State Estimation, Kalman Filters and Vehicle models

Ecurie Aix.

Small introduction

© FSG Andrae

DV Group Ecurie Aix

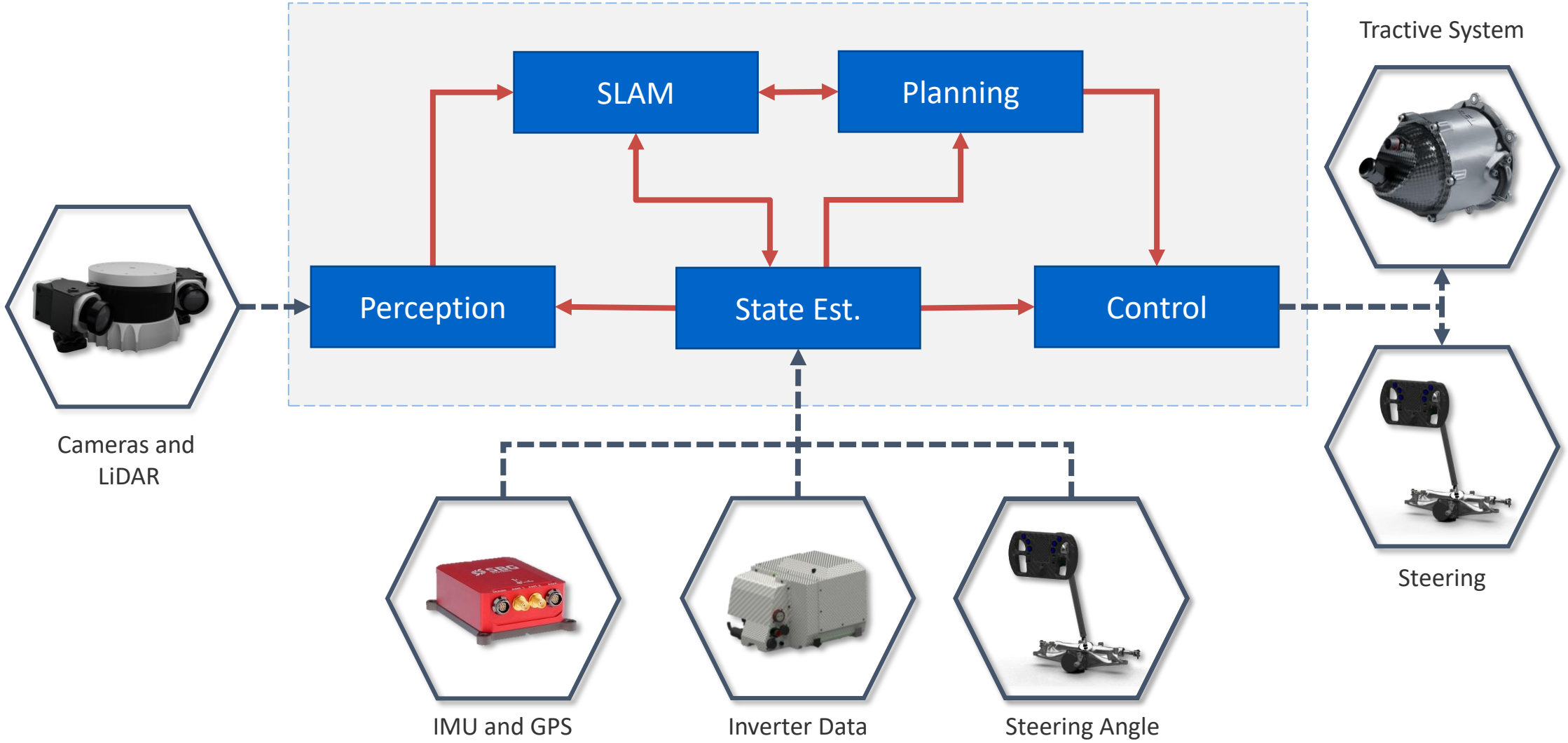
What I did.

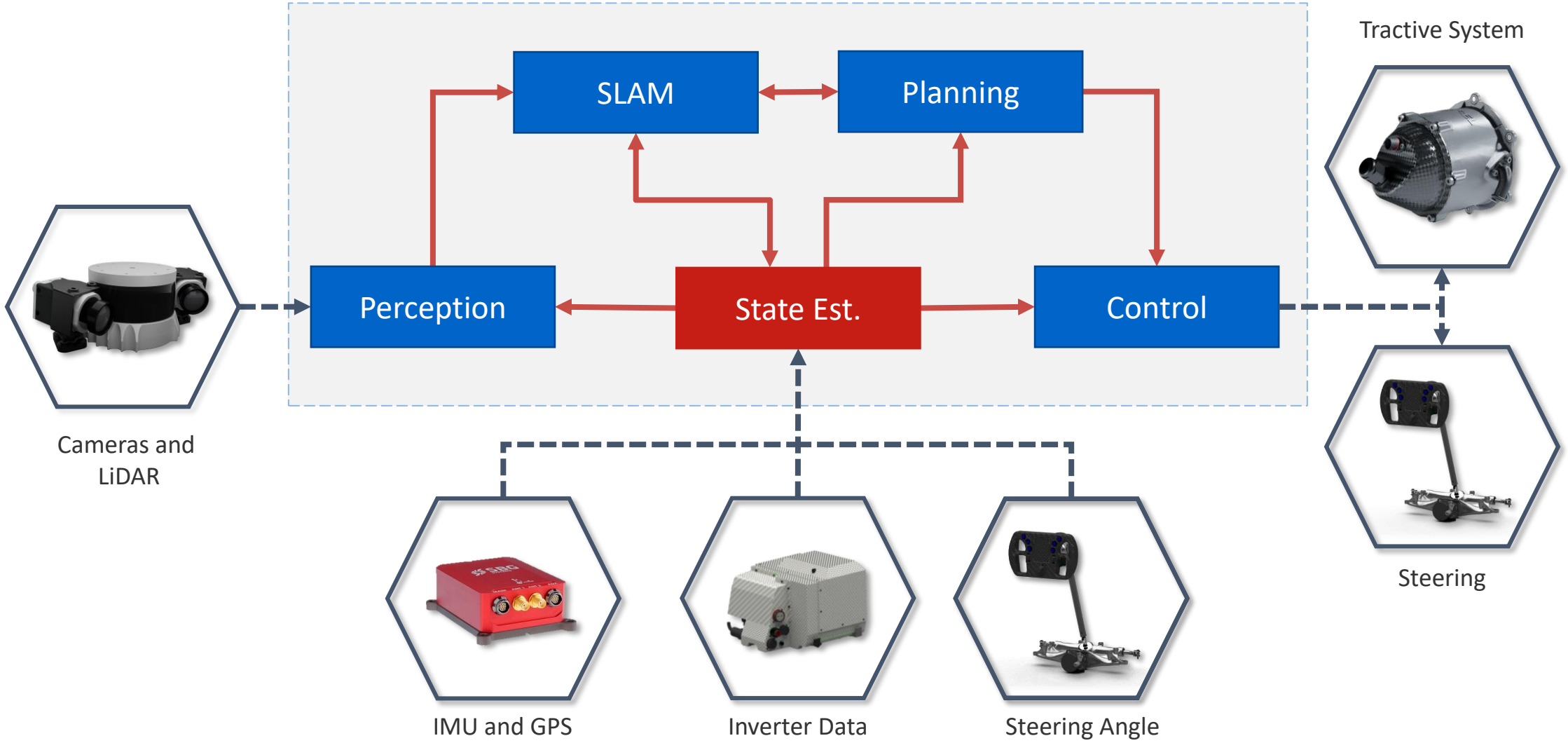
- Started October 2023
- State Estimation
  - Robot Localization
  - Optimizer (FUSE)
  - Implementing UKF
- Ongoing Bachelor Thesis

State Estimation.

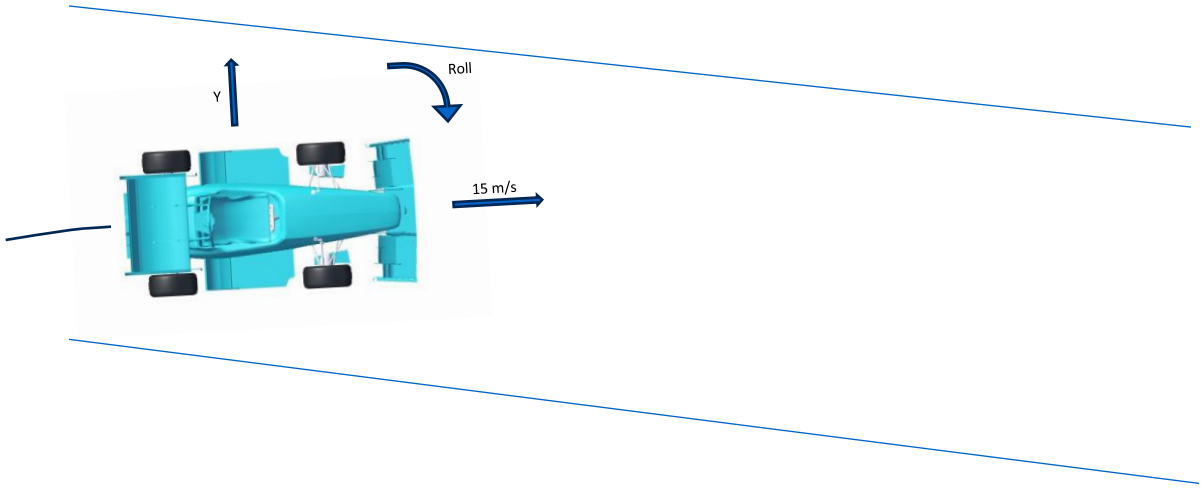
Is it worth it?

© FSG Andrae


SLAM
Planning
Perception
State Est.
Control
Tractive System
Cameras and LiDAR
Steering
IMU and GPS
Inverter Data
Steering Angle

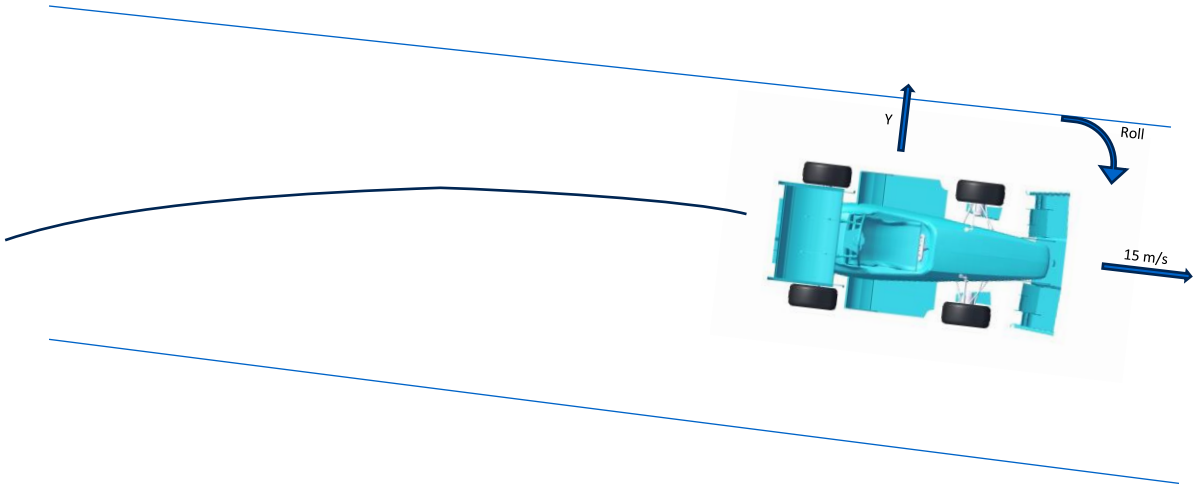

SLAM
Planning
Perception
State Est.
Control
Tractive System
Cameras and LiDAR
Steering
IMU and GPS
Inverter Data
Steering Angle

IS IT NEEDED?



