xi_x \\ \xi_y \\ \xi_\theta \end{bmatrix} = f\left(\alpha_1, \alpha_2, \dot{\alpha}_1, \dot{\alpha}_2\right)$$

$$\begin{bmatrix} \xi_x \\ \xi_y \\ \xi_\theta \end{bmatrix} = A_{\alpha_1,\alpha_2}\left(\dot{\alpha}_1, \dot{\alpha}_2\right)$$

Viscous fluid







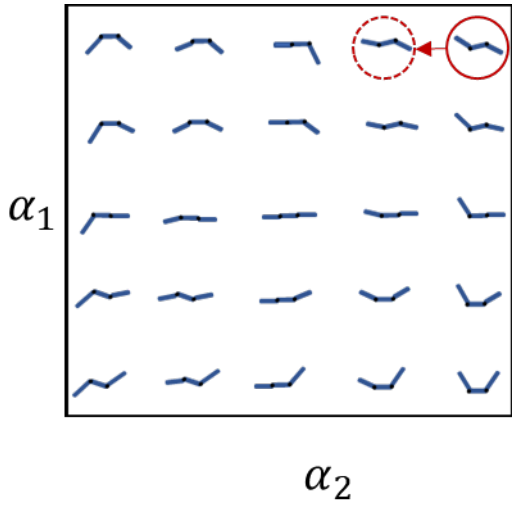
$$\begin{bmatrix} \xi_x \\ \xi_y \\ \xi_\theta \end{bmatrix} = f(\alpha_1, \alpha_2, \dot{\alpha}_1, \dot{\alpha}_2)$$

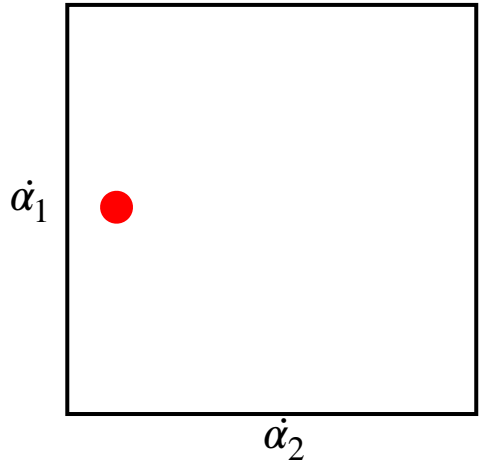
$$\begin{bmatrix} \xi_x \\ \xi_y \\ \xi_\theta \end{bmatrix} = A_{\alpha_1,\alpha_2}(\dot{\alpha}_1, \dot{\alpha}_2)$$

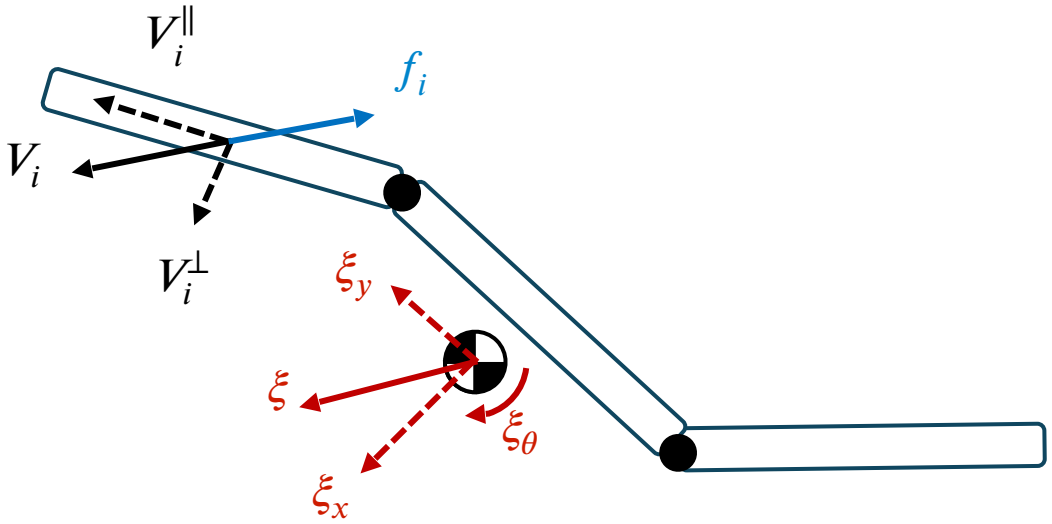
$$\begin{bmatrix} \xi_x \\ \xi_y \\ \xi_\theta \end{bmatrix} = A(\alpha_1, \alpha_2) \begin{bmatrix} \dot{\alpha}_1 \\ \dot{\alpha}_2 \end{bmatrix}$$

Linear connection

Viscous fluid





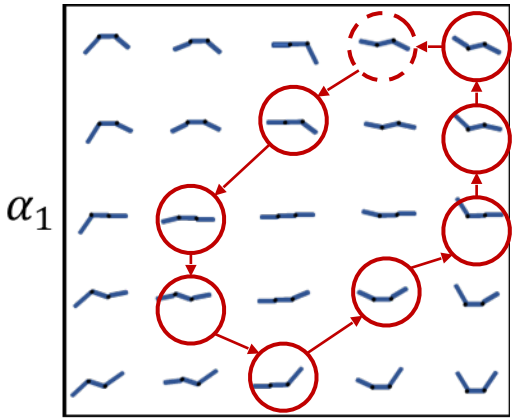


$$\begin{bmatrix} \xi_x \\ \xi_y \\ \xi_\theta \end{bmatrix} = \underbrace{A(\alpha_1,\ \alpha_2)}_{3\times 2\ \text{Connection matrix}} \begin{bmatrix} \dot{\alpha}_1 \\ \dot{\alpha}_2 \end{bmatrix}$$

Linear relationship

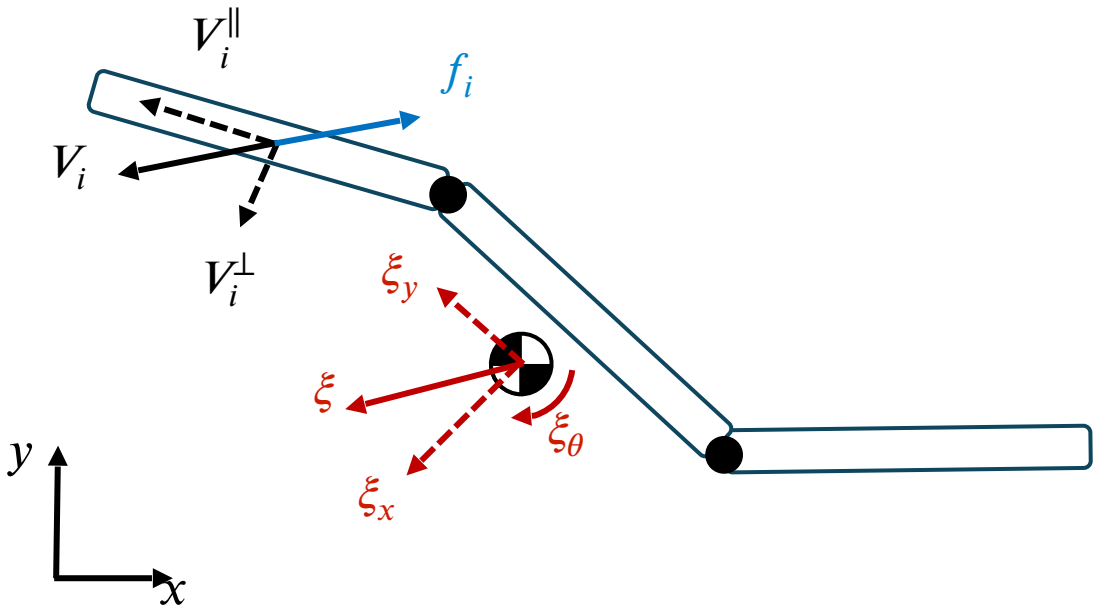
$$\begin{bmatrix} \xi_x \\ \xi_y \\ \xi_\theta \end{bmatrix} = \begin{bmatrix} A_x(\alpha_1, \alpha_2) \\ A_y(\alpha_1, \alpha_2) \\ A_\theta(\alpha_1, \alpha_2) \end{bmatrix} \begin{bmatrix} \dot{\alpha}_1 \\ \dot{\alpha}_2 \end{bmatrix}$$

