Deep Learning for LIGO’s Lock Acquisition

Machine Learning for non-linear dynamic controls.

Peter Ma | Supervisor: Dr. Gabriele Vajente

Caltech LIGO UNIVERSITY OF TORONTO

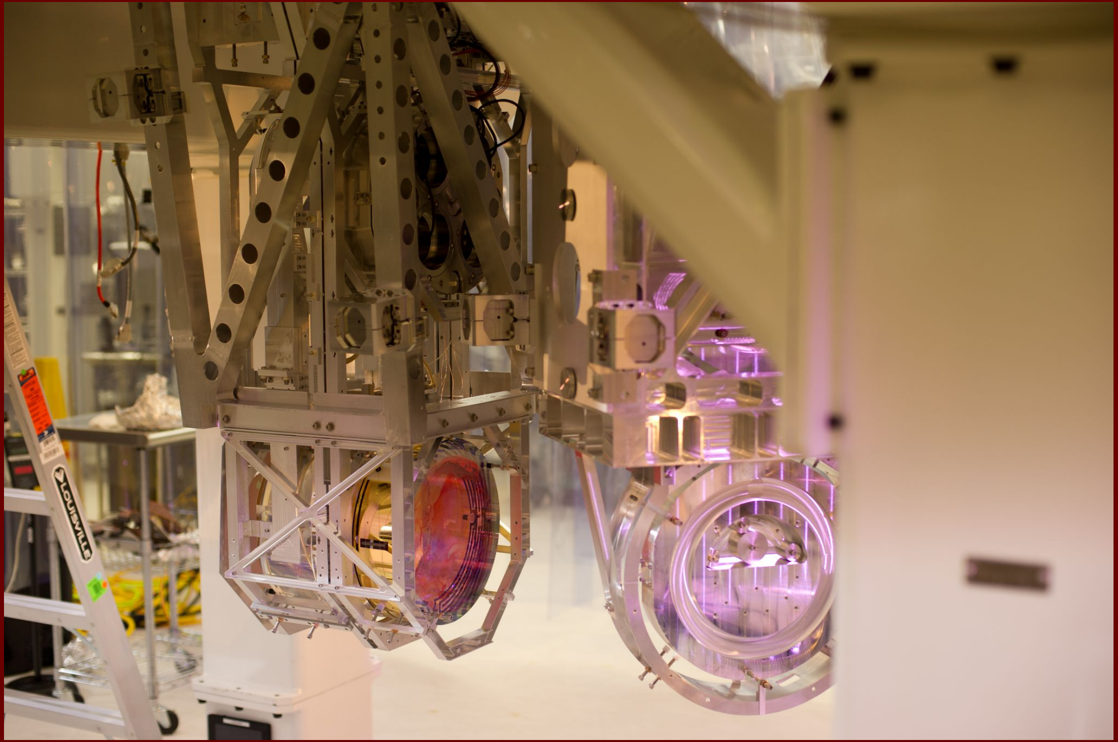
Introduction

What are we controlling?

Goal is to put the detector in a “neutral state”.

Goal is to put the detector in a “neutral state”. We want to be able to control or ***“acquire the lock”*** for these mirrors.

Goal is to put the detector in a “neutral state”. We want to be able to control or ***“acquire the lock”*** for these mirrors. We currently spend **~10%** unable to observe since we’re trying to lock the motions.



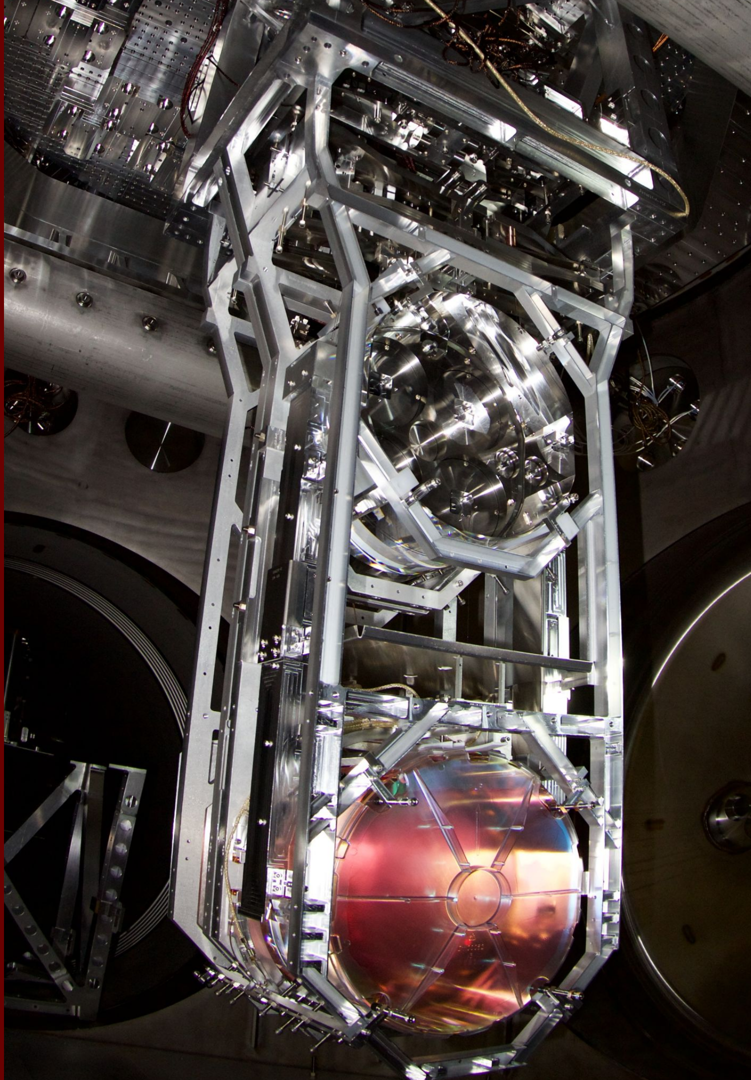
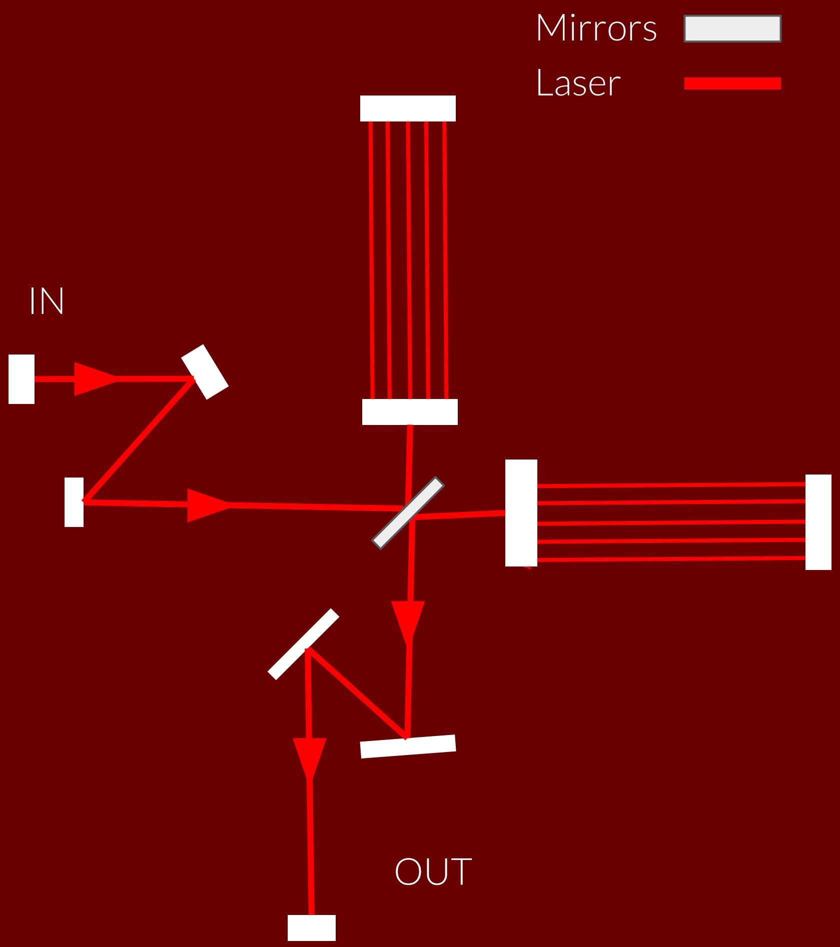


Photo: Caltech/MIT/LIGO Laboratory

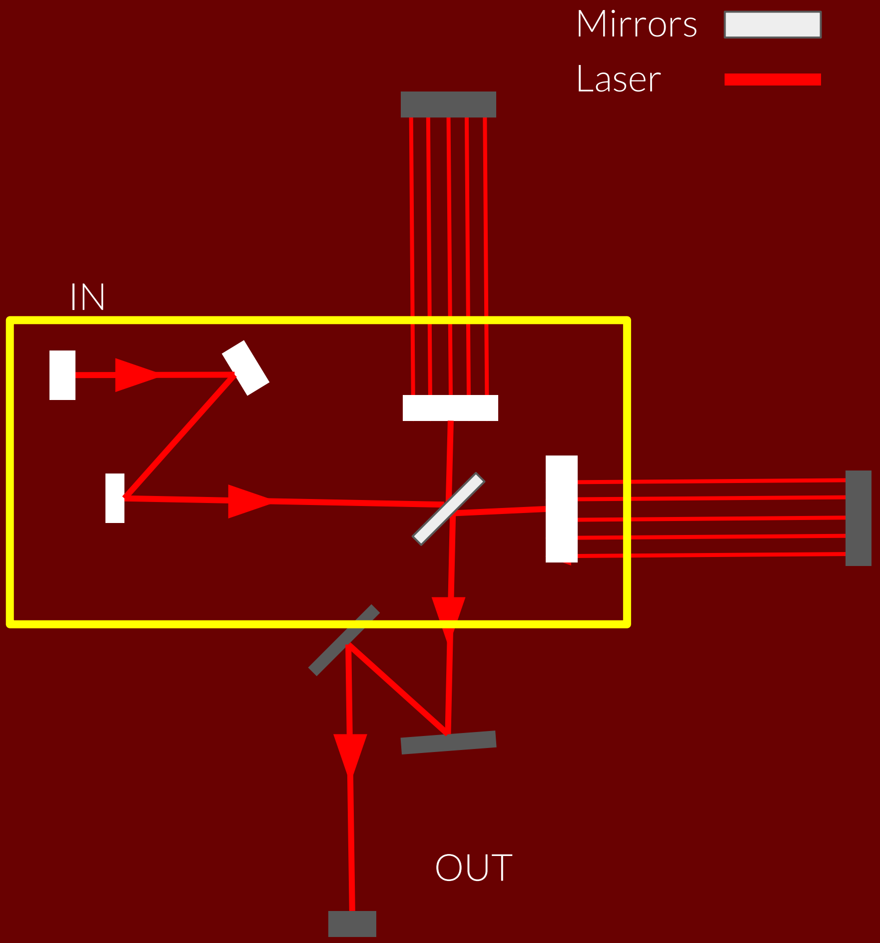
Introduction

LIGO Mirrors



Introduction

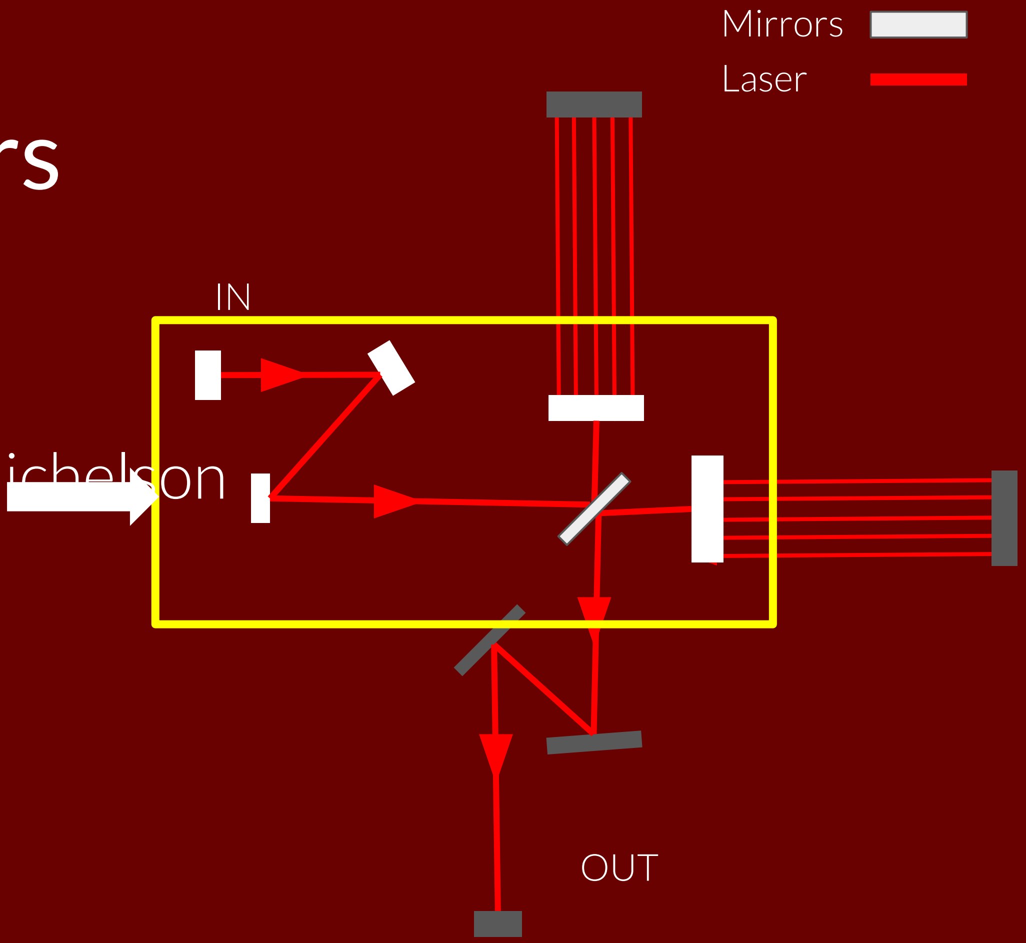
LIGO Mirrors



Introduction

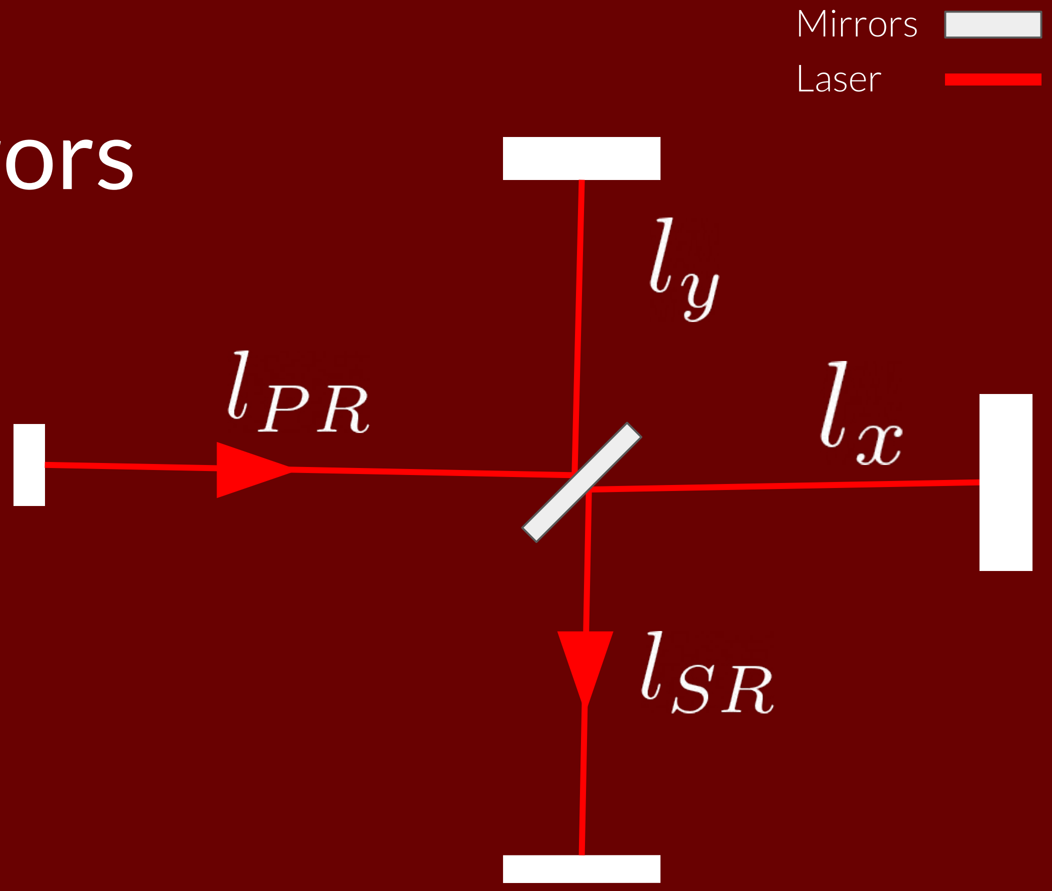
LIGO Mirrors

Power **R**ecycled **M**ichelson
(PRMI)



