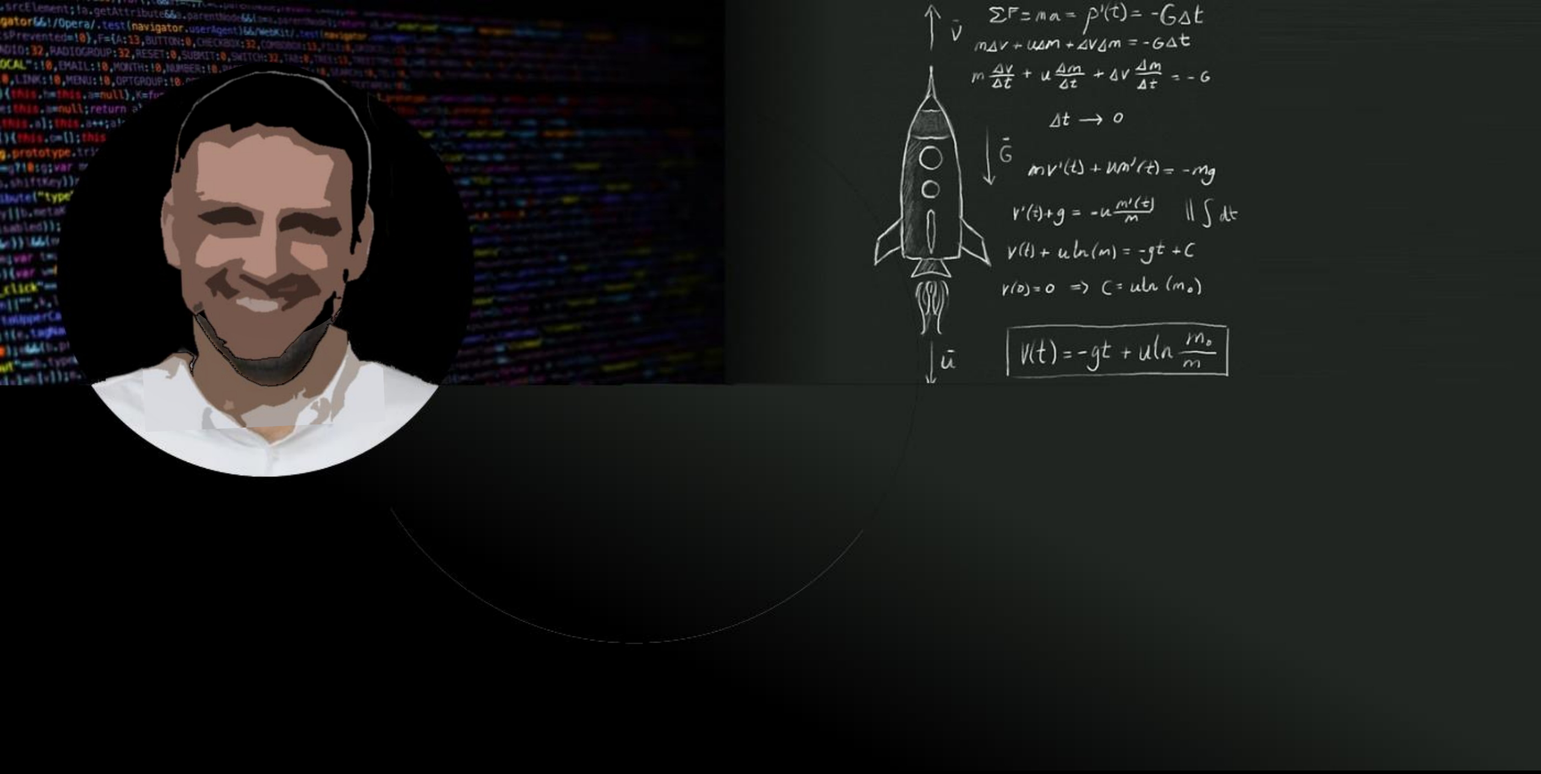

$\bar{V}$
$\Sigma F = ma = p'(t) = -G\Delta t$
$m\Delta v + u\Delta m + \Delta v\Delta m = -G\Delta t$
$m\frac{\Delta v}{\Delta t} + u\frac{\Delta m}{\Delta t} + \Delta v\frac{\Delta m}{\Delta t} = -G$
$\Delta t \longrightarrow 0$
$\bar{G}$
$mv'(t) + um'(t) = -mg$
$v'(t) + g = -u\frac{m'(t)}{m} \quad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