Viscous fluid



Gait:

A close-loop path in the shape space



world frame

$\begin{bmatrix} x_{com} \\ y_{com} \\ \theta_{com} \end{bmatrix}$: the location of the swimmer w.r.t world frame

First order approximation:

$$\begin{bmatrix} \dot{x}_{com} \\ \dot{y}_{com} \\ \dot{\theta}_{com} \end{bmatrix} \approx \begin{bmatrix} \xi_x \\ \xi_y \\ \xi_\theta \end{bmatrix} = \begin{bmatrix} A_x(\alpha_1, \alpha_2) \\ A_y(\alpha_1, \alpha_2) \\ A_\theta(\alpha_1, \alpha_2) \end{bmatrix} \begin{bmatrix} \dot{\alpha}_1 \\ \dot{\alpha}_2 \end{bmatrix}$$

$$\Delta x_{com} = \int_T \xi_x dt = \int_{\partial\phi} A_x(\alpha_1, \alpha_2) d \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix}$$

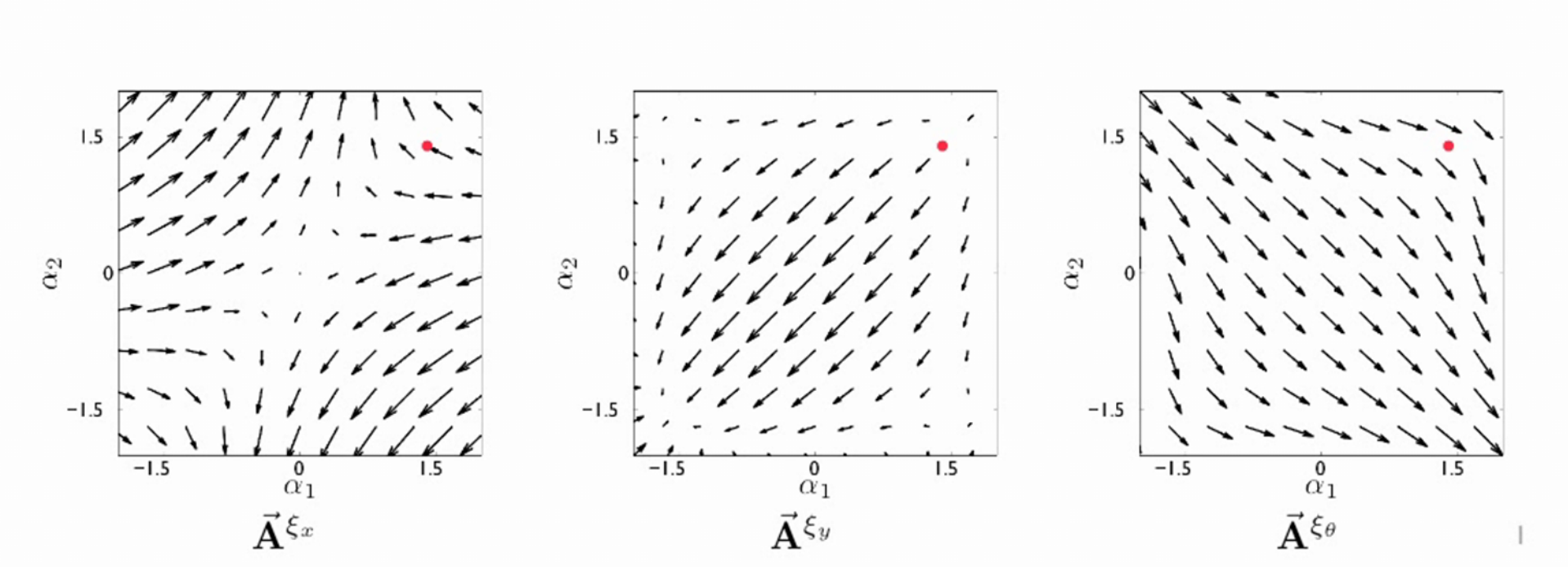
Approximate displacement by line integral over dot product

Geometric Mechanics

$$\xi = A(\alpha)\dot{\alpha}$$

$$\begin{bmatrix} \delta x \\ \delta y \\ \delta \theta \end{bmatrix} = \int_{\partial\phi} A(\alpha) d\alpha$$

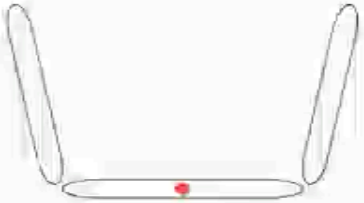
- Shape variable
- Connection vector field