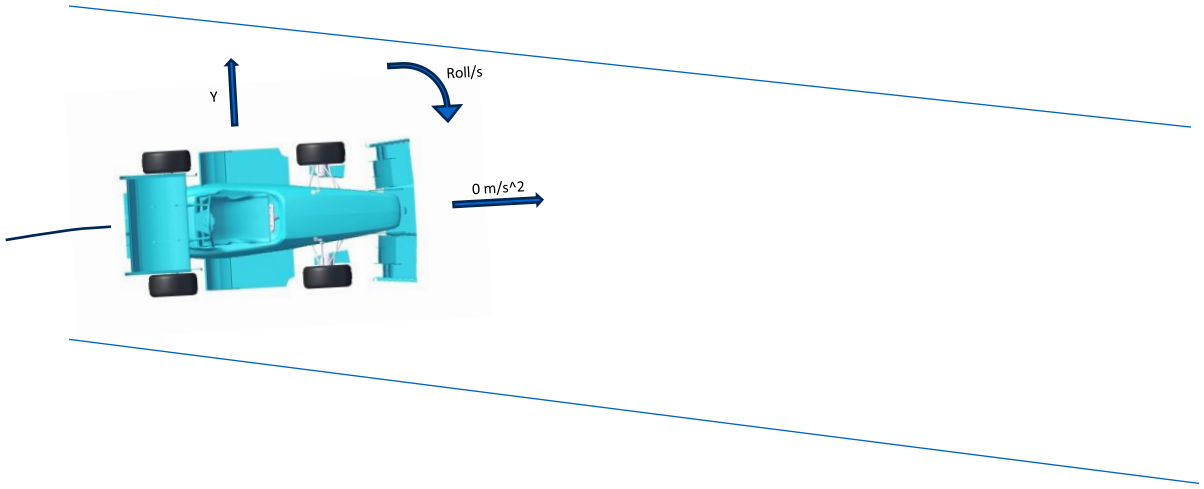
- All information at every time

IS IT NEEDED?



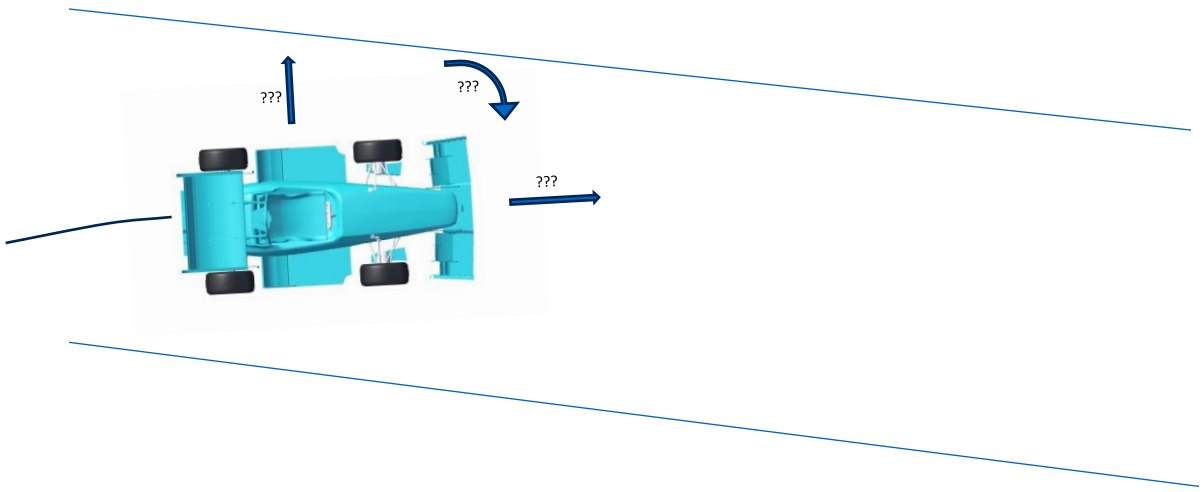
- All information at every time

IS IT NEEDED?



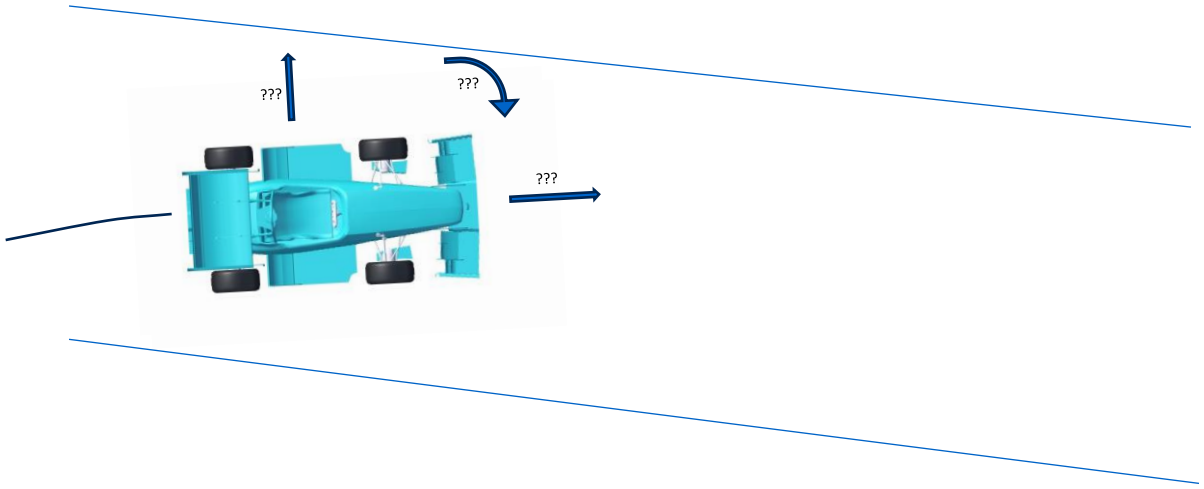
- SLAM 10 Hz
- IMU 180Hz
  - Only accel and yaw rate

IS IT NEEDED?



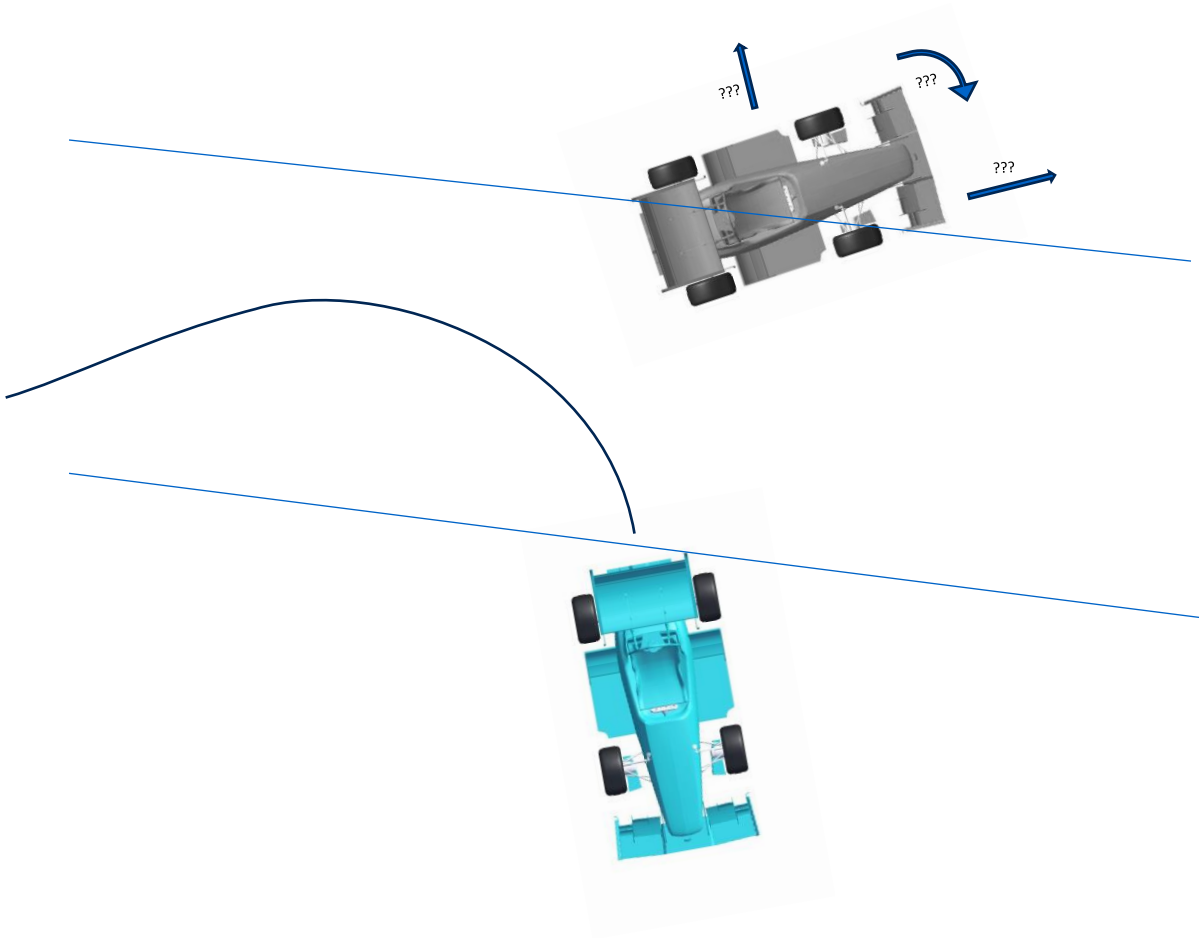
- SLAM 10 Hz
- IMU 180Hz
  - Only accel and yaw rate

IS IT NEEDED?



- SLAM 10 Hz
- IMU 180Hz
  - Only accel and yaw rate

IS IT NEEDED?



- SLAM 10 Hz
- IMU 180Hz
  - Only accel and yaw rate

Filter.

Because nothing is perfect

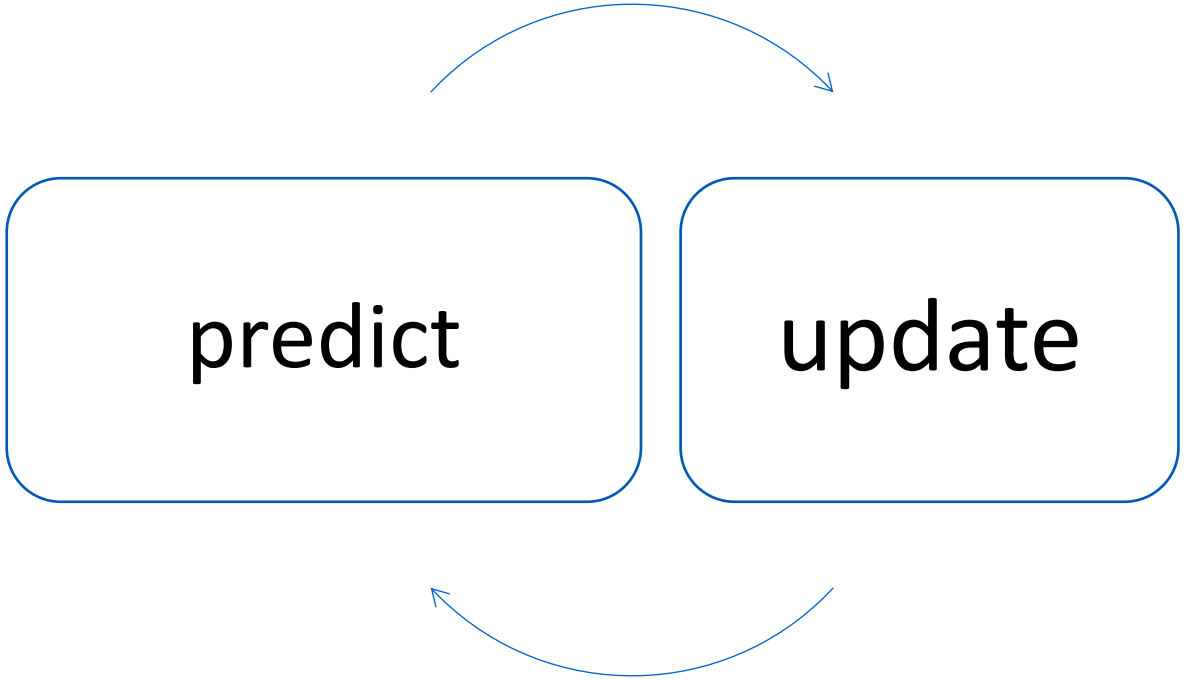
© FSG Mosch

Integrating.