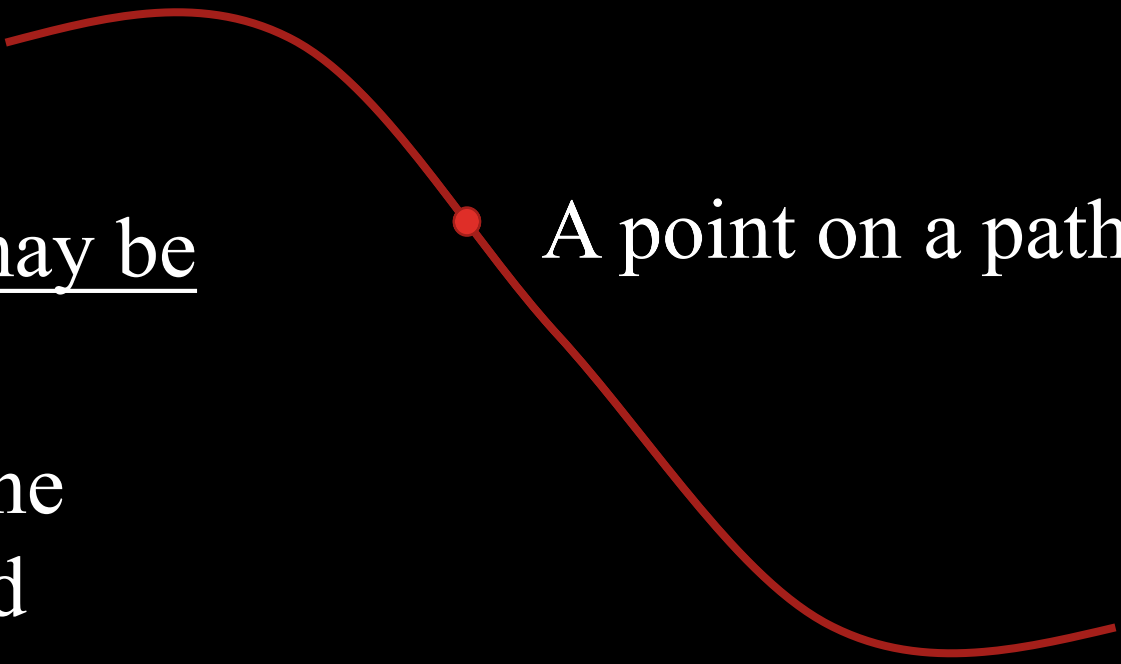
ALGORITHMS PHILOSOPHY

C4dynamics

Many engineers develop their algorithms detached from context.