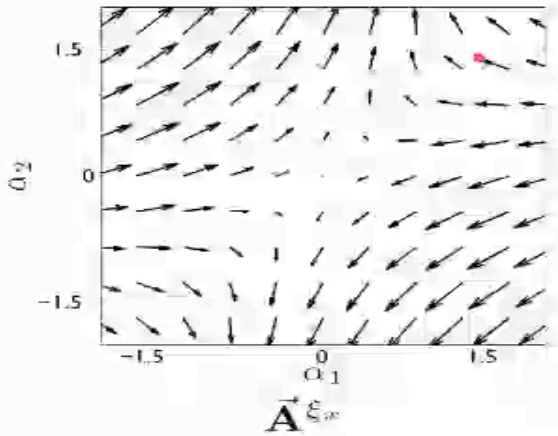


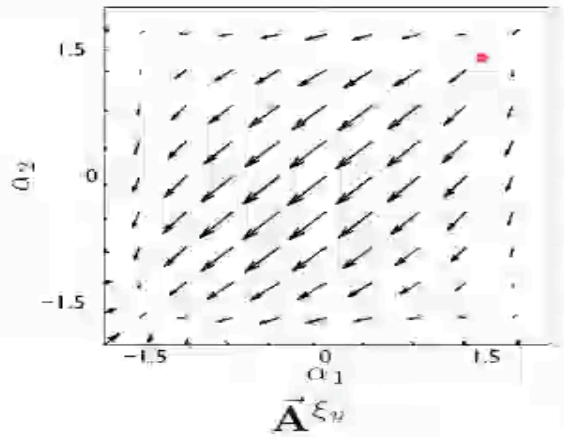
Geometric Mechanics

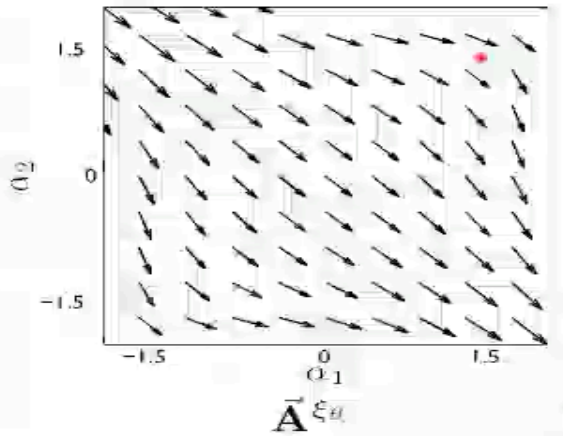
Strokes

- Shape variable
- Connection vector field







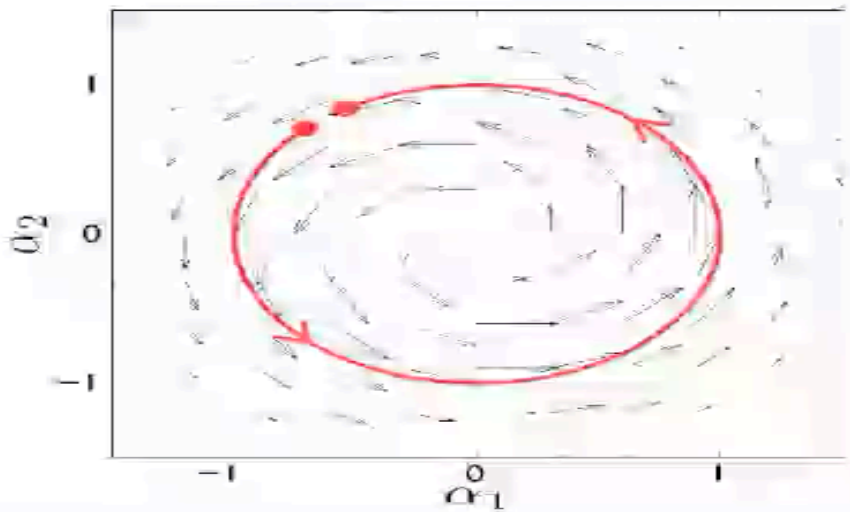


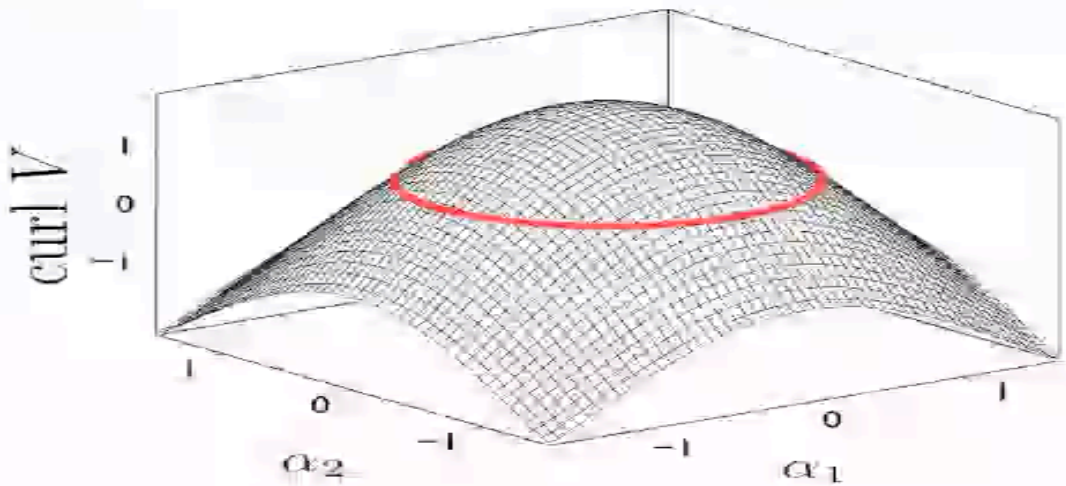
Geometric Mechanics

- Shape variable
- Connection vector field
- Height function

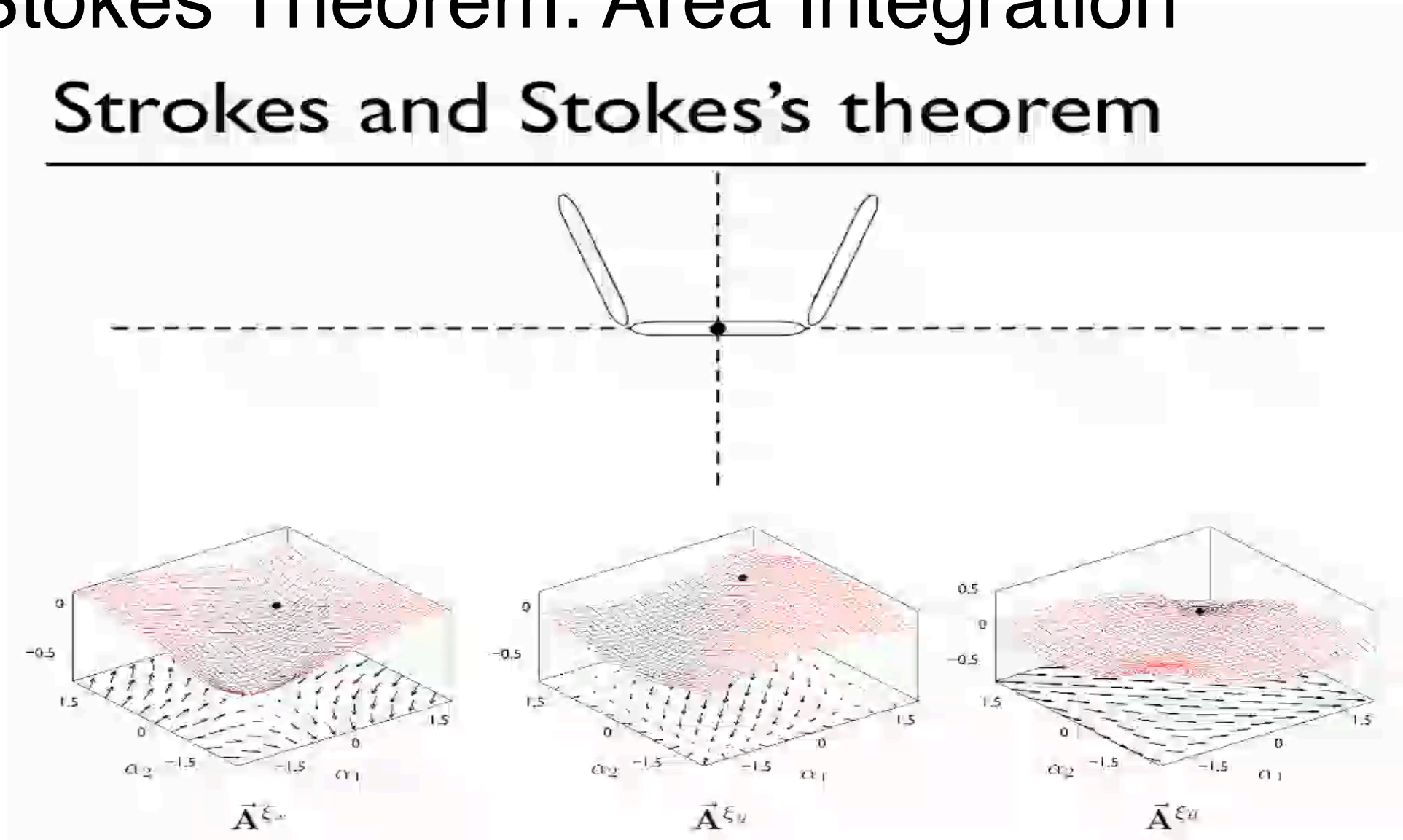
$$\begin{bmatrix} \delta x \\ \delta y \\ \delta \theta \end{bmatrix} = \int_{\partial\phi} A(\alpha)\boldsymbol{d}\alpha = \iint_{\phi} \nabla \times A(\alpha)\boldsymbol{d}\alpha$$

Stokes's theorem (Green's form)

