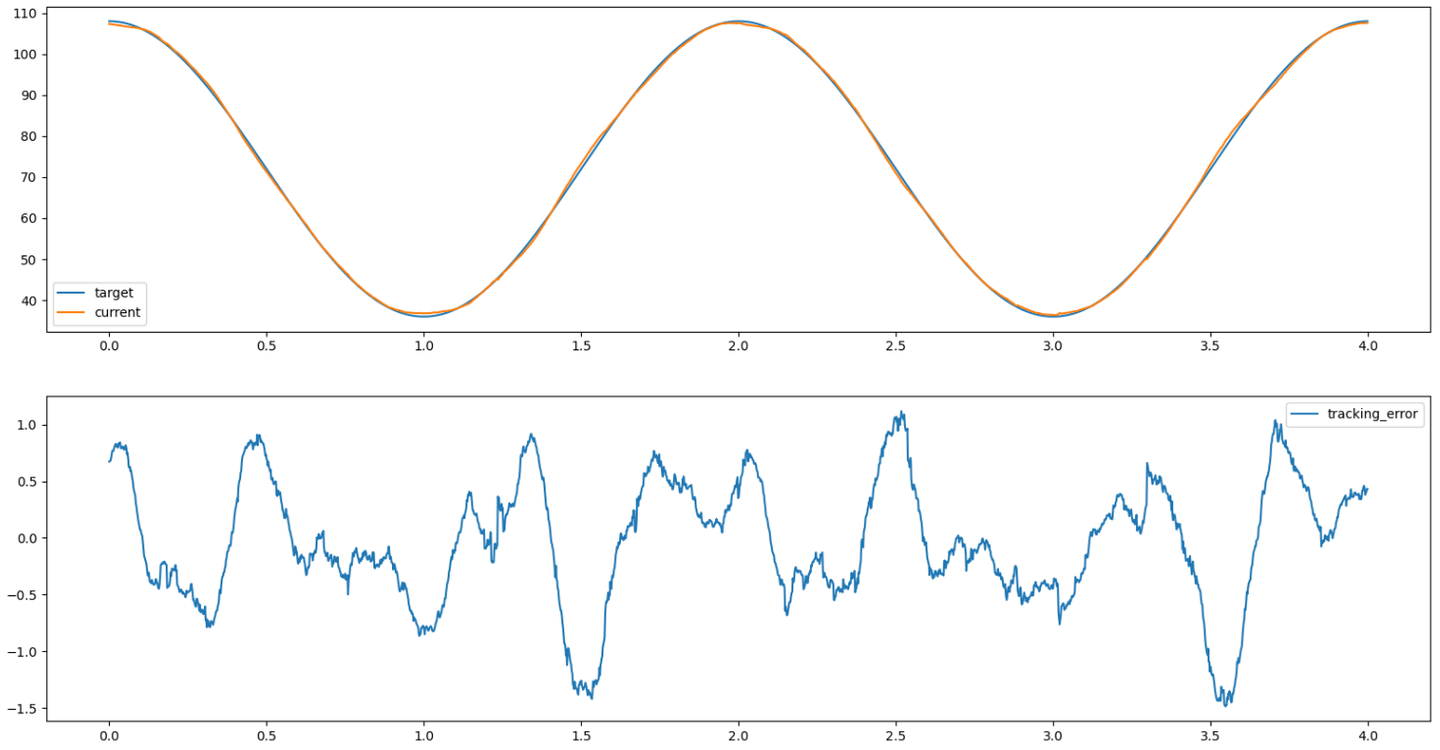
SERVO INTERFACE SEMANTICS, TAYLOR EXPANSIONS IN INVERSE KINEMATICS SOLVING

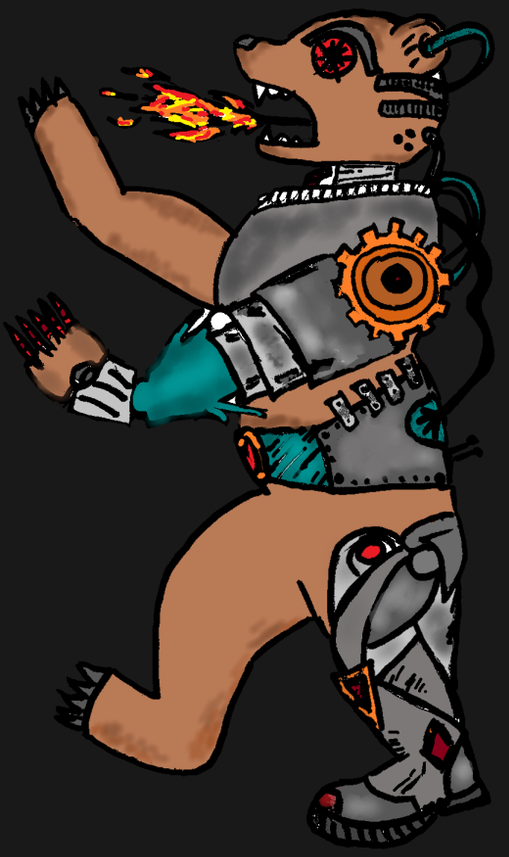
and why you should care

THE GOAL



THE WHO

01.RFC BERLIN



THE WHY

- Need for cheap servos
- while at it, why not make them nice?

THE WHAT

SERVO INTERFACE SEMANTICS

It's all about how we communicate desired motion.

A servo's interface implies it's functionality

EXAMPLE:

Interface semantics of classic actuators:

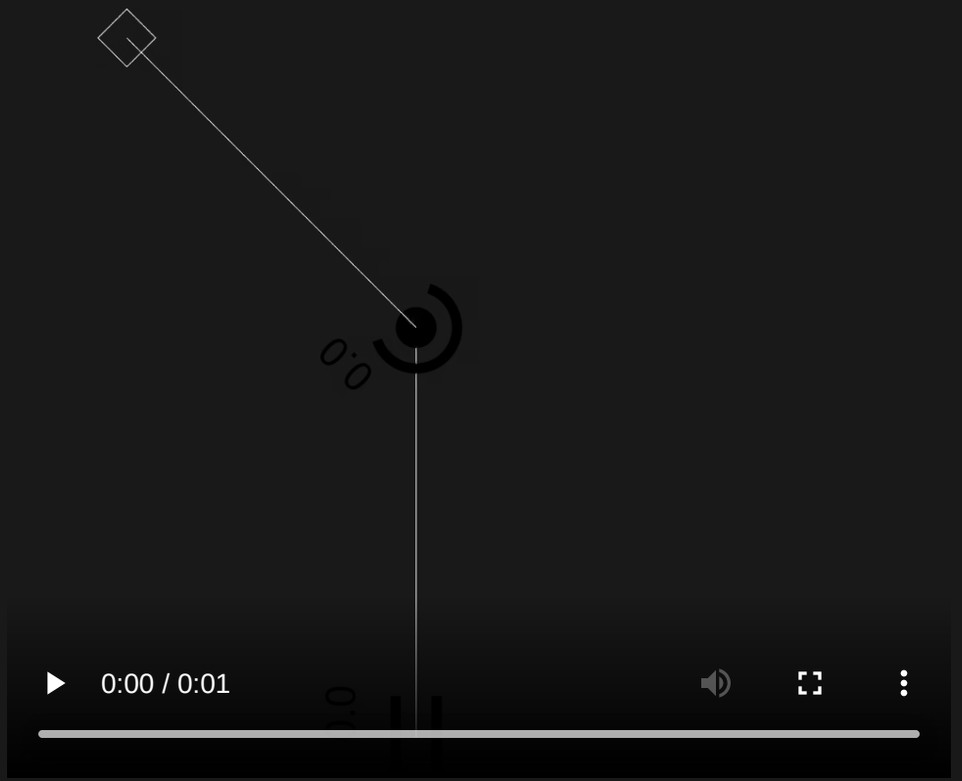
Tell me where to go and how fast to go

THE PROBLEM

The servo's goal position is the goal of the current trajectory

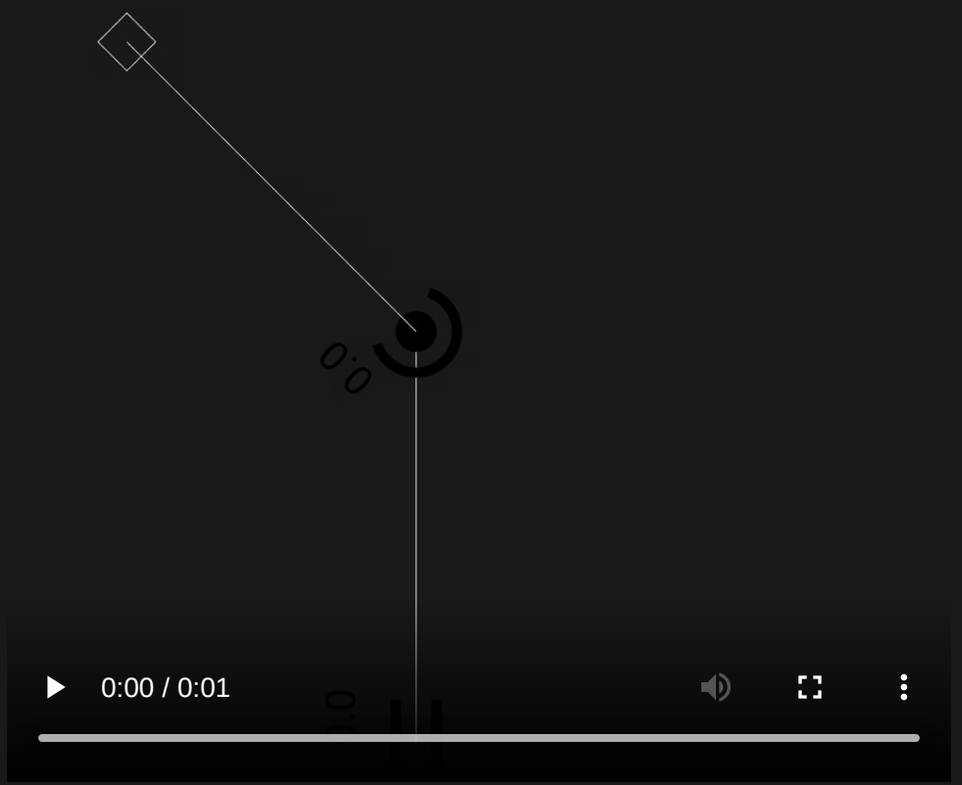
If the trajectory contains a direction change the semantics are contradicting

WHAT WE WANT:



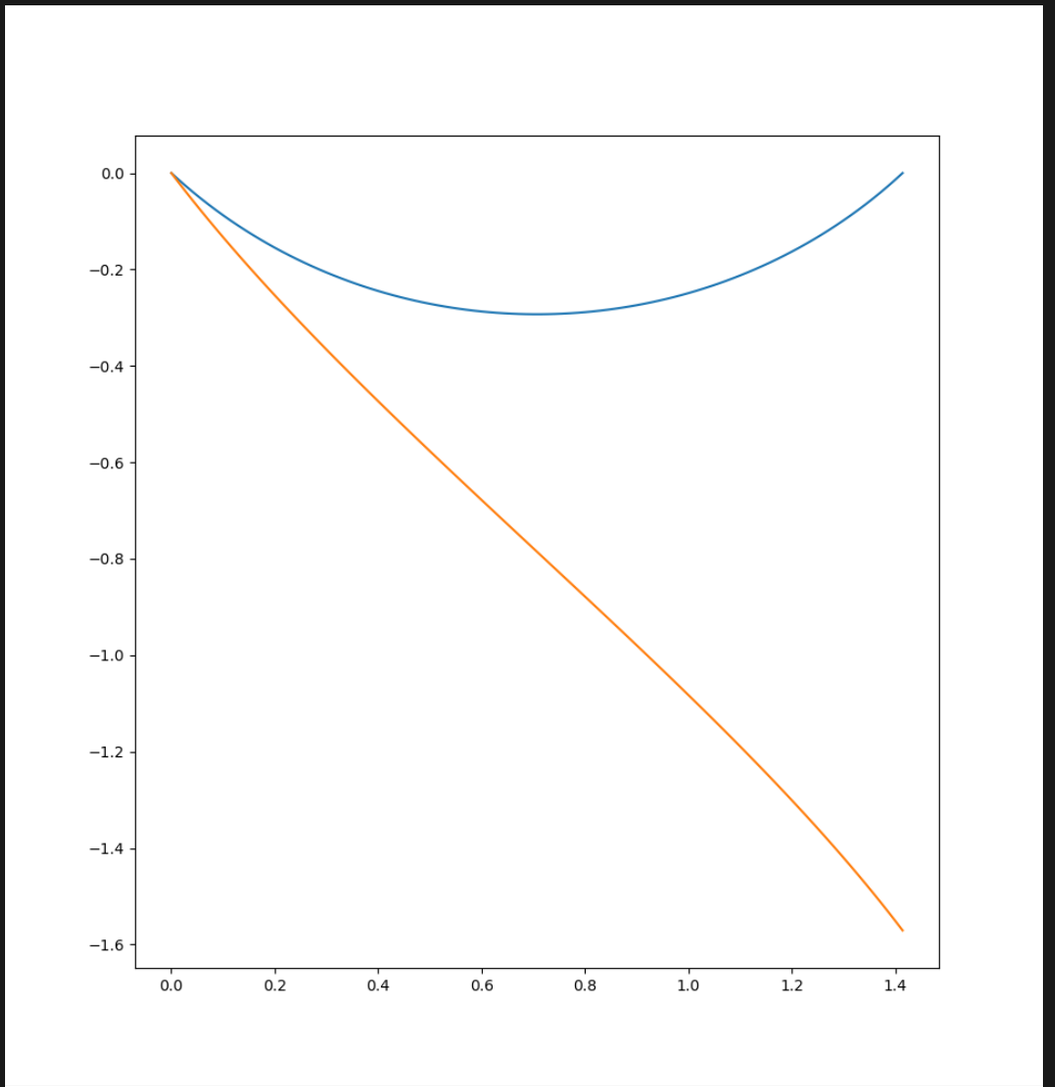
the robot's finger moves along a line

WHAT WE HAVE:



the robot's finger doesn't move along a line

Duh



THE SOLUTION

We can change the interface semantics:

Let a polynomial (of degree N) with respect to time describe the servos trajectory!

$$g(t) = \sum_{i=0}^{N} p_i t^i$$

Neat side effect:

The polynomial integrates super nicely into the servo's PID controller

this:

$$u(t) = K_i * \int_0^t e(t)dt + K_p * e(t) + K_d * \frac{de(t)}{dt}$$

becomes this:

$$u(t) = K_i * \int_0^t e(t)dt + \sum_i K_i * (\frac{d^i g(t)}{dt^i} - \frac{d^i c(t)}{dt^i})$$

Another neat side effect:

The servo can incorporate transmission delays t_d:

$$g(t) = \sum_{i=0}^{N} p_i (t + t_d)^i$$

THE IMPLICATIONS

Now we need to generate polynomials to express motions

This isn't all too hard since we have inverse kinematics

Whoopwhoop

INVERSE KINEMATICS REVISITED

IN A NUTSHELL

IK calculates how a posture should be changed to fulfill tasks

TASK:

- Describes what and how to move
- Expresses motion in task space

$\Theta(t)$	target function
$\psi(q)$	value function
$e(q,t) = \Theta(t) - \psi(q)$	error function

EXAMPLE:

"Move the finger along a line expressed in the coordinate frame of the torso"

$\Theta(t)$: the target position of the finger along the line (in torso coordinates)

$\psi(q)$: the current position of the finger (in torso coordinates)

DERIVATION (THE IDEA):

Find the change in posture Δq which minimizes the task's error function $e(q + \Delta q, t)$

$$\mathcal{L}(\Delta q) = ||e(q + \Delta q, t)||_C^2 + ||\Delta q||_W^2$$
$$\mathcal{L}(\Delta q) = ||\Theta(t) - \psi(q + \Delta q)||_C^2 + ||\Delta q||_W^2$$

DERIVATION (THE ASSUMPTION):

For small changes in the posture $\Delta q \rightarrow 0$ we assume ψ to be linear around q

$$\lim_{\Delta q \to 0} \psi(q + \Delta q) = \psi(q) + J\Delta q$$

$$J = \frac{\delta}{\delta q}\psi(q) = \begin{pmatrix} \frac{\delta}{\delta q_1}\psi(q)_1 & \frac{\delta}{\delta q_2}\psi(q)_1 & \cdots & \frac{\delta}{\delta q_n}\psi(q)_1 \\ \frac{\delta}{\delta q_1}\psi(q)_2 & \frac{\delta}{\delta q_2}\psi(q)_2 & \cdots & \frac{\delta}{\delta q_n}\psi(q)_2 \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\delta}{\delta q_1}\psi(q)_d & \frac{\delta}{\delta q_2}\psi(q)_d & \cdots & \frac{\delta}{\delta q_n}\psi(q)_d \end{pmatrix}$$

DERIVATION (THE ASSUMPTION):

Our loss function now became:

$$\mathcal{L}(\Delta q) = ||\Theta(t) - \psi(q) - J\Delta q||_C^2 + ||\Delta q||_W^2$$
$$\mathcal{L}(\Delta q) = ||e(q,t) - J\Delta q||_C^2 + ||\Delta q||_W^2$$