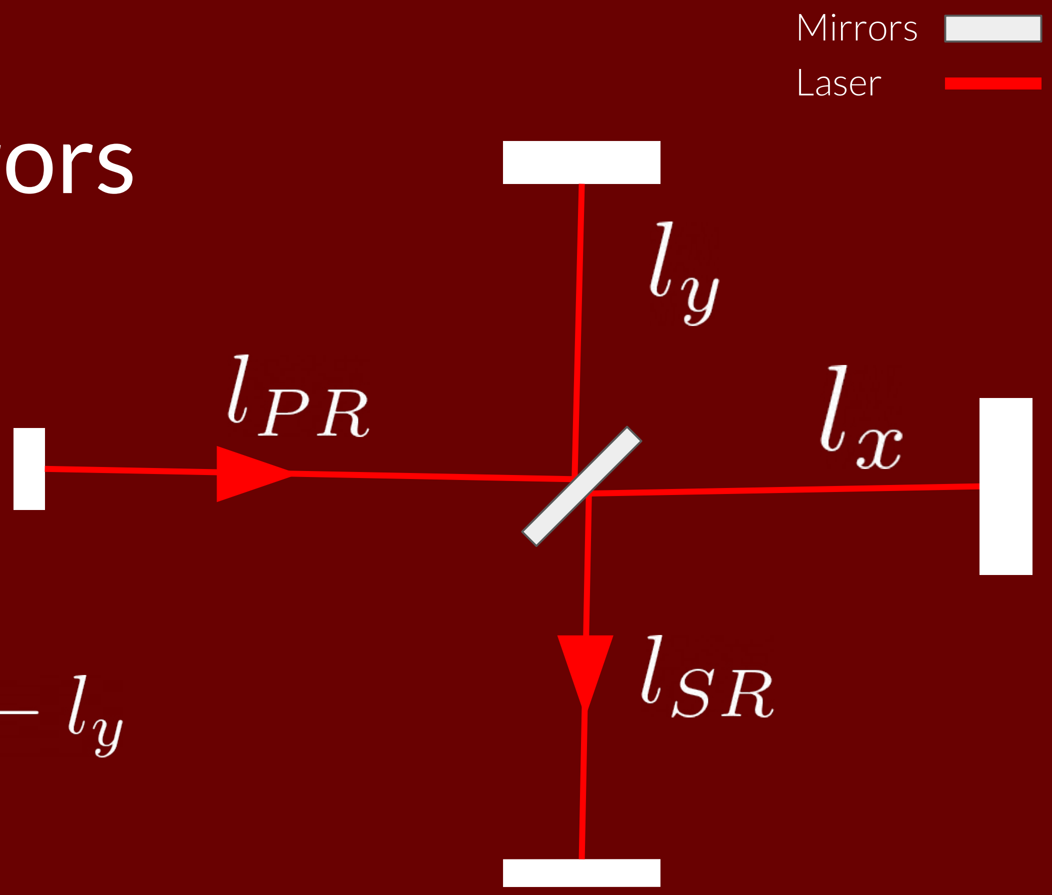
Introduction

LIGO Mirrors



Introduction

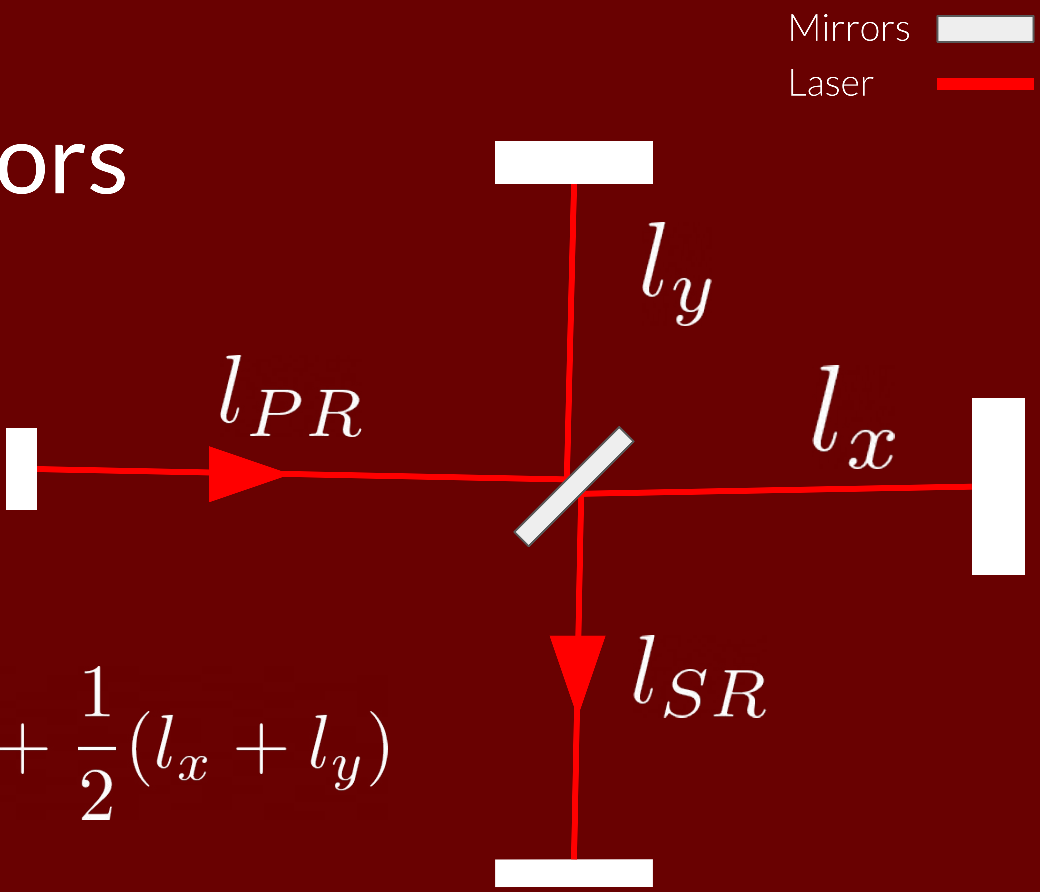
LIGO Mirrors



$$\mathrm{MICH} = l_x - l_y$$

Introduction

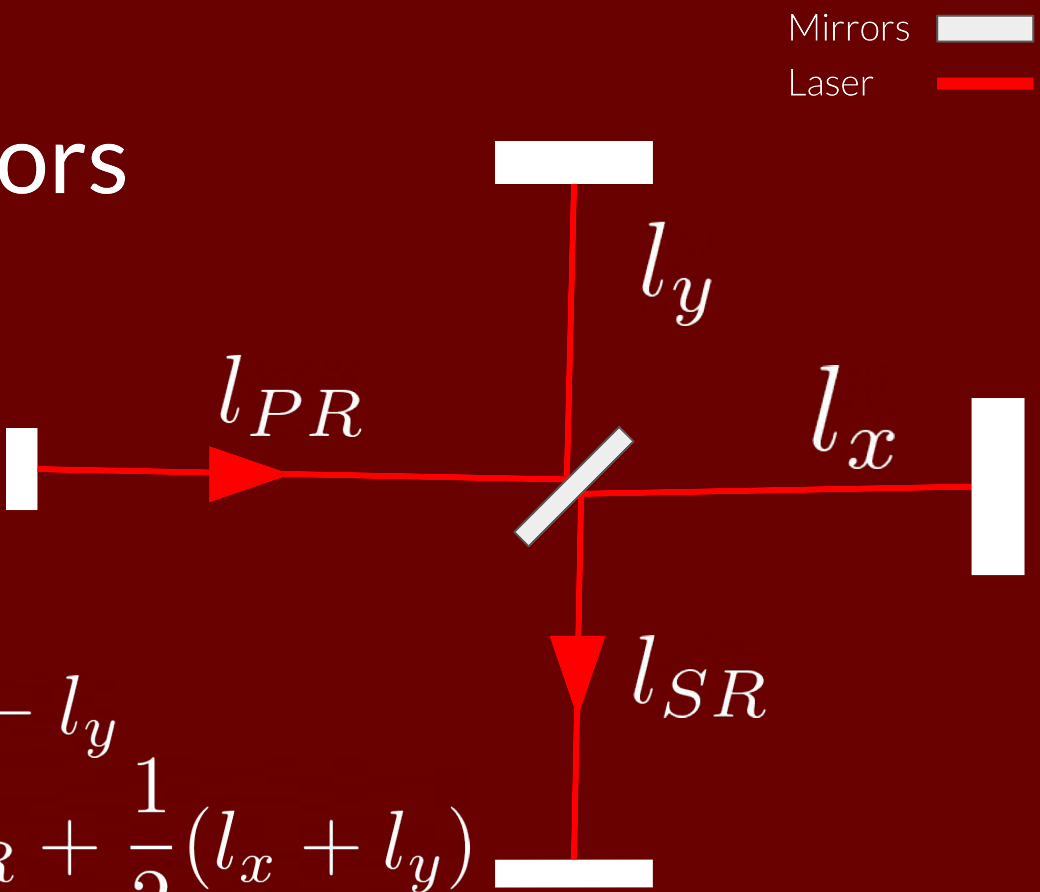
LIGO Mirrors



$$\mathrm{PRCL} = l_{PR} + \frac{1}{2}(l_x + l_y)$$

Introduction

LIGO Mirrors

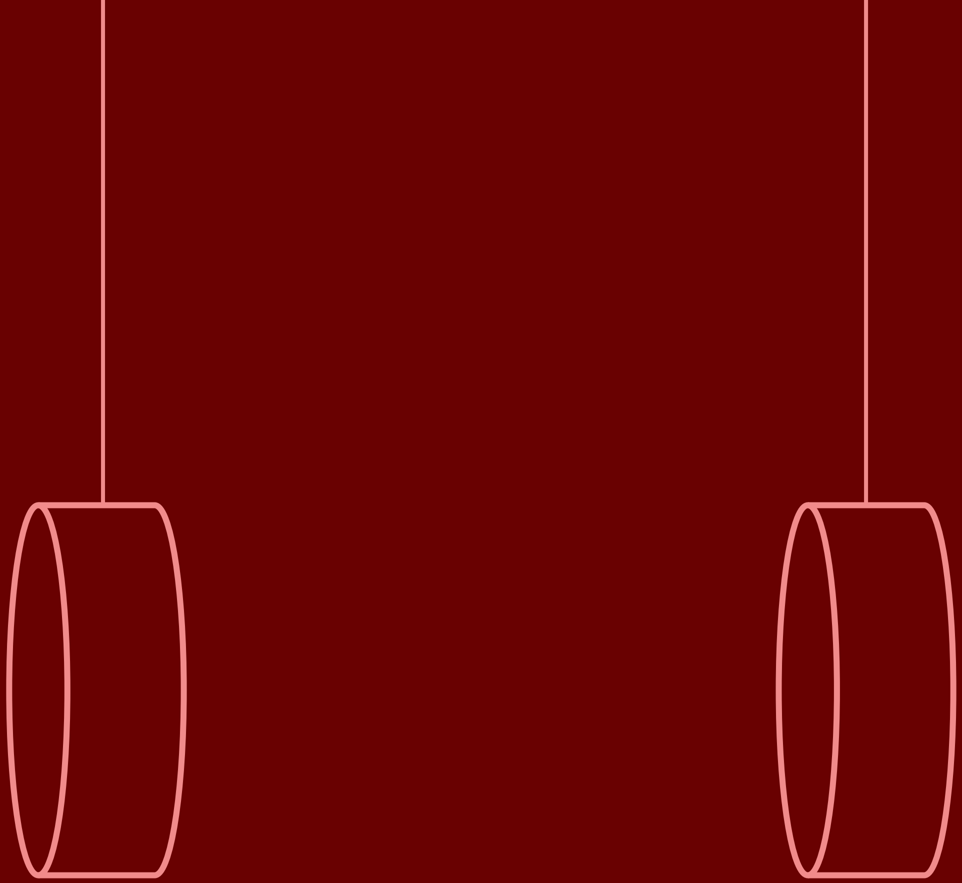


$$\mathrm{MICH} = l_x - l_y$$

$$\mathrm{PRCL} = l_{PR} + \frac{1}{2}(l_x + l_y)$$

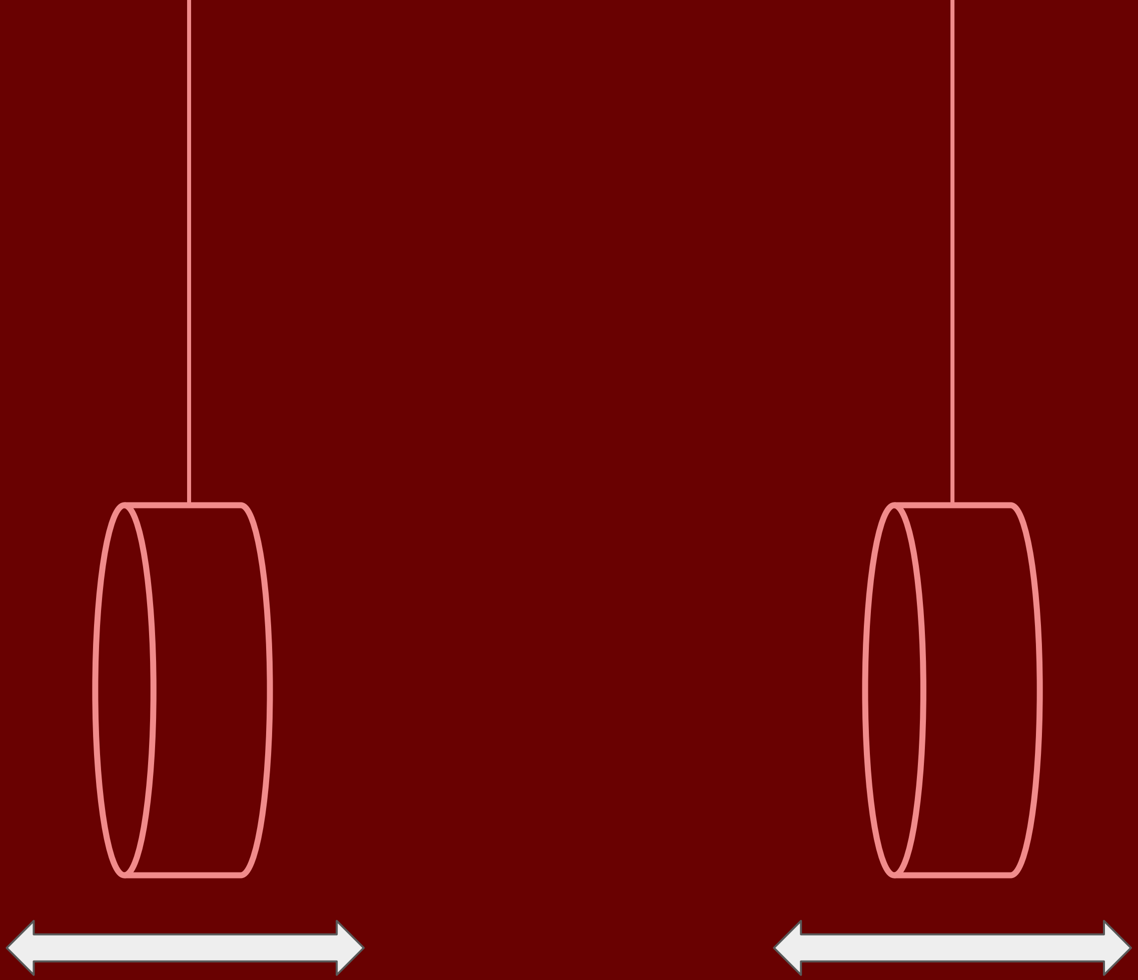
Introduction

LIGO Mirrors



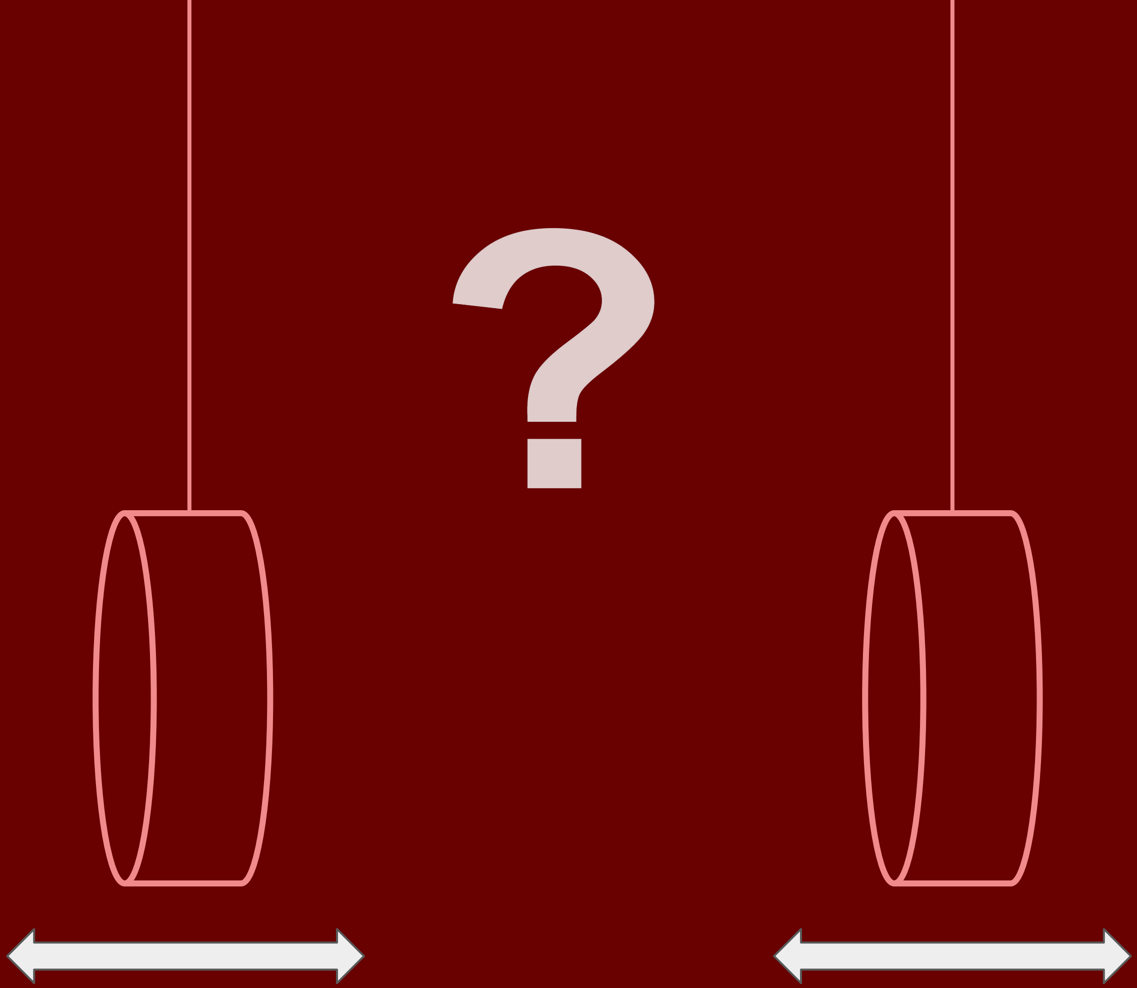
Introduction

LIGO Mirrors

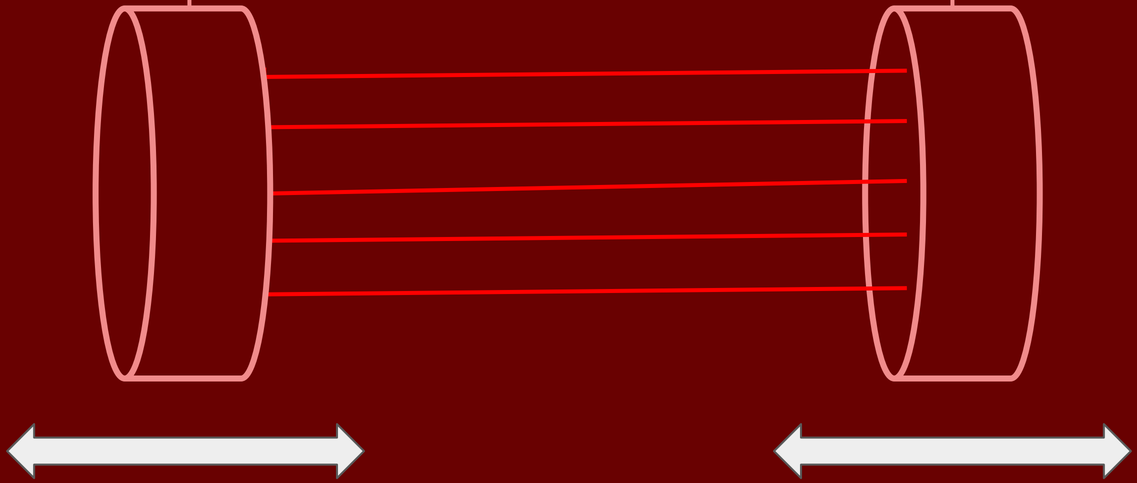


Controlling a pendulums is a solved problem.

Controlling a pendulums is a solved problem. **If we know its position / dynamics**



We have ***Optical Signals*** which are a **non-linear, non-unique** mapping of the **positions.**



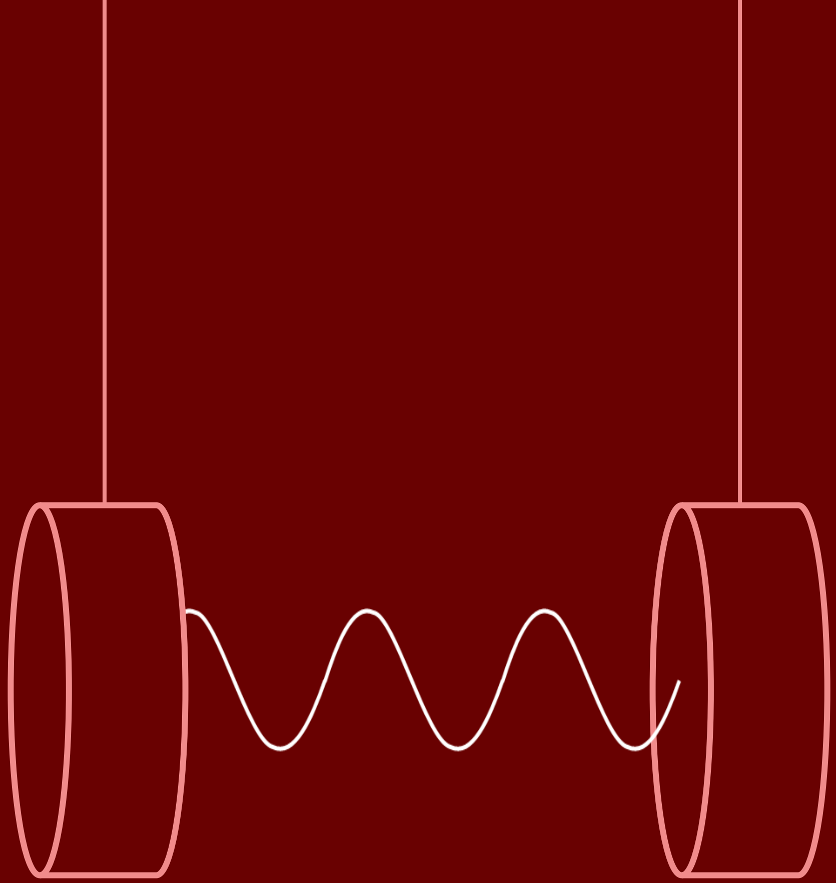
Inverting The Problem: given only the *optical signal* can we predict the positions of the mirrors?

Non-uniqueness

Wrapping

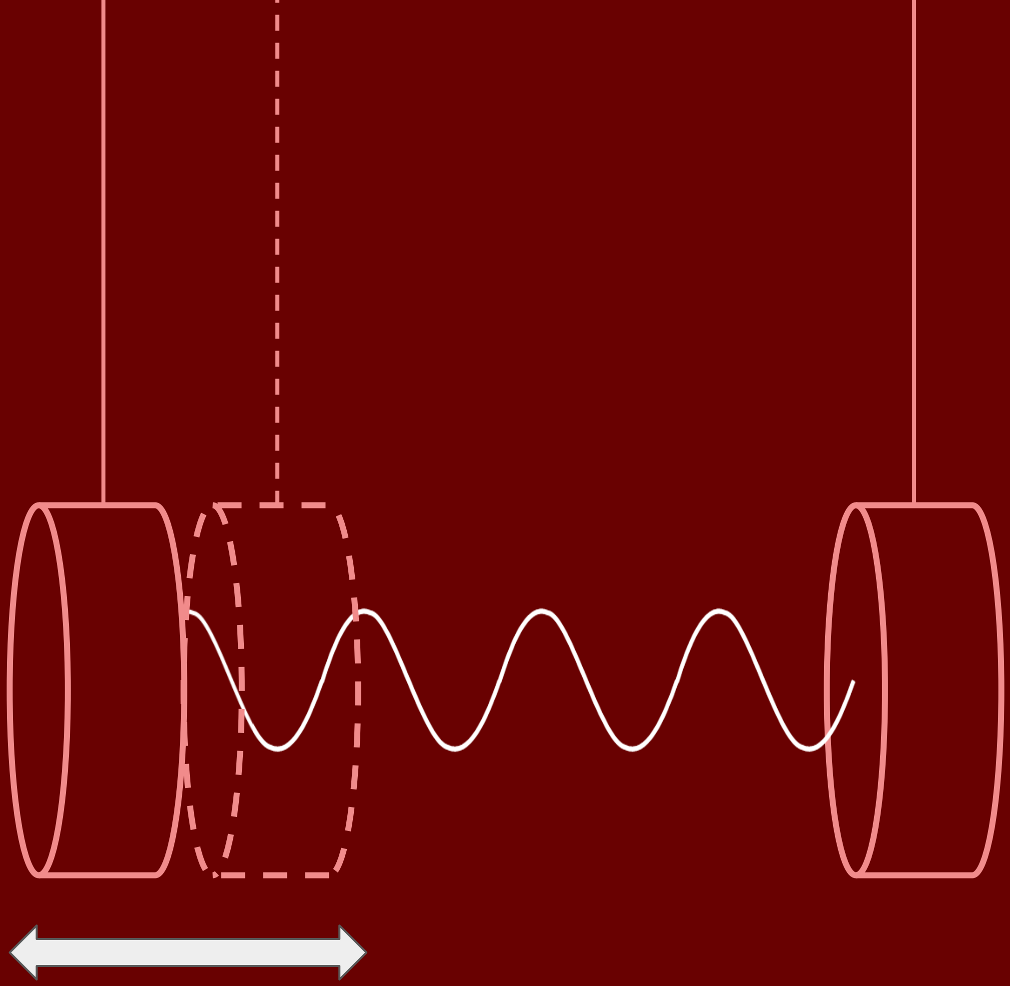
Non-uniqueness

Wrapping



Non-uniqueness

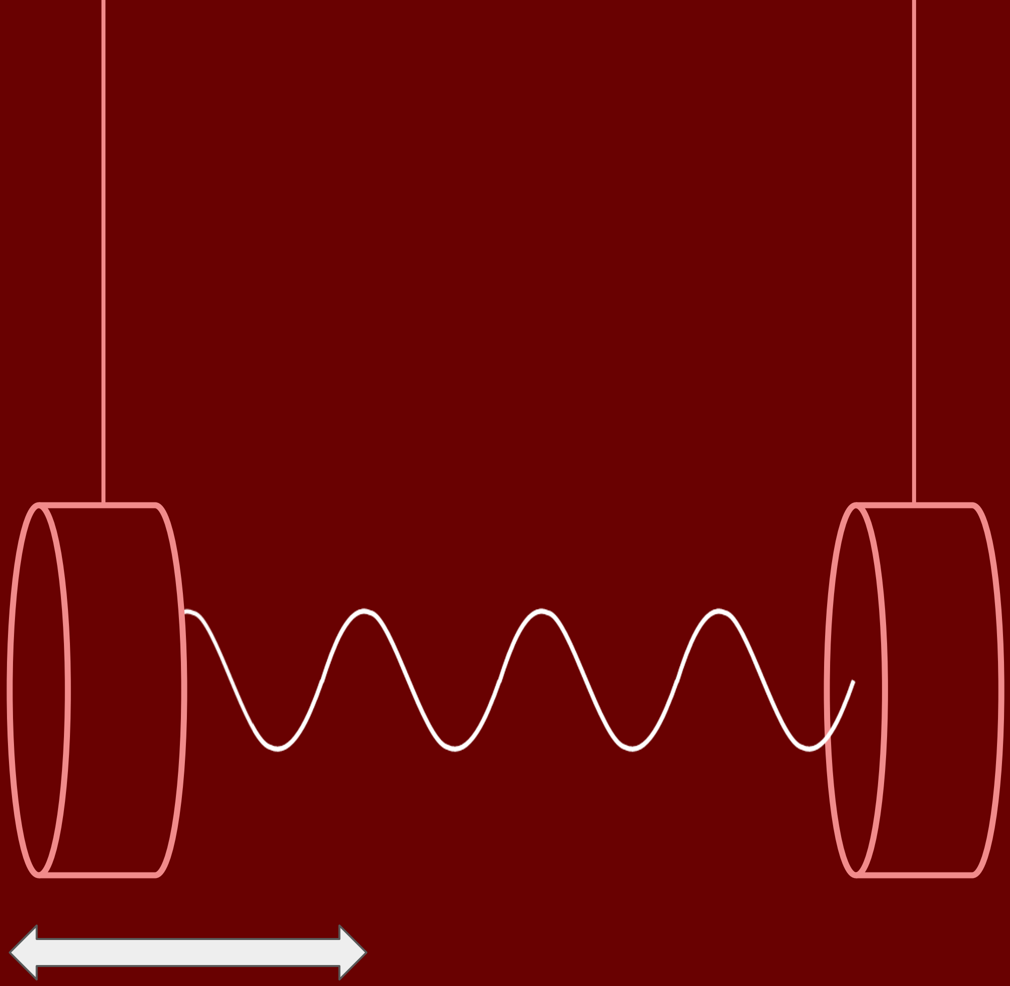
Wrapping



Non-uniqueness

Wrapping

Any *linear adjustment* by a specific amount could result in the same signals.