Most intuitive

1. Take last sensor information
2. Integrate it over time
3. Get new information
4. Overwrite that state

goto 1.

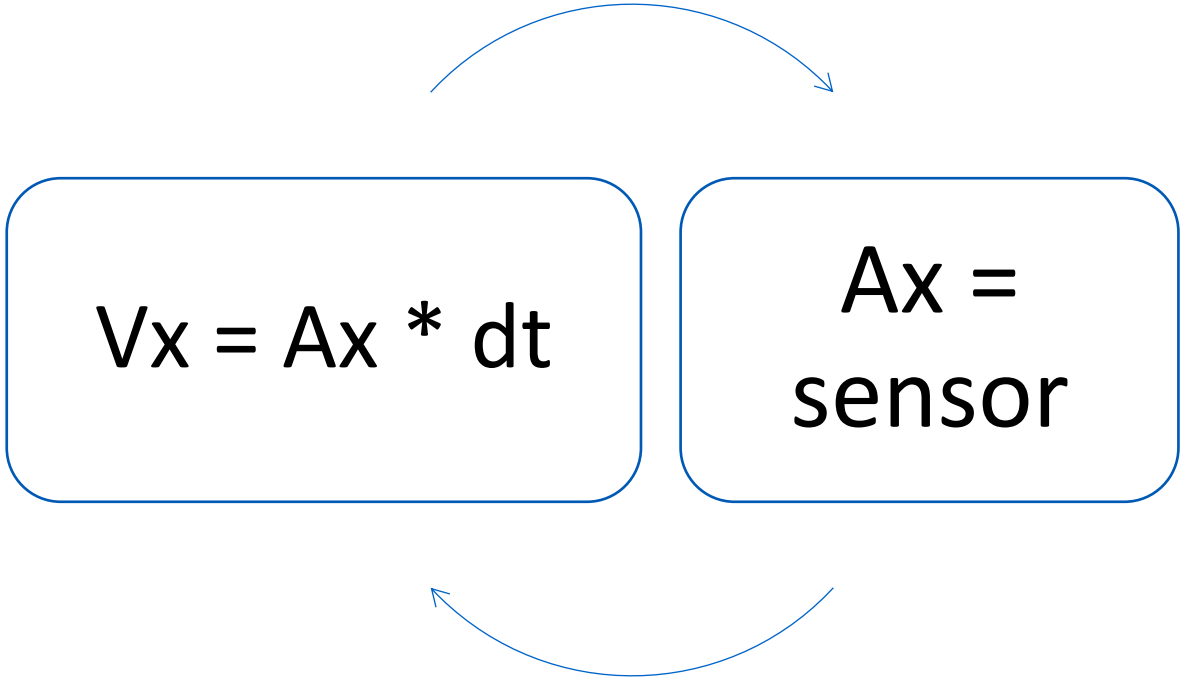


Integrating.

Most intuitive

1. Take last sensor information
2. Integrate it over time
3. Get new information
4. Overwrite that state

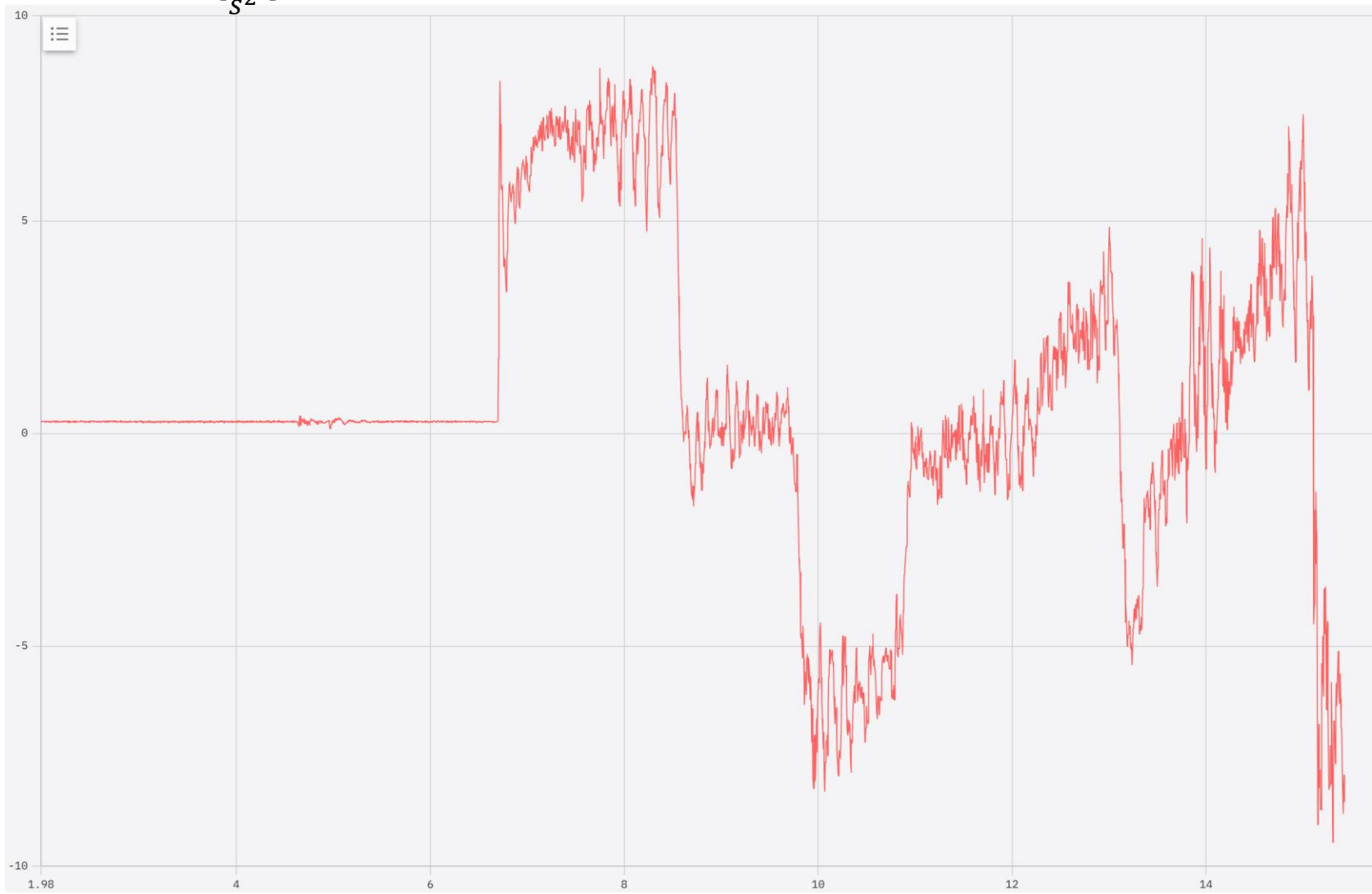
goto 1.



Why is it bad?

- Sensor noise
  - Problem short term

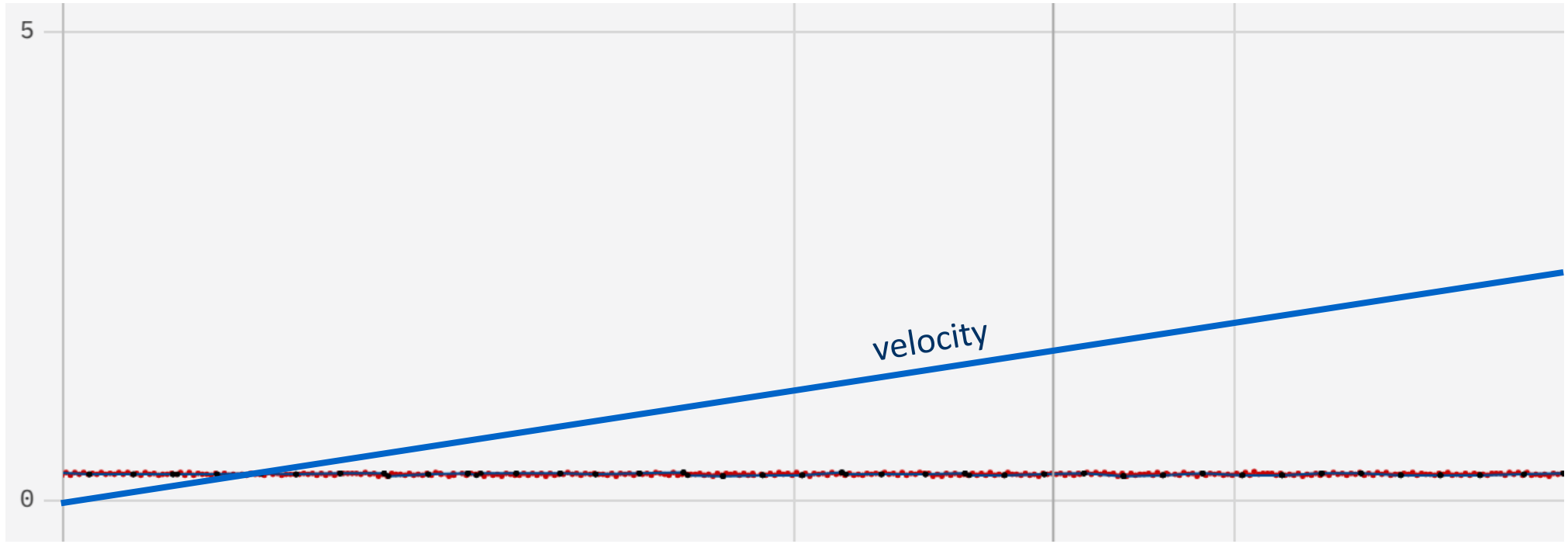
Acceleration [$\frac{m}{s^2}$]



ROS time [s]

Why is it bad?

- Sensor noise
- Sensor drift
  - Problems long term



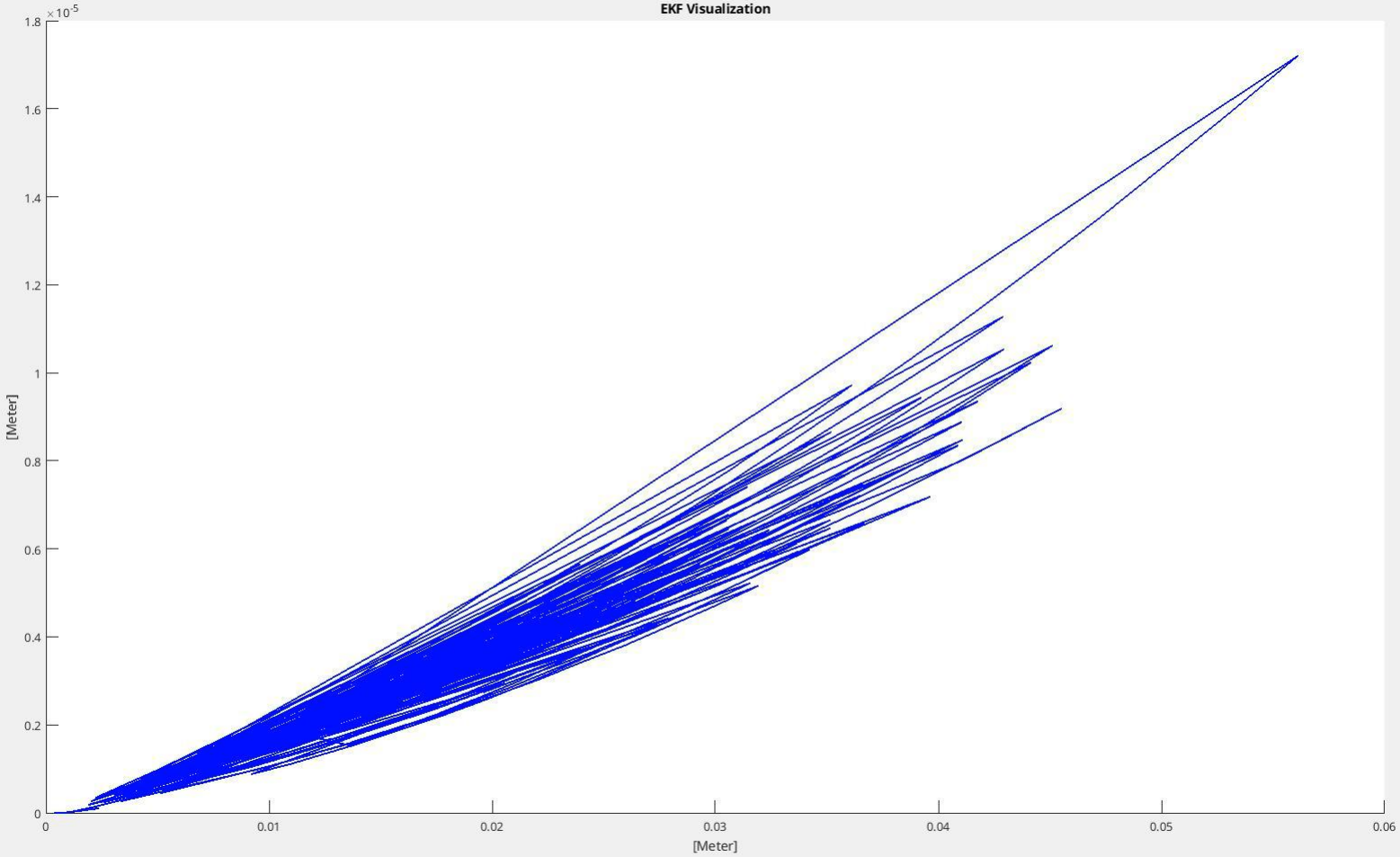
Why is it bad?