DERIVATION (THE WORK):

$$
\begin{aligned}
\frac{\delta}{\delta \Delta q}\mathcal{L}(\Delta q) = 0 &= \frac{\delta}{\delta \Delta q}\left[||e(q,t) - J\Delta q||_C^2 + ||\Delta q||_W^2\right] \\
&= \frac{\delta}{\delta \Delta q}\left[(e(q,t) - J\Delta q)^T C(e(q,t) - J\Delta) + \Delta q^T W \Delta q\right] \\
&= \frac{\delta}{\delta \Delta q}\left[(e(q,t)^T - \Delta q^T J^T) C(e(q,t) - J\Delta q) + \Delta q^T W \Delta q\right] \\
&= \frac{\delta}{\delta \Delta q}\left[e(q,t)^T C e(q,t) - e(q,t)^T C J \Delta q - \Delta q^T J^T C e(q,t) + \Delta q^T J^T C J \Delta q + \Delta q^T W \Delta q\right] \\
&= \frac{\delta}{\delta \Delta q}\left[e(q,t)^T C e(q,t) - 2e(q,t)^T C J \Delta q + \Delta q^T J^T C J \Delta q + \Delta q^T W \Delta q\right] \\
&= 2J^T C^T J \Delta q - 2J^T C^T e(q,t) + 2W^T \Delta q \\
&= J^T C^T J \Delta q - J^T C^T e(q,t) + W^T \Delta q \\
&= (J^T C^T J + W^T)\Delta q - J^T C^T e(q,t) \\
J^T C^T e(q,t) &= (J^T C^T J + W^T)\Delta q \\
\Delta q &= (J^T C^T J + W^T)^{-1} J^T C^T e(q,t) \\
\Delta q &= W^{T^{-1}} J^T (J W^{T^{-1}} J^T + C^{T^{-1}})^{-1} e(q,t)
\end{aligned}
$$

DERIVATION (THE RESULT):

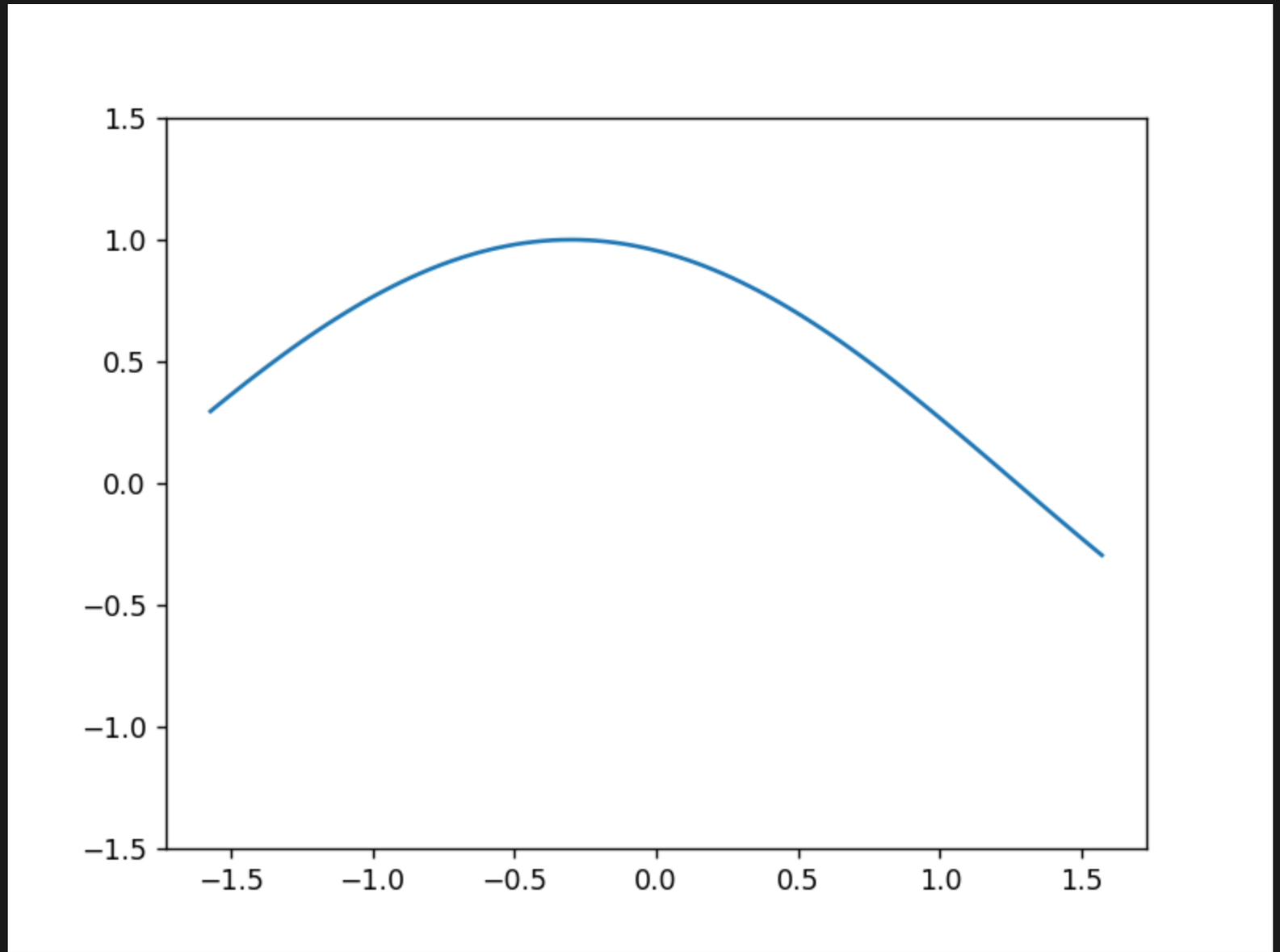
The Δq that minimizes $\mathcal{L}$ can be calculated like so:

$$\Delta q = W^{T^{-1}} J^T (J W^{T^{-1}} J^T + C^{T^{-1}})^{-1} e(q,t)$$

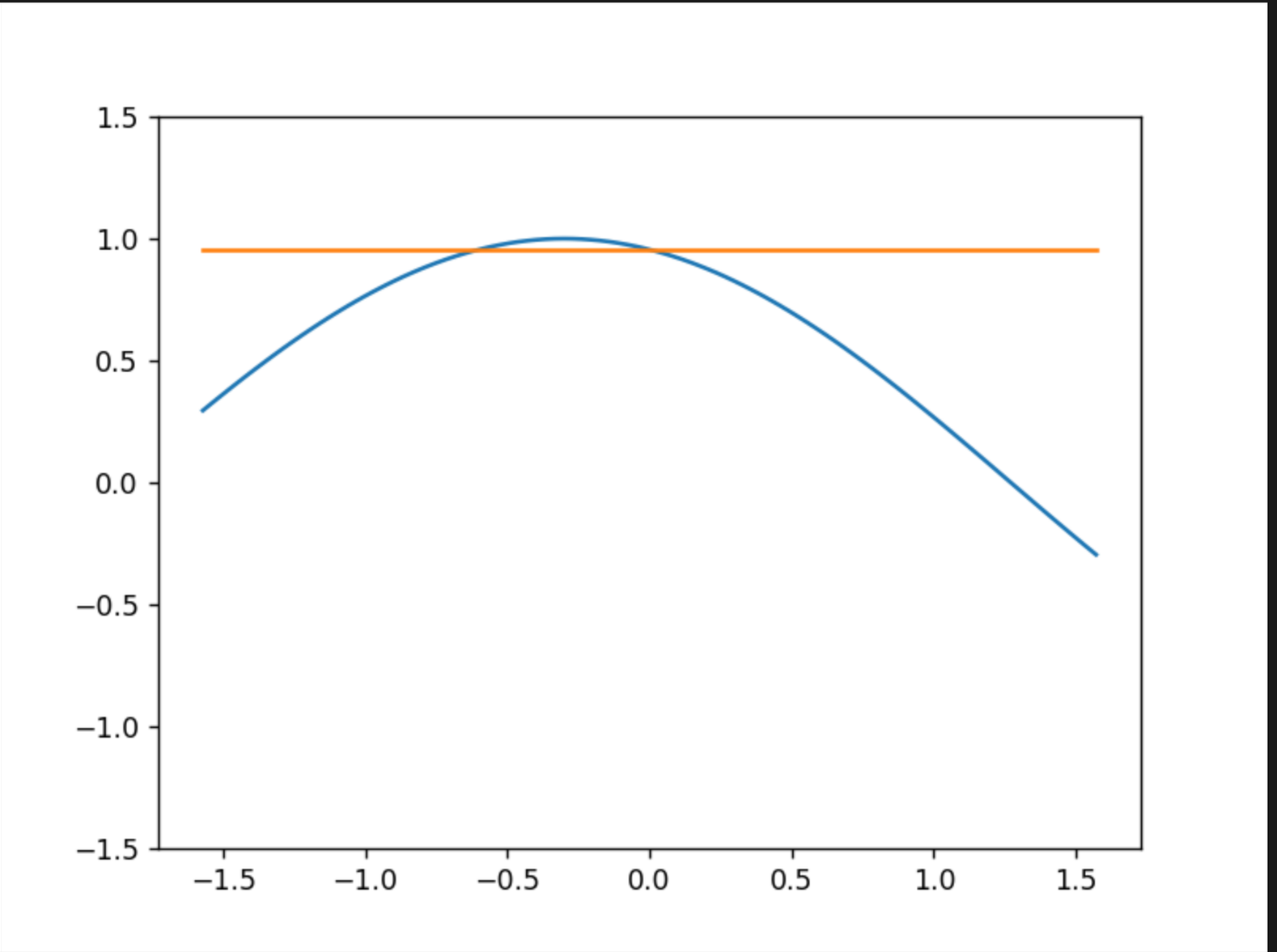
Shorthand (with $W = I$ and $C \to \infty$):