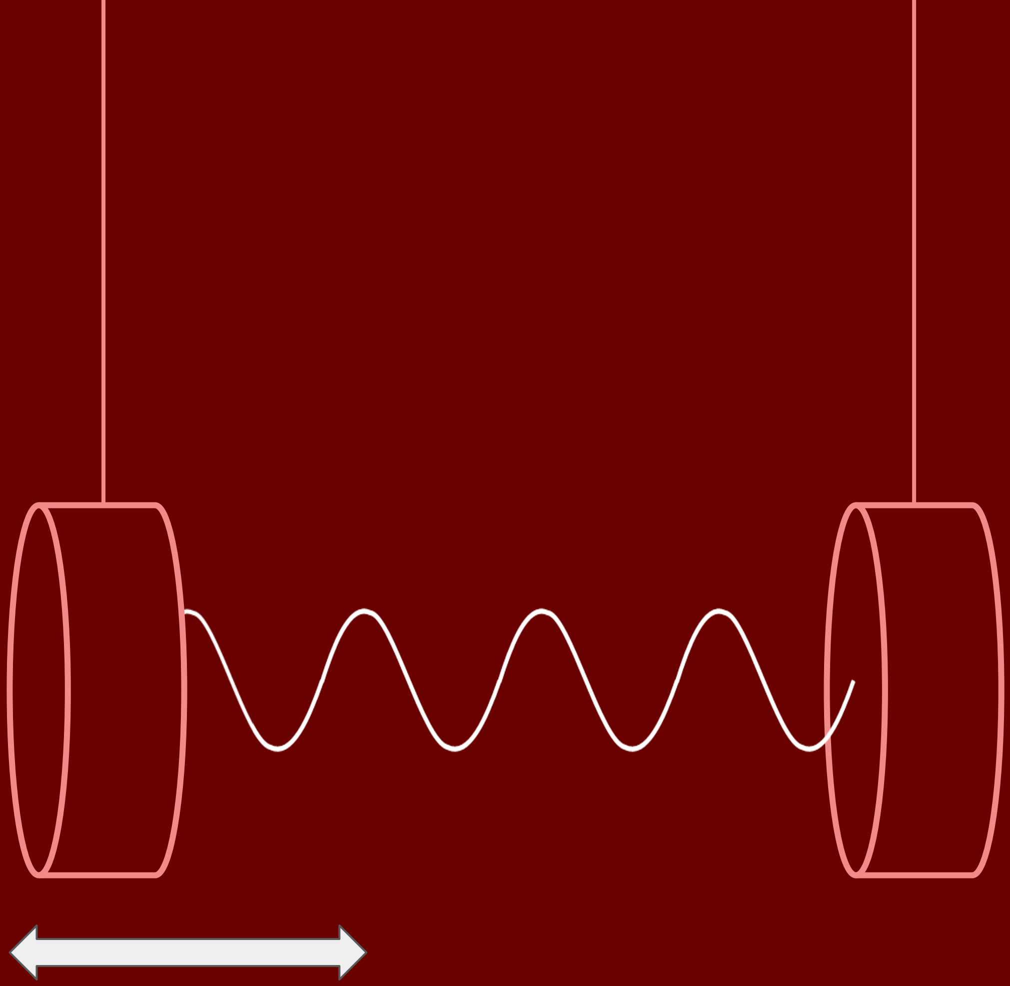


Non-uniqueness

Wrapping

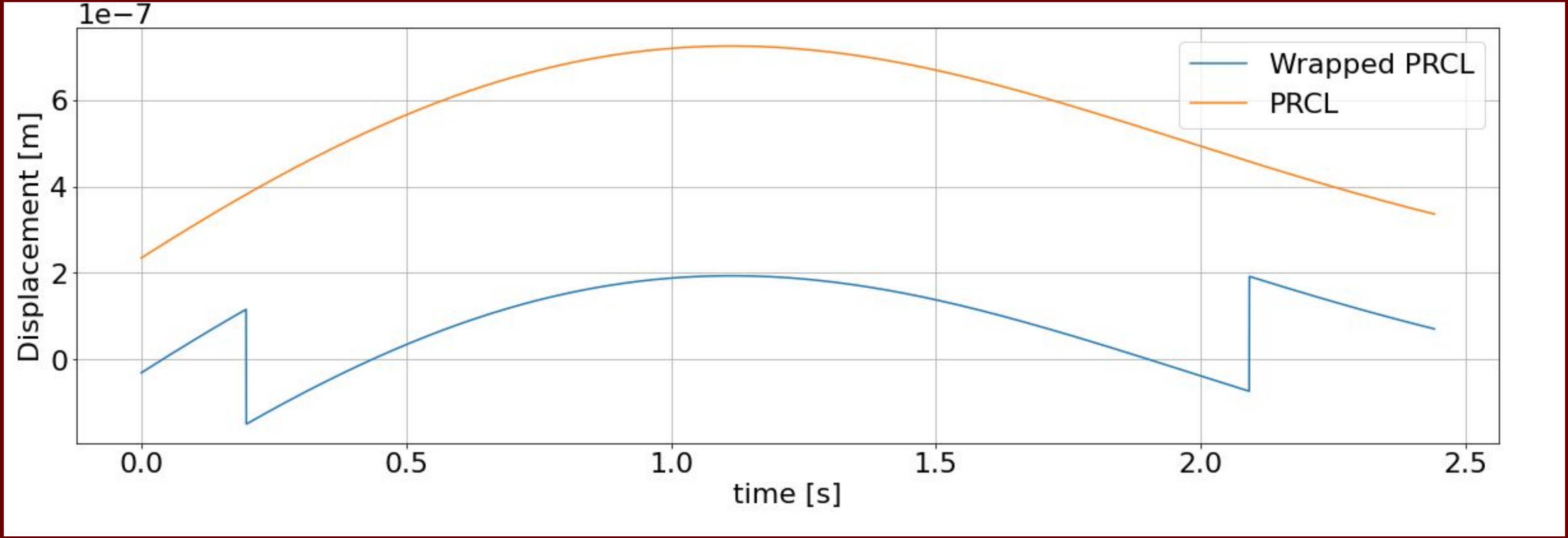
Any *linear adjustment* by a specific amount could result in the same signals.

We can make solutions unique linearly adjust all positions to a specific range!