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

A POINT



A point on a path

It may be

Car

Plane

Bird

…

C4dynamics

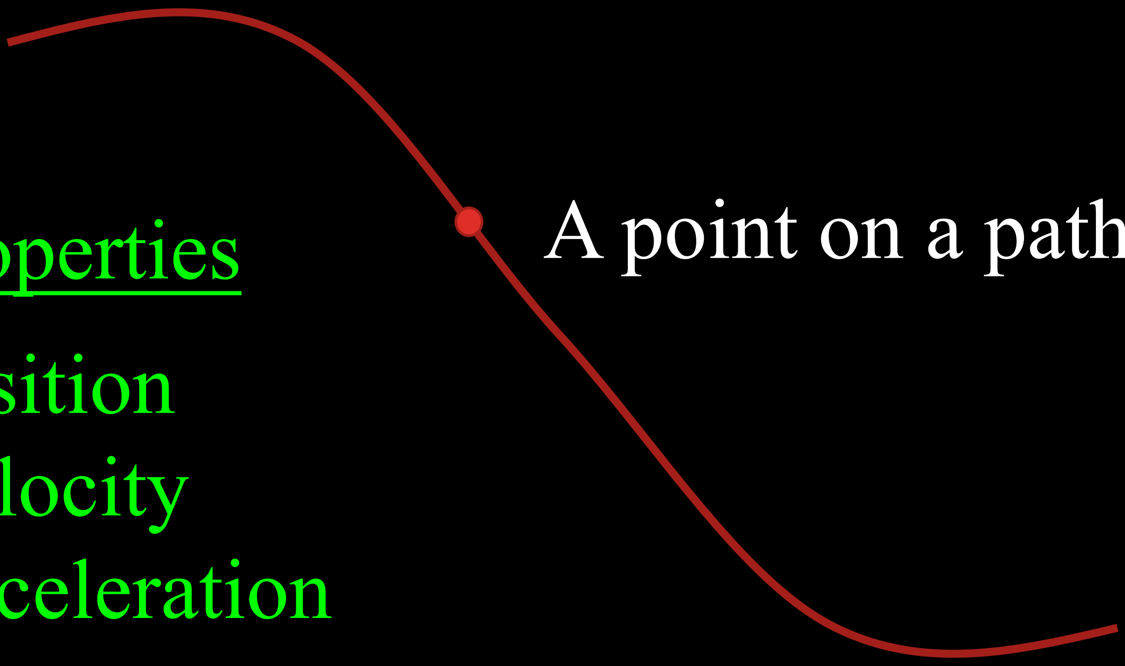
A POINT

Properties

Position