$$\Delta q = J^\dagger e(q,t)$$

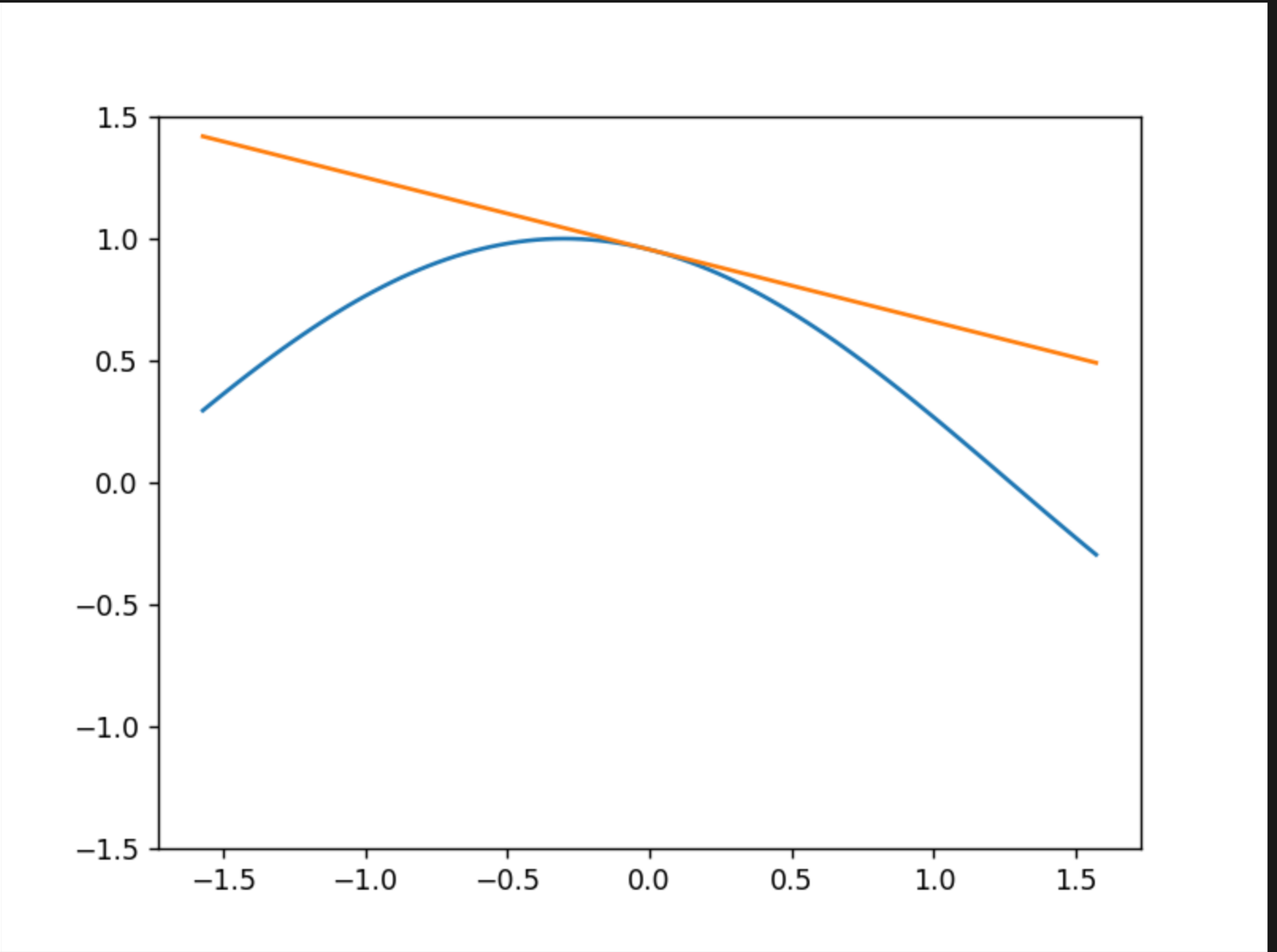
TAYLOR EXPANSIONS:



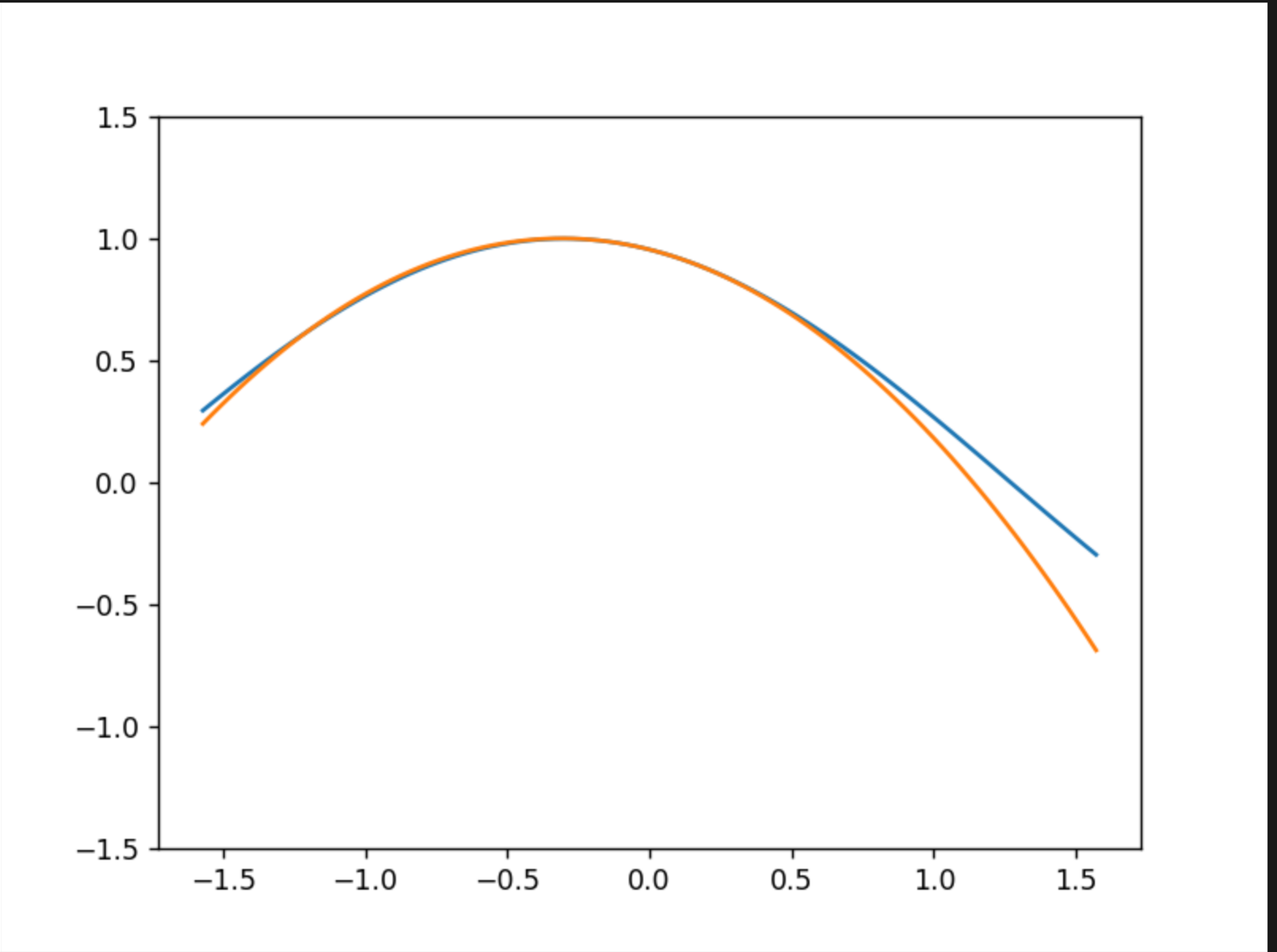
some function



some function with a constant approximation



some function with a linear approximation



some function with a square approximation

PUTTING IT ALL TOGETHER

THIS IS WHAT WE WANT:

$$g(t) = \sum_{i=0}^{N} p_i t^i$$

BROOK TAYLOR TO THE RESCUE:

$$g(t) = \sum_{i=0}^{N} p_i t^i$$

$$p_i = \frac{1}{i!} \frac{d^i \Delta q}{dt^i}$$

BROOK TAYLOR TO THE RESCUE:

$$p_i = \frac{1}{i!}\frac{d^i \Delta q}{dt^i}$$

$$p_i = \frac{1}{i!}\frac{d^i}{dt^i}(J^\dagger e(q,t))$$

$$p_i = \frac{1}{i!}\frac{d^i}{dt^i}(J^\dagger(\Theta(t) - \psi(q)))$$

$$p_i = \frac{1}{i!}J^\dagger(\frac{d^i}{dt^i}\Theta(t) - \frac{d^i}{dt^i}\psi(q))$$