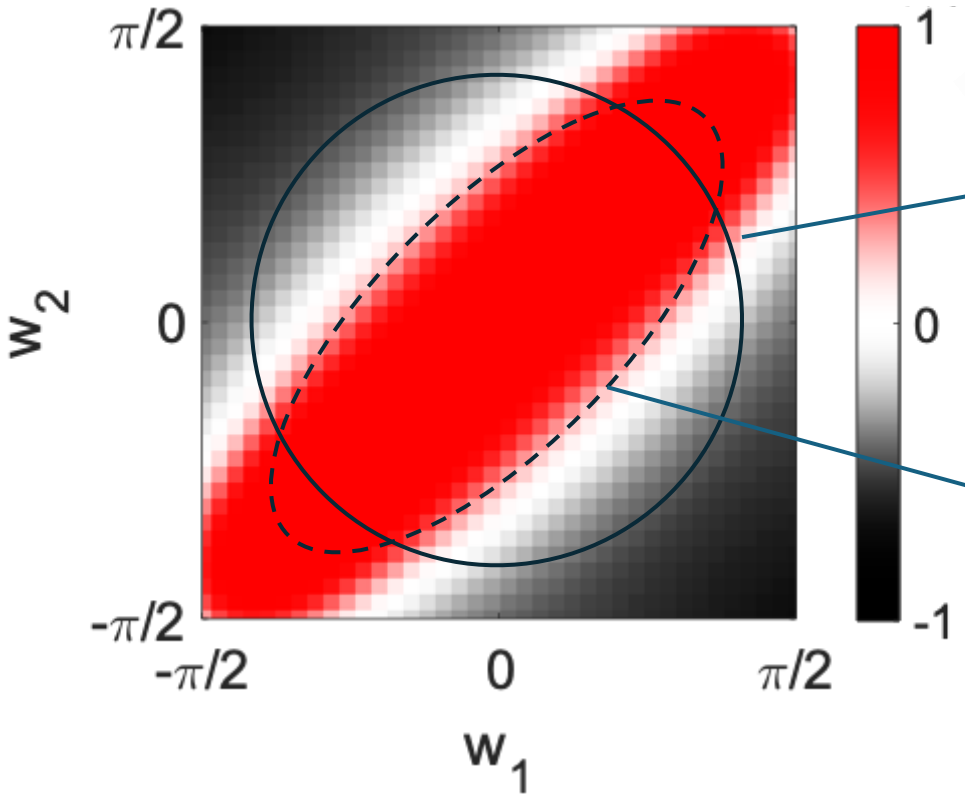




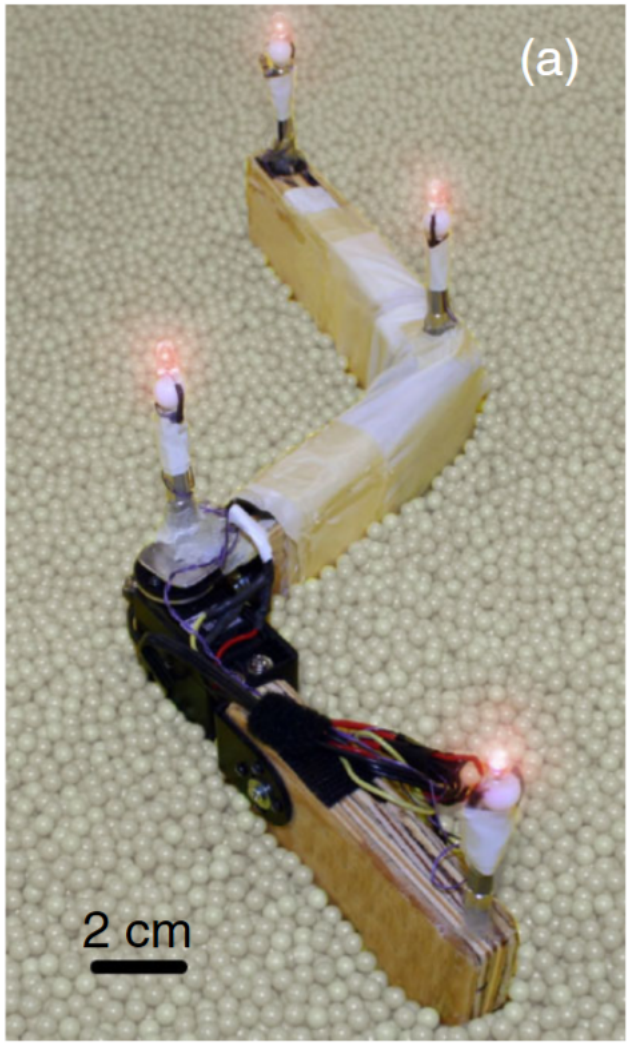
Stokes Theorem: Area Integration



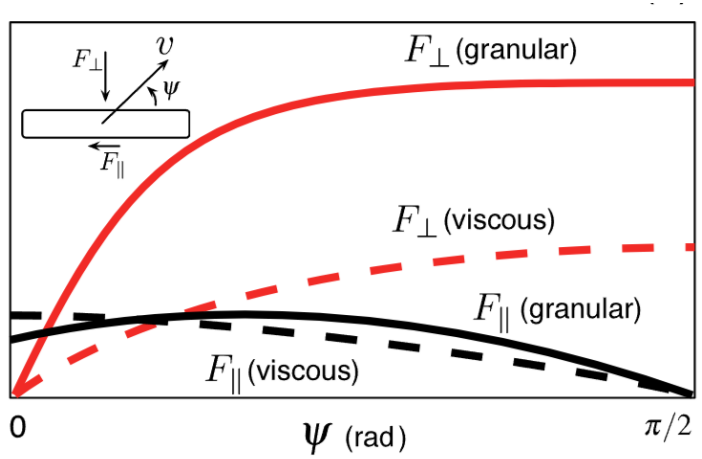
Height function



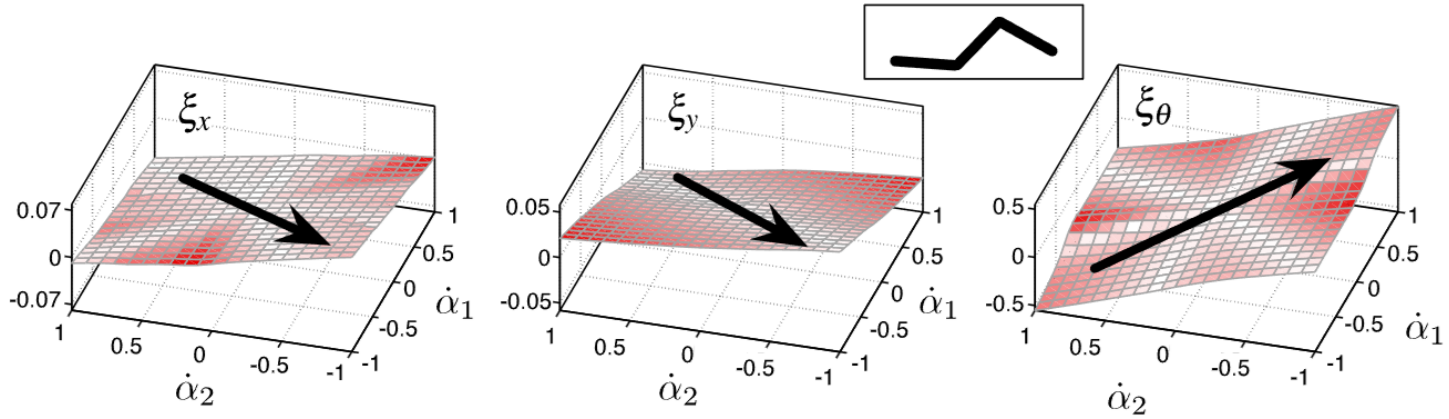
In granular media



Hatton, et al. *PRL,* 2013.