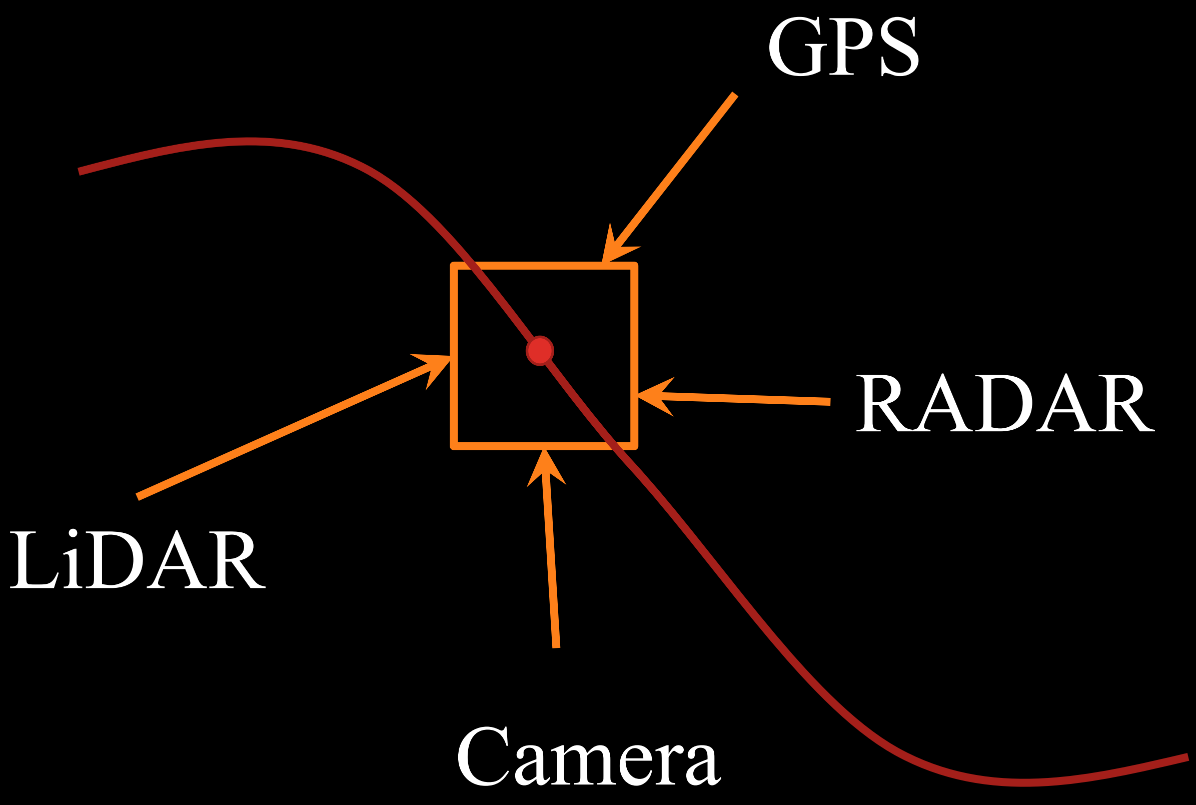
Velocity

Acceleration



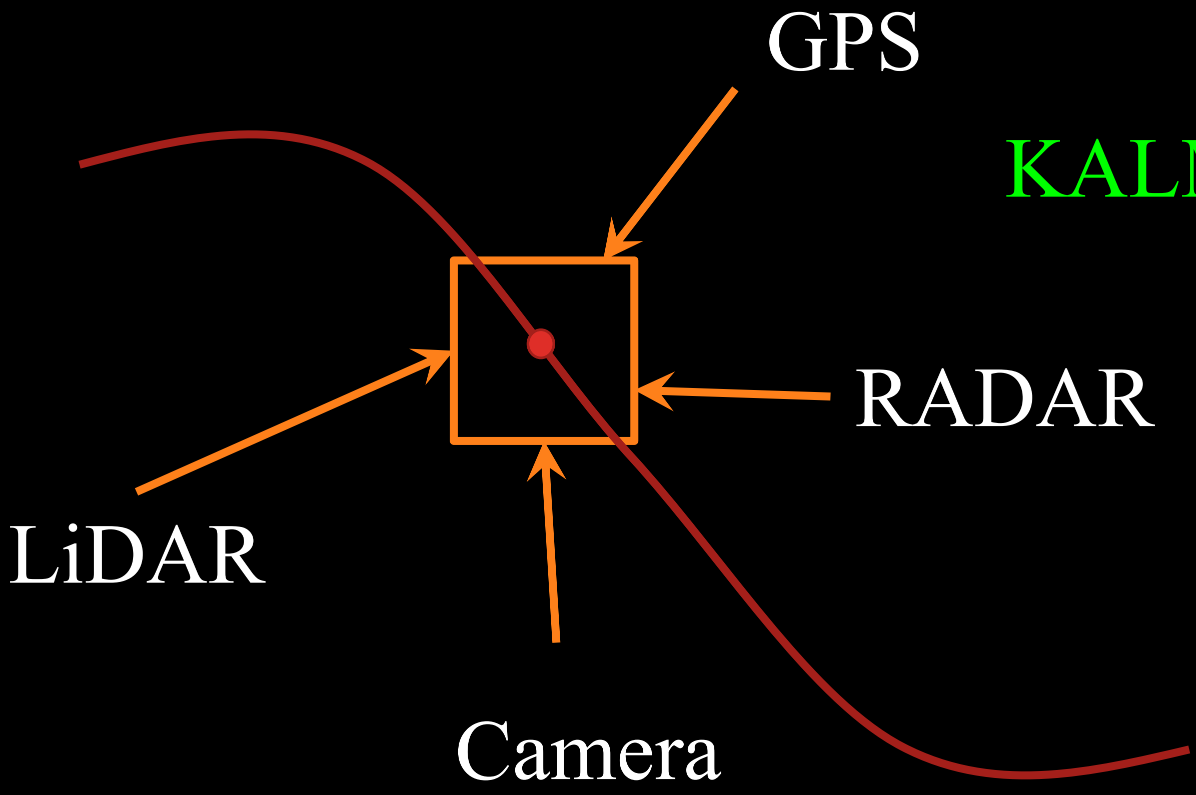
A point on a path

UPDATE FROM SENSORS



C4dynamics

DIFFERENT ALGORITHMS