THE MOST IMPORTANT EQUATIONS IN THIS PRESENTATION:

$$g(t) = \sum_{i=0}^{N} p_i t^i$$

$$p_i = \frac{1}{i!} J^\dagger \left(\frac{d^i}{dt^i} \Theta(t) - \frac{d^i}{dt^i} \psi(q) \right)$$

WHAT DOES THAT MEAN?

$\frac{d^i}{dt^i}\Theta(t)$ the i-th derivative of the task trajectory

$\frac{d^i}{dt^i}\psi(q)$ the i-th derivative of the "current value function".
I.e., how the system continues to move.

WHAT THE $\frac{d^i}{dt^i}\psi(q)$?

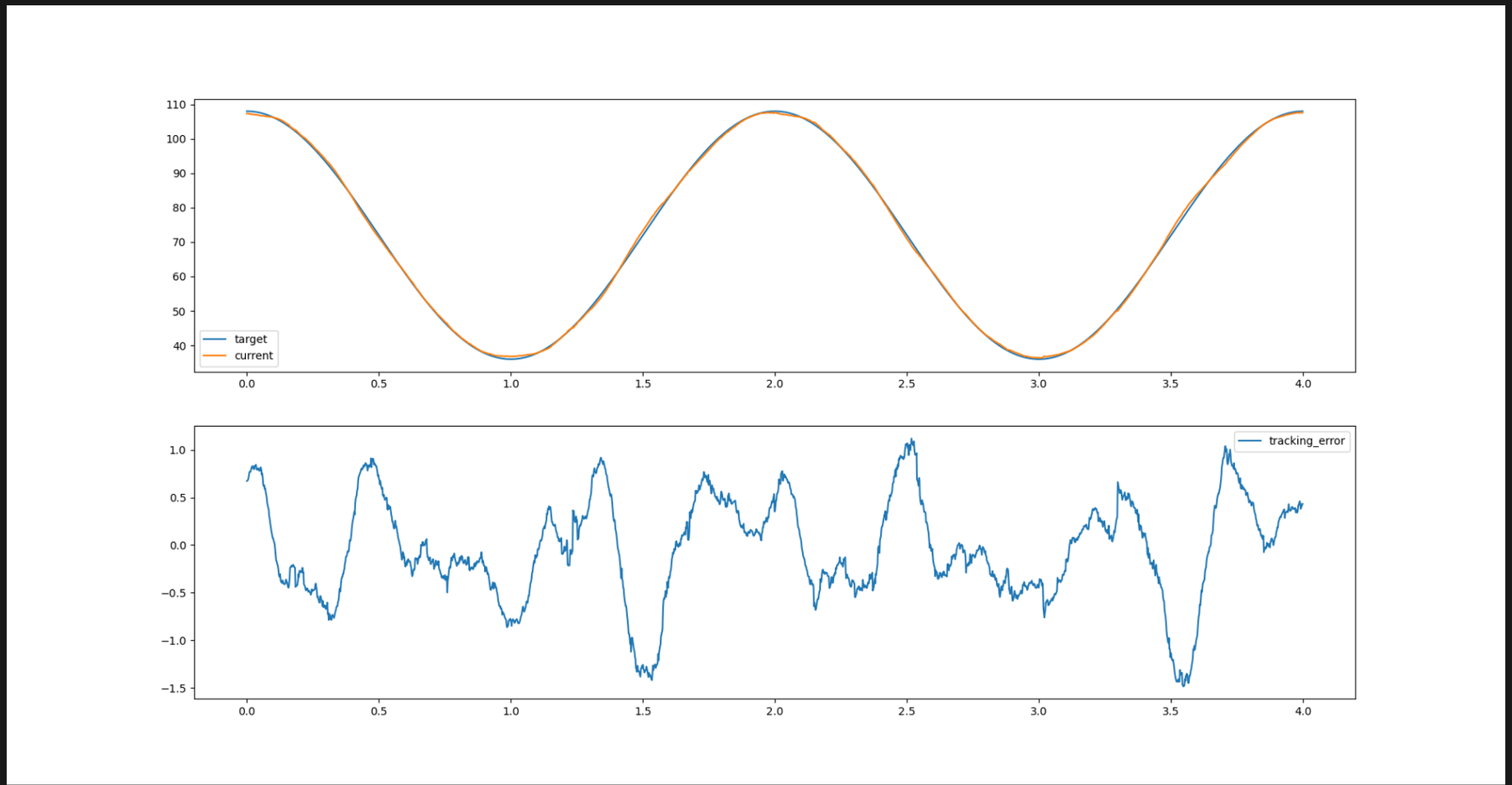
From the second derivative and onward of $\frac{d^i}{dt^i}\psi(q)$ tend to be nontrivial!

Depending on the choice of the task's reference frame and the types of joints you'd need to incorporate centripetal, Euler and Coriolis accelerations

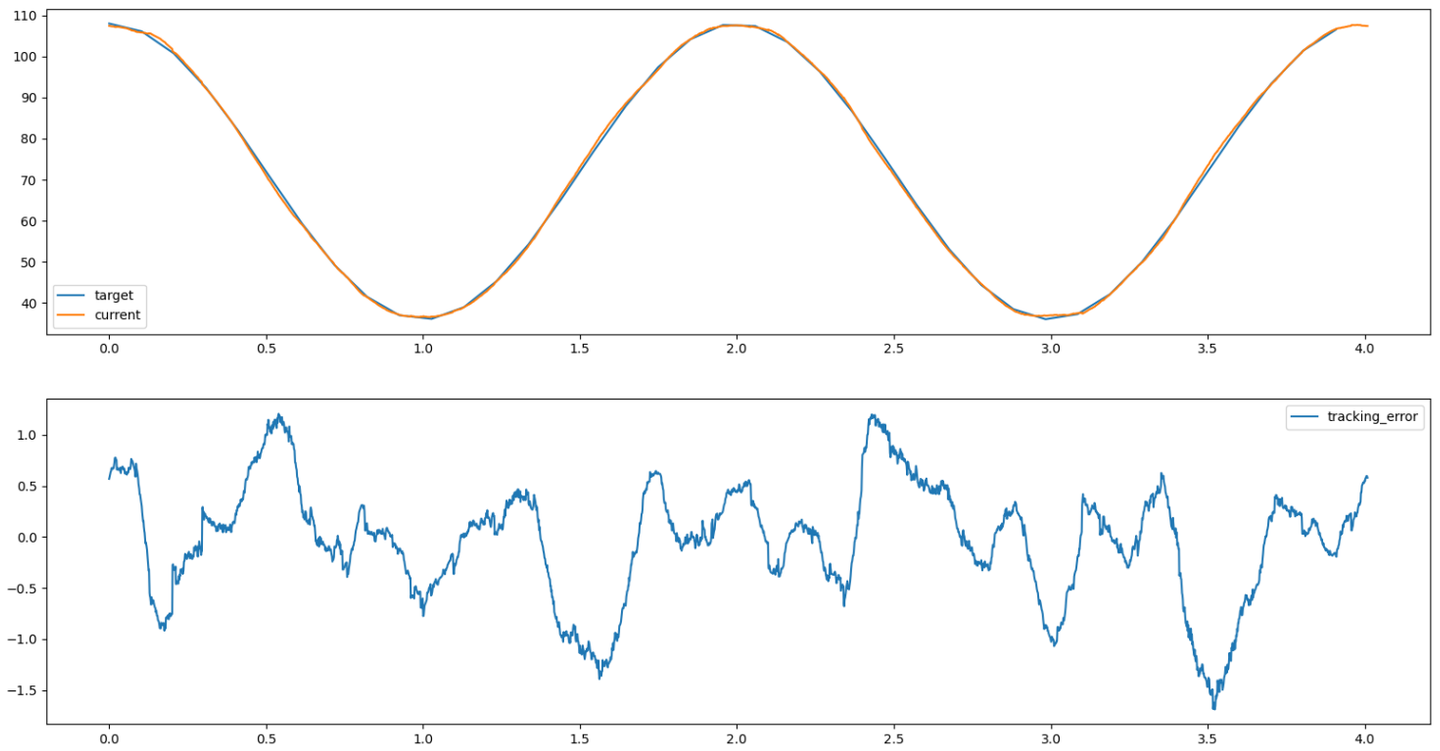
WHY YOU SHOULD CARE

- safely reduce the motion execution frequency
- better tracking performance of the actuators
- easily extendable for different actuator types (e.g., speed driven actuators)
- get rid of the hack to find a suitable "target position" for classic actuators