- Sensor noise
- Sensor drift

- More and accurate sensors are expensive.
- But there is no perfect sensor!

- Plus Vehicle limits
  - Space, Weight, Calculations, CAN, etc.

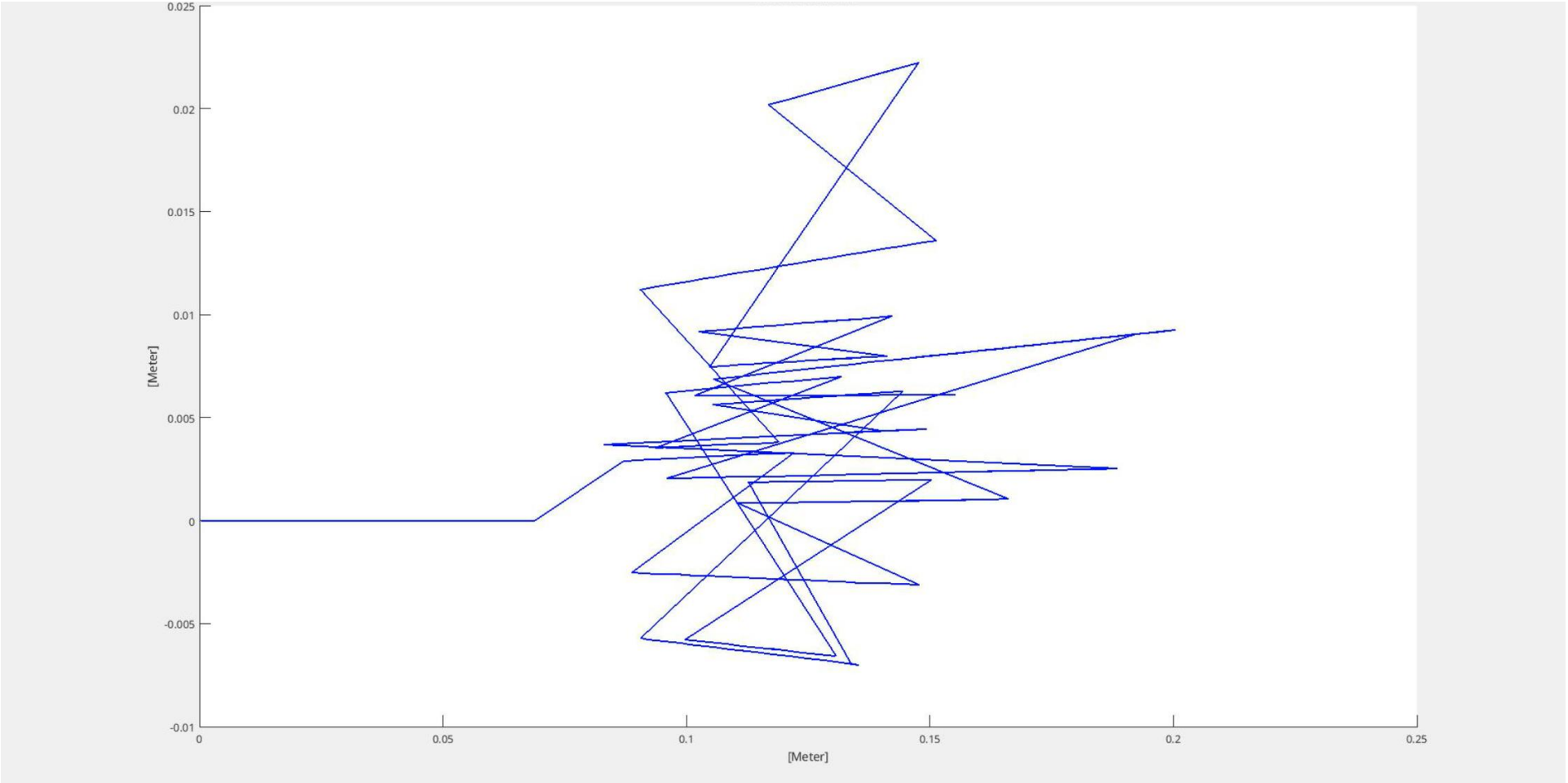
Why is it bad?



- Just IMU (x only)
- + SLAM(starting)

Why is it bad?

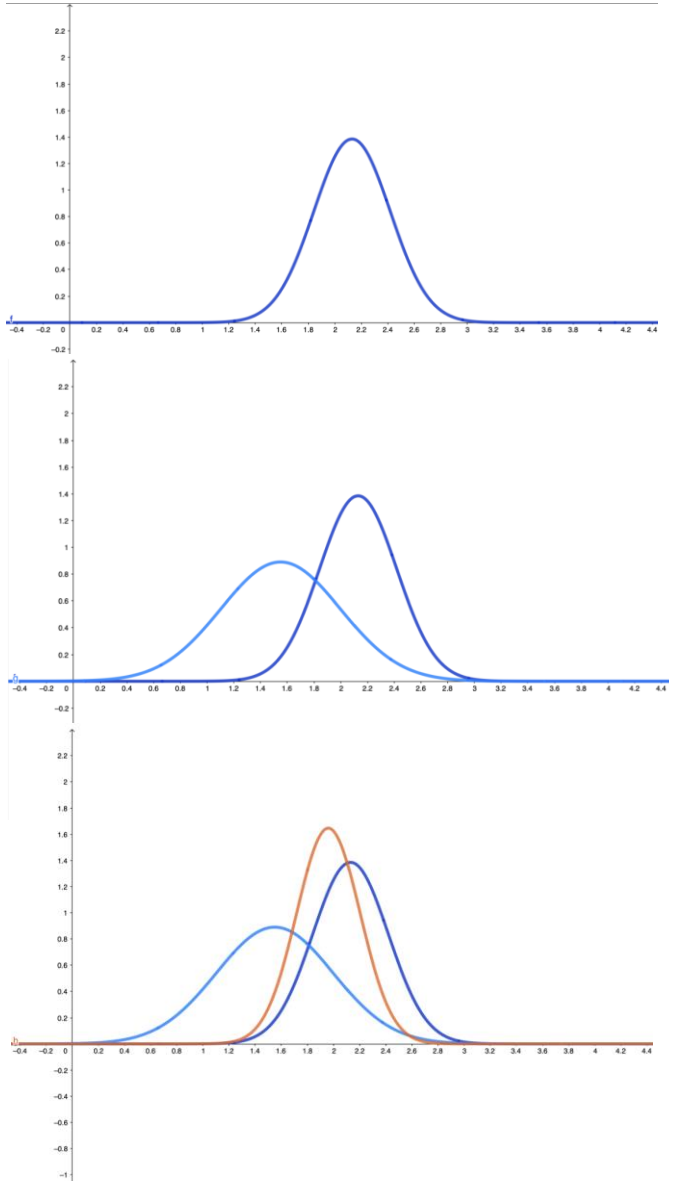
Car standing still
SLAM jumps
IMU bias



- Just IMU (x only)
- + SLAM(starting)

Kalman Filter.

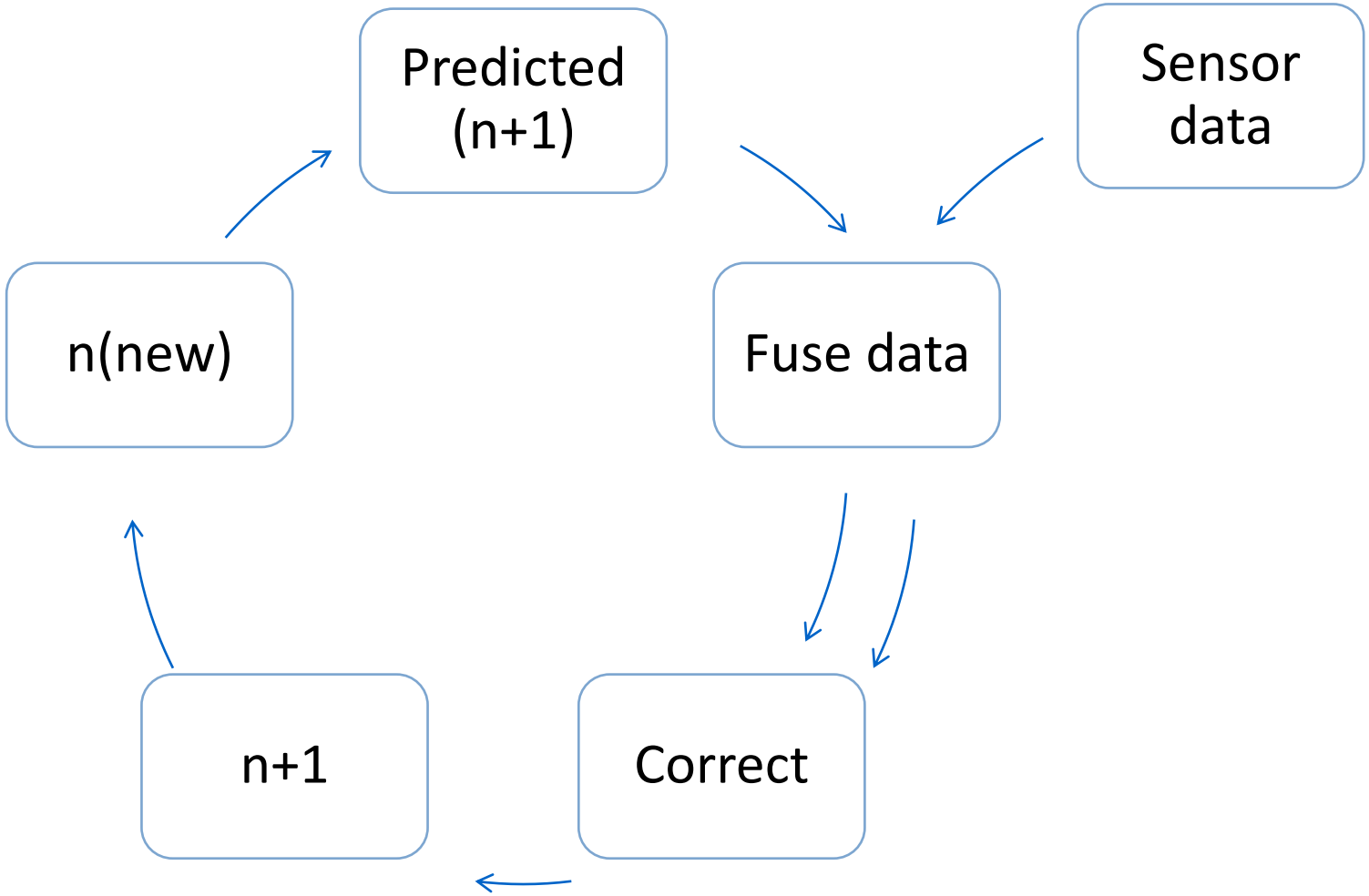
- Sensor noise -> Gaussian distribution
- Not all Sensors perfect
  - also corrects velocity with Position updates