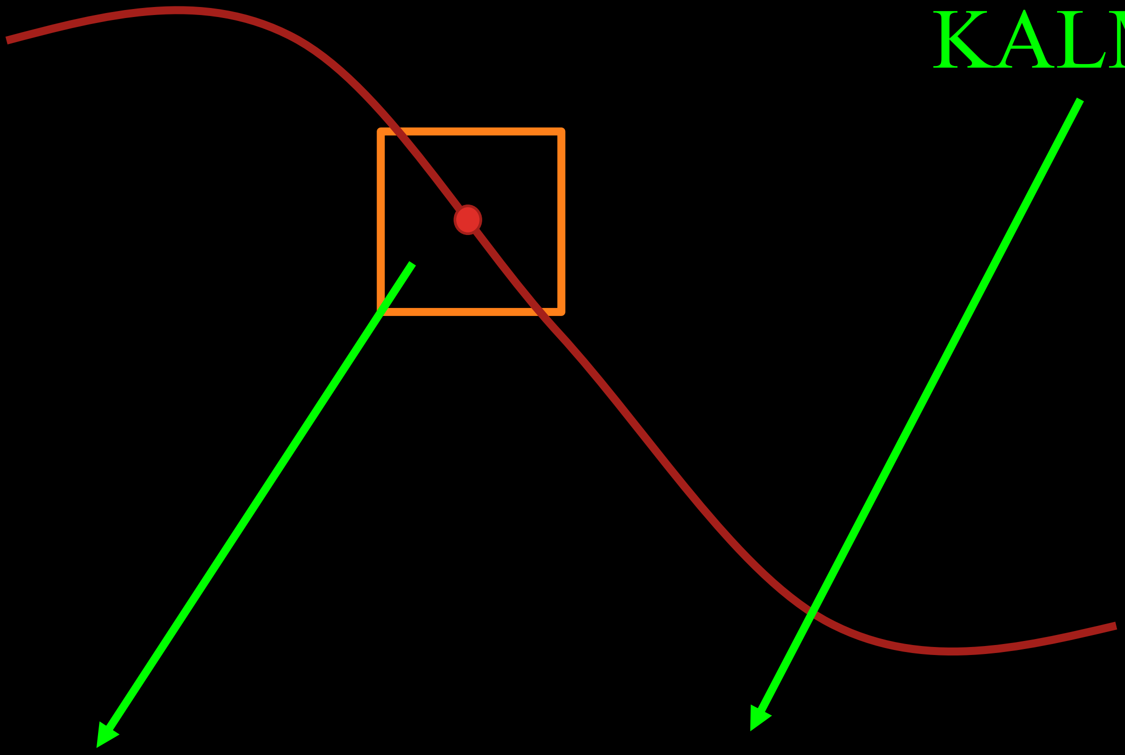


KALMAN

SLAM

POINT'S POINT OF VIEW

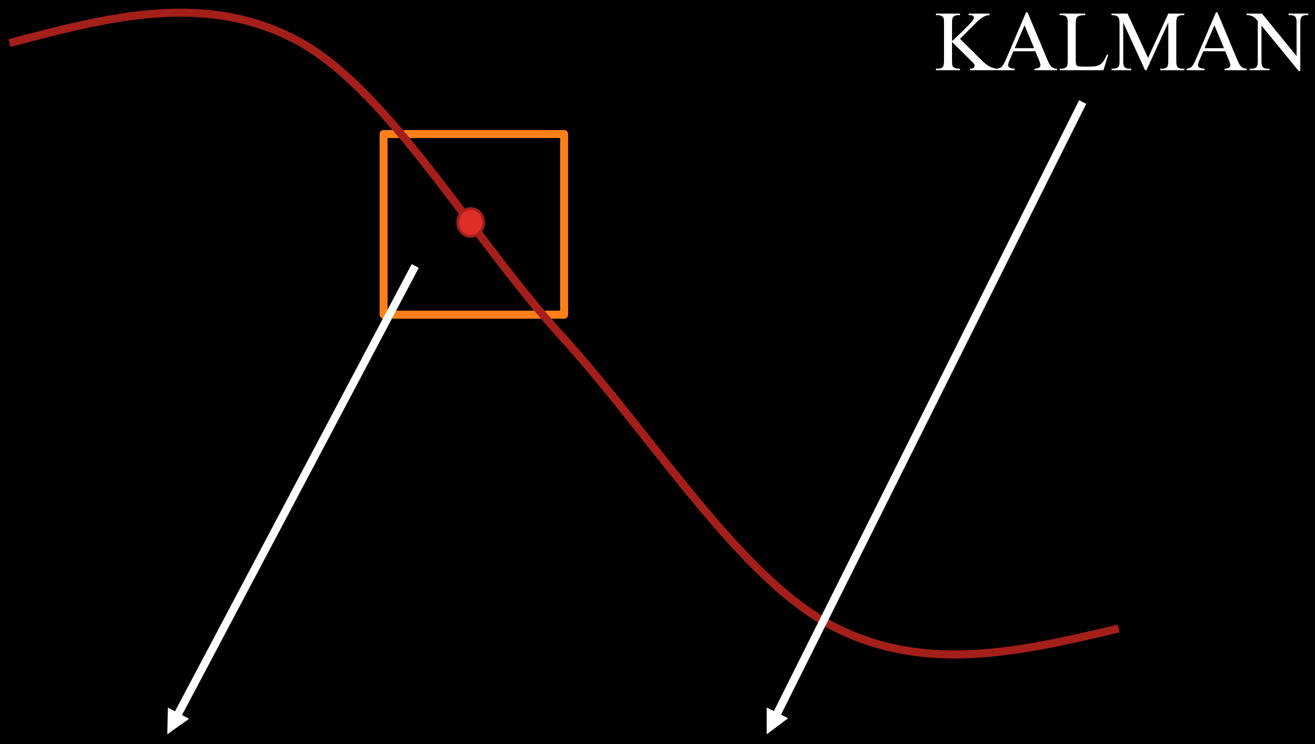
KALMAN



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

C4dynamics

POINT'S POINT OF VIEW