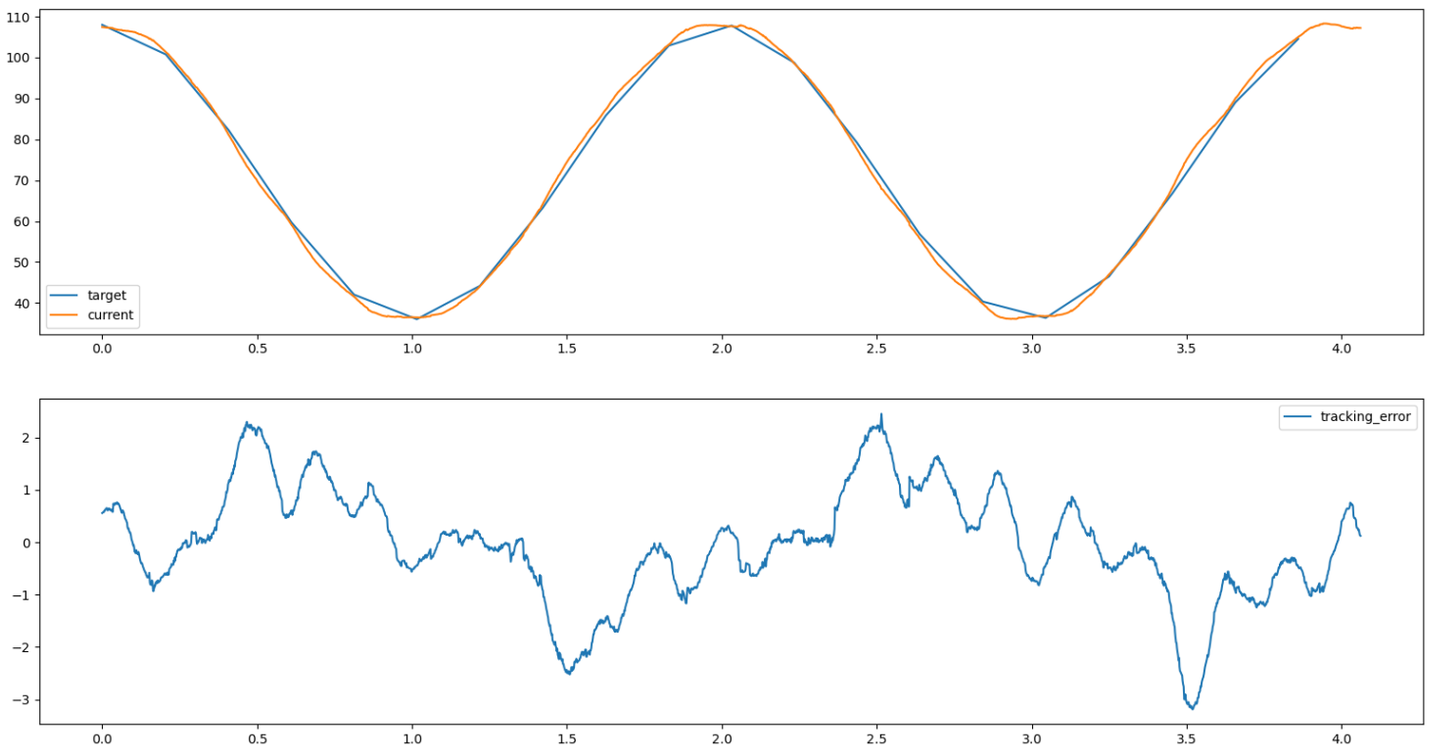
TRACKING PERFORMANCE:



update rate 500Hz

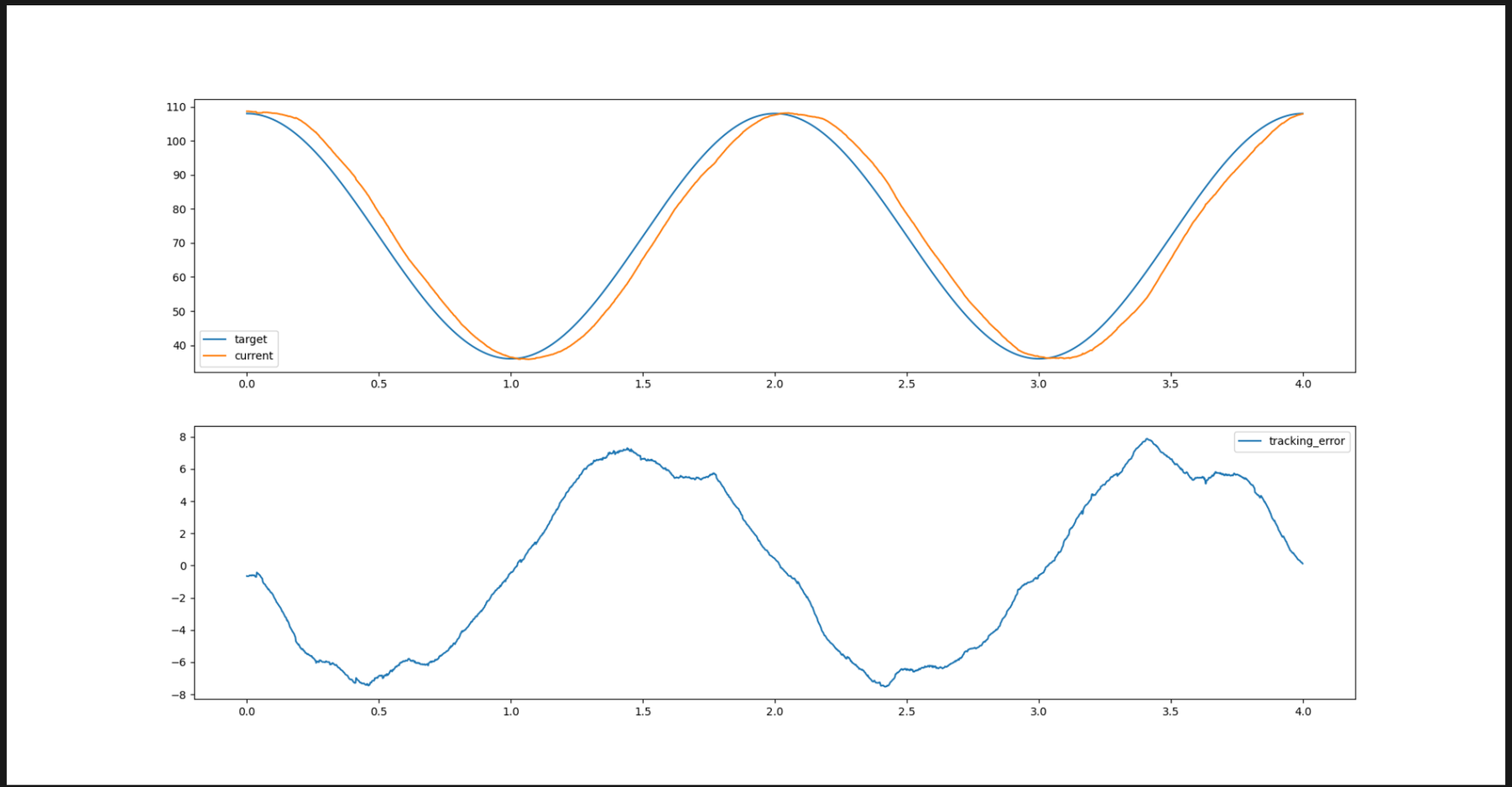


update rate 10Hz

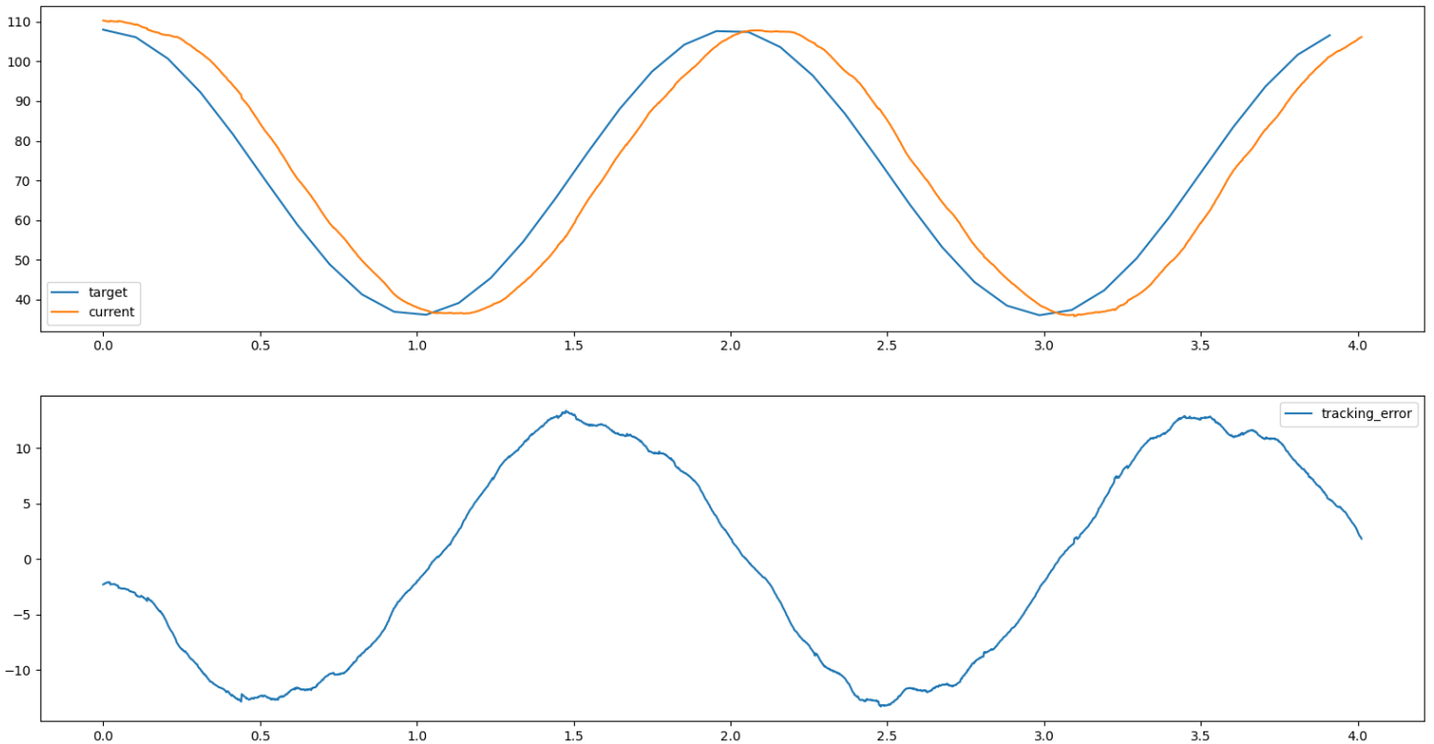


update rate 5Hz

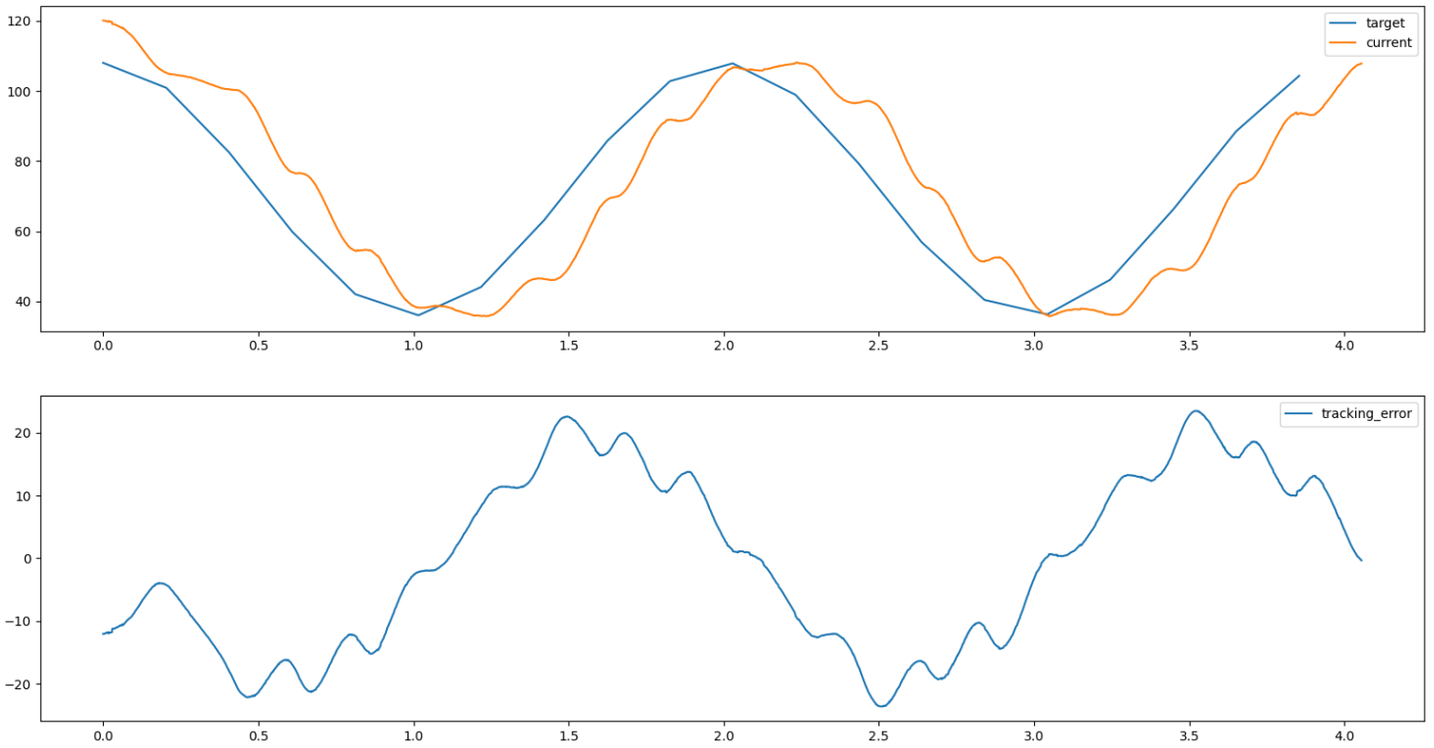
TRACKING PERFORMANCE (NO ACC AND VEL):