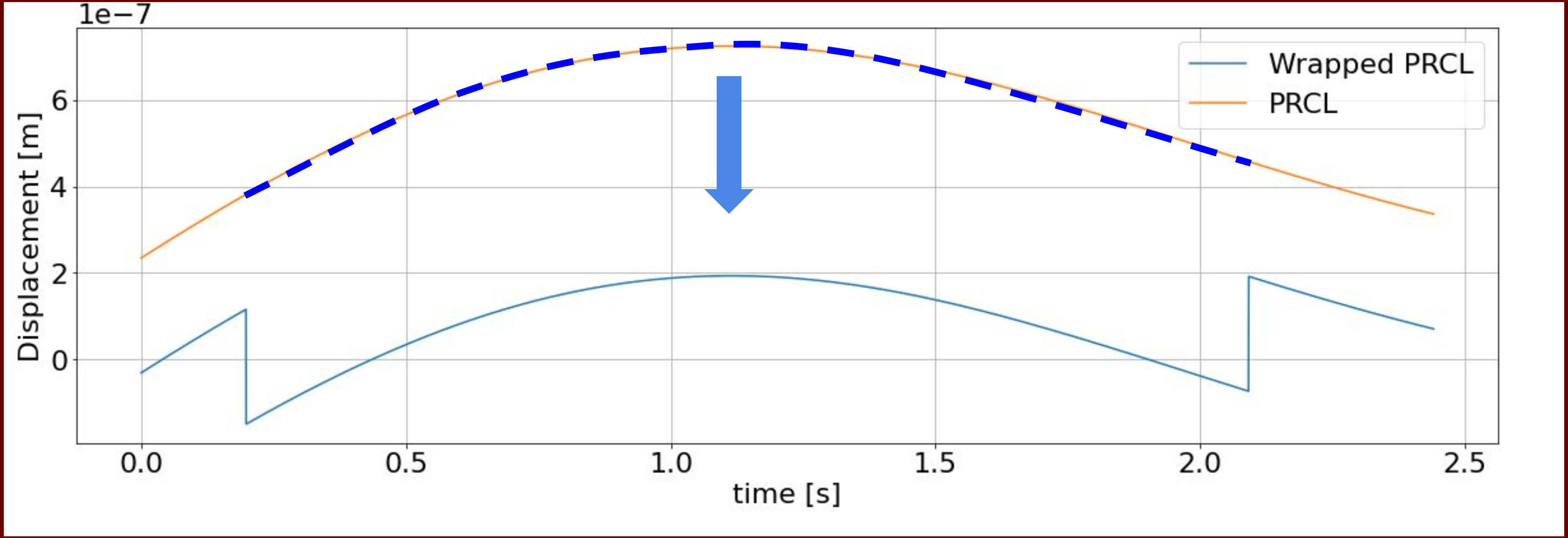
Non-uniqueness

Wrapping



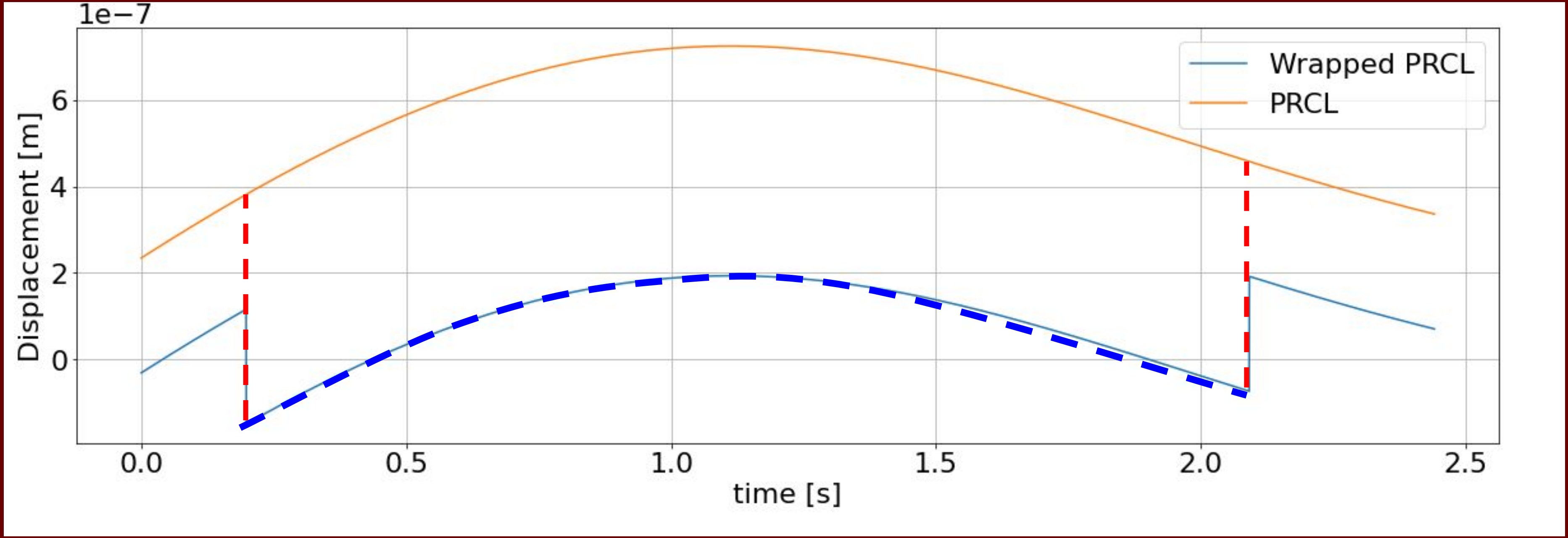
Non-uniqueness

Wrapping

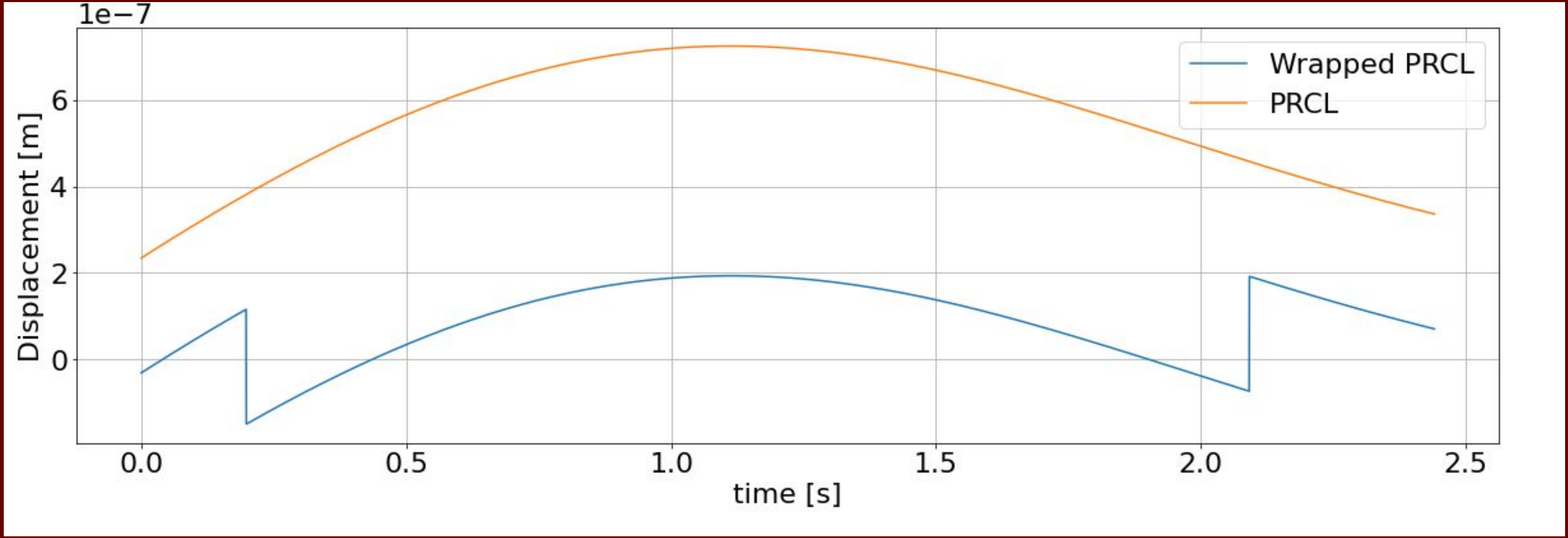


Non-uniqueness

Wrapping

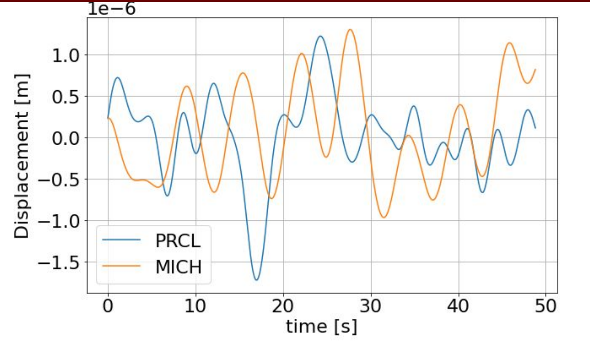


Non-uniqueness
Wrapping



Non-uniqueness
Wrapping

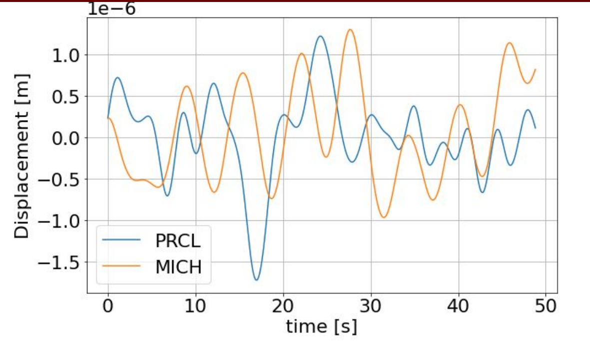
Simulate Motions

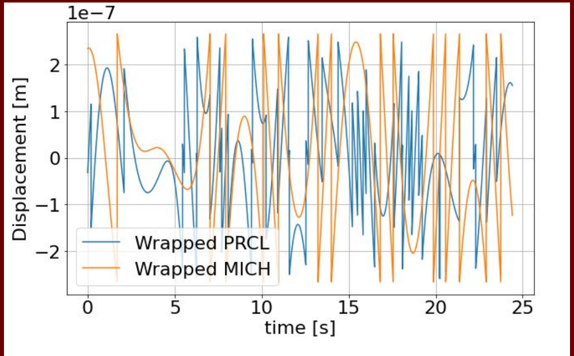


Non-uniqueness

Wrapping

Simulate Motions → Wrap data

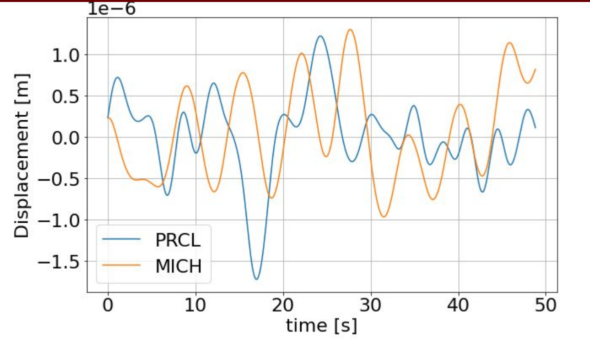


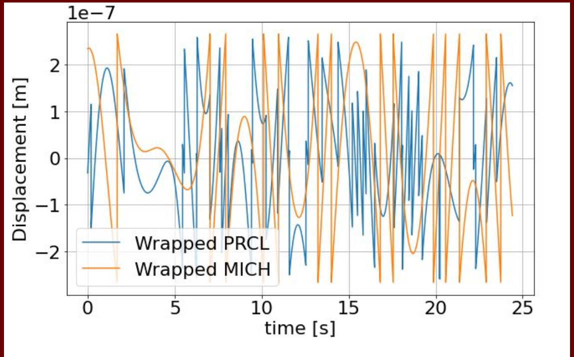


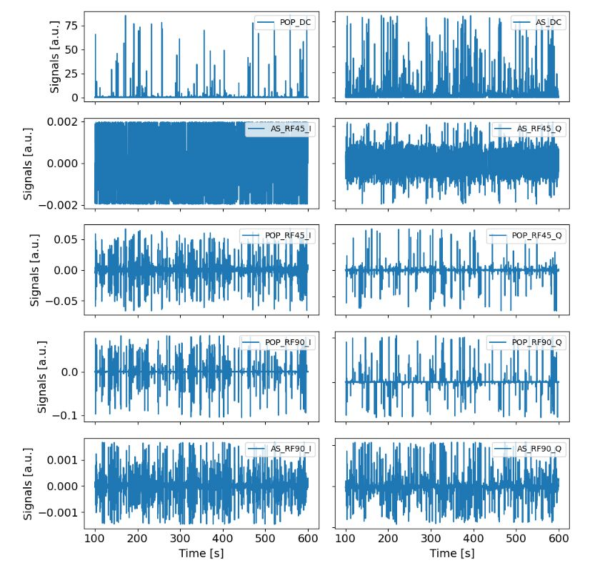
Non-uniqueness

Wrapping

Simulate Motions → Wrap data → Get Signals

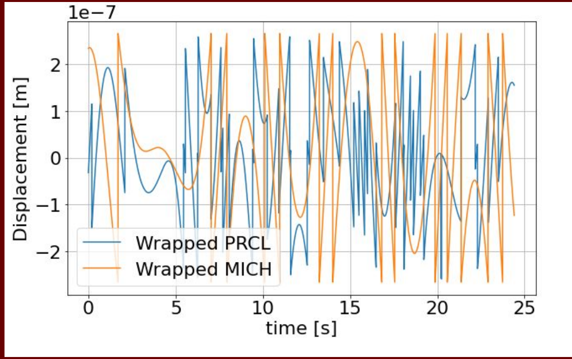




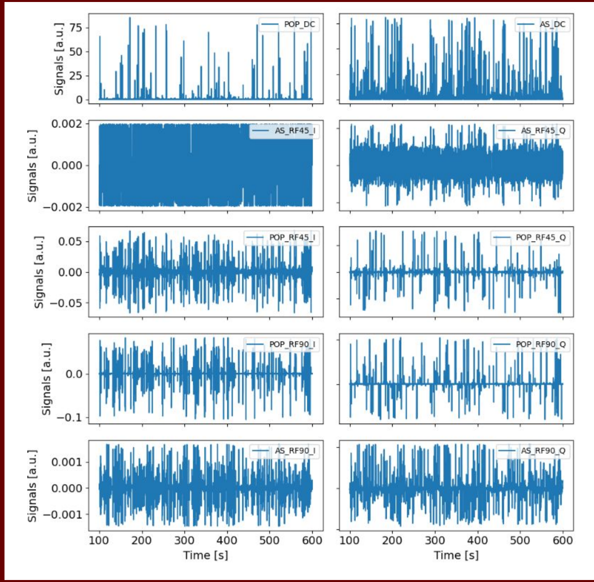


Non-uniqueness
Wrapping

Wrap data

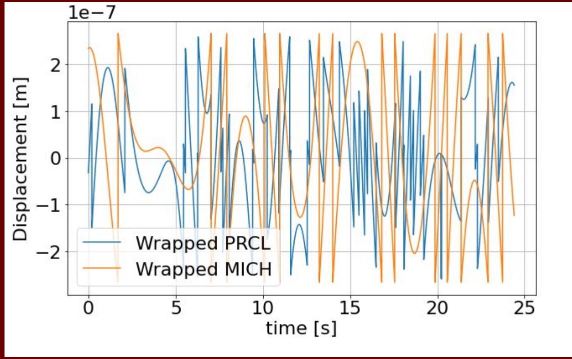


Signals



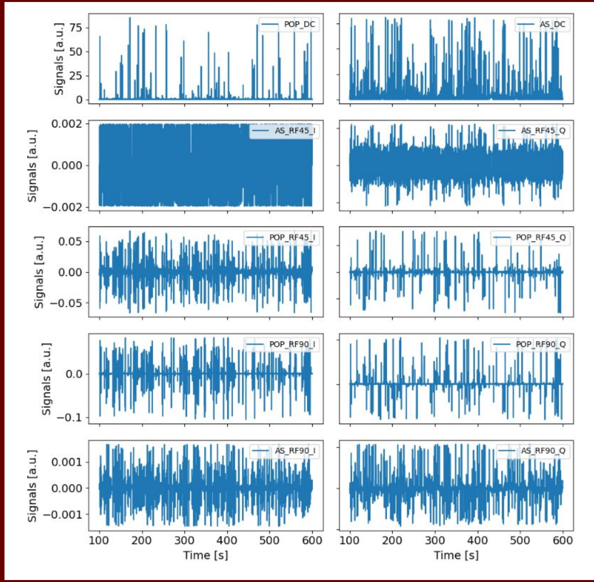
Non-uniqueness
Wrapping

Wrap data



ML Model

Signals



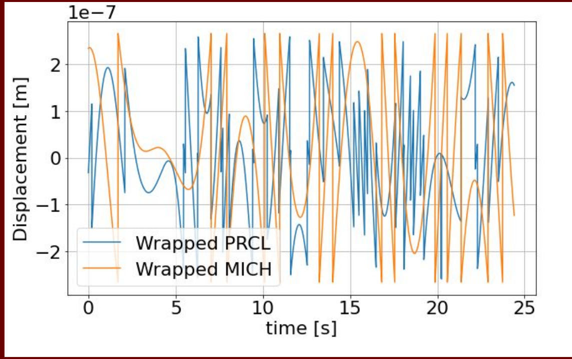
Non-uniqueness

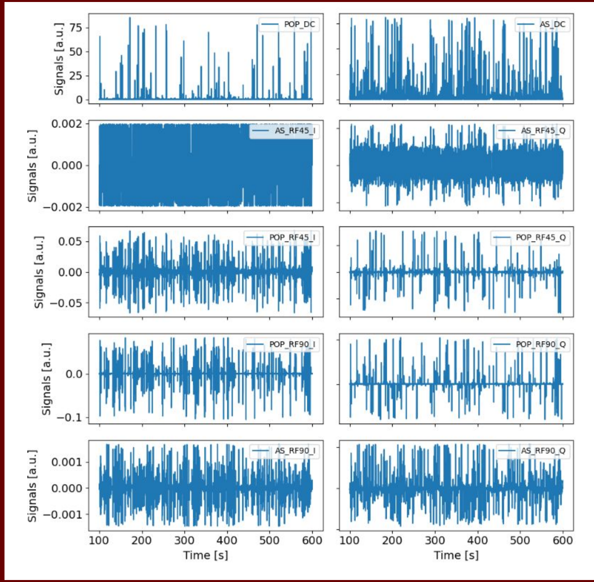
Wrapping

Wrap data

Signals

<u>ML Model</u>

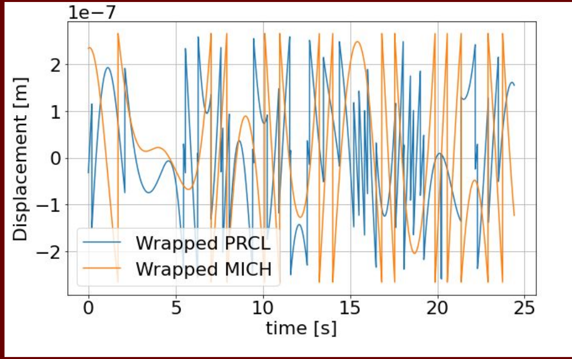




Predicting Wrapped Positions

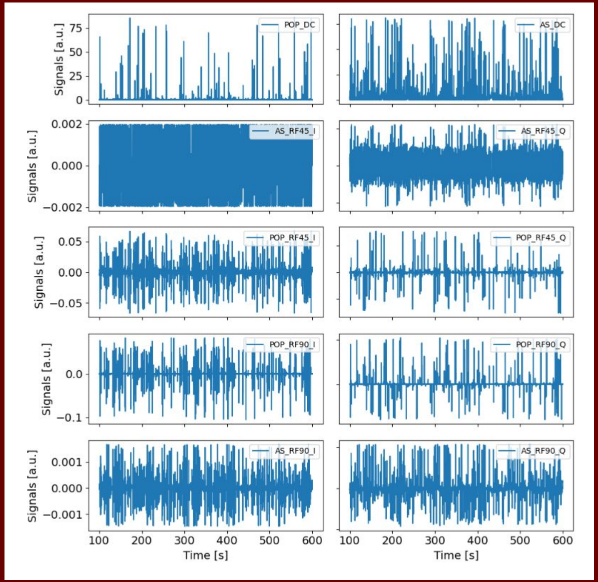
Deep Learning Model

Wrap data



Gated Recurrent Units

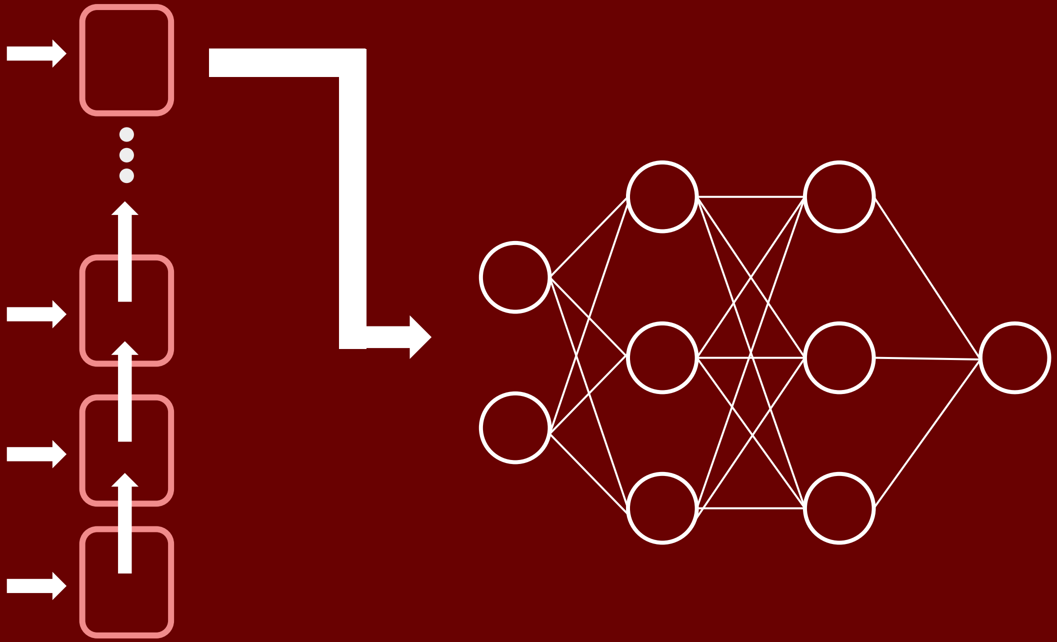
Signals



Predicting Wrapped Positions

Deep Learning Model

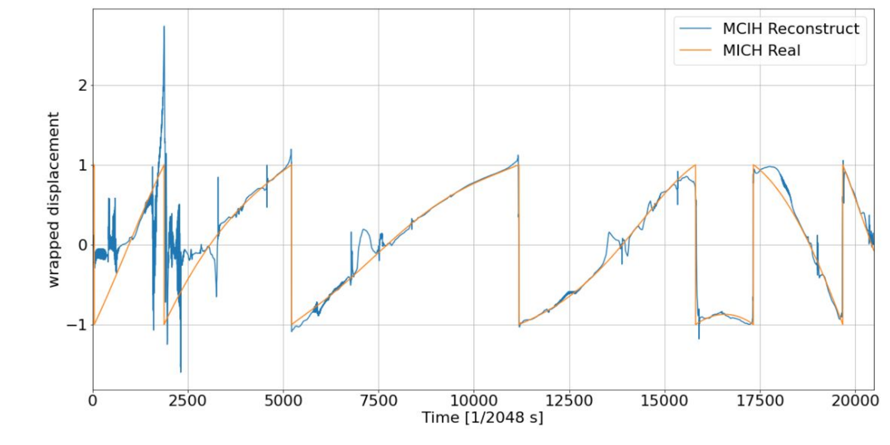
Signals

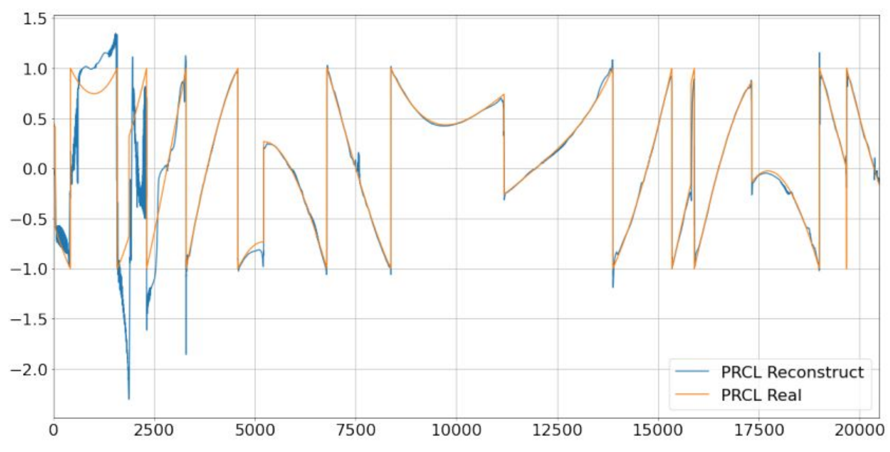


Wrapped PRCL and MICH

Predicting Wrapped Positions

Deep Learning Model

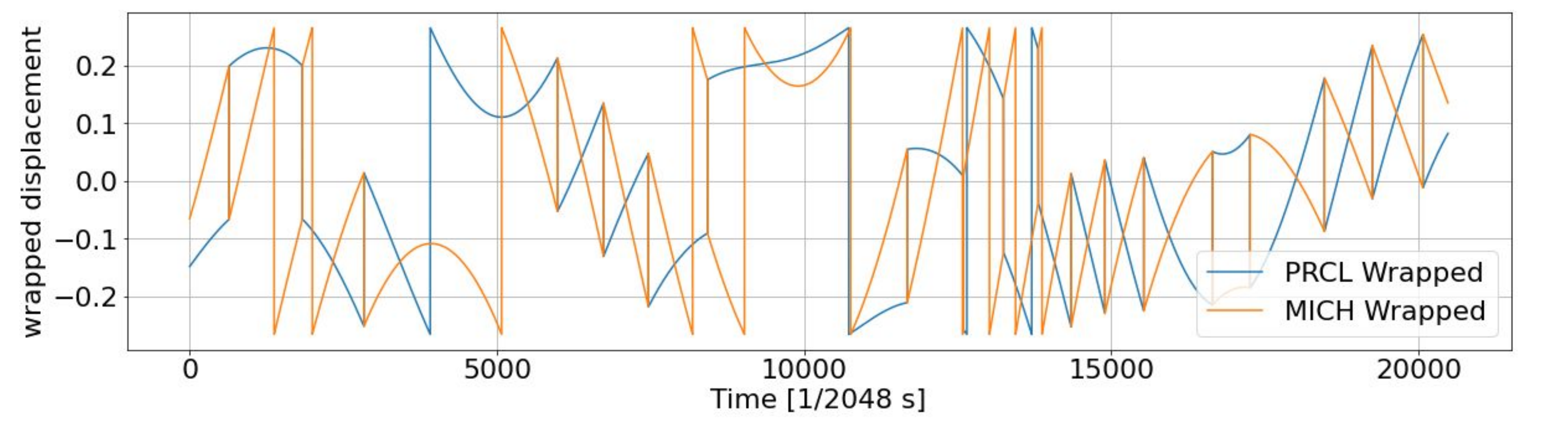




Problem: These are NOT real motions of the mirrors.

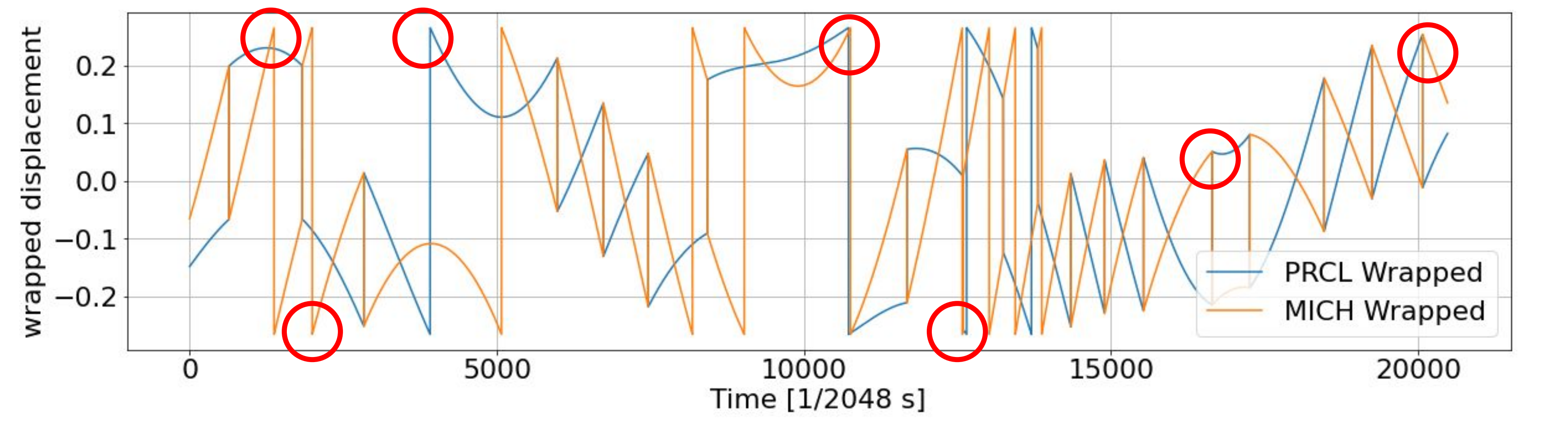
Non-uniqueness

Wrapping



Non-uniqueness

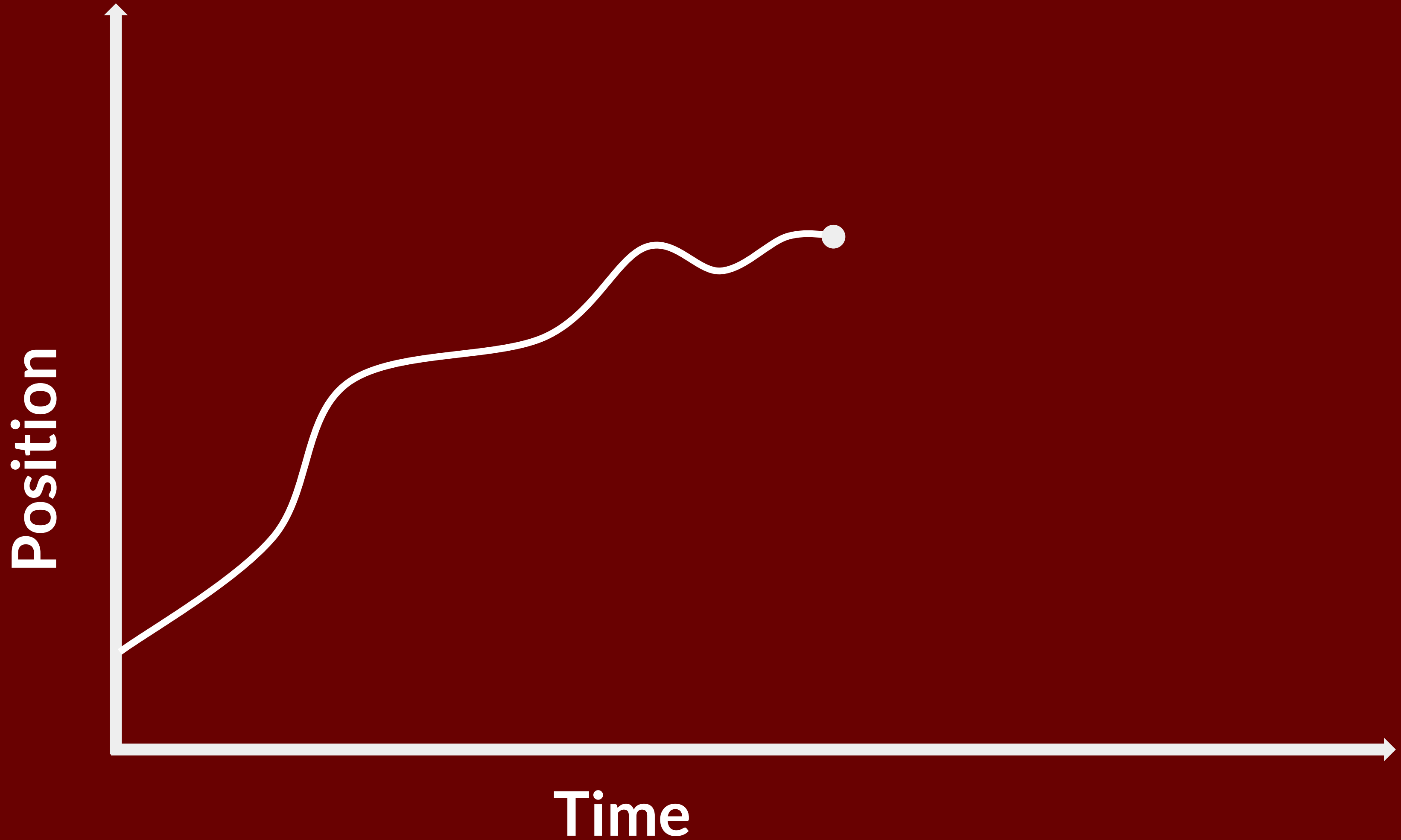
Wrapping

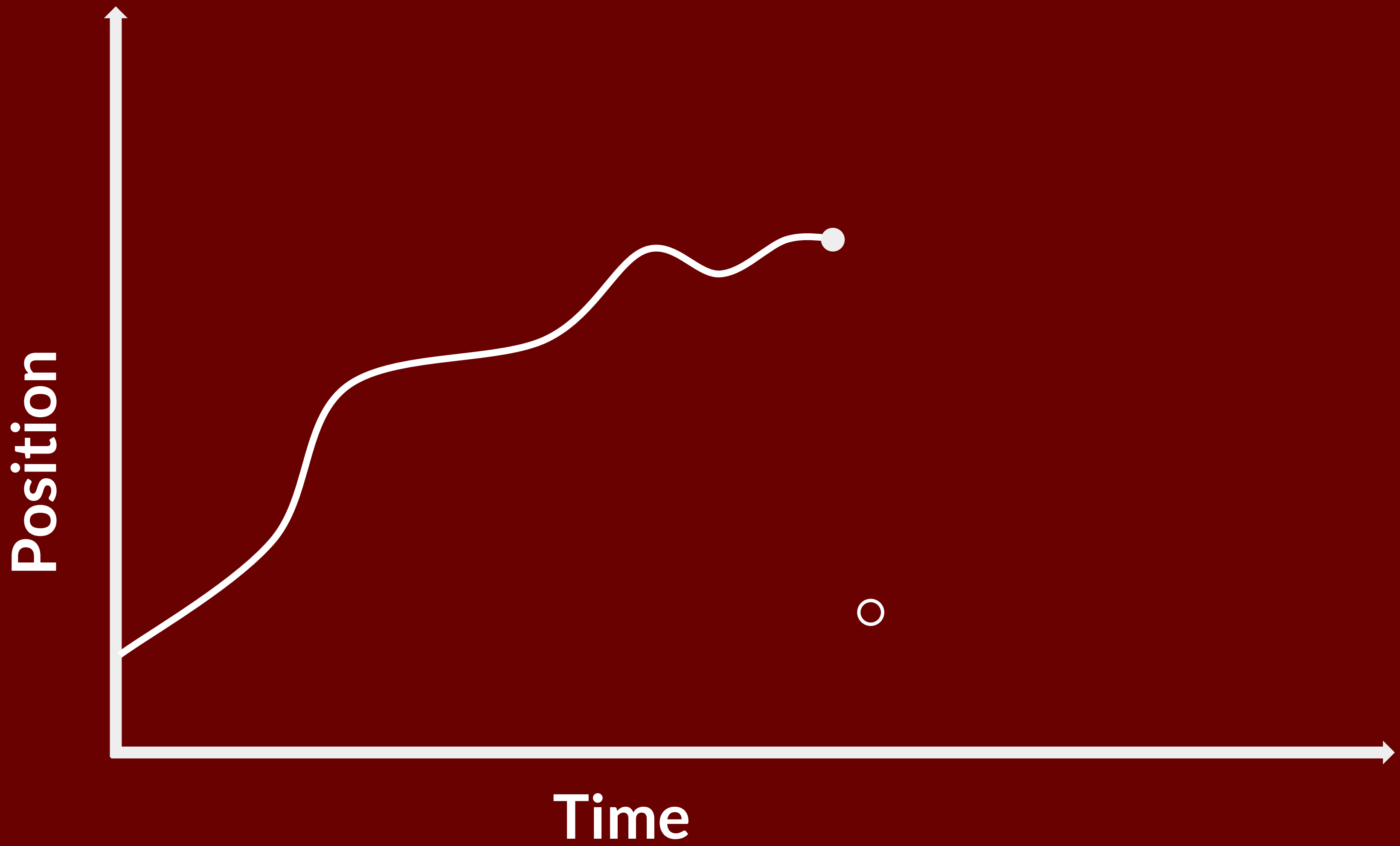


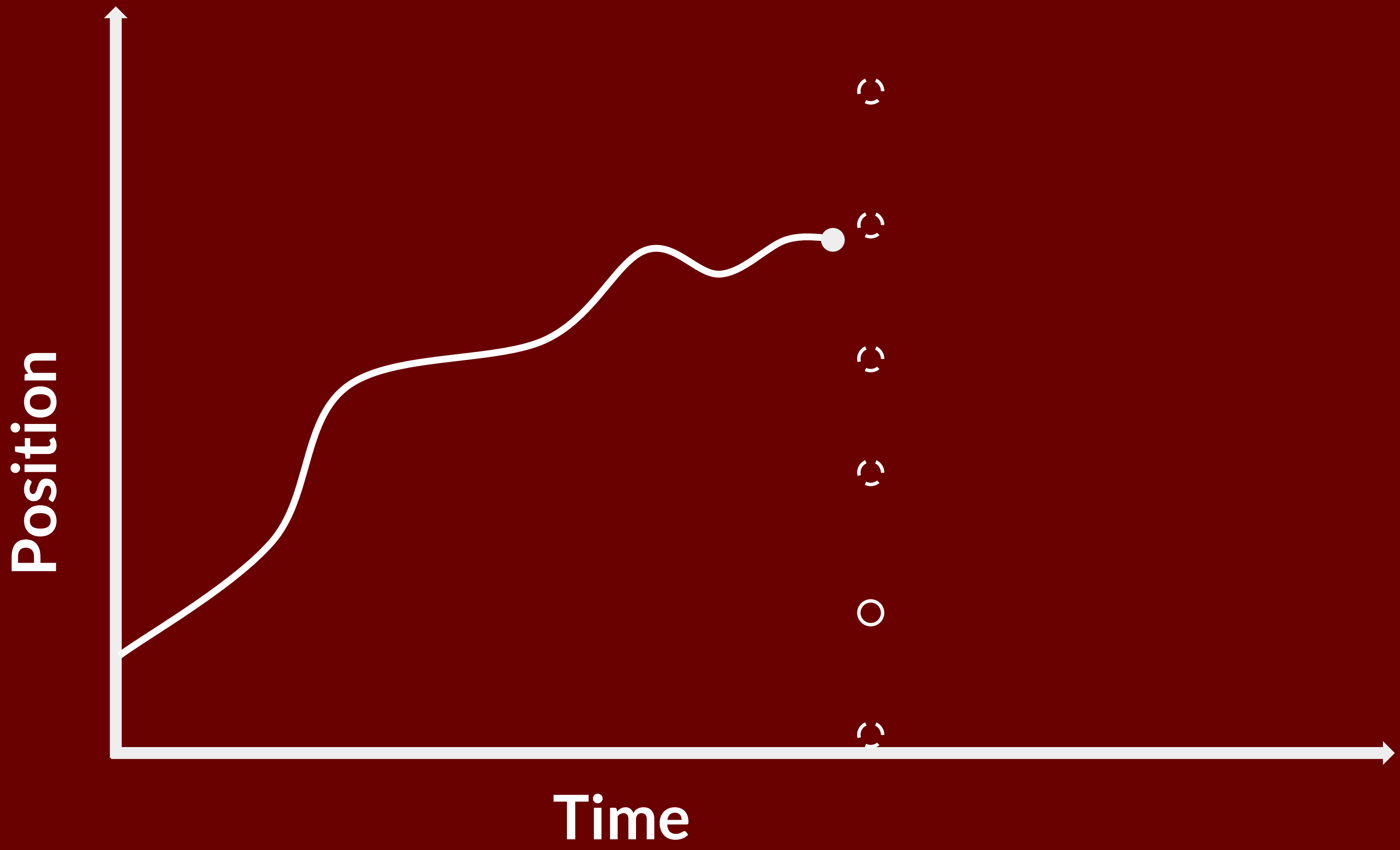
Problem: These are NOT real motions of the mirrors. We need to unwrap them.