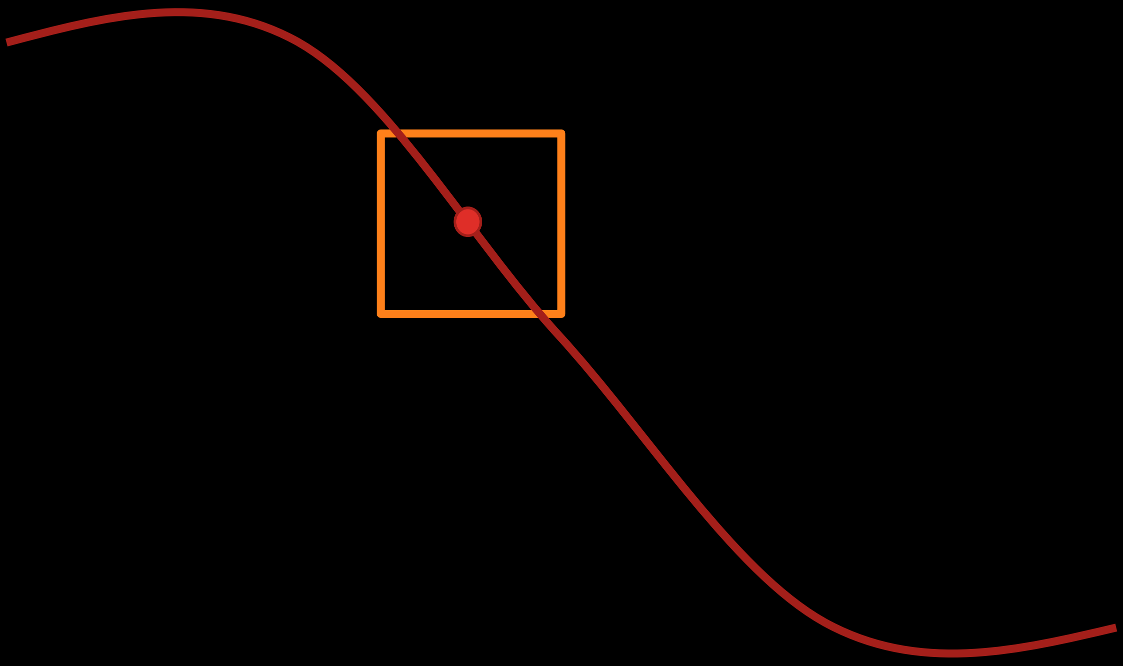
```
data_point.update(my_algo, new_data)
```

Bound together!

C4dynamics

So..

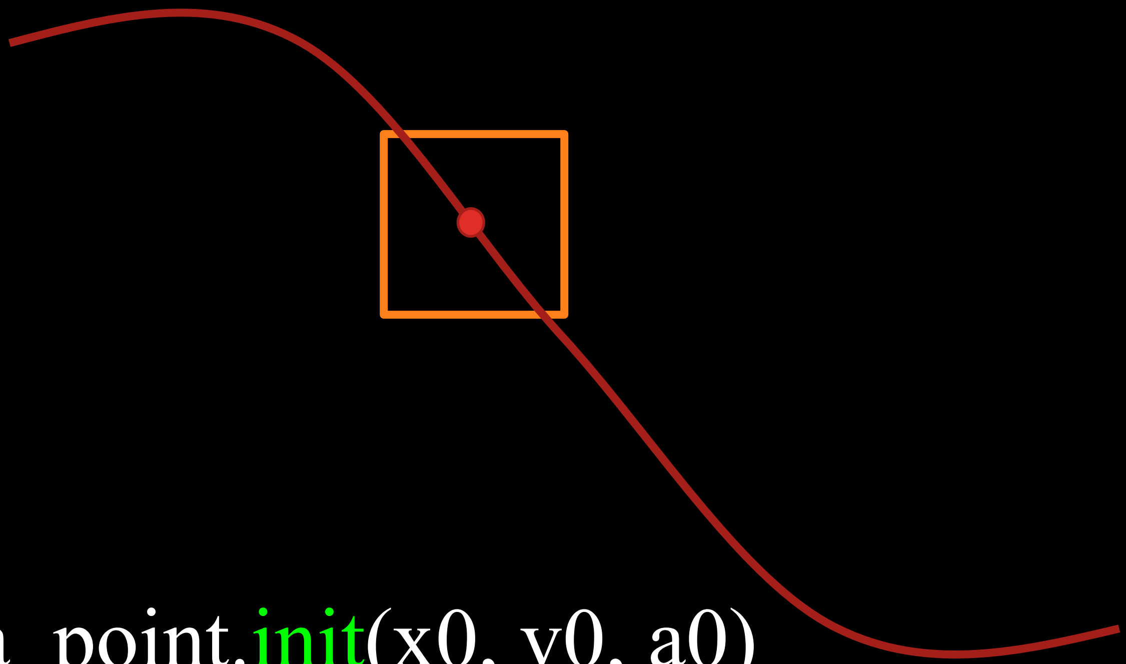
DO EVERYTHING FROM ONE PLACE!