update rate 500Hz



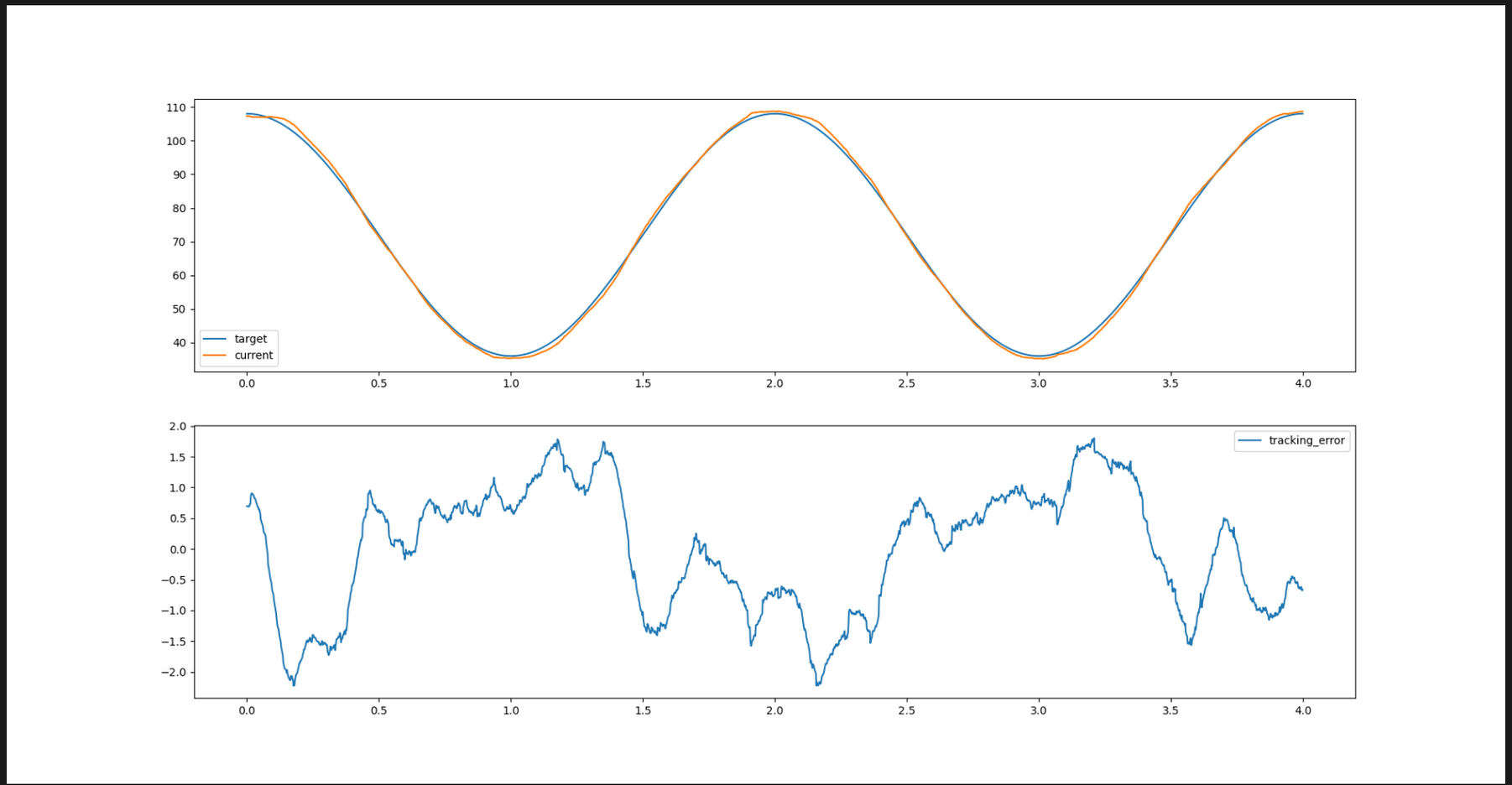
update rate 10Hz



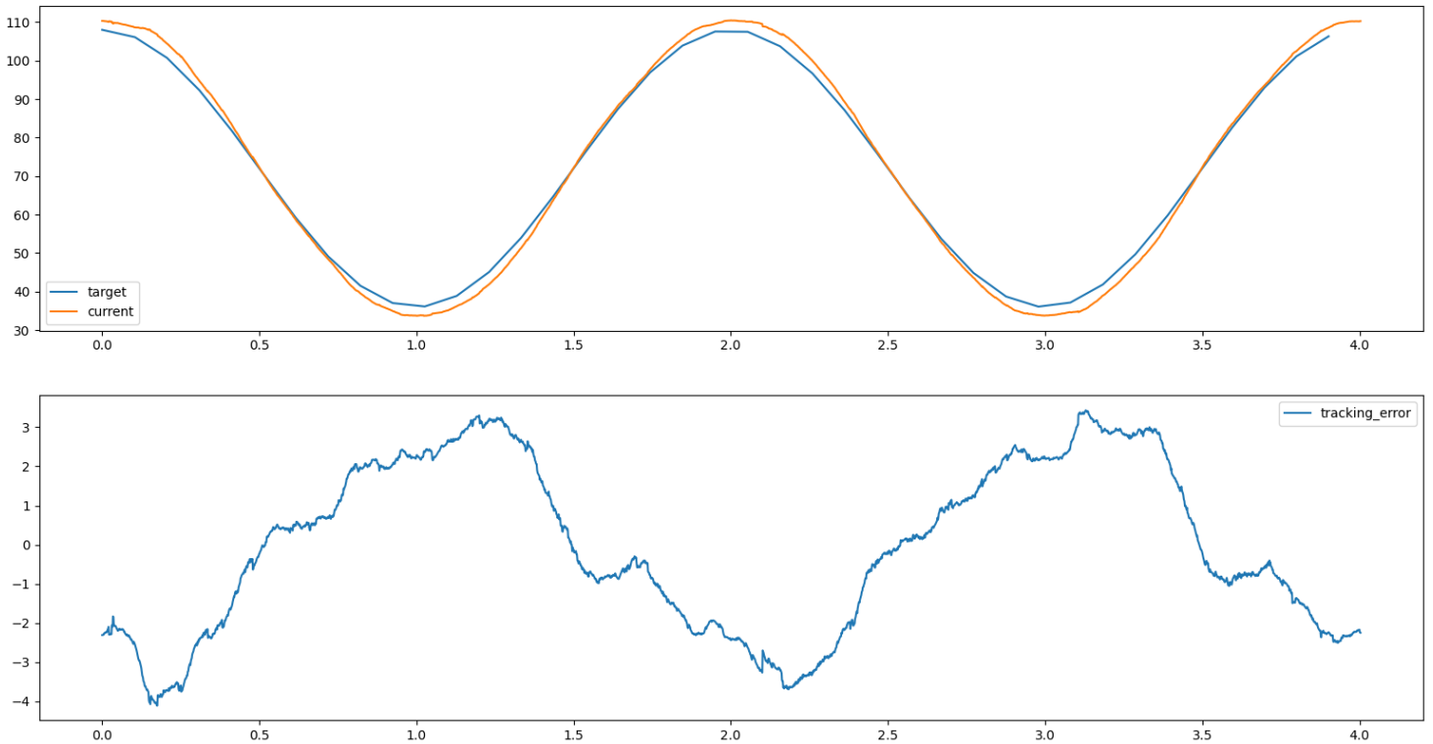
update rate 5Hz

DEMO

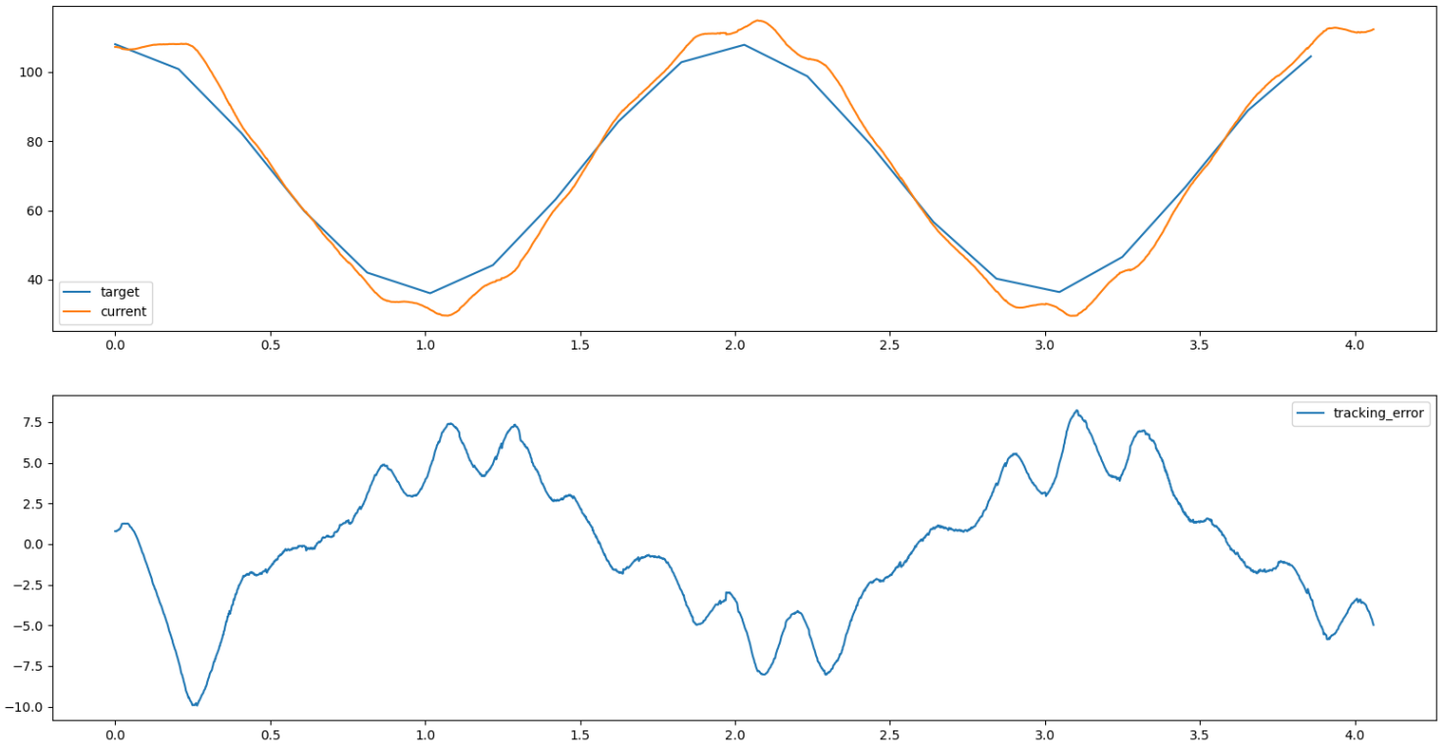
TRACKING PERFORMANCE (NO ACC):



update rate 500Hz



update rate 10Hz



update rate 5Hz