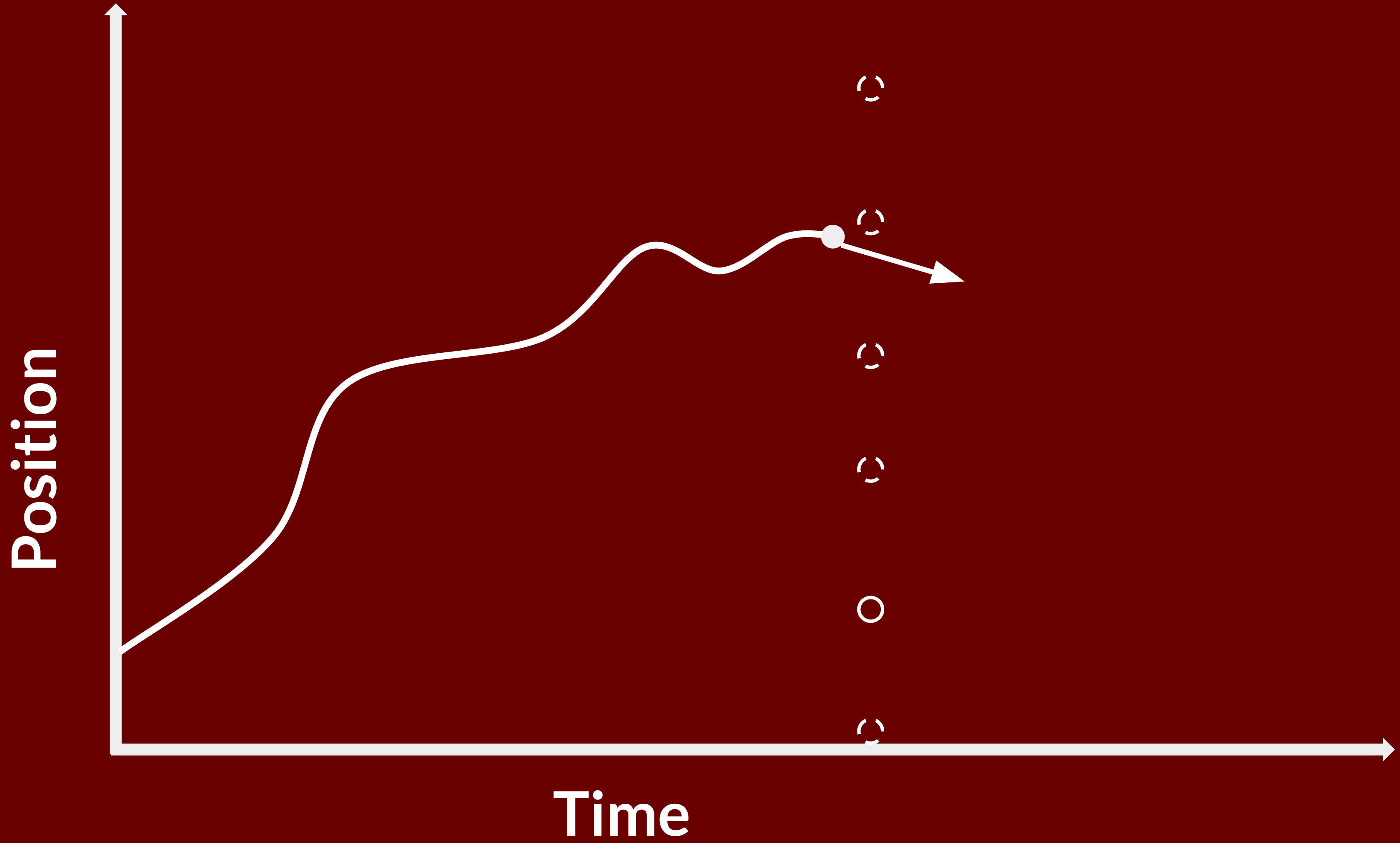
We use the a **Kalman Filter.**

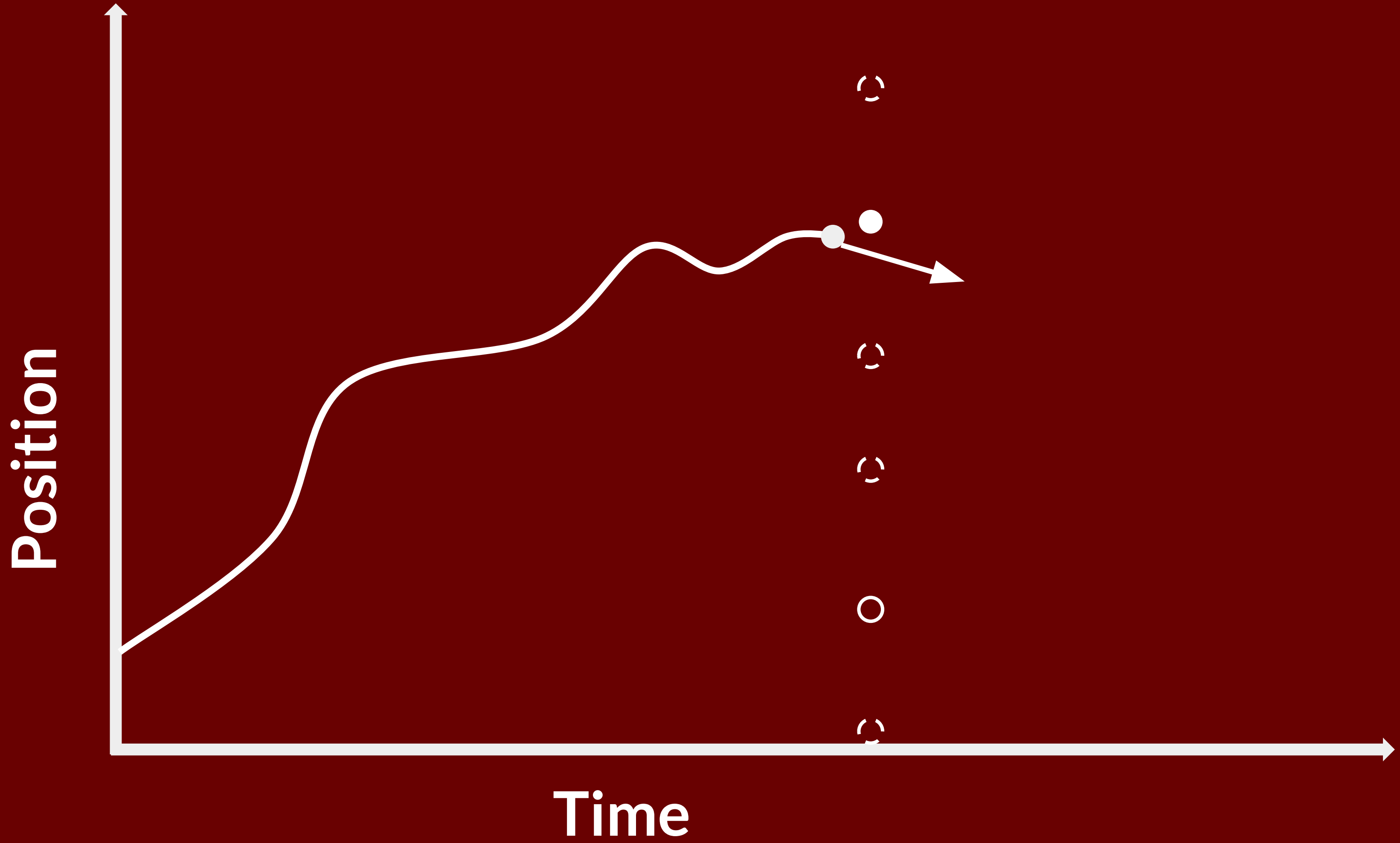
Imagine we had **position** AND **velocity** of the mirrors.


Position
Time

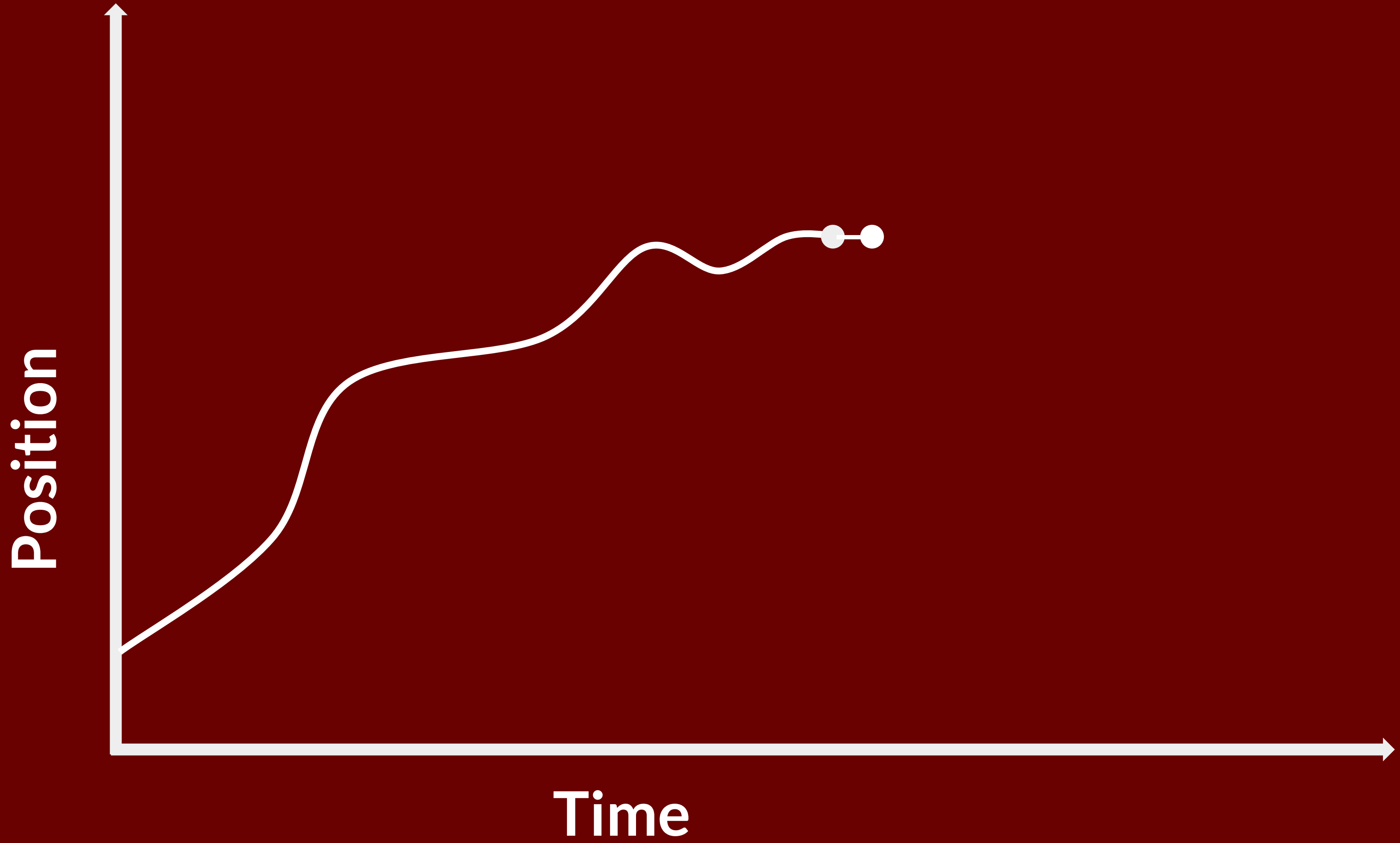

Position
Time


Position
Time


Position
Time



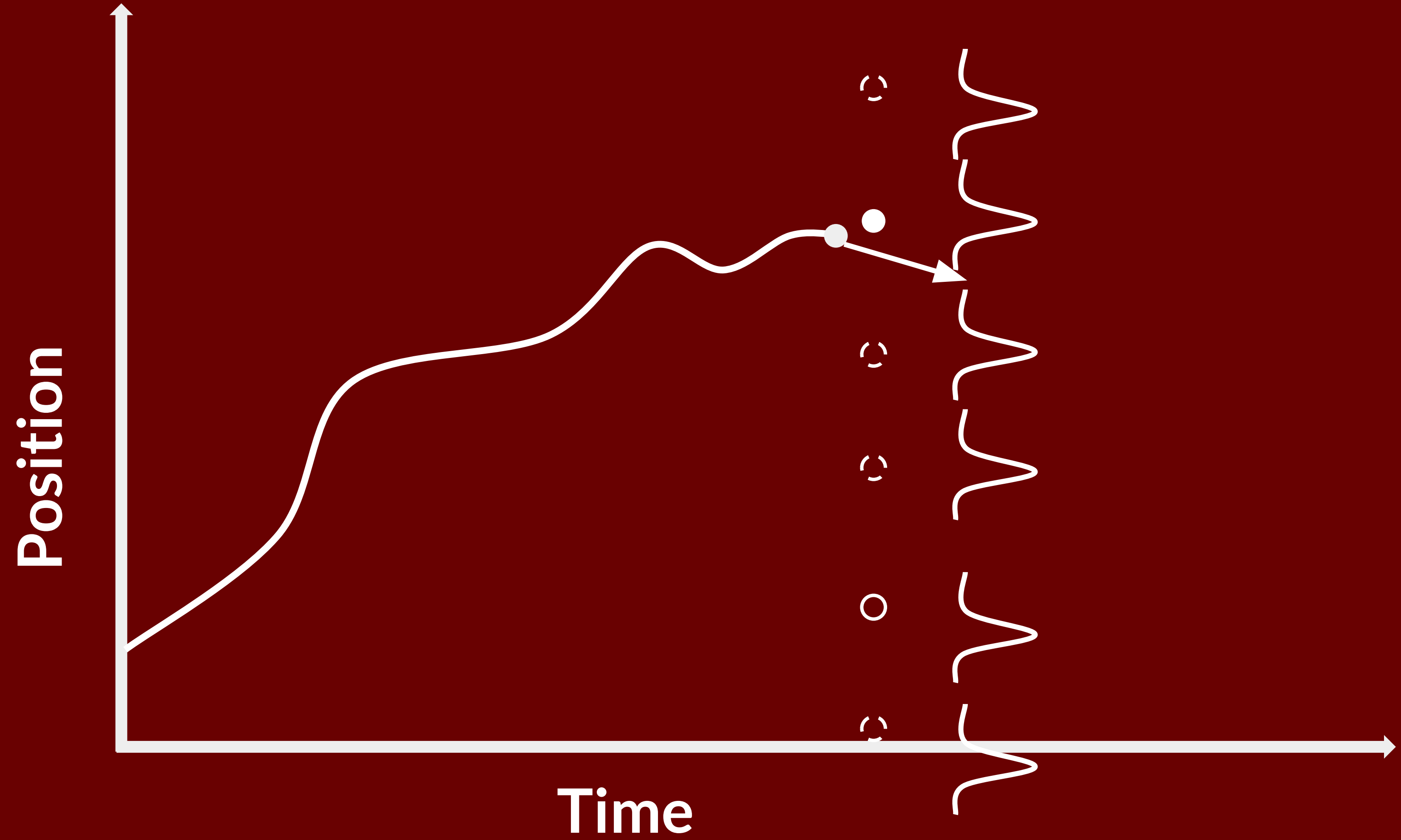
Position
Time


Position
Time

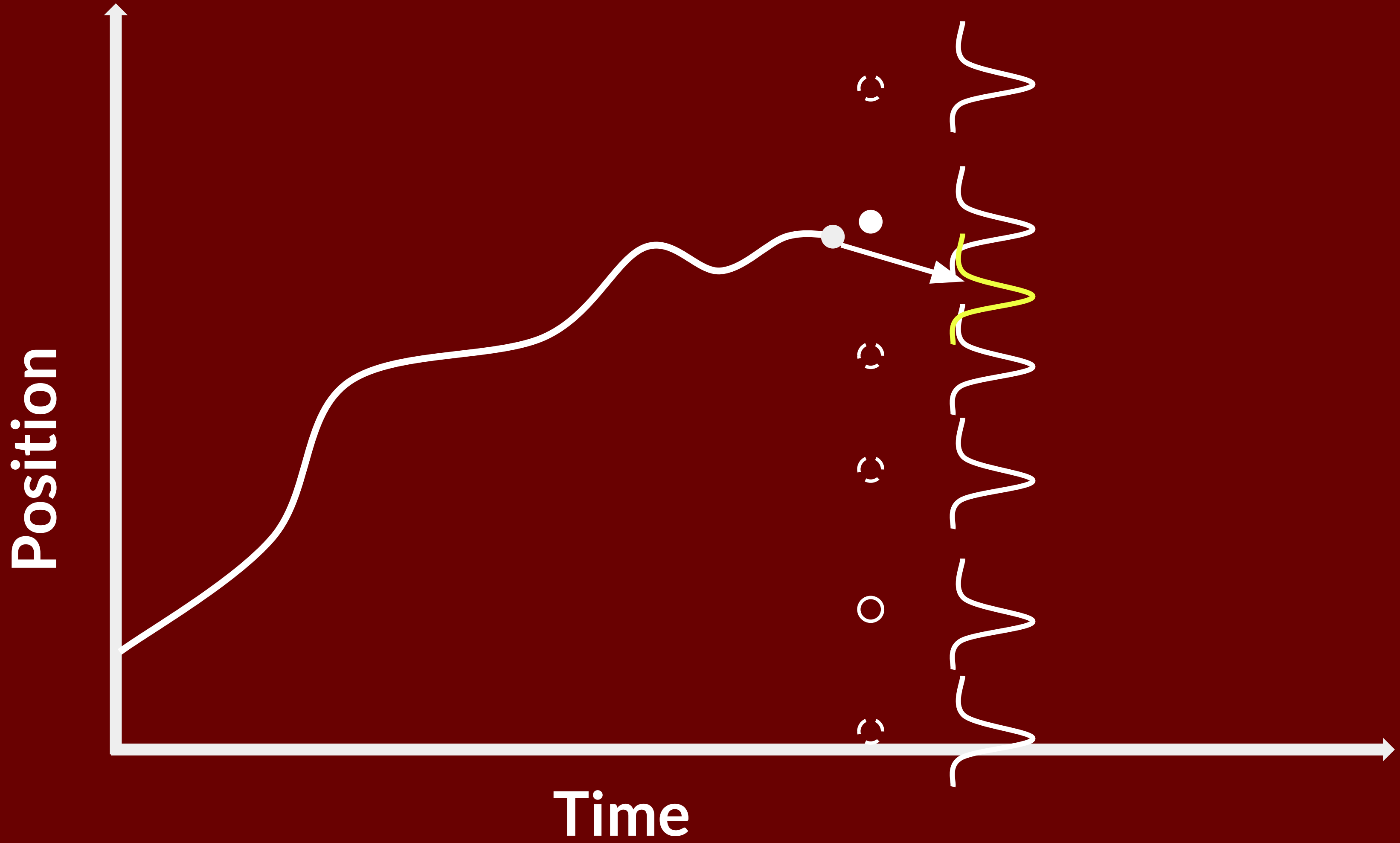
How do we combine Velocity and Position Information?