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)
```

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

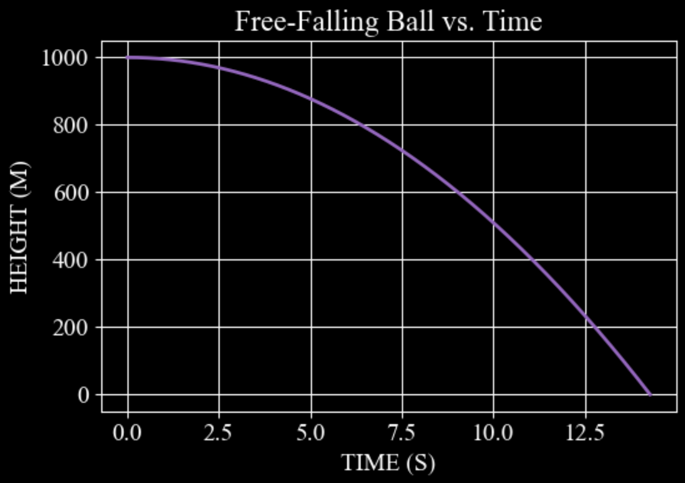
This is Algorithm Context!

Want to work with cool algorithm framework?

Download now C4dynamics and run freefall.py

Follow the instructions there:

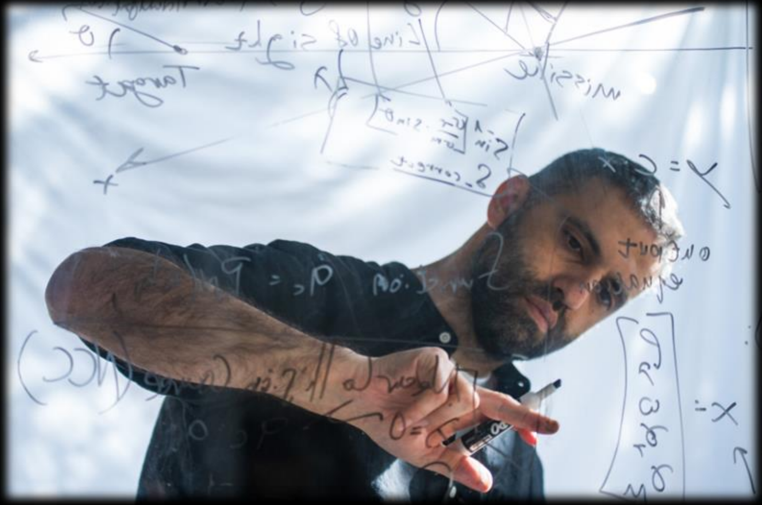
https://github.com/C4dynamics/C4dynamics/blob/main/examples/freefall.py

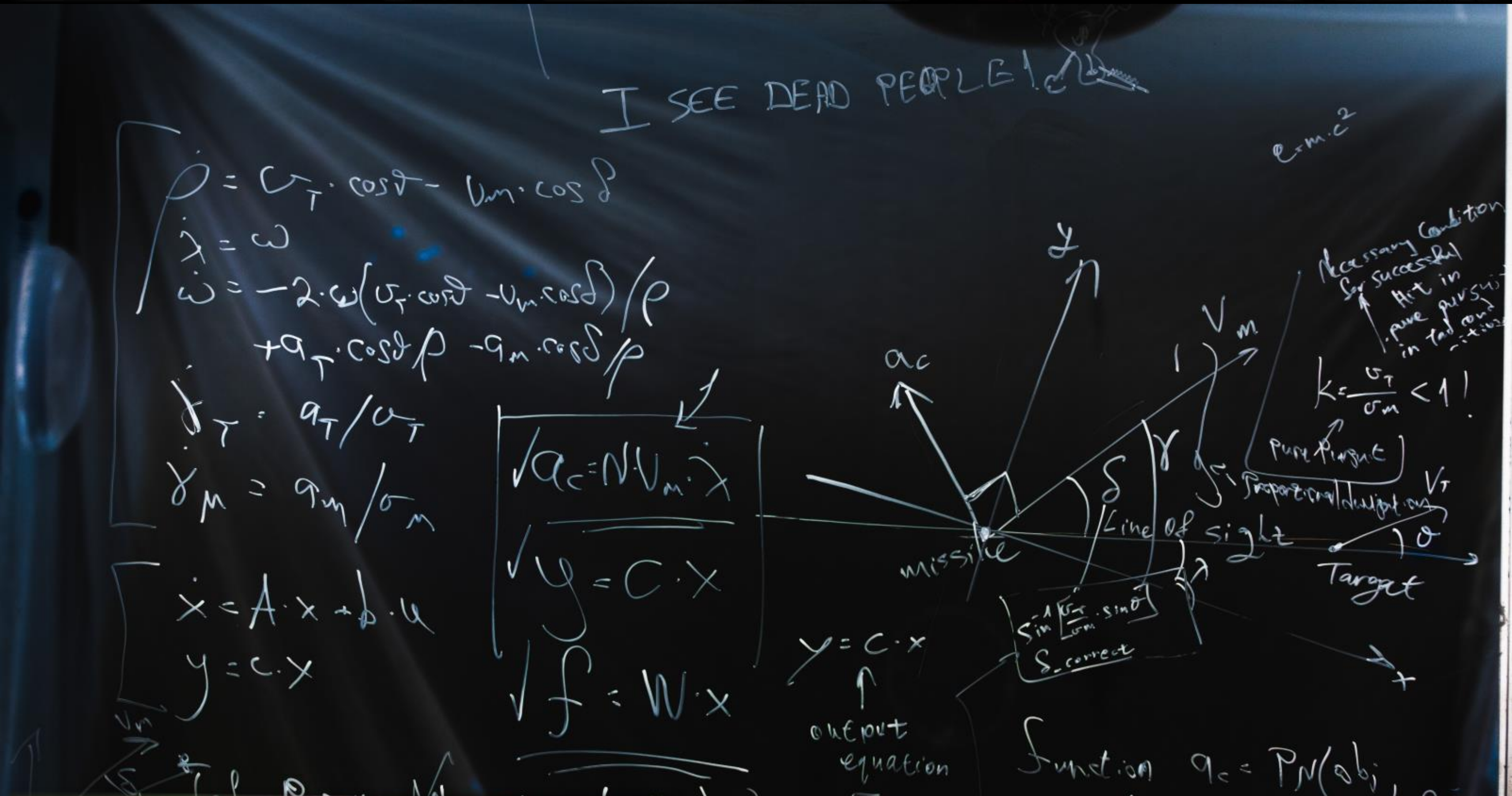


C4dynamics

A cutting-edge, high-standard algorithms development framework

C4dynamics




I SEE DEAD PEOPLE!
e=m·c²
$\dot{\rho} = v_T \cdot \cos\vartheta - v_m \cdot \cos\delta$
$\dot{\lambda} = \omega$
$\dot{\omega} = -2 \cdot \omega(v_T \cdot \cos\vartheta - v_m \cdot \cos\delta)/\rho + a_T \cdot \cos\vartheta/\rho - a_m \cdot \cos\delta/\rho$
$\dot{\gamma}_T = a_T / v_T$
$\dot{\gamma}_M = a_m / v_m$
$\dot{x} = A \cdot x + b \cdot u$
$y = C \cdot x$
$a_c = N \cdot V_m \cdot \dot{\lambda}$
$y = C \cdot x$
$f = W \cdot x$
a_c
y
V_m
δ
γ
V_T
ϑ
Line of sight
missile
Target
Necessary Condition for successful Hit in pure pursuit in tail conditions
$k = \frac{v_T}{v_m} < 1$!
Pure Pursuit
Proportional Navigation
$\sin^{-1}\left(\frac{v_T}{v_m} \cdot \sin\vartheta\right)$
δ_correct
$y = C \cdot x$
output equation
function a_c = PN(obj,

Gavriel Weinberger

C4dynamics