



Handwritten physics equations and a rocket diagram:

Diagram of a rocket with velocity \vec{v} and acceleration \vec{a} .

$$\Sigma F = ma = \dot{p}(t) = -G \Delta t$$
$$m \Delta v + u \Delta m + \Delta v \Delta m = -G \Delta t$$
$$m \frac{\Delta v}{\Delta t} + u \frac{\Delta m}{\Delta t} + \Delta v \frac{\Delta m}{\Delta t} = -G$$
$$\Delta t \rightarrow 0$$
$$m v'(t) + u m'(t) = -mg$$
$$v'(t) + g = -u \frac{m'(t)}{m} \quad || \int dt$$
$$v(t) + u \ln(m) = -gt + C$$
$$v(0) = 0 \Rightarrow C = u \ln(m_0)$$
$$v(t) = -gt + u \ln \frac{m_0}{m}$$

ALGORITHMS PHILOSOPHY



C4dynamics

Many engineers develop their algorithms detached from context.

If you are engaged with physical systems, follow this..



C4dynamics

A POINT

It may be

Car

Plane

Bird

...

A point on a path



C4dynamics

A POINT

Properties

Position

Velocity

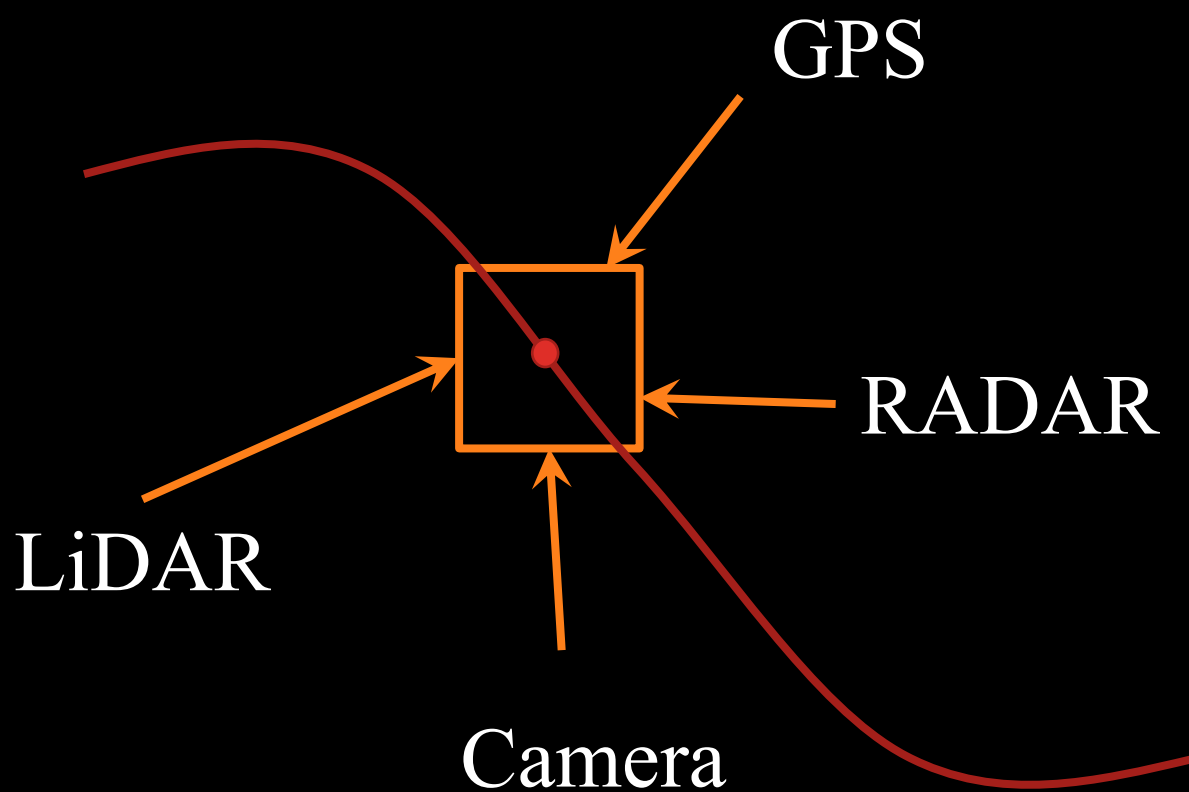
Acceleration

A point on a path

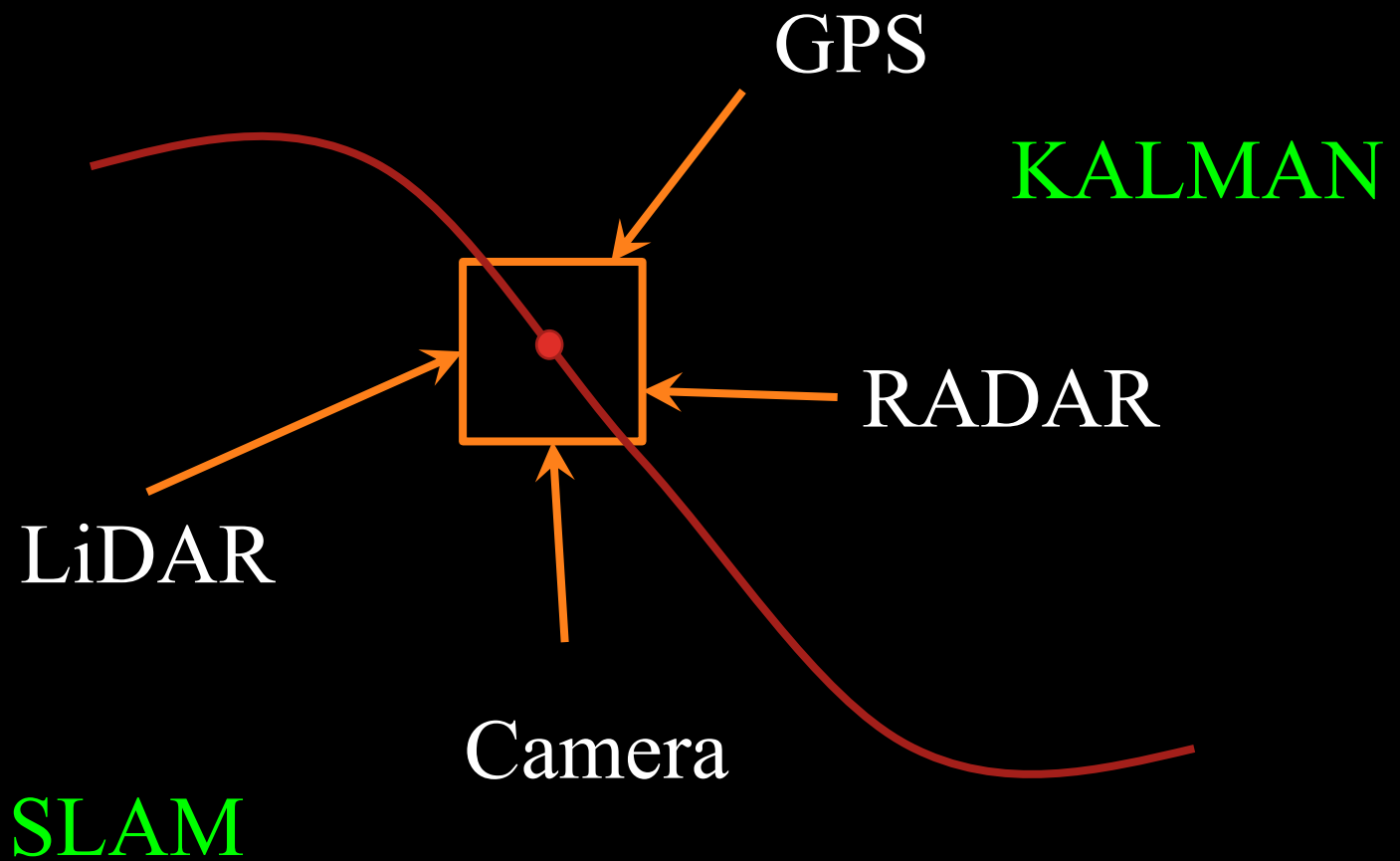
A red curved line representing a path, starting from the left, curving upwards, then downwards, and finally curving back upwards towards the right. A small red dot is placed on the downward-sloping part of the curve, representing a point on the path.