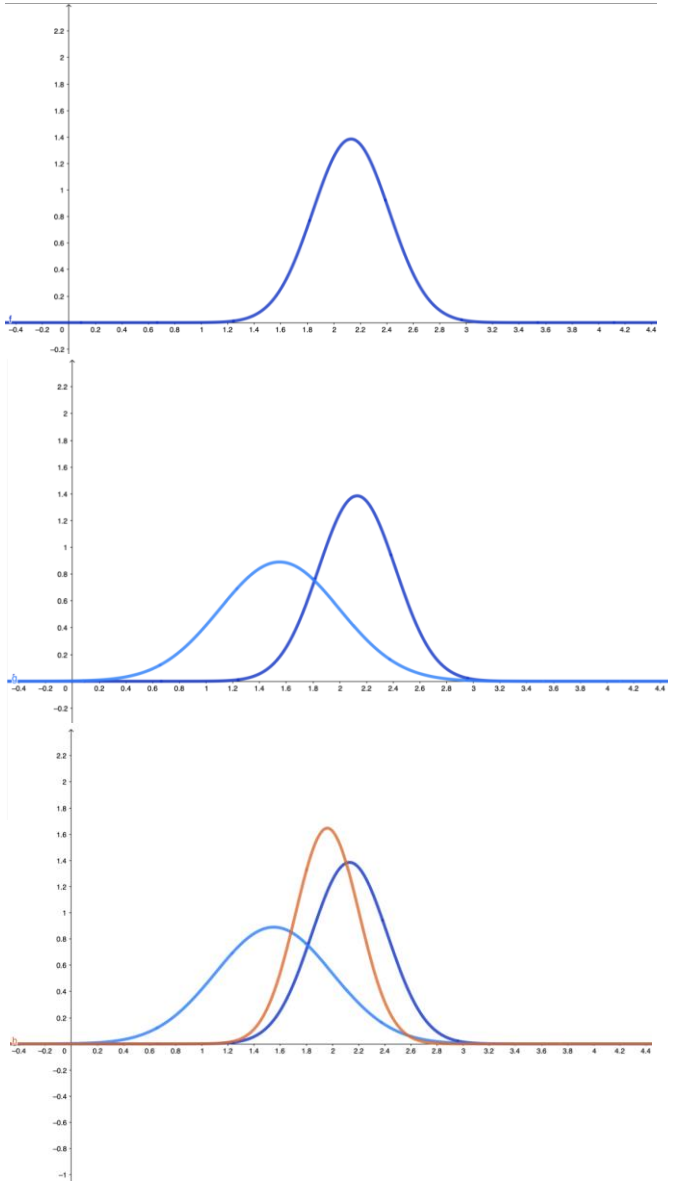


- Tries to fuse all given info
- Interpolate and predict

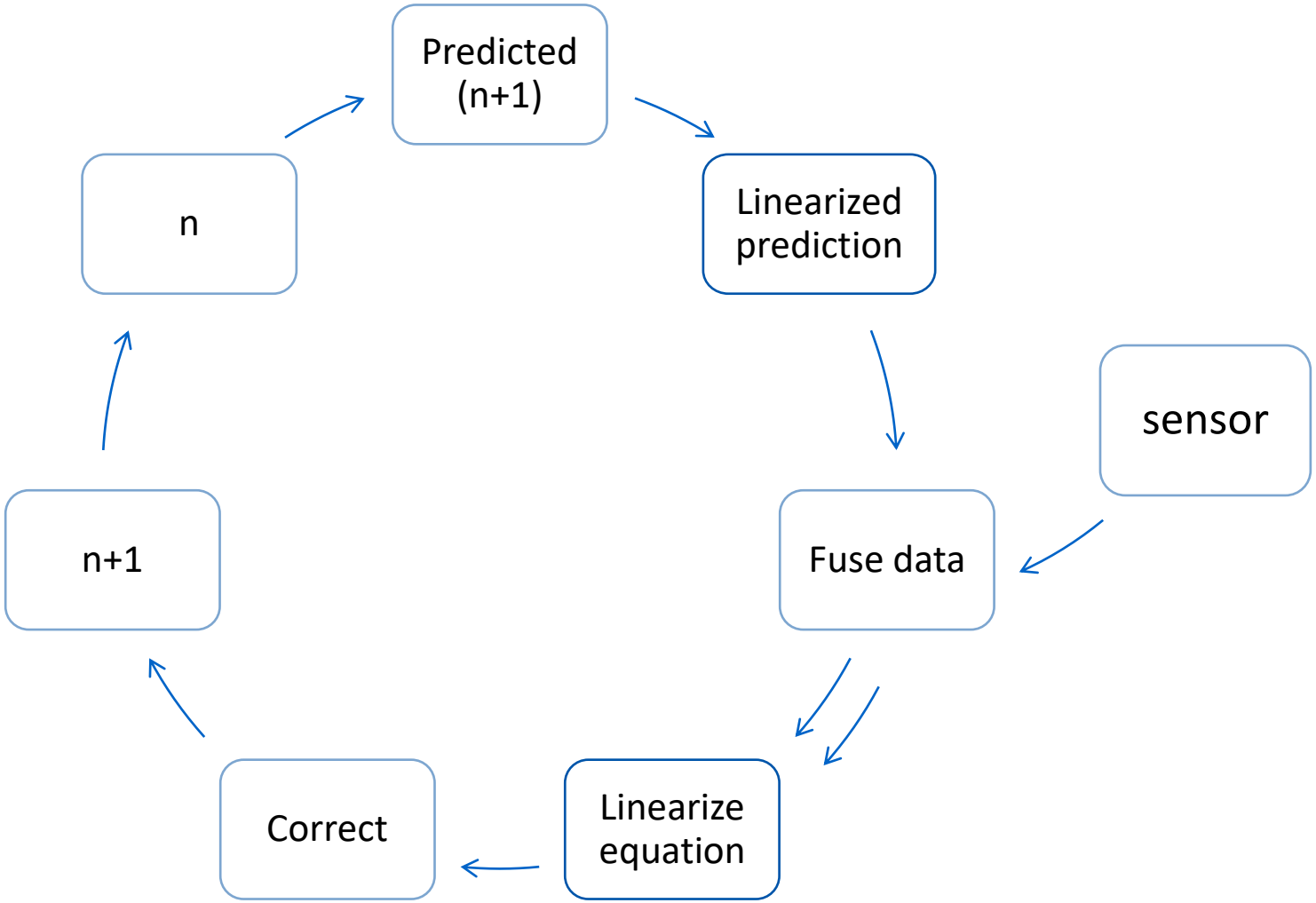
- We see that later on!

Kalman Filter.

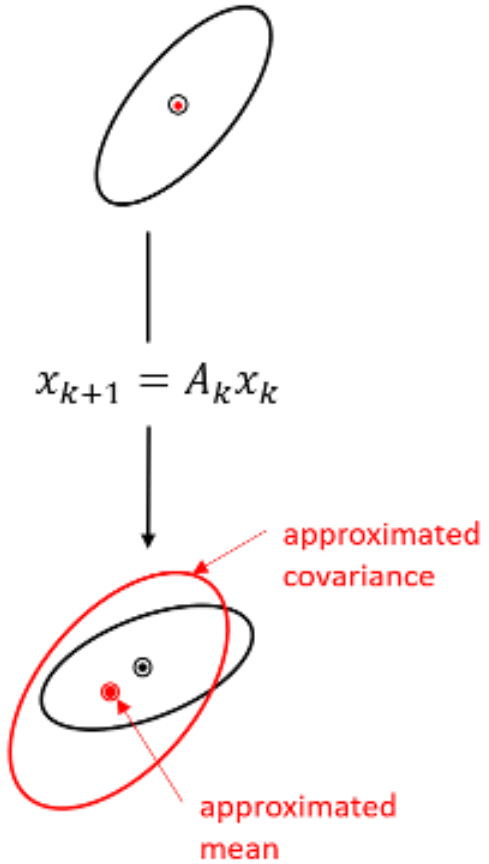




EKF.



Linear Transformation (EKF)



EKF.

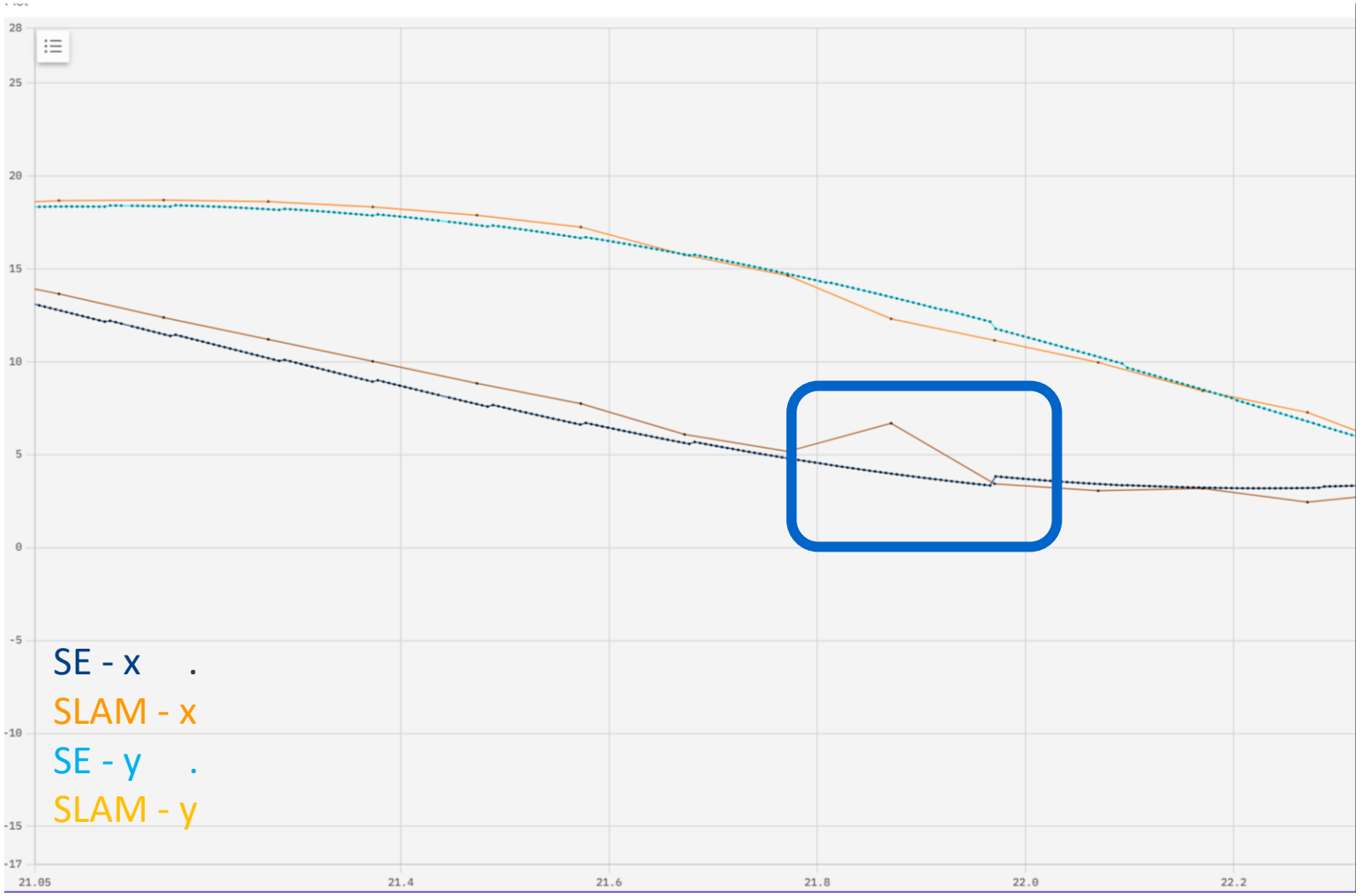
- Linearization around work point
  - Taylor expansion ending at first depth -> linear approximation
  - Using Jacobian

- Highly non-linear systems:
  - Still only an approximation
  - -> problems and errors