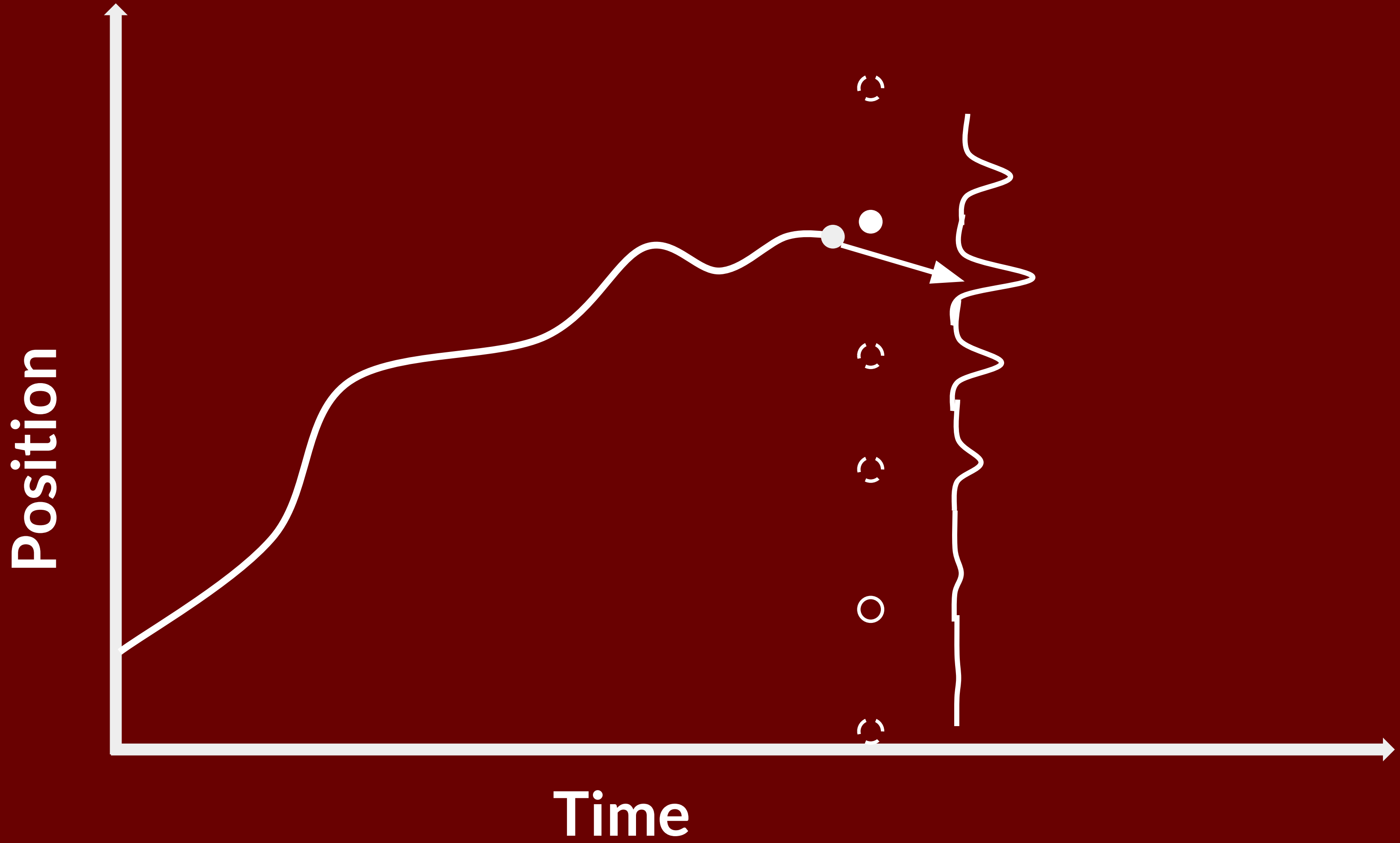
How do we combine Velocity and Position Information?

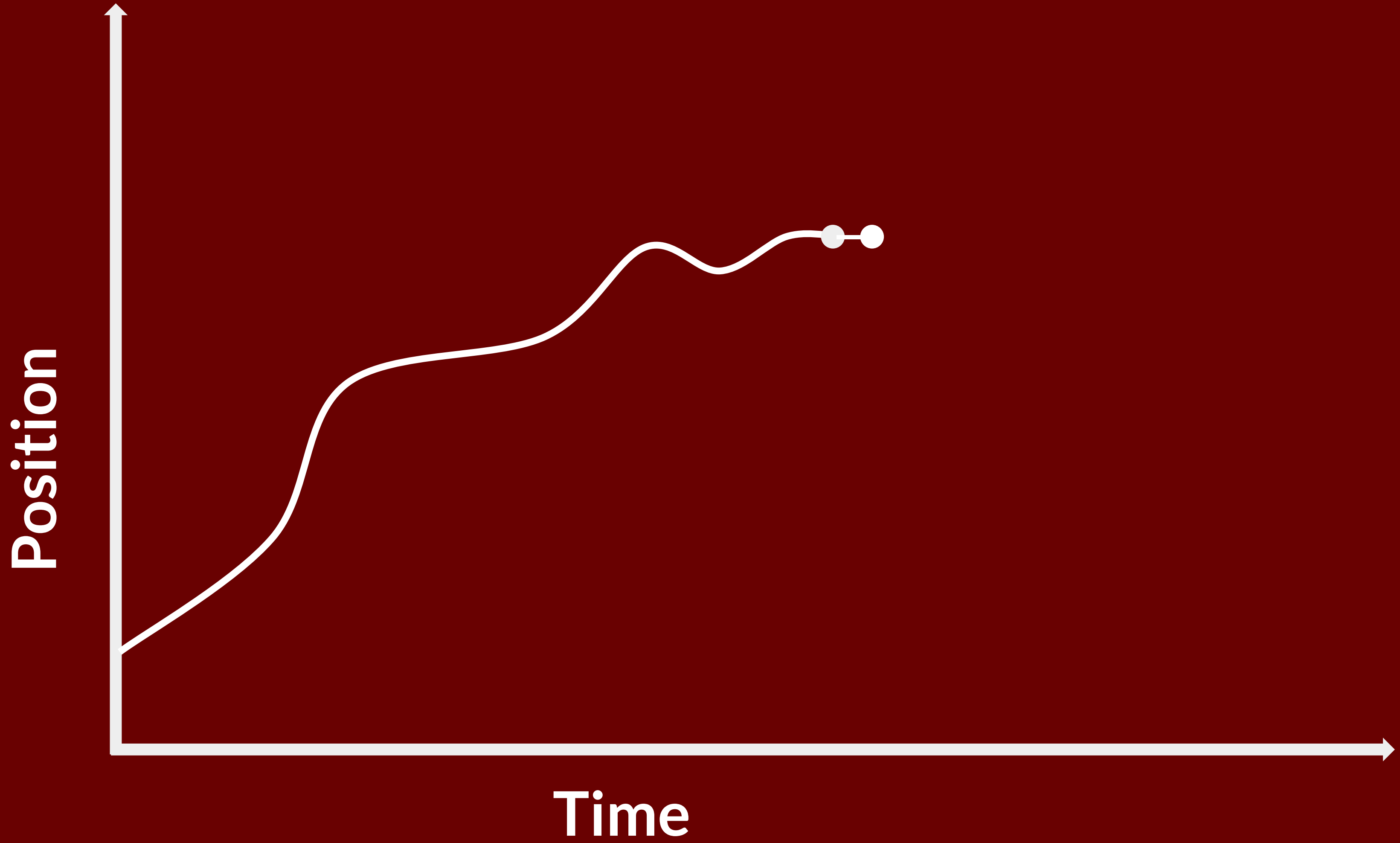
We need uncertainties for each measurement modeled as Gaussians to then multiply distributions together.



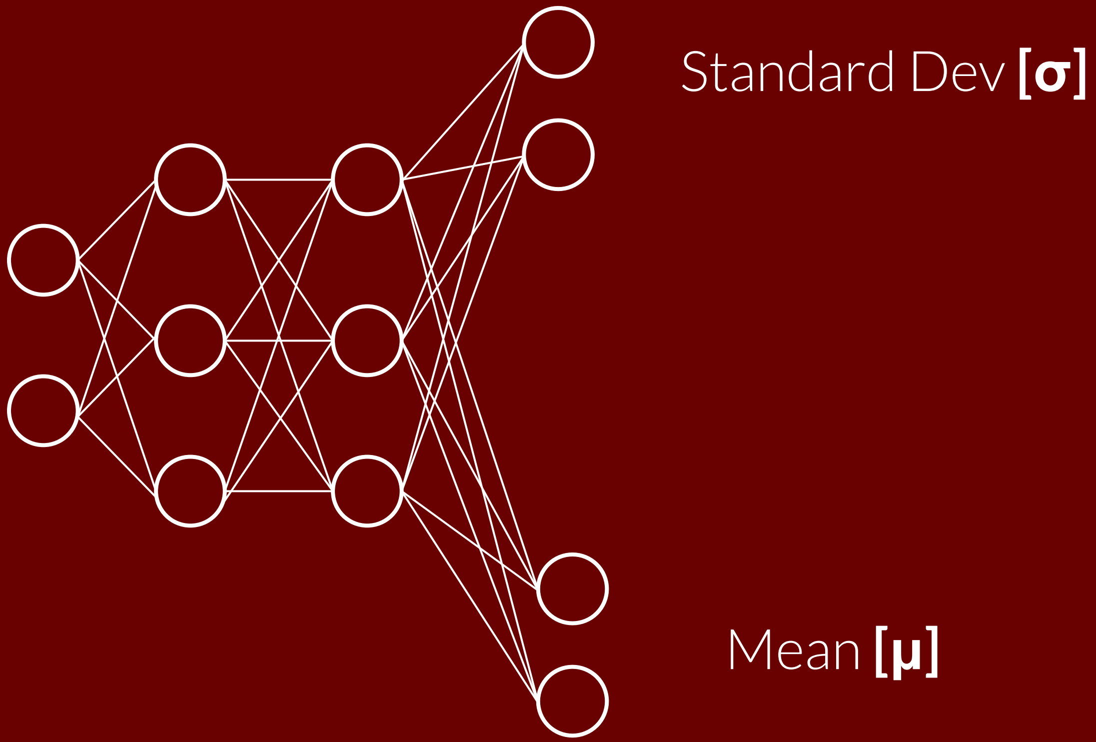
Position
Time


Position
Time


Position
Time


Position
Time

How do we naturally produce uncertainties with Neural Nets?


Standard Dev **[σ]**
Mean **[μ]**

Standard Dev **[σ]**

Maximize $P(y|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{1}{2\sigma^2}(-(y-\mu)^2)}$

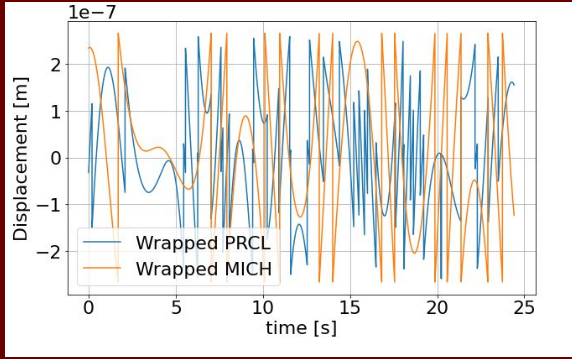
Mean **[μ]**

y Is the target position

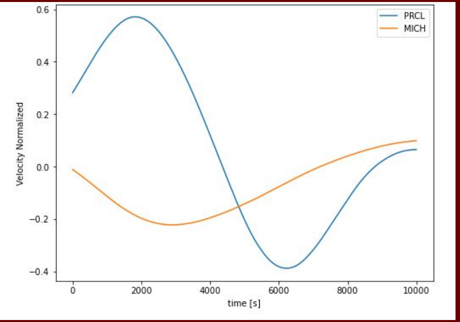
How do we naturally predict **velocities**?

Velocity

Position



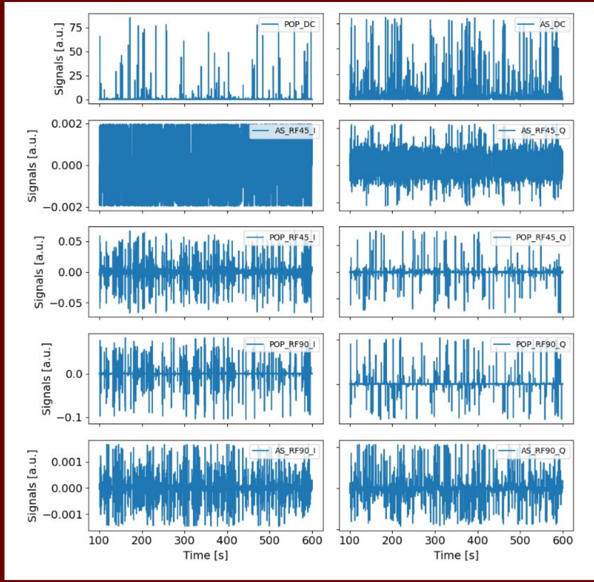
Velocity



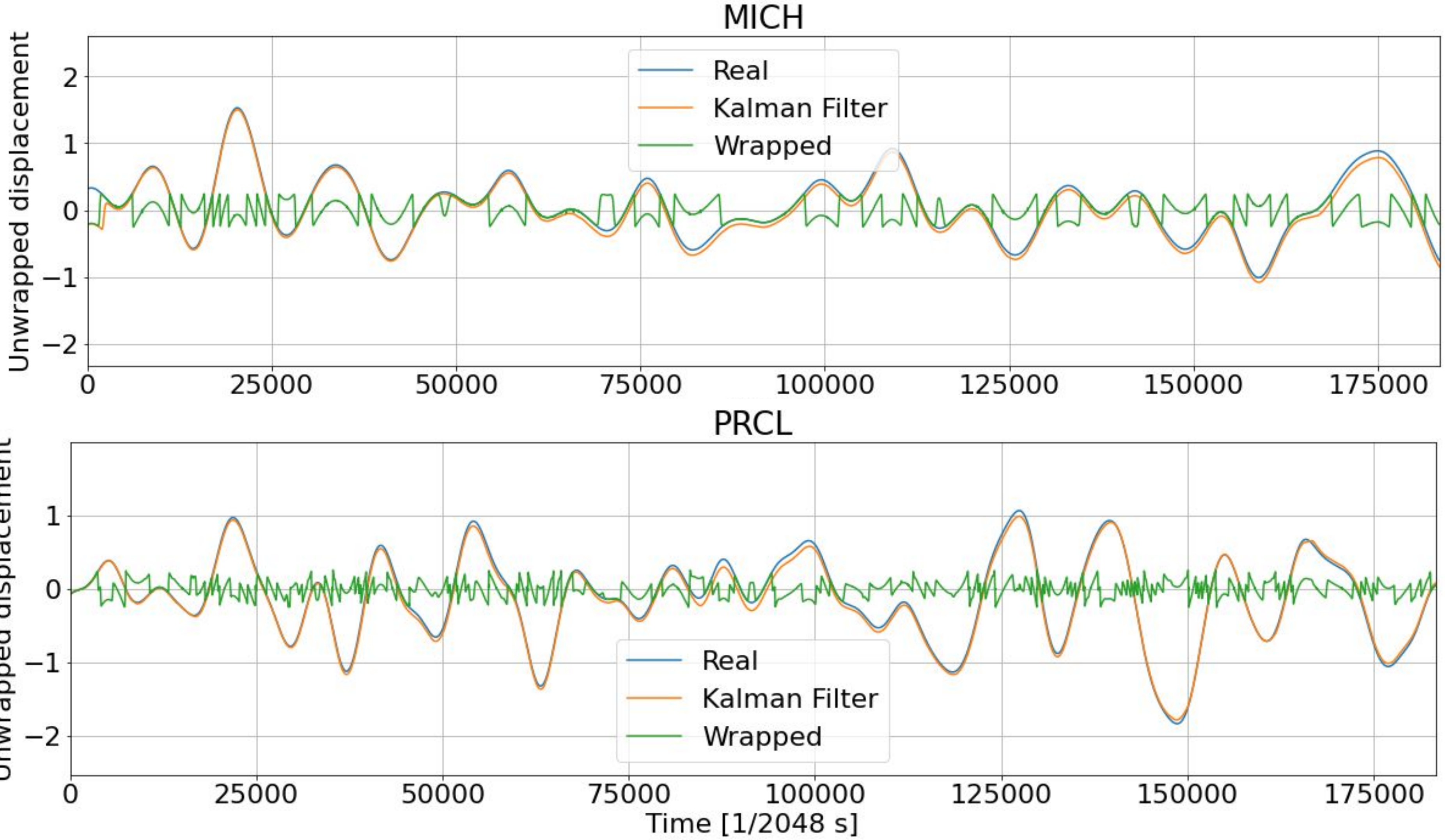
ML Model

ML Model

Signals



Okay now let’s combine everything together!



MICH
Real
Kalman Filter
Wrapped
Unwrapped displacement
2
1
0
−1
−2
0
25000
50000
75000
100000
125000
150000
175000
PRCL
Real
Kalman Filter
Wrapped
Unwrapped displacement
1
0
−1
−2
0
25000
50000
75000
100000
125000
150000
175000
Time [1/2048 s]

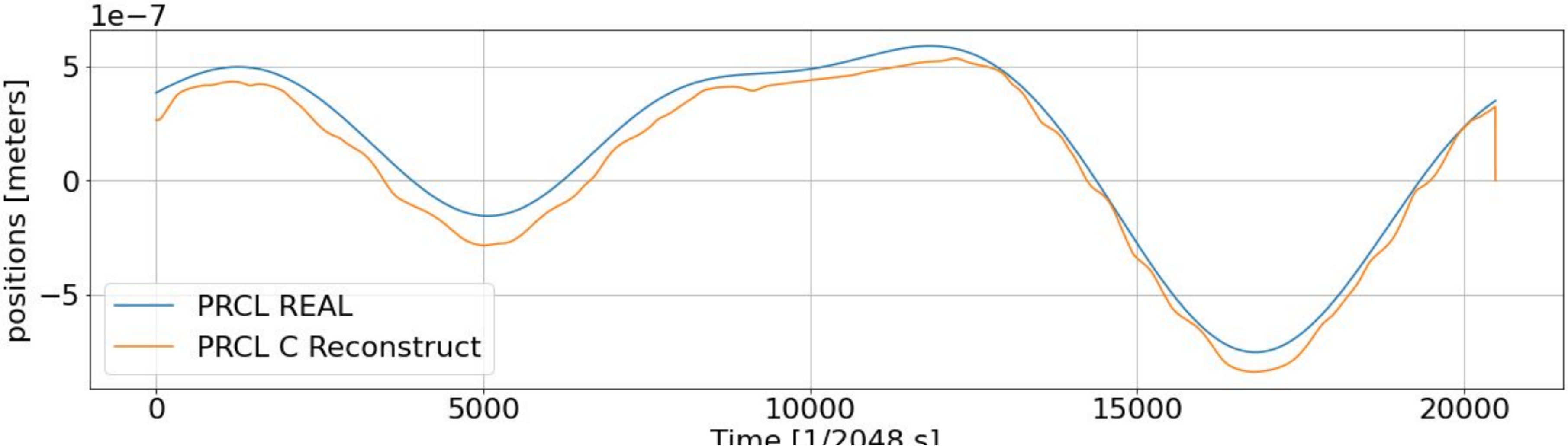
This took **9 hours** to run. We need to shrink this down to **90 sec…**

1. Make models 200x smaller.

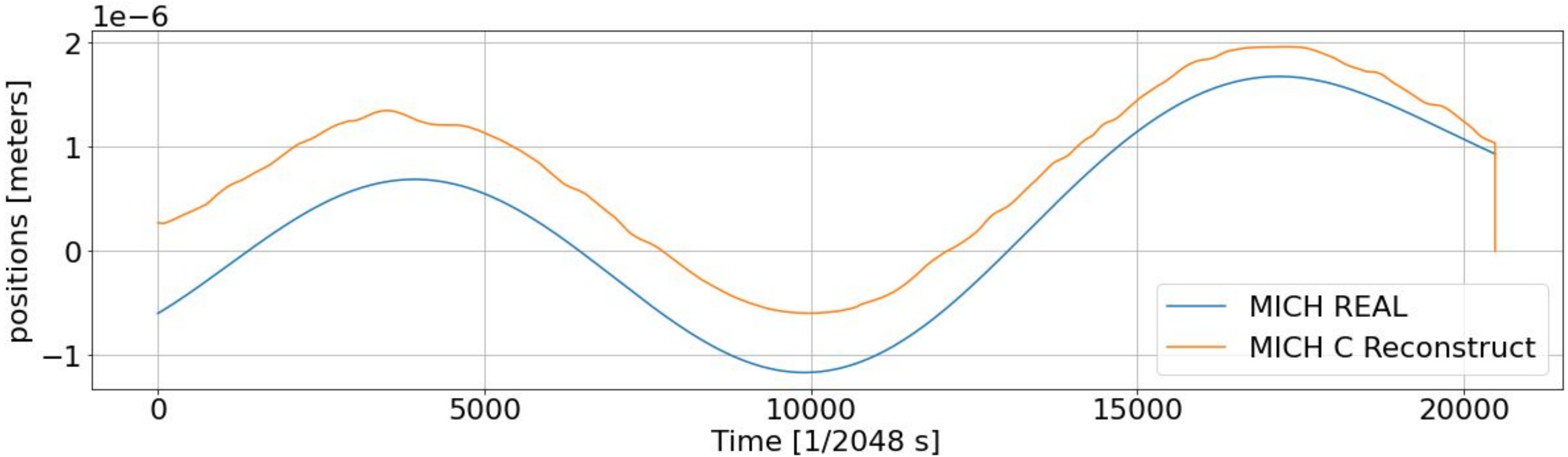
1. Make models 200x smaller.
2. Convert everything into pure C

1. Make models 200x smaller.
2. Convert everything into pure C
3. Cry.

1. Make models 200x smaller.
2. Convert everything into pure C
3. Cry.
4. Pray everything works



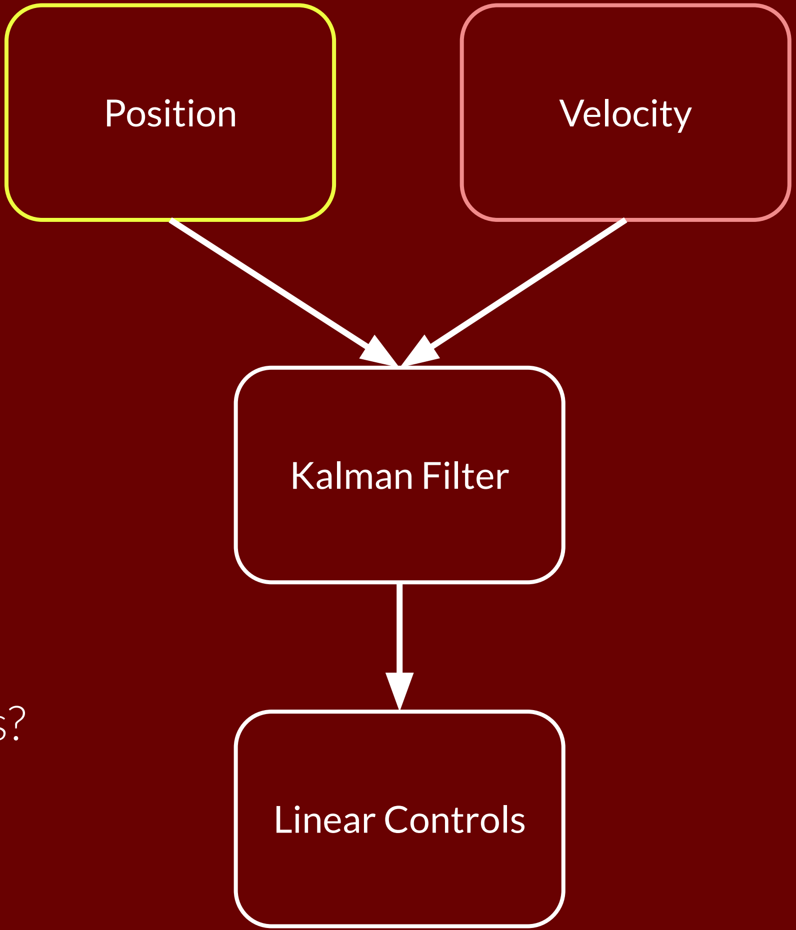
1e−7
positions [meters]
PRCL REAL
PRCL C Reconstruct
Time [1/2048 s]



1e−6
positions [meters]
MICH REAL
MICH C Reconstruct
Time [1/2048 s]

Interpretable

- Lots of physics baked in
- Uncertainties!!
- Well understood controls
- Well understood Kalman Filter
- Did I mention we have uncertainties?