C4dynamics

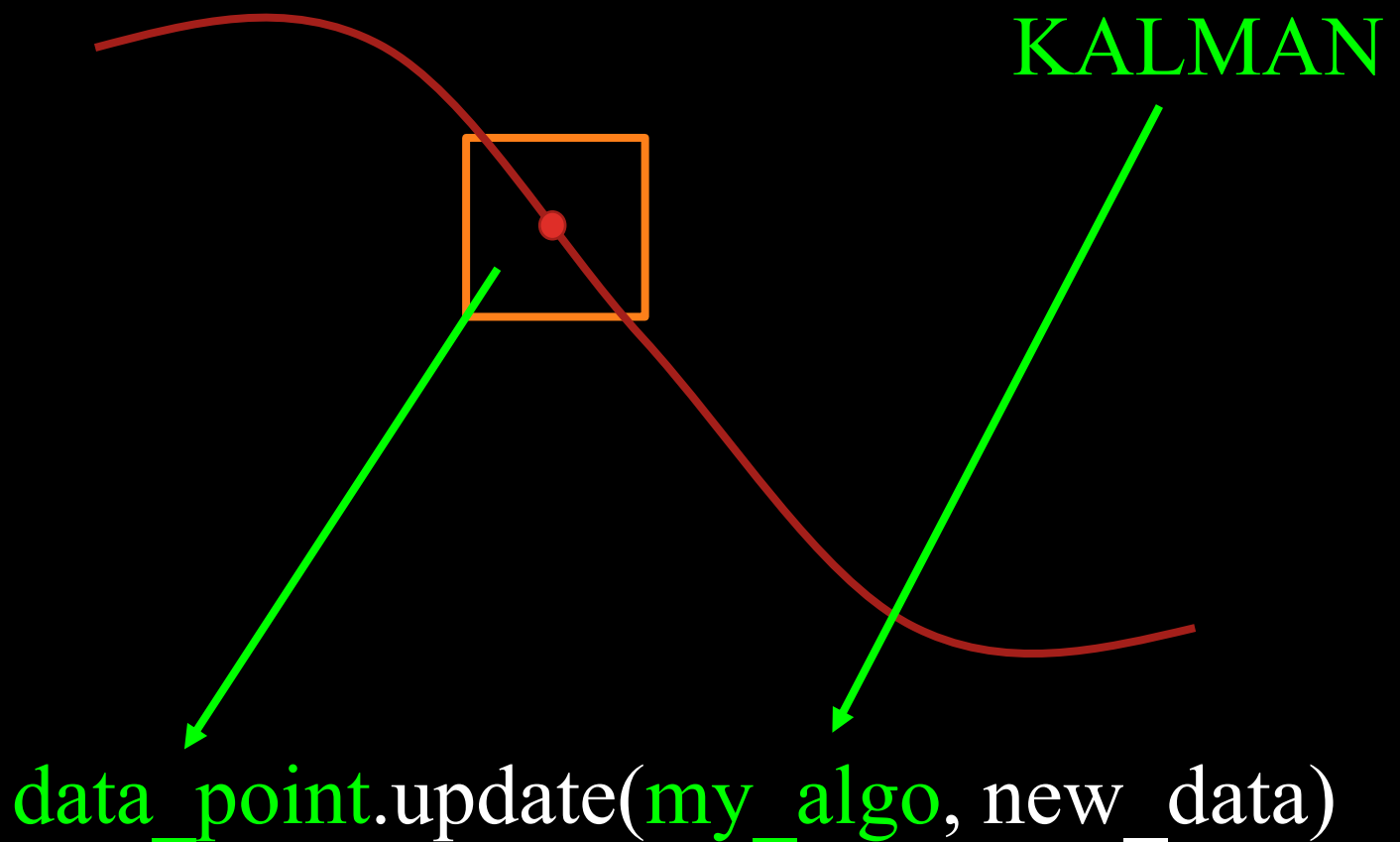
UPDATE FROM SENSORS



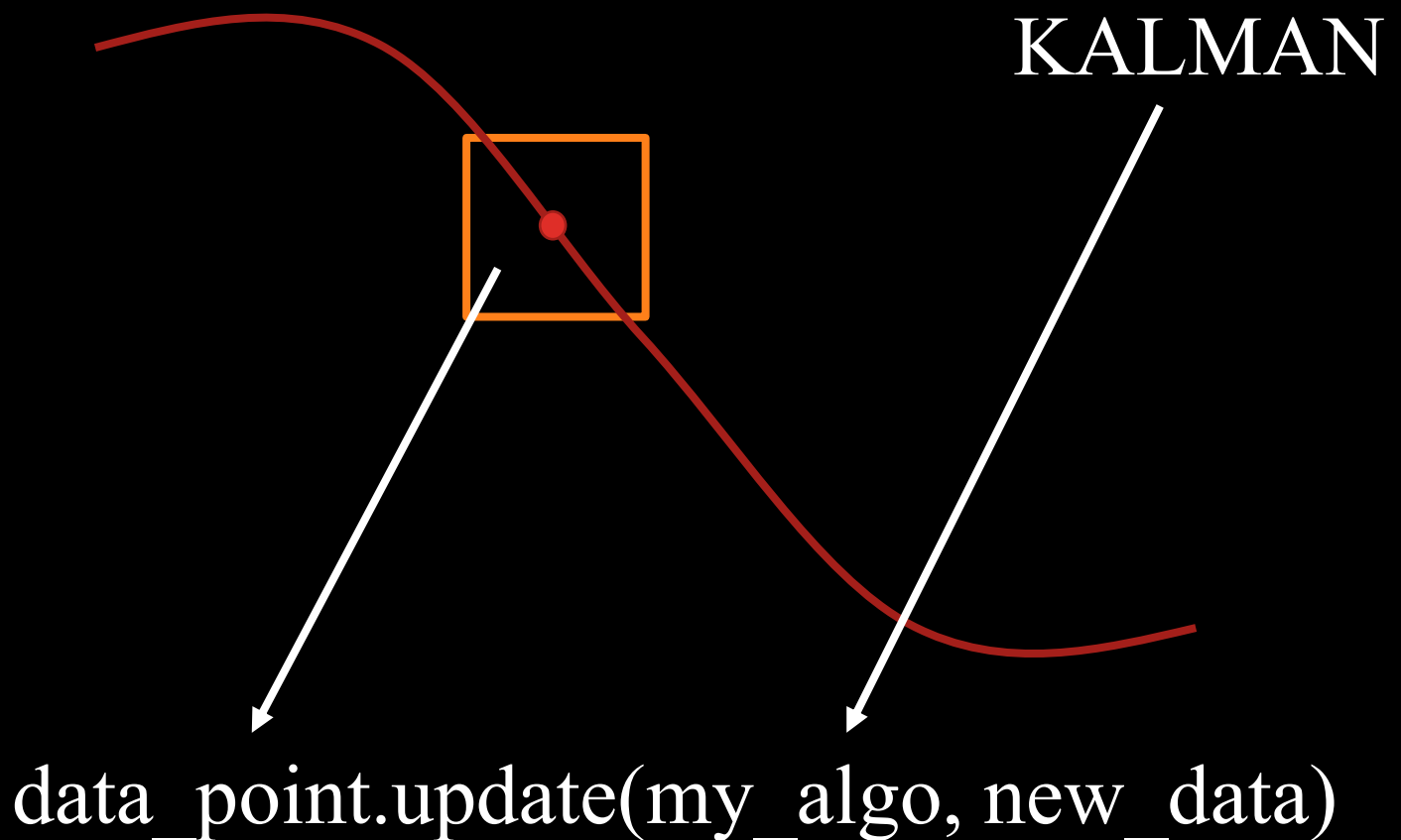
DIFFERENT ALGORITHMS



POINT'S POINT OF VIEW



POINT'S POINT OF VIEW



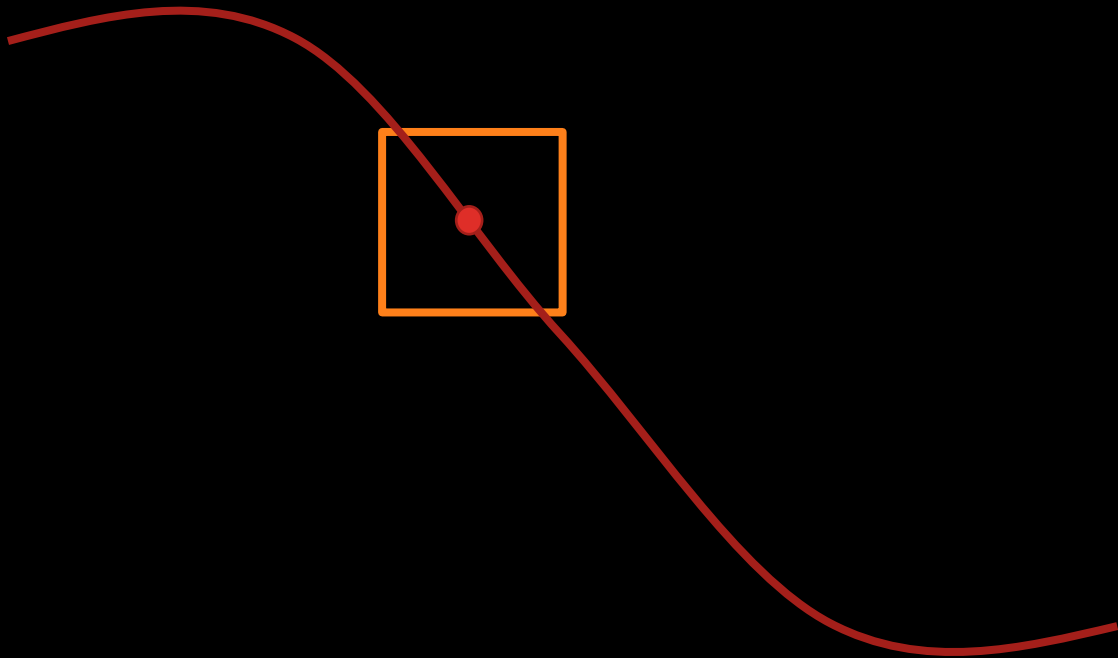