EKF

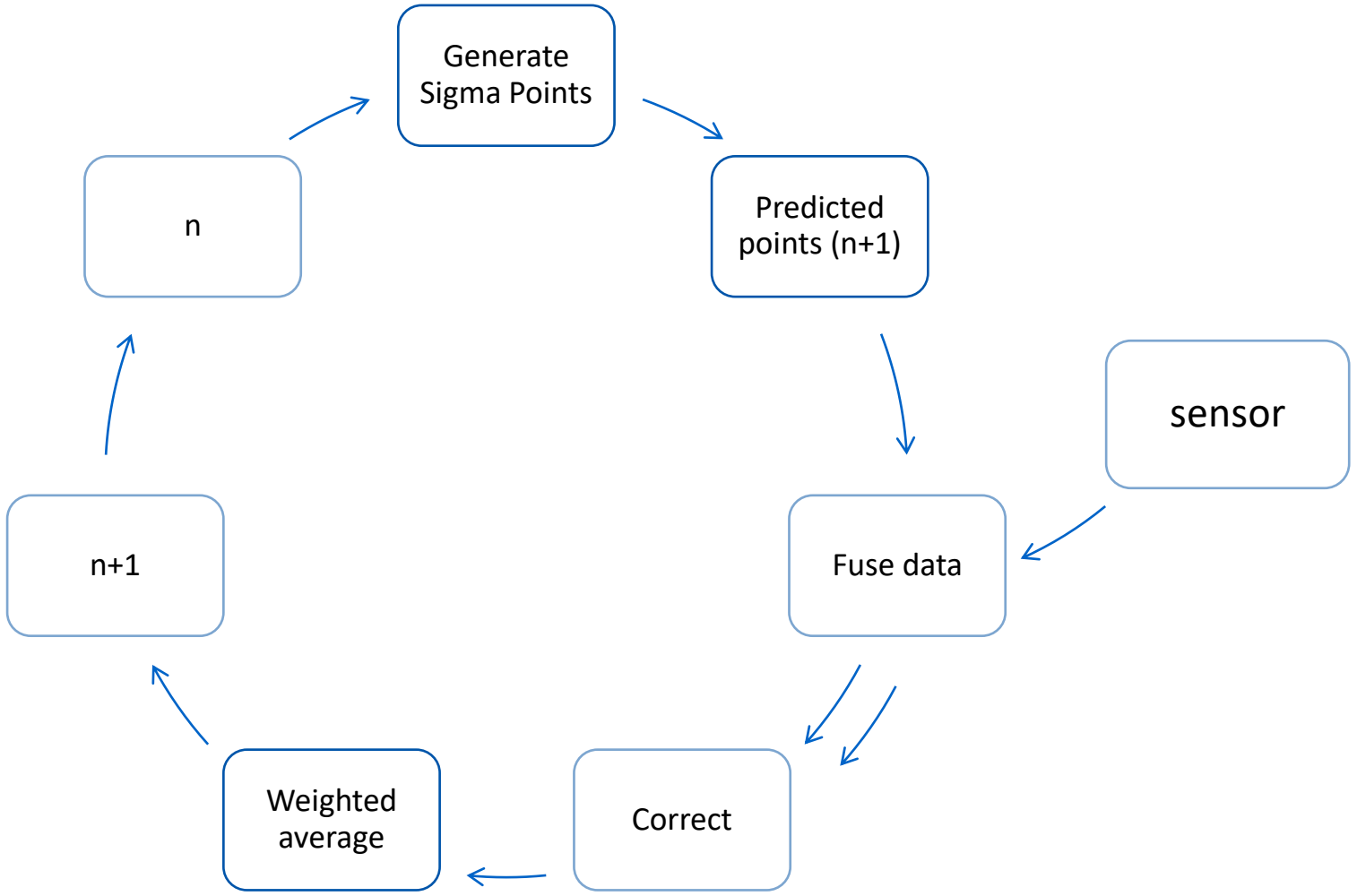
- SLAM is also not perfect
- Missallignment not trusted 100%

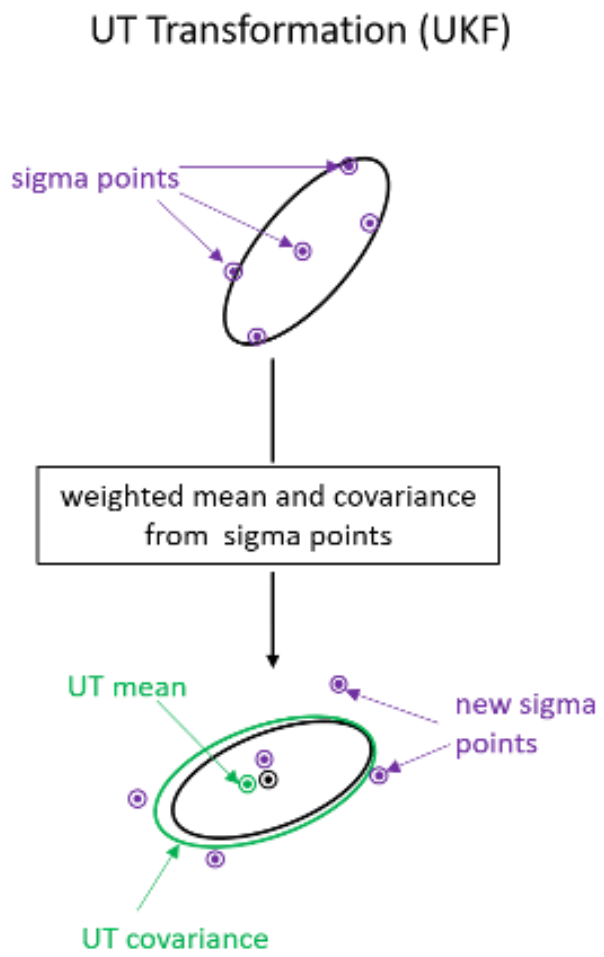
Global pos [m]



ROS time [s]

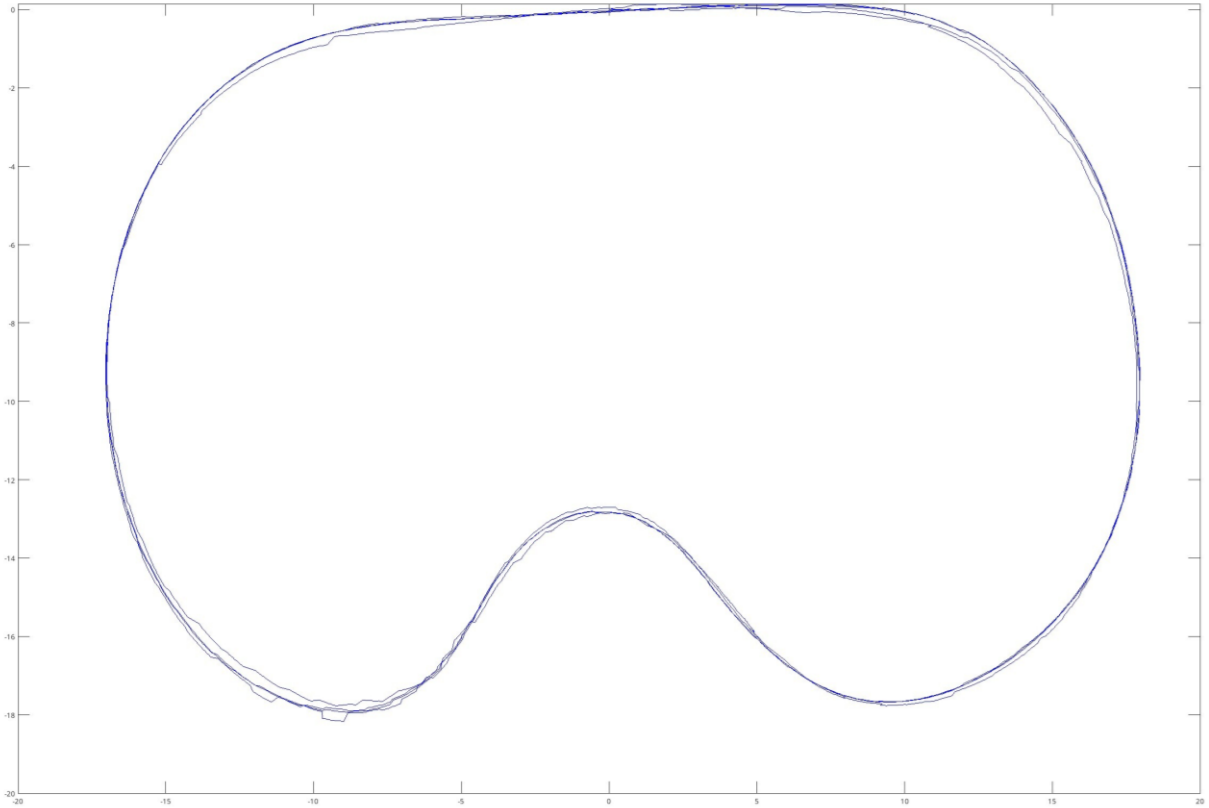
UKF.


Generate Sigma Points
Predicted points (n+1)
sensor
Fuse data
Correct
Weighted average
n+1
n


UT Transformation (UKF)
sigma points
weighted mean and covariance from sigma points
UT mean
new sigma points
UT covariance

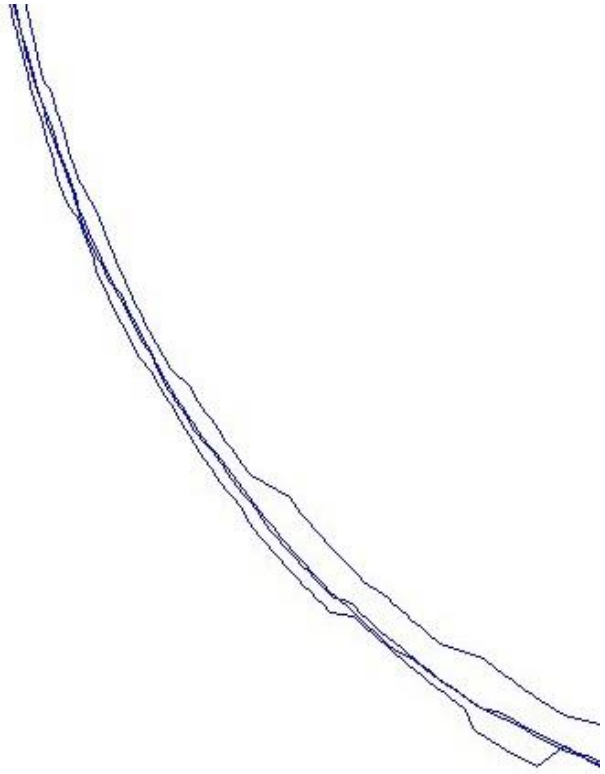
Traps you can fall into.

- The UKF needs more computational power.
- (nearly) Linear models (point mass) -> no gain!