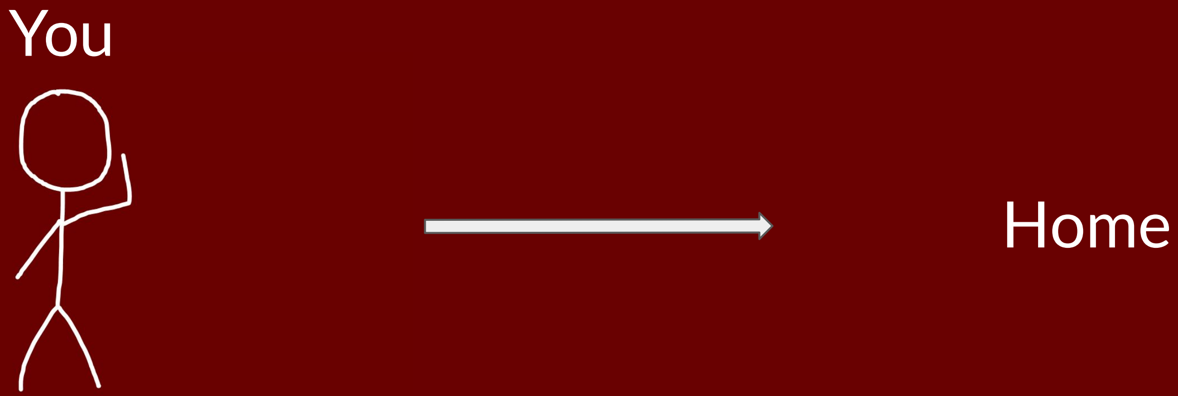


What if we disregard all of this.

What if we disregard all of this. And just learn *everything*? Is this even possible?

What if we disregard all of this. And just learn *everything*? Is this even possible?

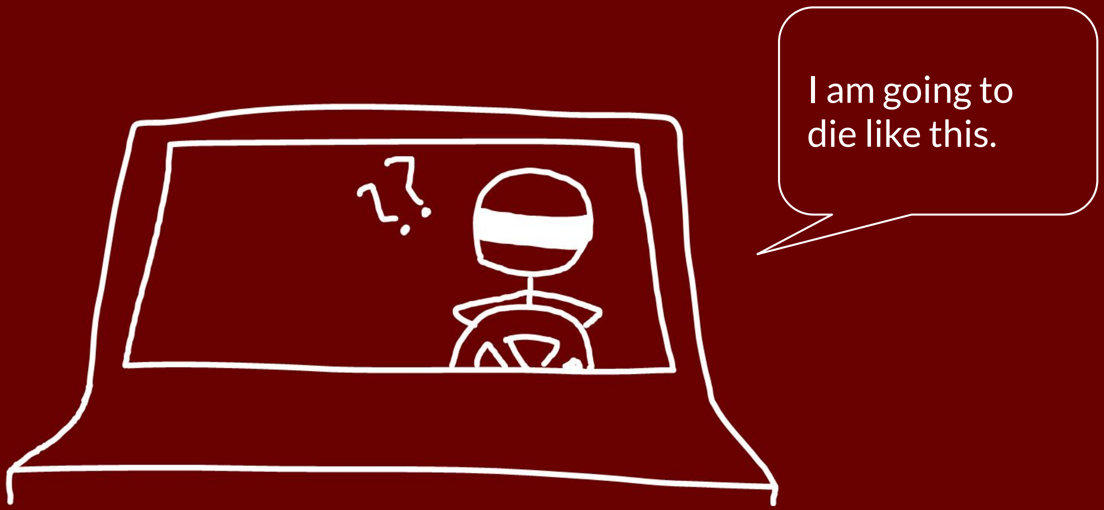
Warning: Exploratory section not super well documented but cool nonetheless :)

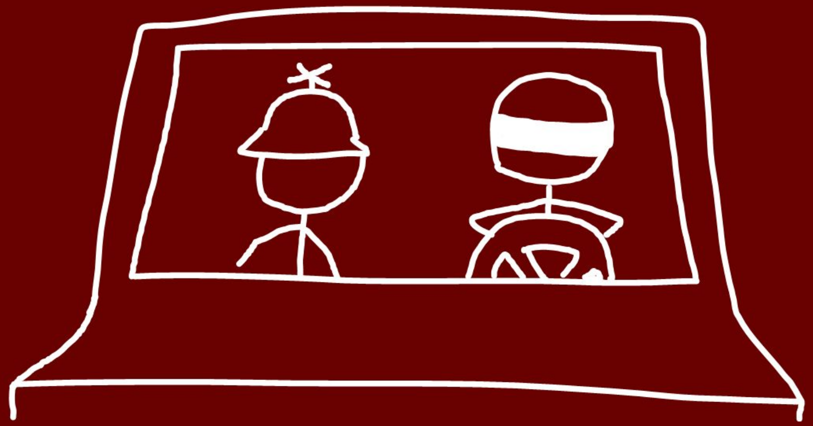

You
Home

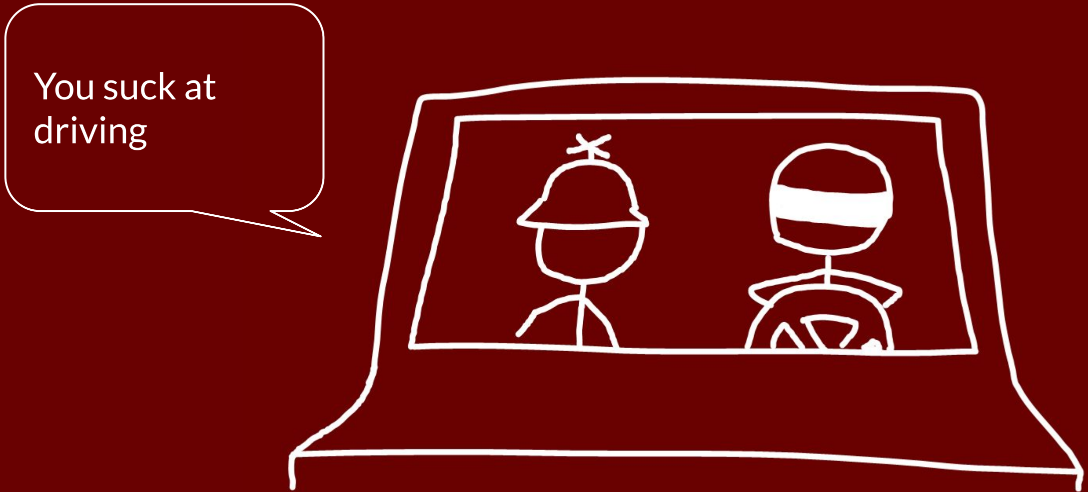

Let's go home.

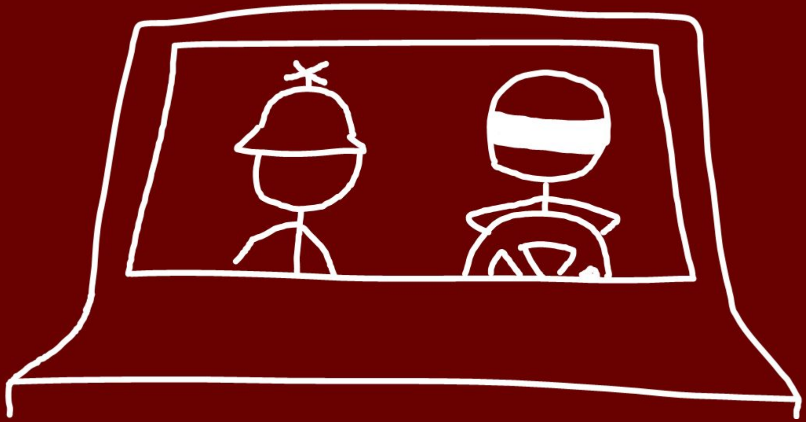

I don’t know
where to go

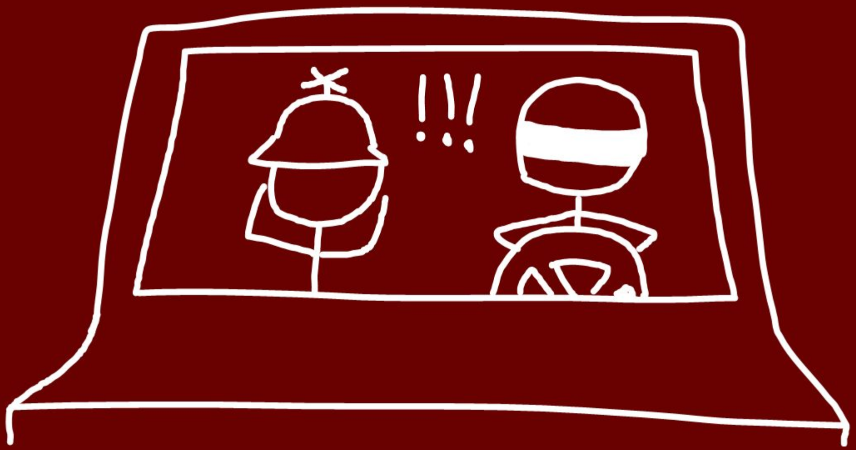

I also don’t know how to drive.


I am going to die like this.



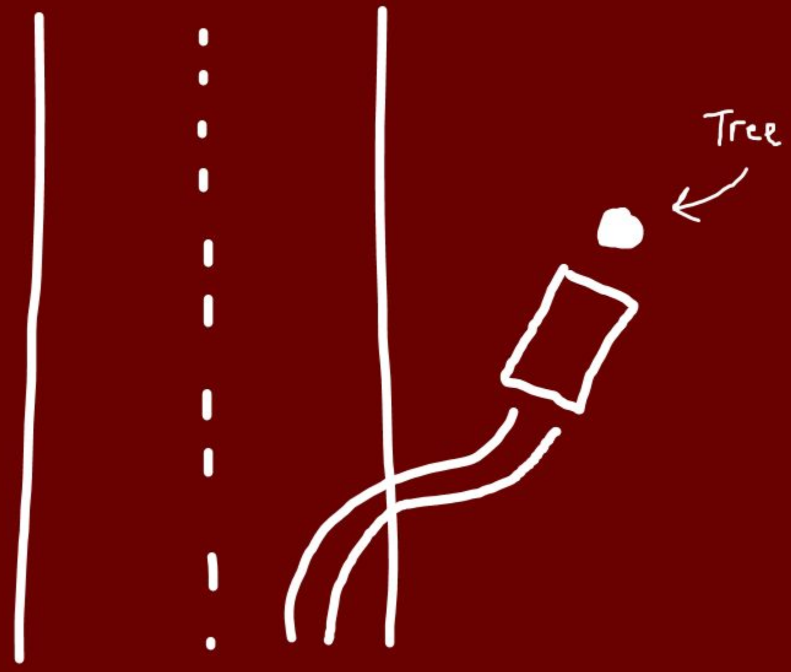

You suck at driving



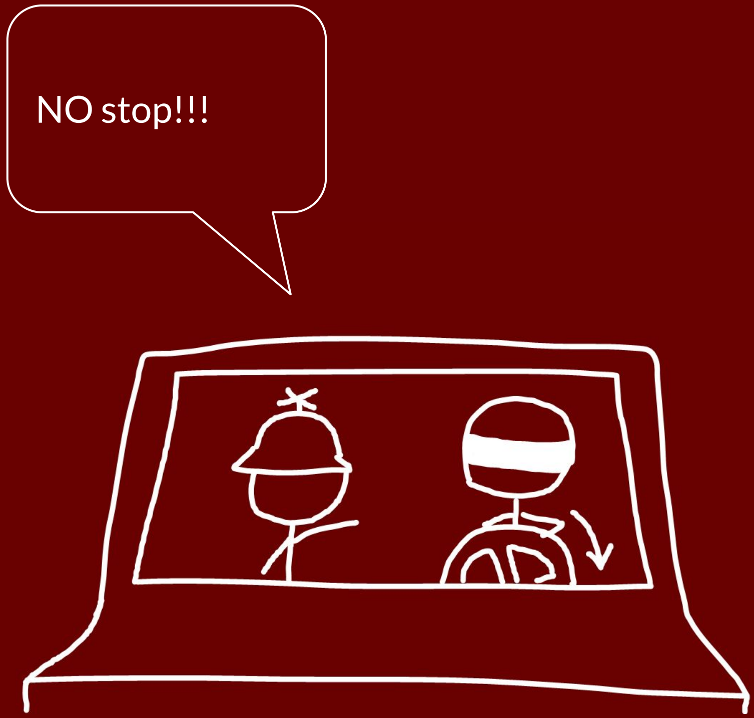


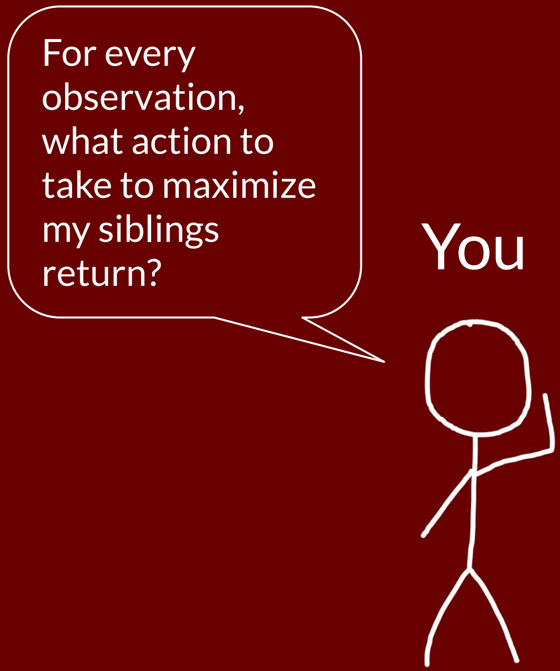

WATCH OUT THERES a tree!!!

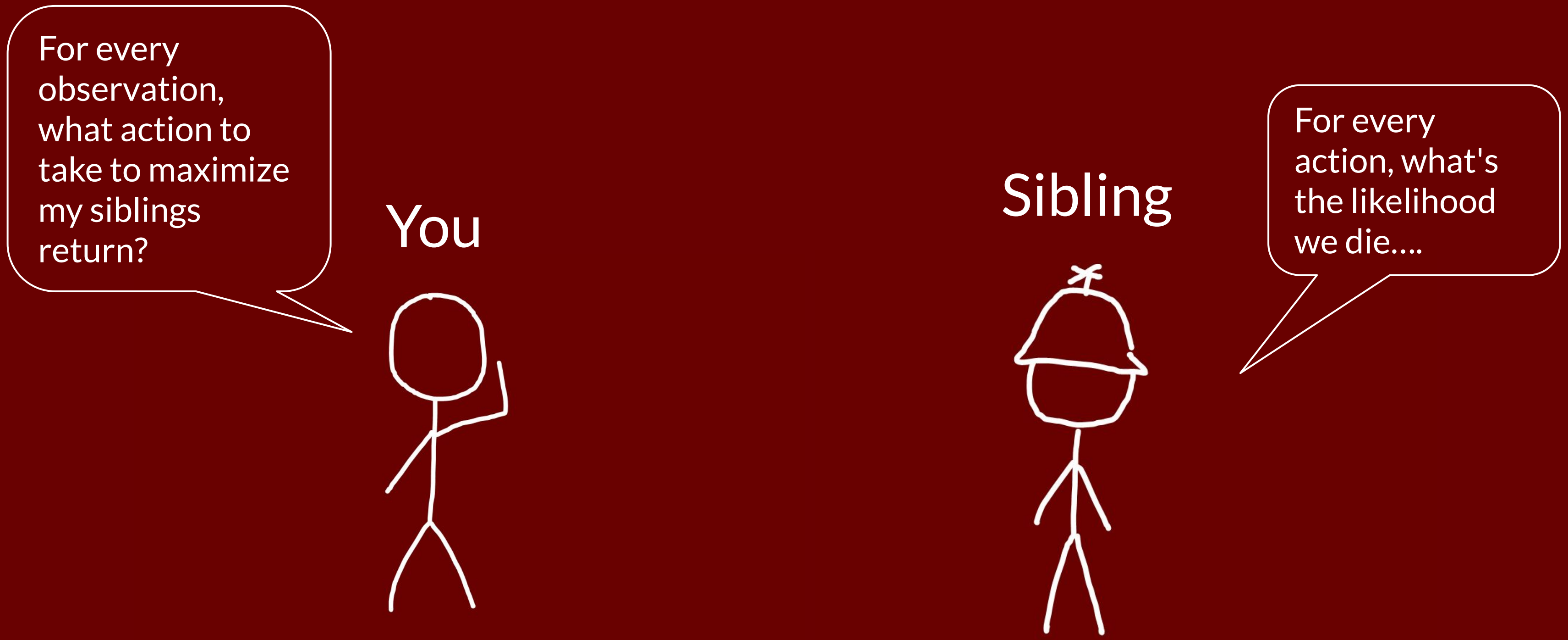

WATCH OUT THERES a tree!!!
!!!
A what???

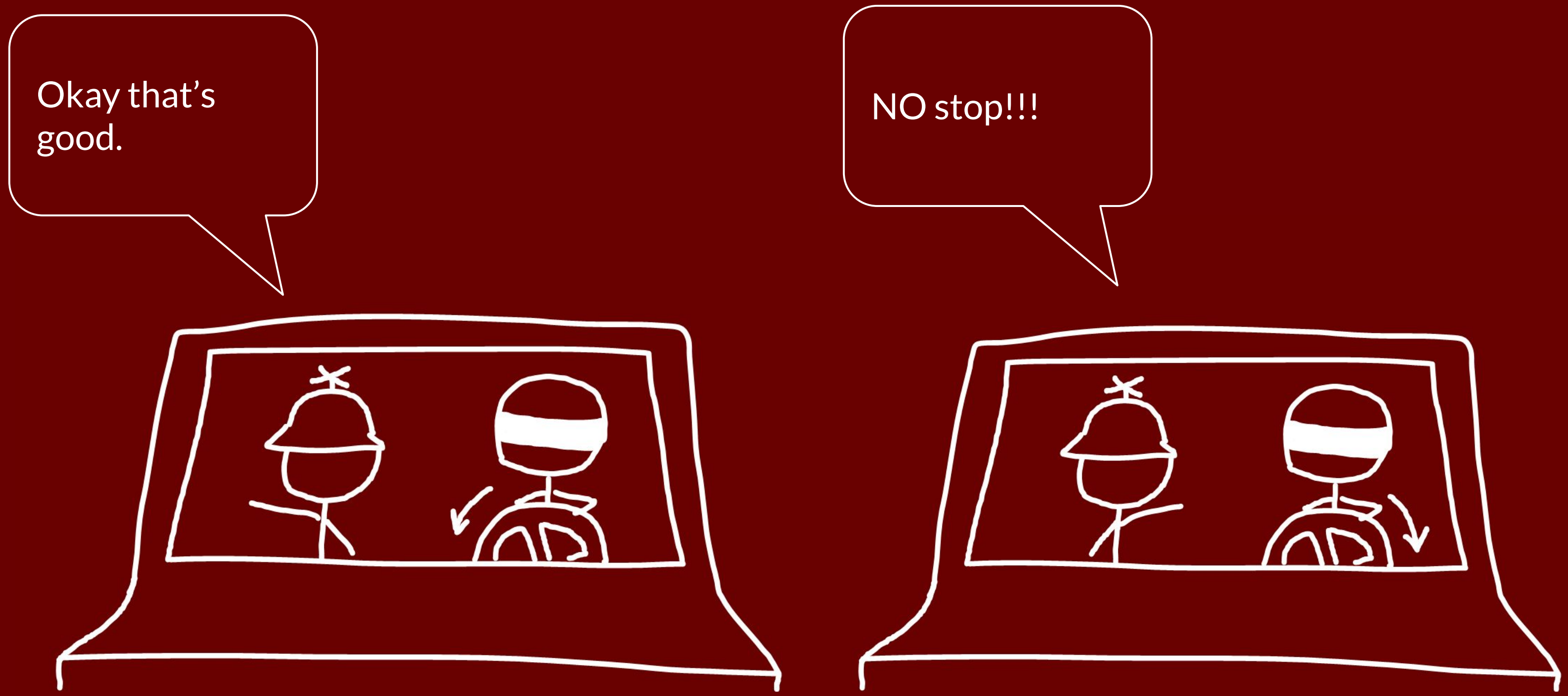

Tree


Okay that’s good.


NO stop!!!