Binds the algorithm with the physical effect!



C4dynamics

So..

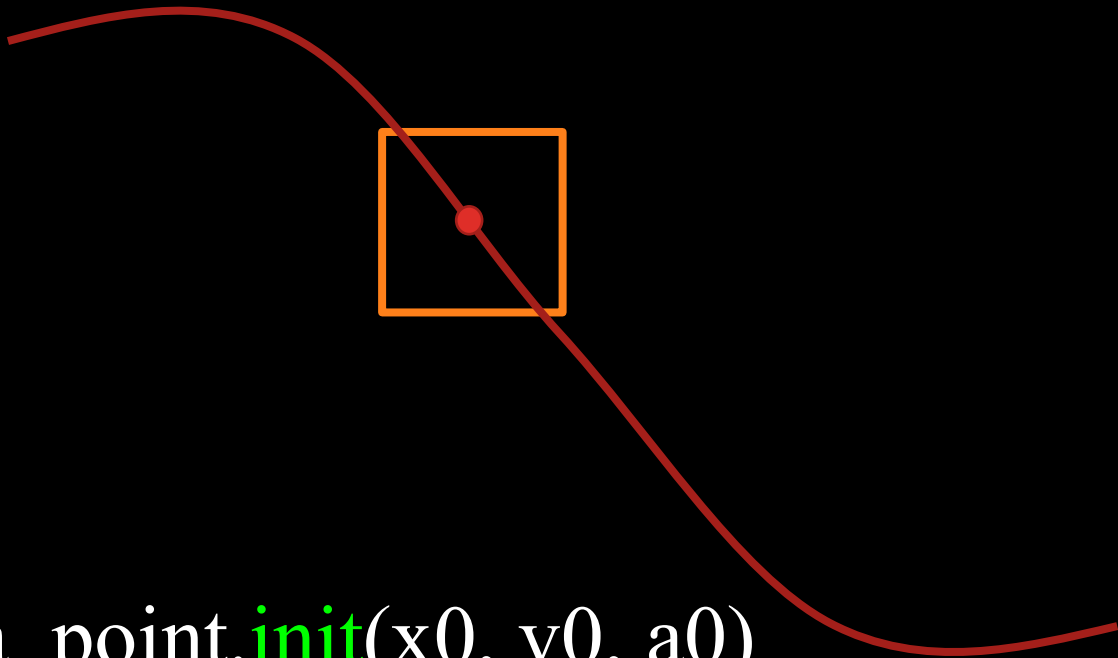
**DO EVERYTHING FROM
ONE PLACE!**



C4dynamics

So..

DO EVERYTHING FROM ONE PLACE



```
data_point.init(x0, v0, a0)
```

```
new_data = my_sensor.measure()
```

```
data_point.update(my_algo, new_data)
```

```
data_point.plot(t0, tf)
```



C4dynamics

Whatever your algorithm is,
always look from the object's
point of view.

This is Algorithm Context!