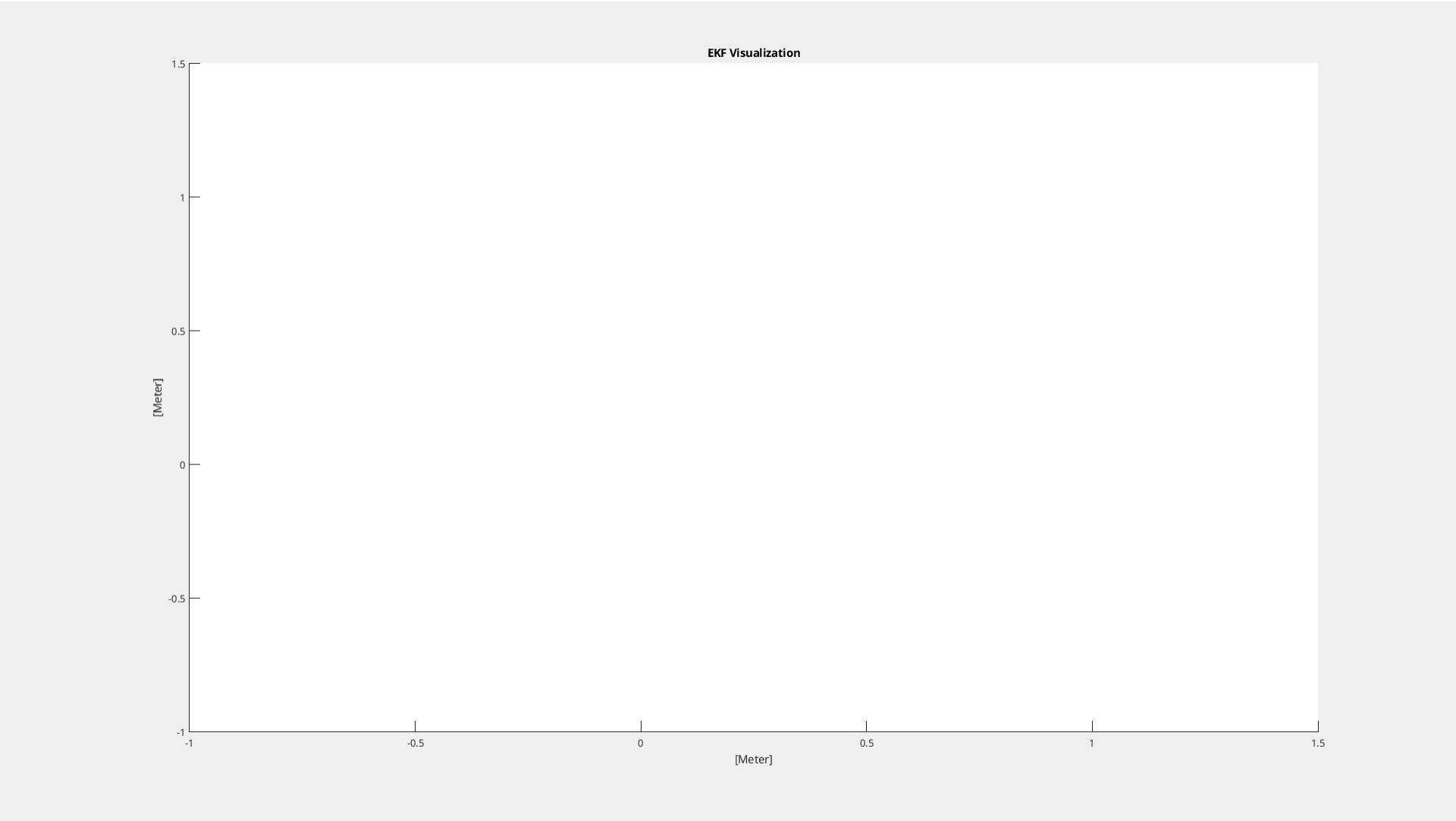


Traps you can fall into.

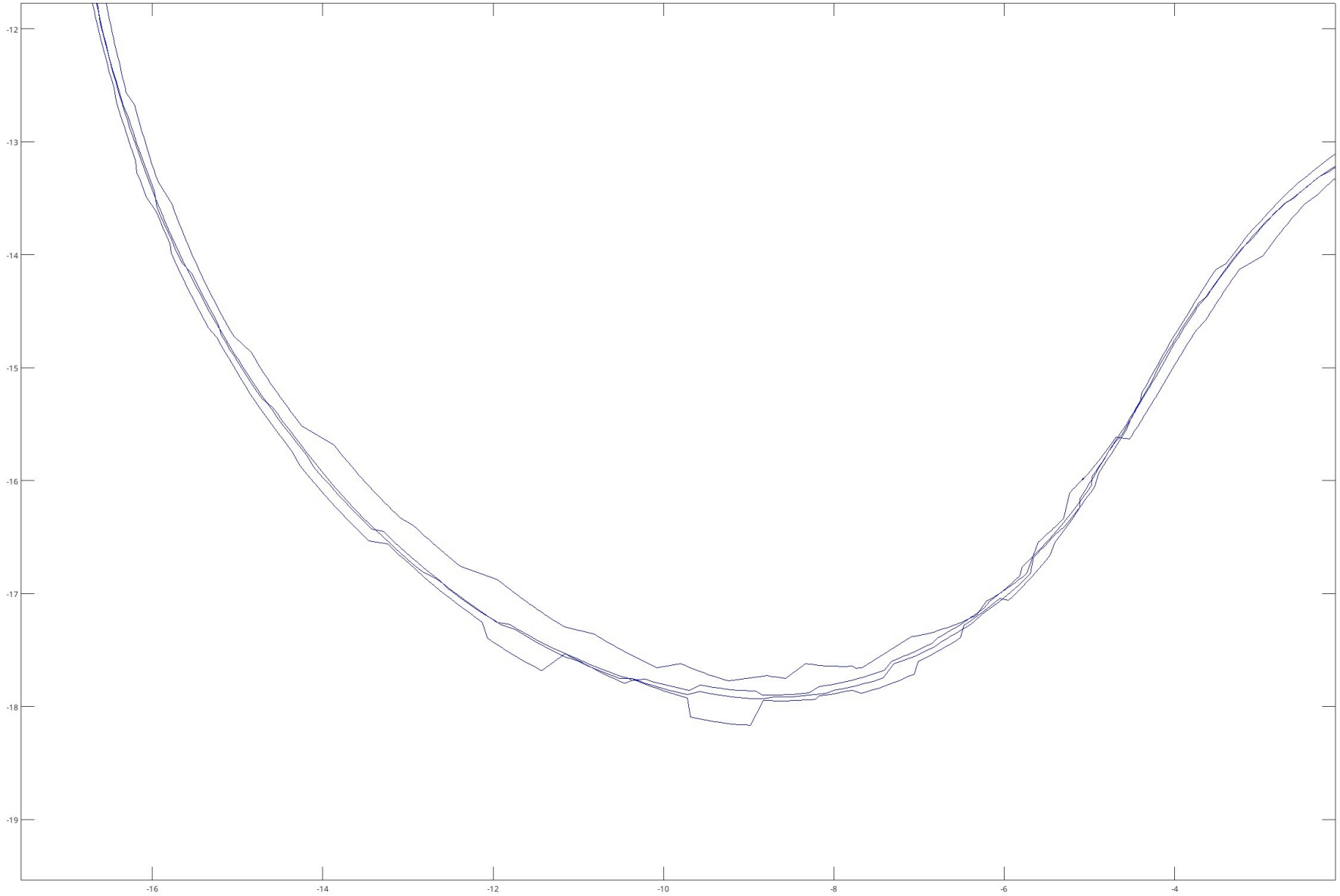
- The UKF needs more computational power.
- (nearly) Linear models (point mass) -> no gain!



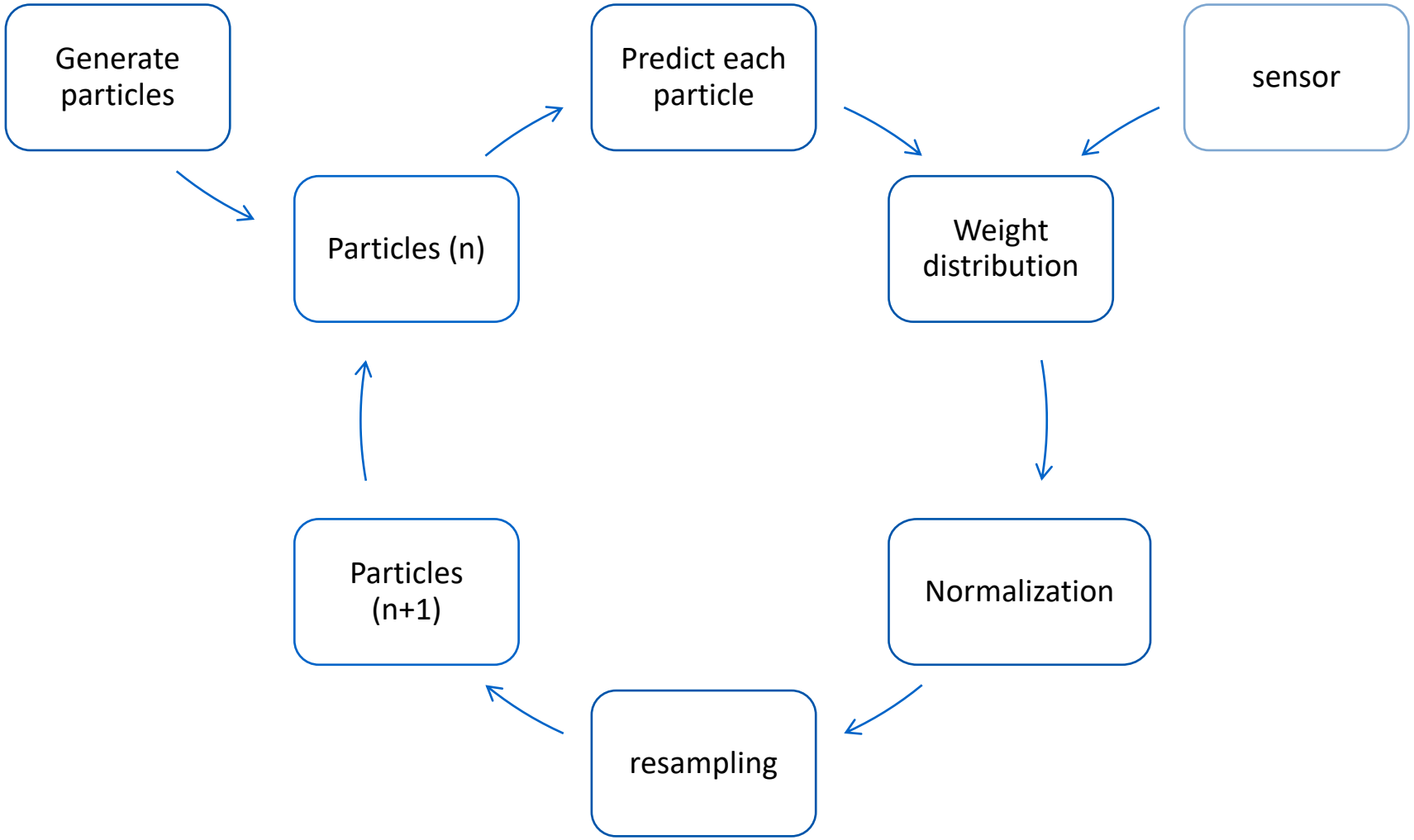
Still not perfect.

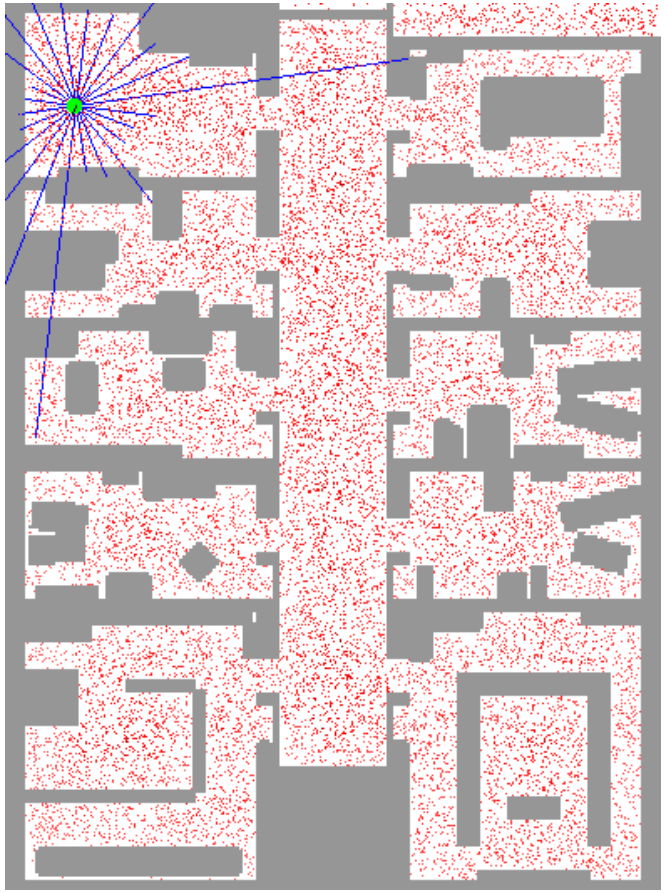


Still not perfect.



Particle Filter.





Optimizer.