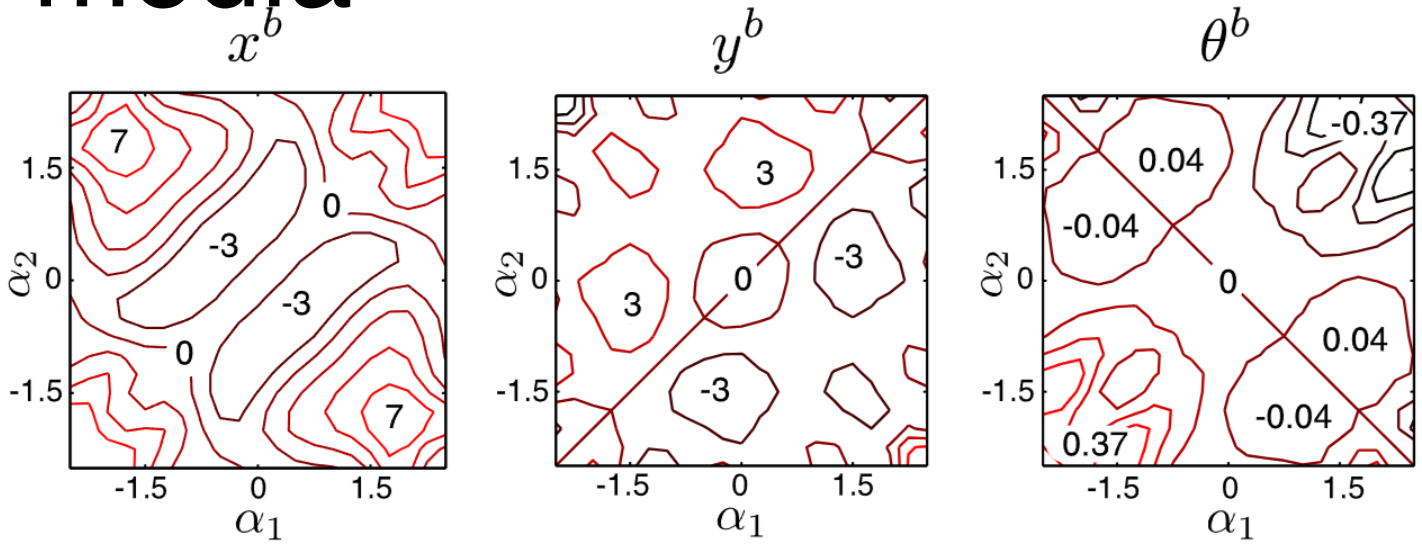


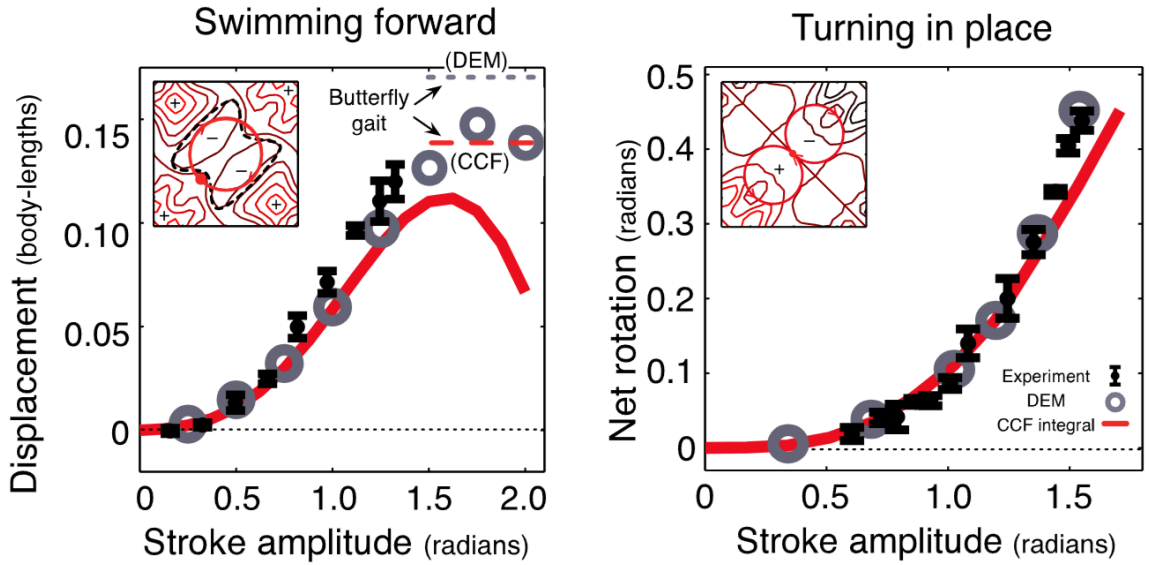
Granular media resistive force theory (RFT)



Empirical linear relationship $\dot{g}_p \approx A(\alpha)\dot{\alpha}$

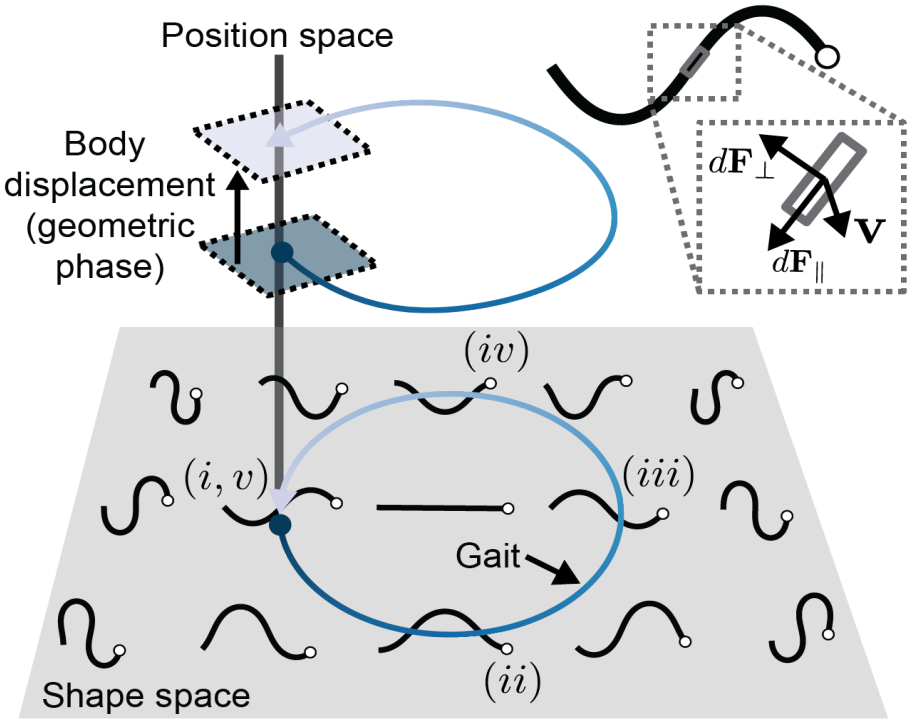
In granular media





Hatton, et al. *PRL,* 2013.