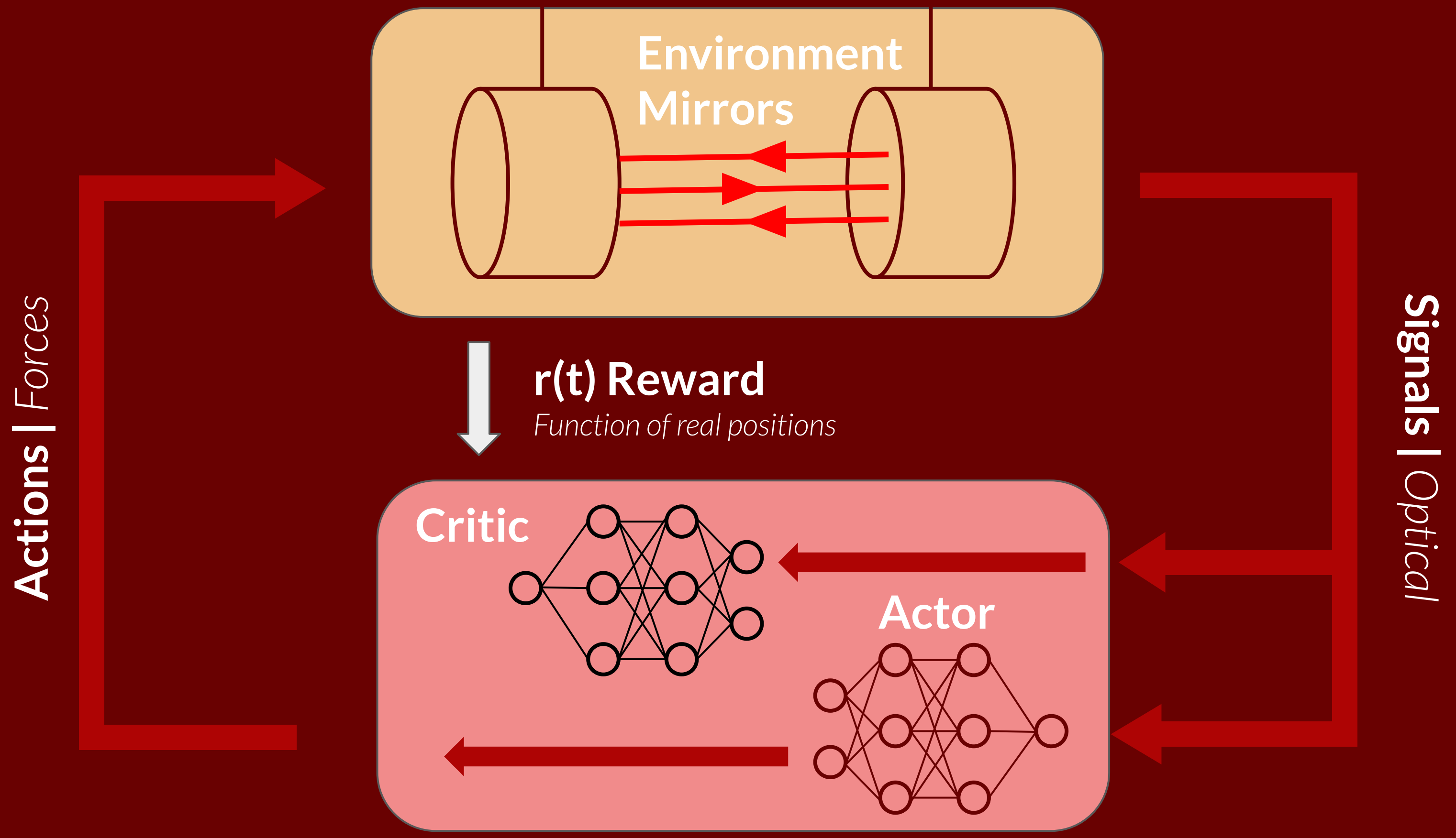

For every observation, what action to take to maximize my siblings return?
You

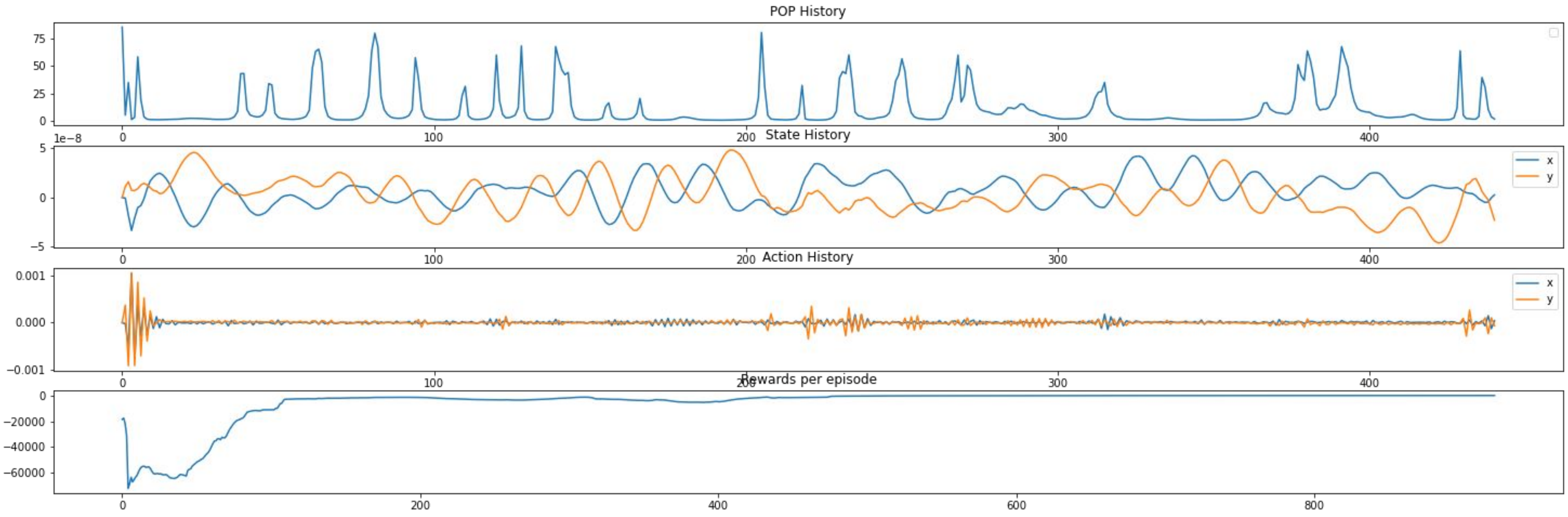

For every observation, what action to take to maximize my siblings return?
You
Sibling
For every action, what's the likelihood we die....


Okay that's good.
NO stop!!!

Actor and **Critic** setup.

Actor and **Critic** setup. ***Deep Deterministic Policy Gradient*** (DDPG)


Environment
Mirrors
r(t) Reward
Function of real positions
Critic
Actor
Actions | Forces
Signals | Optical



POP History
State History
x
y
1e−8
Action History
Rewards per episode

Summary

State-estimator

- Developed probabilistic ML model for Position, Velocity est. and Kalman Filter
- Developed fast C version of models

Reinforcement Learning

- Demonstrated under certain situations DDPG could potentially solve control problem

Next Steps

- Apply our technique to locking the mirrors
- Exploring further our RL experiments!

Thank you!

Big thanks to Gabriele for mentoring me through the weeks and the SURF program for supporting this work!

Caltech LIGO

Questions?

Personal Website: Peterma.ca

Email: Peterxy.ma@gmail.com

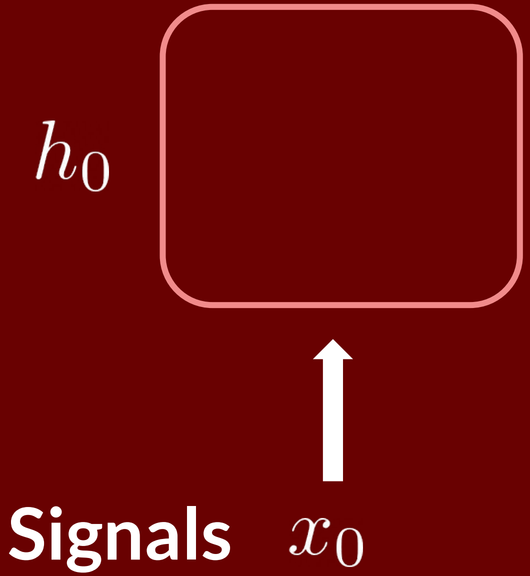
Twitter: @peterma02

Github: @Petchma

Caltech LIGO UNIVERSITY OF TORONTO 遠馬印驤

Predicting Wrapped Positions

Deep Learning Model



Predicting Wrapped Positions

Deep Learning Model

h_0

$$h_0W + Ux_0 + b$$

Signals x_0

Predicting Wrapped Positions

Deep Learning Model

h_0

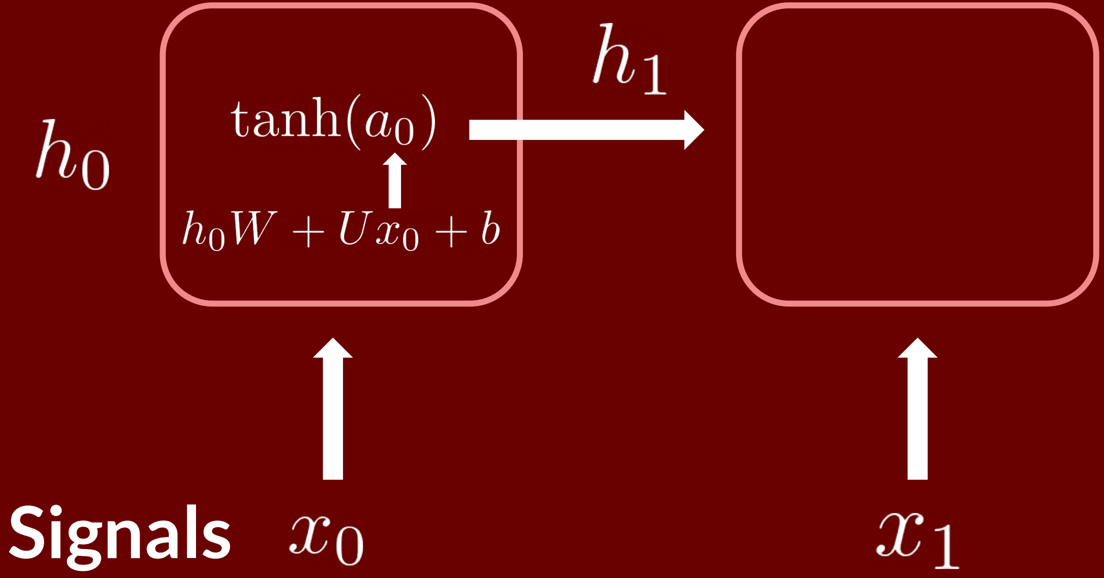
$\tanh(a_0)$

$h_0 W + U x_0 + b$

Signals x_0

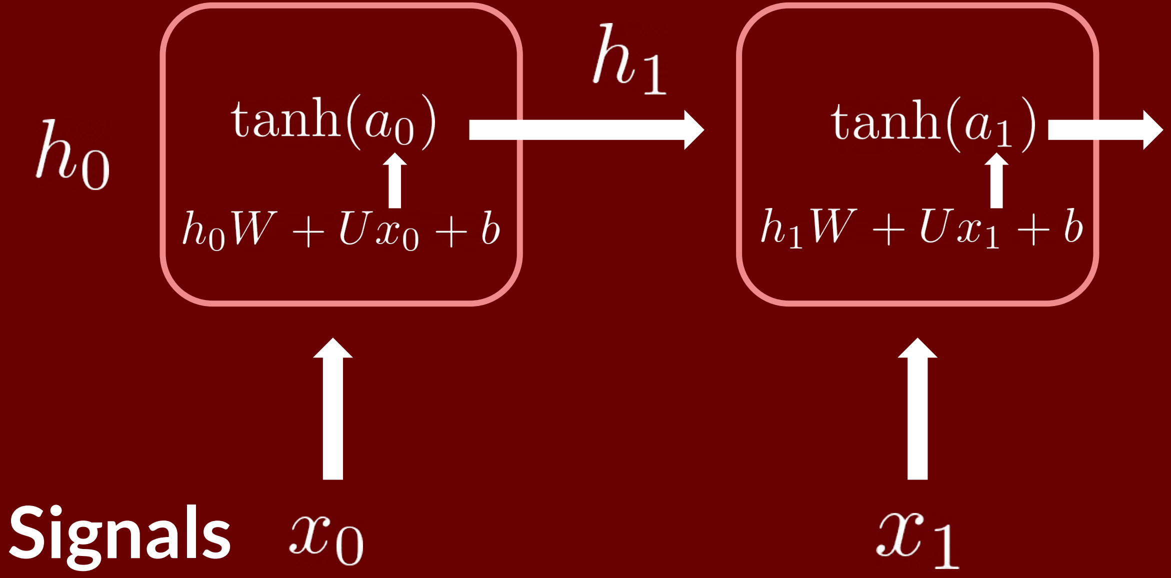
Predicting Wrapped Positions

Deep Learning Model



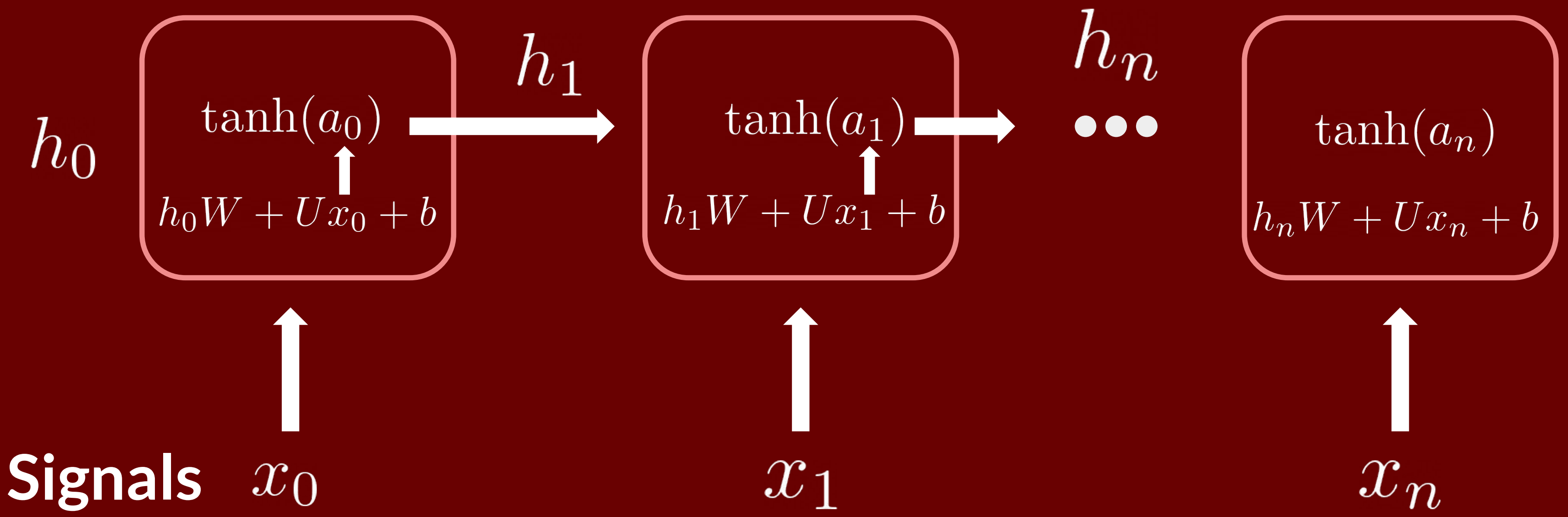
Predicting Wrapped Positions

Deep Learning Model



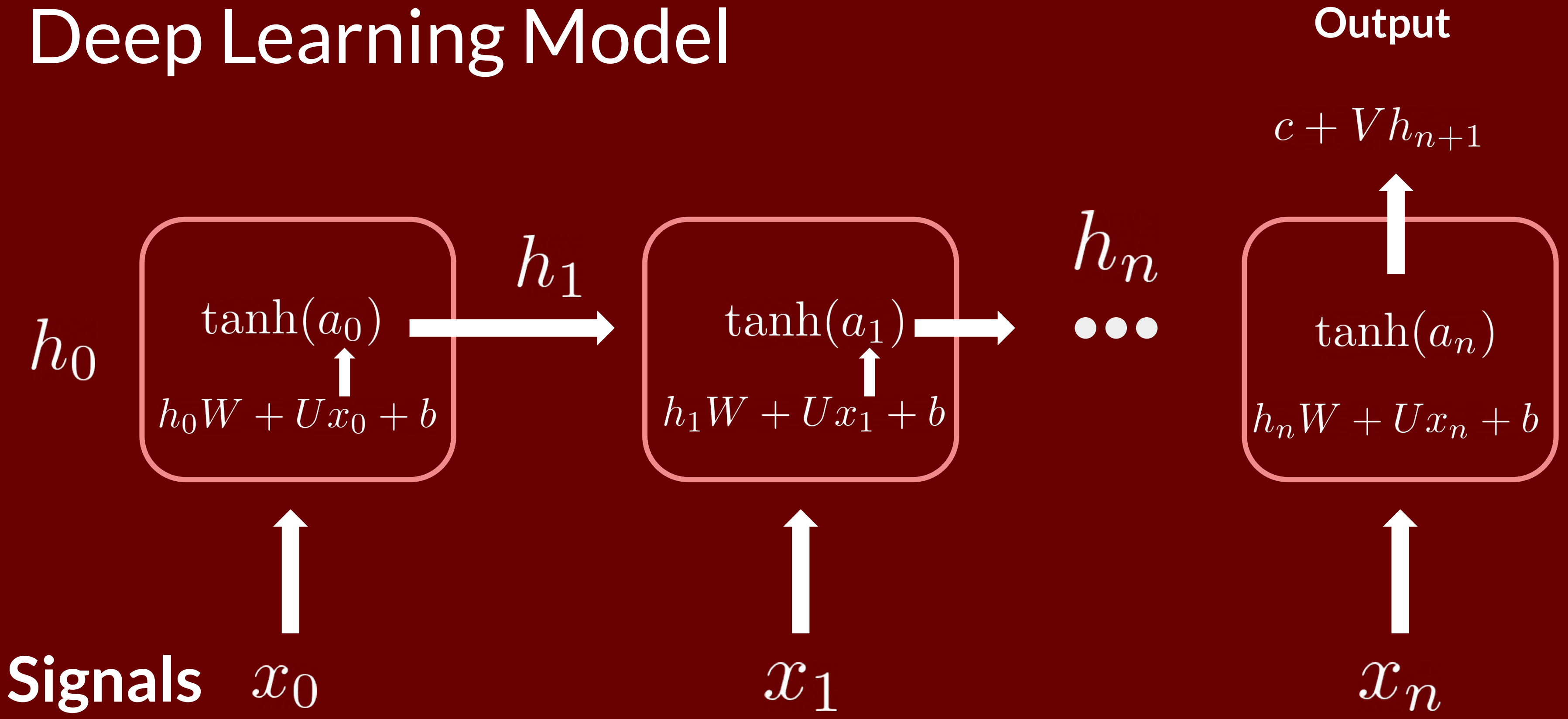
Predicting Wrapped Positions

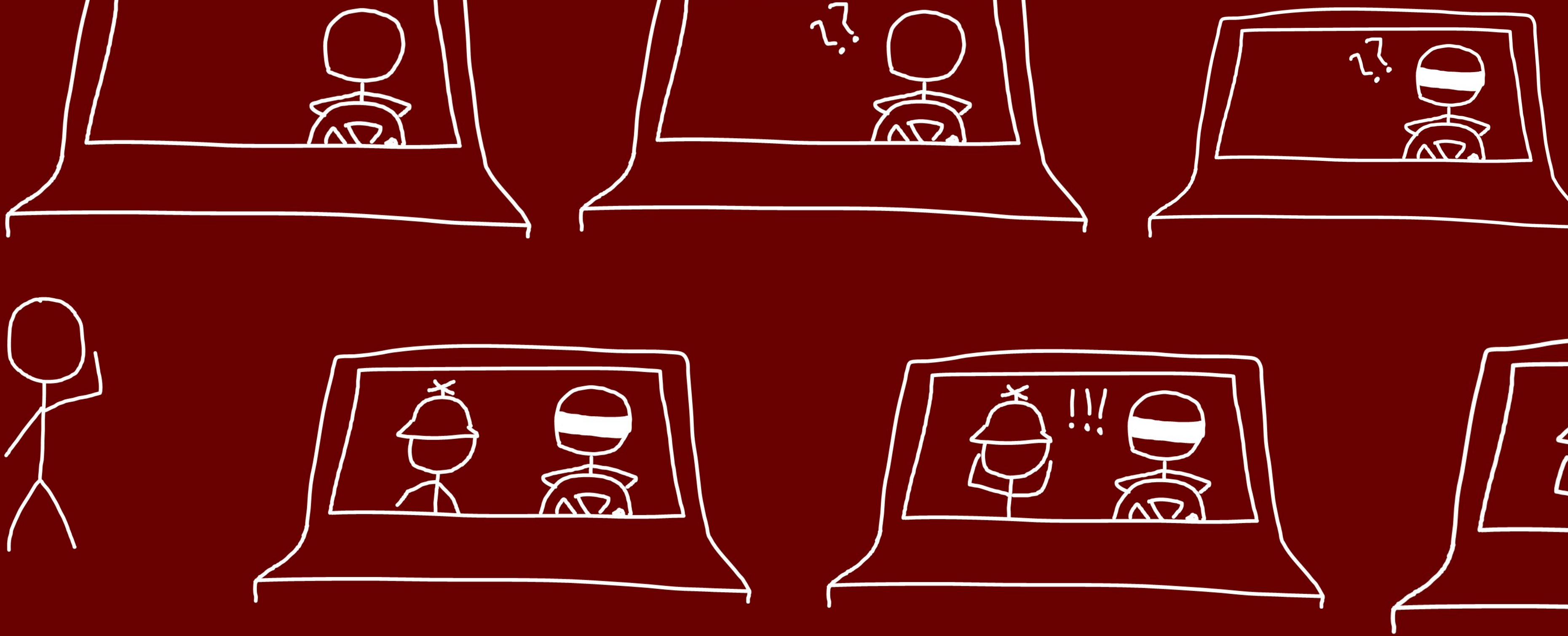
Deep Learning Model



Predicting Wrapped Positions

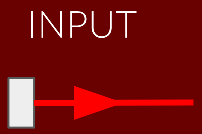
Deep Learning Model





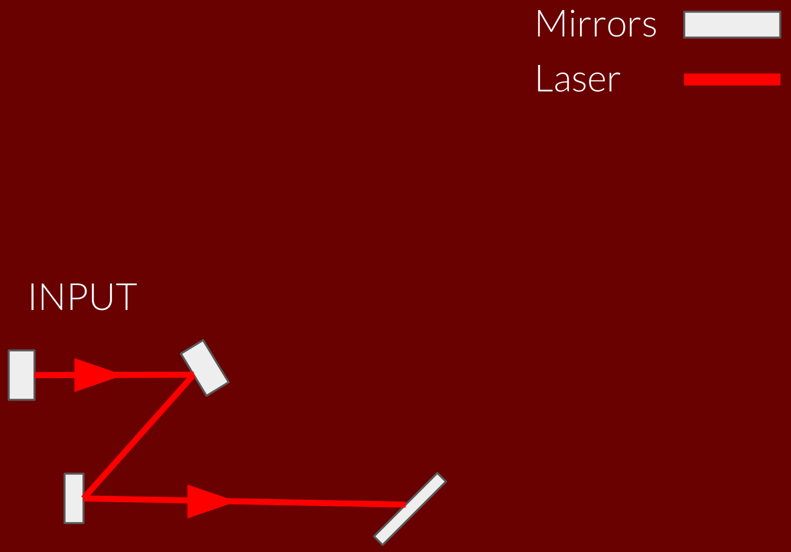
Introduction

LIGO Mirrors



Introduction

LIGO Mirrors



Introduction

LIGO Mirrors