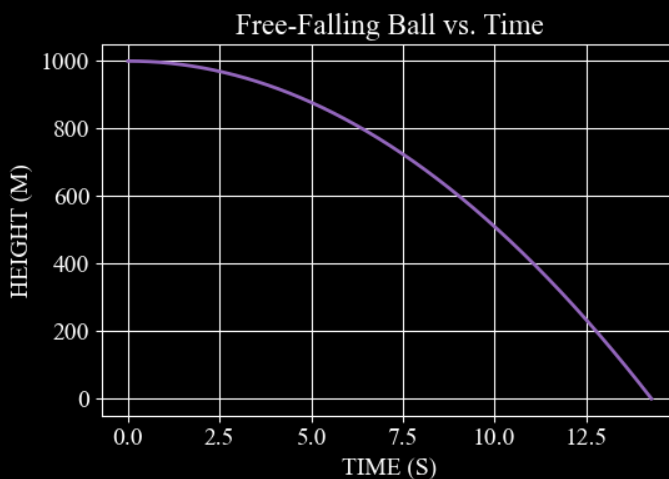
C4dynamics

Want to work with cool algorithm framework?

Download now C4dynamics and run freefall.py

Follow the instructions there:

<https://github.com/C4dynamics/C4dynamics/blob/main/examples/freefall.py>

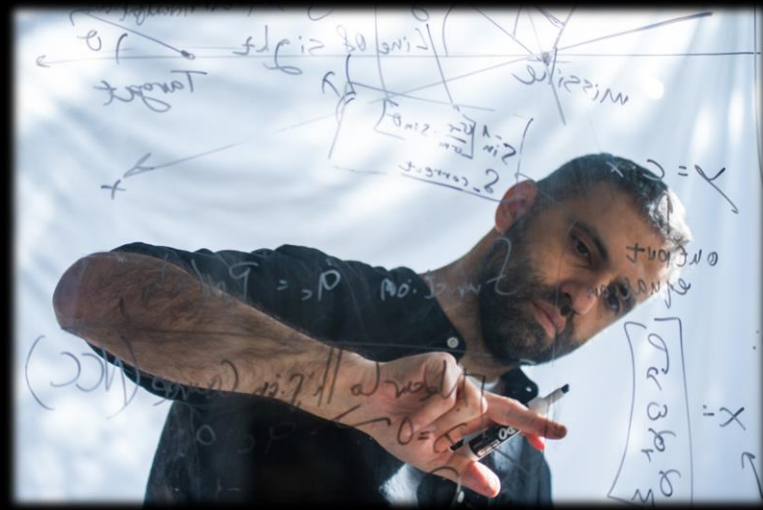


C4dynamics

A cutting-edge, high-standard algorithms development framework



C4dynamics



I SEE DEAD PEOPLE!

$$\rho = c_T \cdot \cos \delta - v_m \cdot \cos \beta$$

$$\dot{\lambda} = \omega$$

$$\dot{\omega} = -2 \cdot \omega (c_T \cdot \cos \delta - v_m \cdot \cos \beta) / \rho + a_T \cdot \cos \delta / \rho - a_m \cdot \cos \beta / \rho$$

$$\delta_T = a_T / \omega_T$$

$$\delta_m = a_m / \omega_m$$

$$\dot{x} = A \cdot x + b \cdot u$$

$$y = c \cdot x$$

$$\sqrt{a_c} = N \cdot v_m \cdot \dot{x}$$

$$\sqrt{y} = c \cdot x$$

$$\sqrt{f} = W \cdot x$$

Diagram illustrating missile guidance geometry:

- Coordinate system with x and y axes.
- Missile position and velocity vector v_m .
- Target position and velocity vector v_T .
- Line of sight between missile and target.
- Angles δ and β relative to the line of sight.
- Acceleration vectors a_c and a_m .
- Equation $y = c \cdot x$ labeled as "output equation".
- Equation $S = A \cdot \frac{v_m \cdot \sin \delta}{\omega_m}$ labeled as $S_{correct}$.
- Condition $k = \frac{v_T}{\omega_m} < 1$ labeled as "Necessary Condition for successful hit in pure pursuit in tail round".
- Function $q_c = P_N(\phi_i)$.
- Substitution $\lambda = 0 \rightarrow q_c = 0$.
- Equation $\int_{-\infty}^{\infty} \dots$.



Gavriel Weinberger



C4dynamics