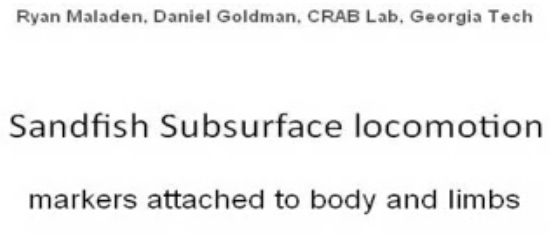
Higher dimensions



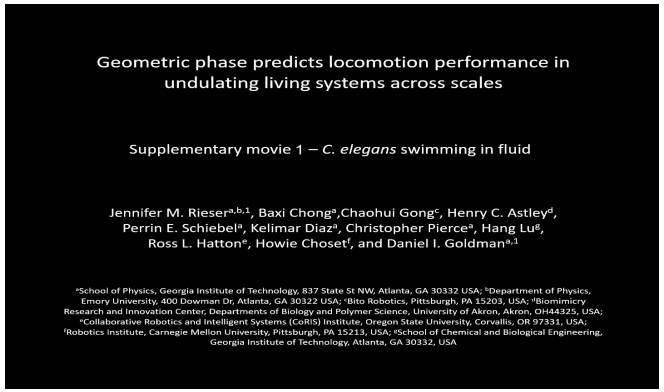
$$k(s,t) = \omega_1(t)\beta_1(s) + \omega_2(t)\beta_2(s)$$

$k(s,t)$: Curvature along body

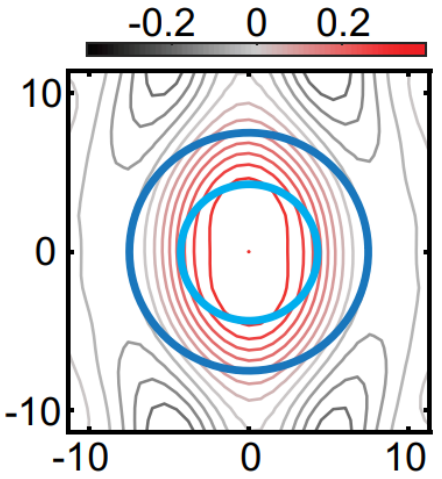
$\beta(s)$: Two principal components

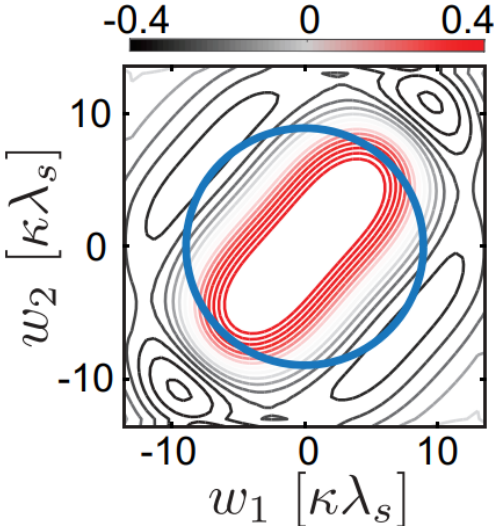


Sandfish swims in sand,
Maladen et al., *Science* 2009



C. elegans swimming,
Rieser et al., *PNAS* 2024





Tutorial: Model 3-link swimmer

```
% Number of gait cycles
period = 10;

% Mapping from velocity to force on each link
K = diag([1;10]);

% Link length (unit: cm)
L = 2;
```

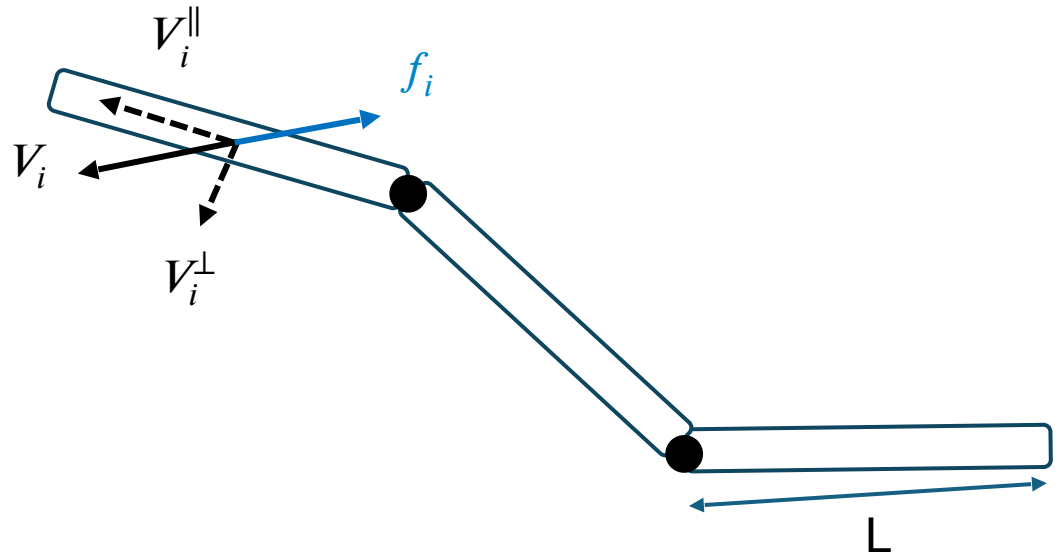
period: the number of cycles being simulated

MATLAB Function *M = diag(V):*

Input V: a vector,
Output: a square matrix with the elements of V on the main diagonal

$$f_i = -\mathbf{C}V_i = -\begin{bmatrix} C_{\parallel} & \\ & C_{\perp} \end{bmatrix} \begin{bmatrix} V_i^{\parallel} \\ V_i^{\perp} \end{bmatrix}$$

K: mapping from the velocity to force: Equations



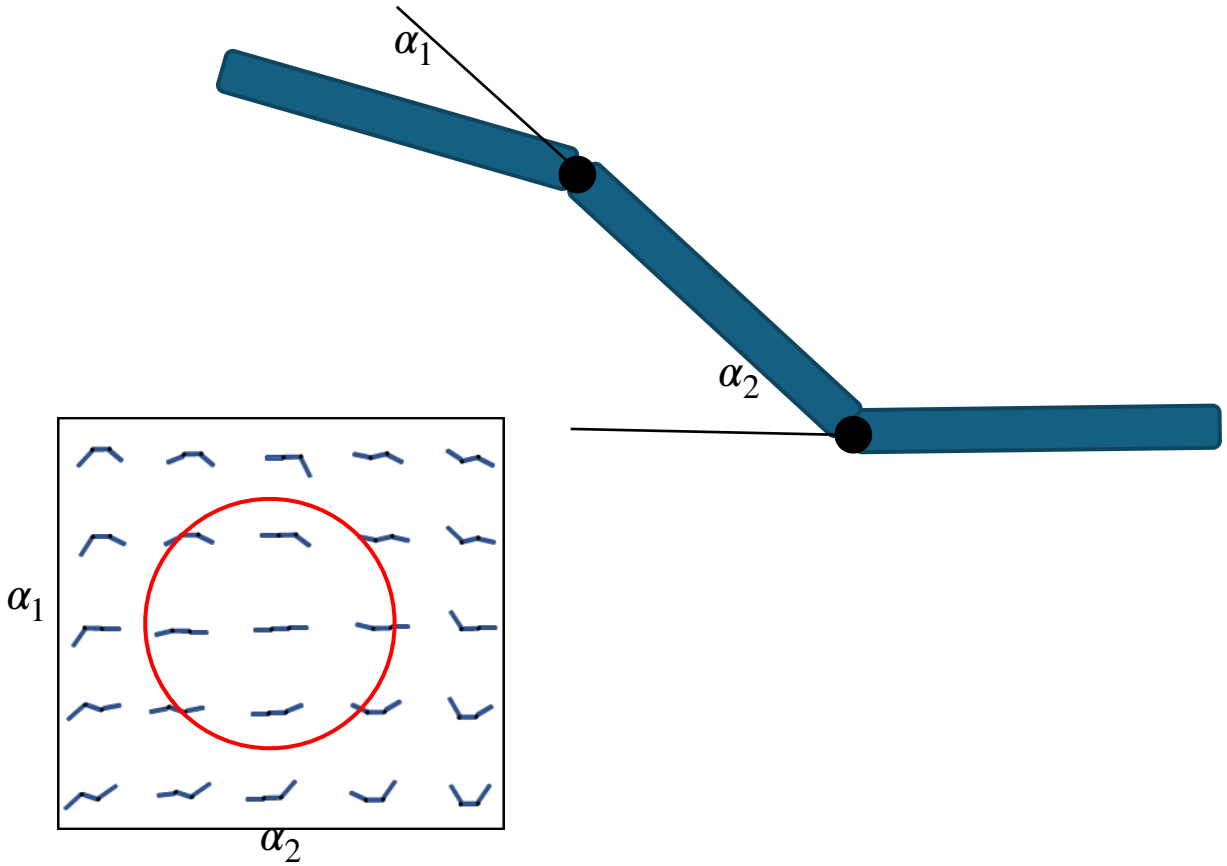
```
% Body joint angle as a function of time
% alpha is a function of time: at each input 't', ]
alpha = @(t)        [pi/2*sin(t); pi/2*sin(t+pi/2)];

% Body joint anglular velocity as a function of time
d_alpha = @(t)      [pi/2*cos(t); pi/2*cos(t+pi/2)];
```

A self-defined function

A = alpha(t)

Input: t
Output: 2x1 vector
A = $[\frac{\pi}{2}\sin(t), \frac{\pi}{2}\cos(t)]$



Joint angle prescribed as a function of time:

$$\boldsymbol{\alpha}(t) = \begin{bmatrix} \alpha_1(t) \\ \alpha_2(t) \end{bmatrix} = \begin{bmatrix} \frac{\pi}{2}\sin(t) \\ \frac{\pi}{2}\cos(t) \end{bmatrix}$$

Show should the gait path look like?

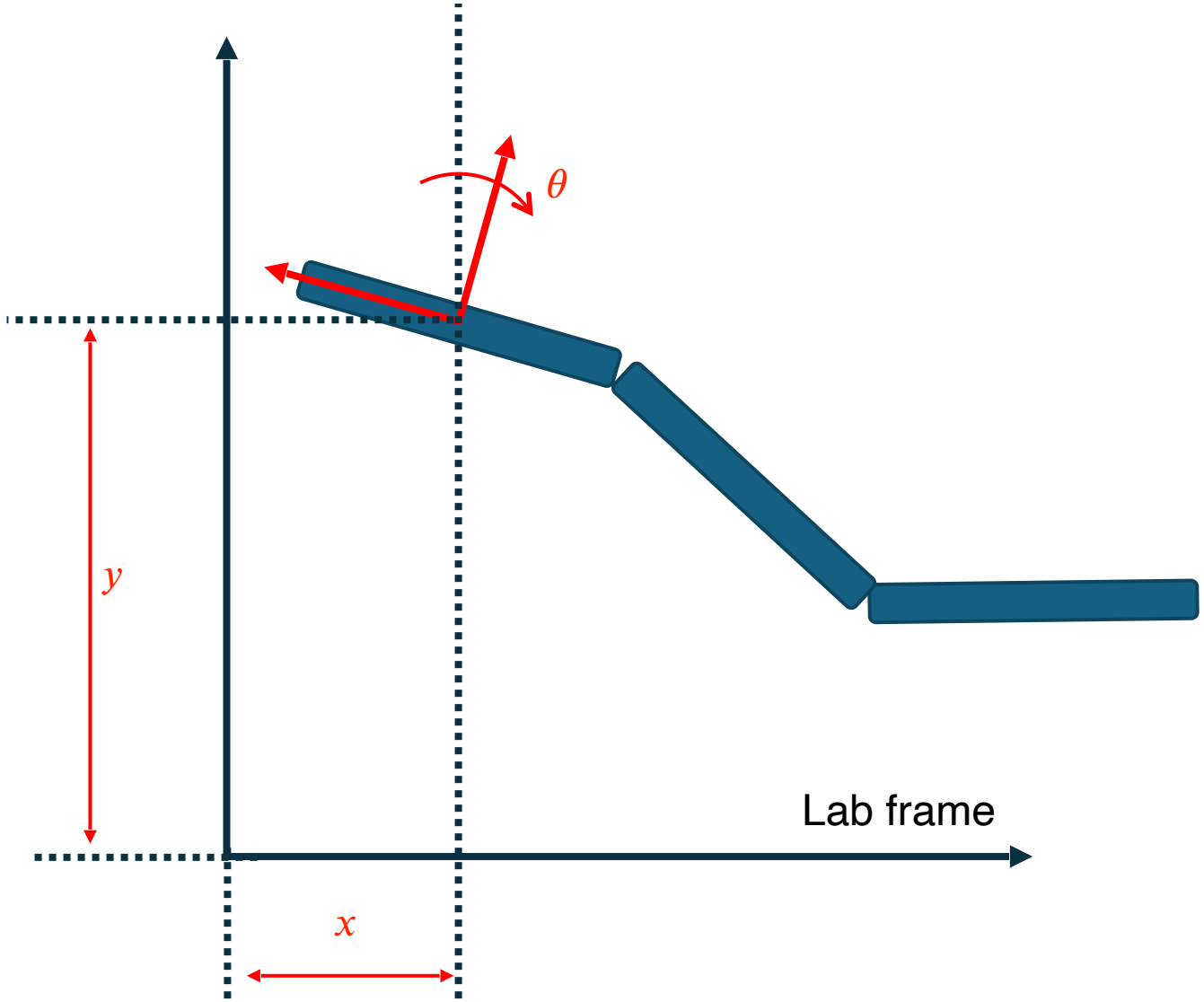
```
17      % current time
18 -    t = 0;
19
20      % Time incremental
21 -    dt = 2*pi/100;
22
23      % Starting robot position
24 -    g_h = eye(3);
```

Line 18: set the current time to be 0

Line 21: sample 100 points over a cycle

Line 24: set the starting position to be I_3

In general: $g_h = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & x \\ \sin(\theta) & \cos(\theta) & y \\ 0 & 0 & 1 \end{bmatrix}$



```
26      % Generate code to make the plot
27 -    figure()
28 -    hold on;
29 -    axis equal;
30 -    axis([-10 10 -10 10]*L);
31 -    xlabel('X (cm)','fontsize',15);
32 -    ylabel('Y (cm)','fontsize',15);
```

Line 27: create a figure

Line 28: holds the current plot, so that subsequent plot commands add to the existing ones

Line 29: set the aspect ratio of the plot is 1:1

Line 30: set the upper bound and lower bound of the plot

Line 31: add label to the x axis. Set the font size to be 15

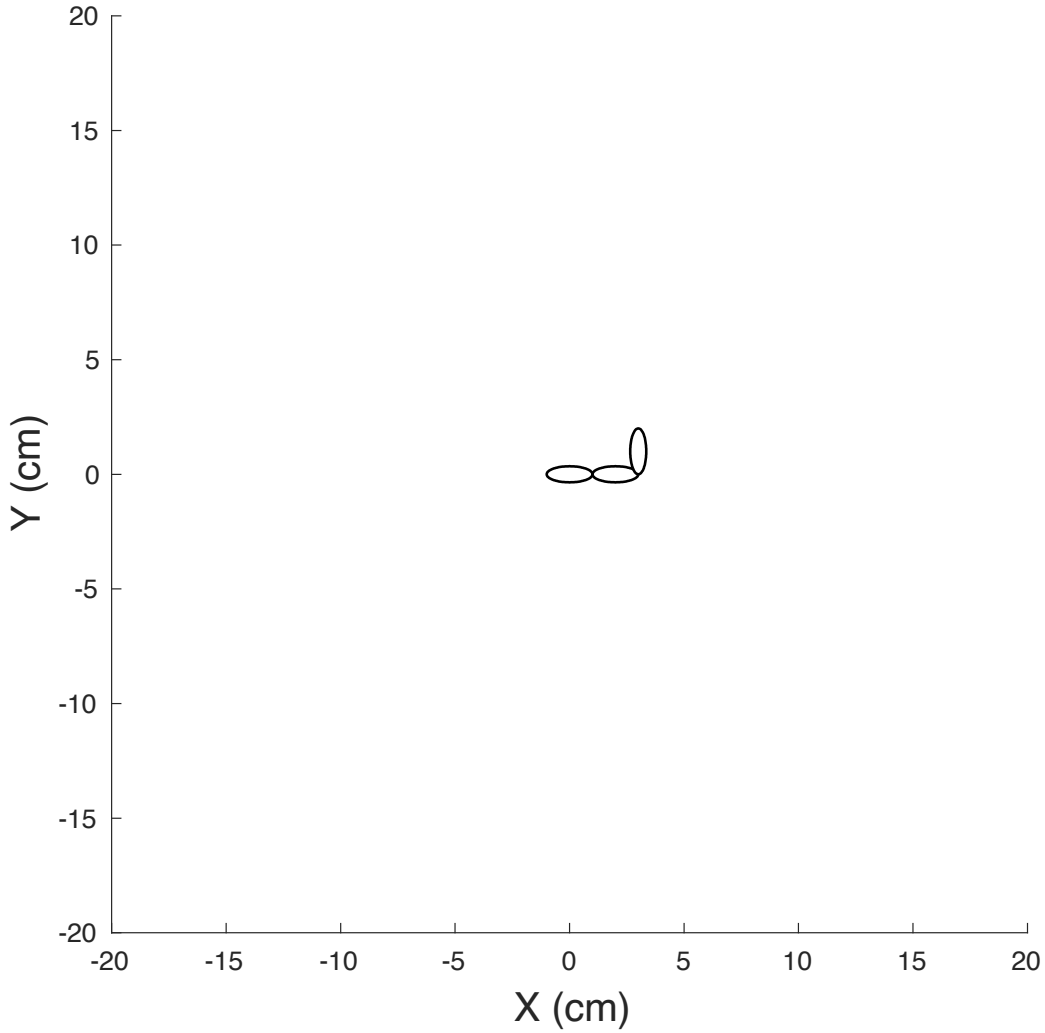
Line 32: add label to the y axis. Set the font size to be 15

```
34      % Plot the robot at the initial position
35 -    [h] = drawActiveFrameSnake(g_h,alpha(t),L);
```

Plot the robot at the initlal position

Take three input:

g_h: the position of the head
alpha(t): joint angles
L: link length



```
37 - while t < period*2*pi
38
39       % update time
40 -     t = t + dt;
41
42       % compute the body velocity at time t
43 -     xi = computeBodyVelocity(alpha(t), d_alpha(t), K, L);
44
45       % Update the robot position
46 -     g_h = g_h*expm(dt*xiHat(xi));
47
48
49
50       % Delete the plot of robot at previous position
51 -     for i = 1:size(h,2)
52 -         delete(h{i});
53 -     end
54
55       % Plot the robot at the current position
56 -     [h] = drawActiveFrameSnake(g_h,alpha(t),L);
57
58       % Get the drawings now
59 -     drawnow;
60 - end
```

Line 37, 40, 60:

Create a loop, repeat Line 42-59 over 10 periods

```
37 - while t < period*2*pi
38
39      % update time
40 -    t = t + dt;
41
42      % compute the body velocity at time t
43 -    xi = computeBodyVelocity(alpha(t), d_alpha(t), K, L);
44
45      % Update the robot position
46 -    g_h = g_h*expm(dt*xiHat(xi));
47
48
49
50      % Delete the plot of robot at previous position
51 -    for i = 1:size(h,2)
52 -        delete(h{i});
53 -    end
54
55      % Plot the robot at the current position
56 -    [h] = drawActiveFrameSnake(g_h,alpha(t),L);
57
58      % Get the drawings now
59 -    drawnow;
60 - end
```

Line 43: compute the body velocity at time t

```
37 - while t < period*2*pi
38
39      % update time
40 -    t = t + dt;
41
42      % compute the body velocity at time t
43 -    xi = computeBodyVelocity(alpha(t), d_alpha(t), K, L);
44
45      % Update the robot position
46 -    g_h = g_h*expm(dt*xiHat(xi));
47
48
49
50      % Delete the plot of robot at previous position
51 -    for i = 1:size(h,2)
52 -        delete(h{i});
53 -    end
54
55      % Plot the robot at the current position
56 -    [h] = drawActiveFrameSnake(g_h,alpha(t),L);
57
58      % Get the drawings now
59 -    drawnow;
60 - end
```

Line 46: update the head position according to the calculated body velocity

```
37 -  while t < period*2*pi
38
39        % update time
40 -      t = t + dt;
41
42        % compute the body velocity at time t
43 -      xi = computeBodyVelocity(alpha(t), d_alpha(t), K, L);
44
45        % Update the robot position
46 -      g_h = g_h*expm(dt*xiHat(xi));
47
48
49
50        % Delete the plot of robot at previous position
51 -      for i = 1:size(h,2)
52 -          delete(h{i});
53 -      end
54
55        % Plot the robot at the current position
56 -      [h] = drawActiveFrameSnake(g_h,alpha(t),L);
57
58        % Get the drawings now
59 -      drawnow;
60 -  end
```

Line 51-53: delete what we draw on line 35

```
37 -  while t < period*2*pi
38
39        % update time
40 -      t = t + dt;
41
42        % compute the body velocity at time t
43 -      xi = computeBodyVelocity(alpha(t), d_alpha(t), K, L);
44
45        % Update the robot position
46 -      g_h = g_h*expm(dt*xiHat(xi));
47
48
49
50        % Delete the plot of robot at previous position
51 -      for i = 1:size(h,2)
52 -          delete(h{i});
53 -      end
54
55        % Plot the robot at the current position
56 -      [h] = drawActiveFrameSnake(g_h,alpha(t),L);
57
58        % Get the drawings now
59 -      drawnow;
60 -  end
```

Line 56: draw the robot at the current time

```
37 -  while t < period*2*pi
38
39        % update time
40 -      t = t + dt;
41
42        % compute the body velocity at time t
43 -      xi = computeBodyVelocity(alpha(t), d_alpha(t), K, L);
44
45        % Update the robot position
46 -      g_h = g_h*expm(dt*xiHat(xi));
47
48
49
50        % Delete the plot of robot at previous position
51 -      for i = 1:size(h,2)
52 -          delete(h{i});
53 -      end
54
55        % Plot the robot at the current position
56 -      [h] = drawActiveFrameSnake(g_h,alpha(t),L);
57
58        % Get the drawings now
59 -      drawnow;
60 -  end
```

Line 59: literally draw it now (as MATLAB doing simulation)

Otherwise: all drawing are created all together after the simulation finished

```
62 -    displacement.xy = norm(g_h(1:2,3))/(3*L);
63 -    close(gcf)
```

Line 62: read the displacement after 10 cycles

Line 63: close the figure