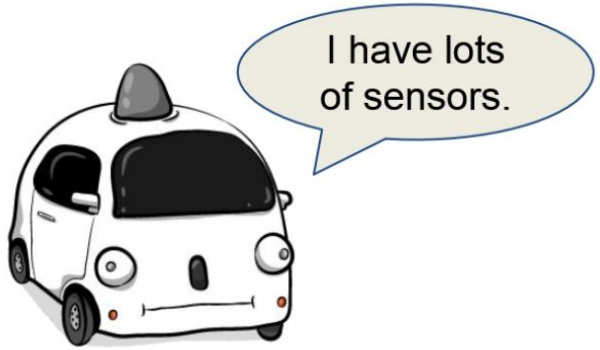
- Instead of approximating state with Kalman Gain
- Minimizing error and distance

- Possible Accuracy in Complex Systems

- Optimization complex in high dimensional states
  - -> hard to keep update rate

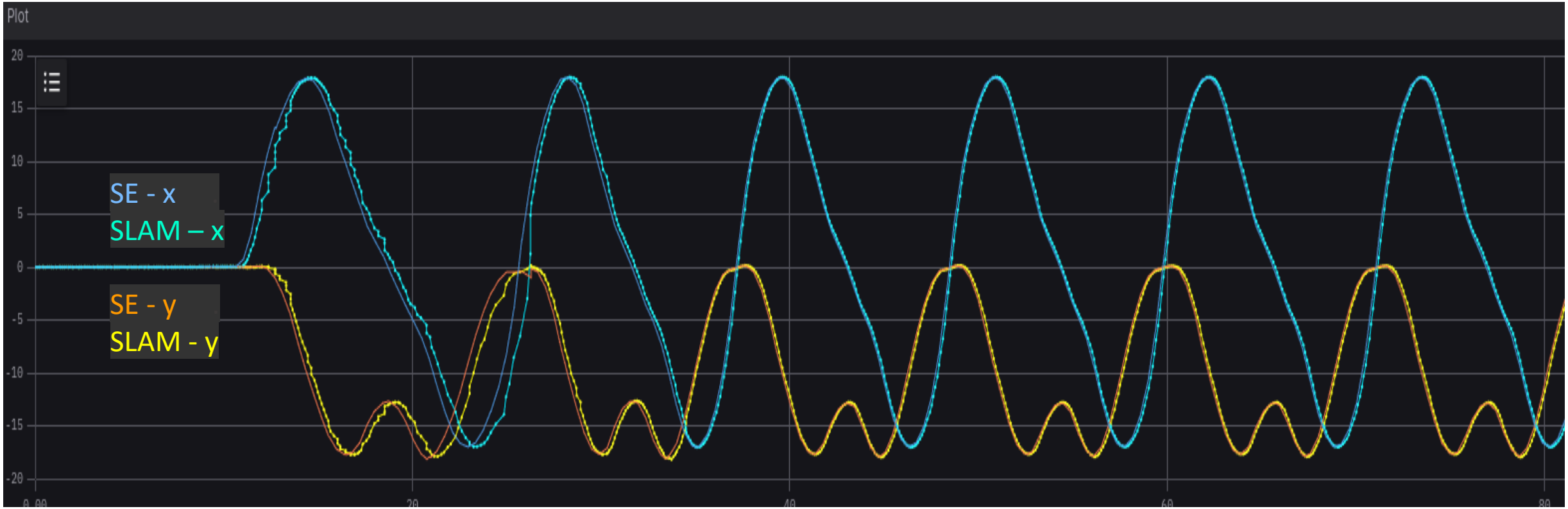
FUSE

- Current development by locus robotics
  - Also worked on robot_localization
- Optimization instead EKF/UKF
- Sadly, not much documentation
- Hardware limits



Optimizer.

Global pos [m]



ROS time [s]

Modeling.

It can be fun (trust me)

© FSG Wintermantel

DIFFERENCE TO OTHER MODELS.

Parameter and Inputs matter

- Think what information/inputs you have
- Same model type with different inputs
  - Motor speed or IMU accel

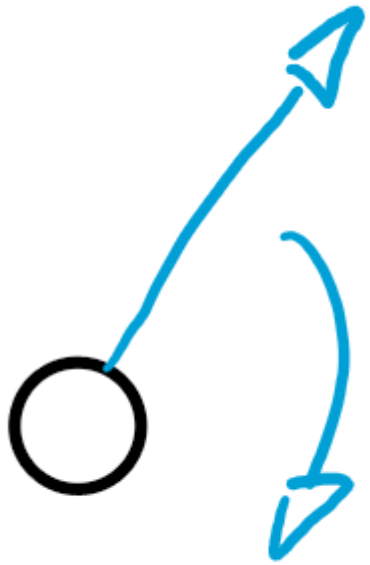
- What do you need?
  - Accuracy
  - Computational / time restrains

JUST A POINT.

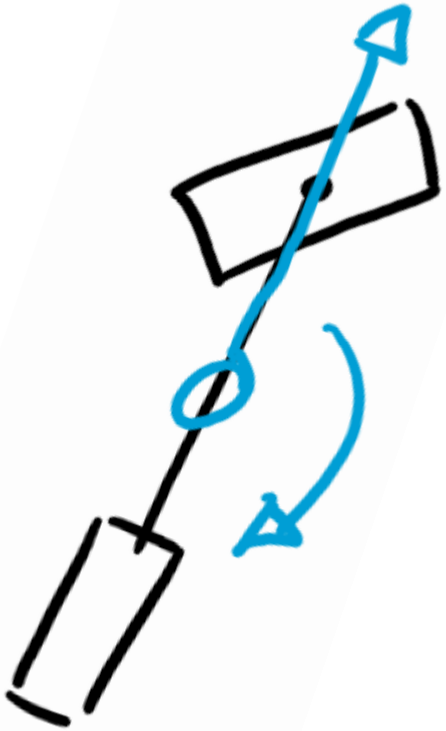
Don’t judge my drawing skills

- Really easy to implement
- Nearly linear
- Not much information needed

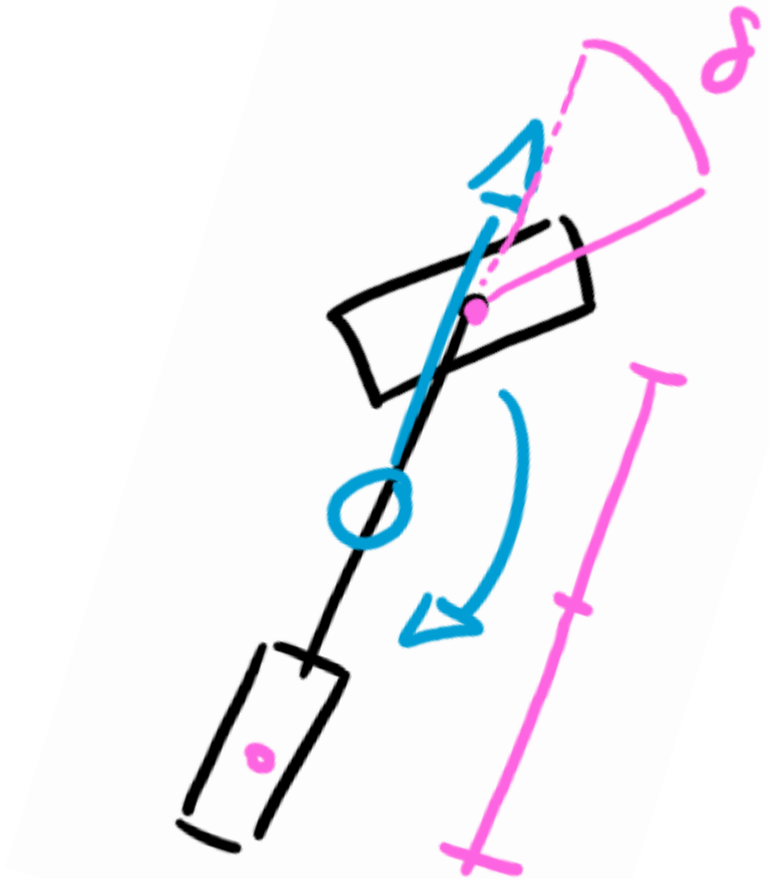
- $pos_x = pos_x +$
  - $\cos(yaw) * \left(vel_x * dt + \frac{1}{2}\, acc_x * dt^2\right) +$
  - $\sin(yaw) * \left(vel_y * dt + \frac{1}{2}\, acc_y * dt^2\right)$
- $yaw = yaw + \dot{yaw} * dt$



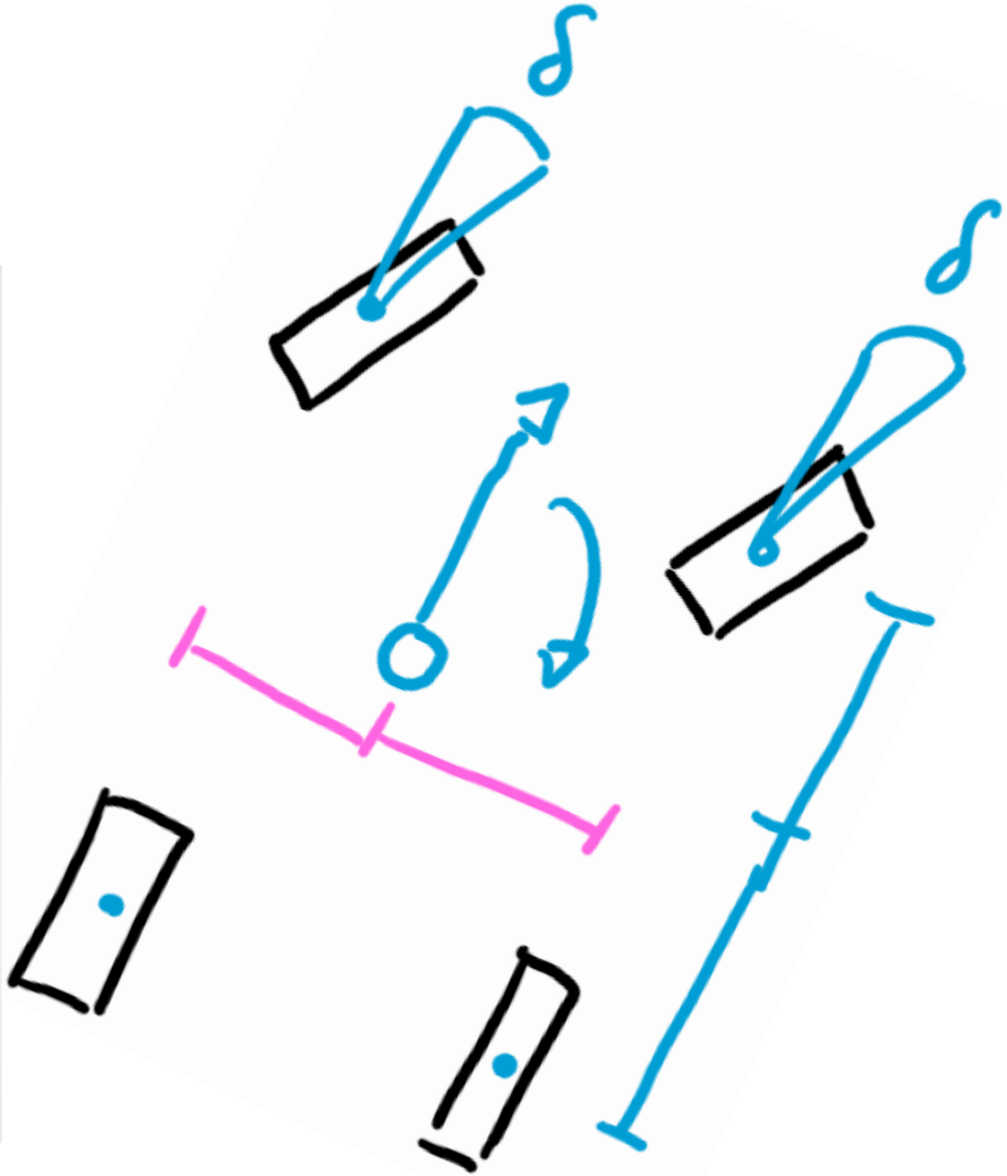
BICYCLE.



BICYCLE.

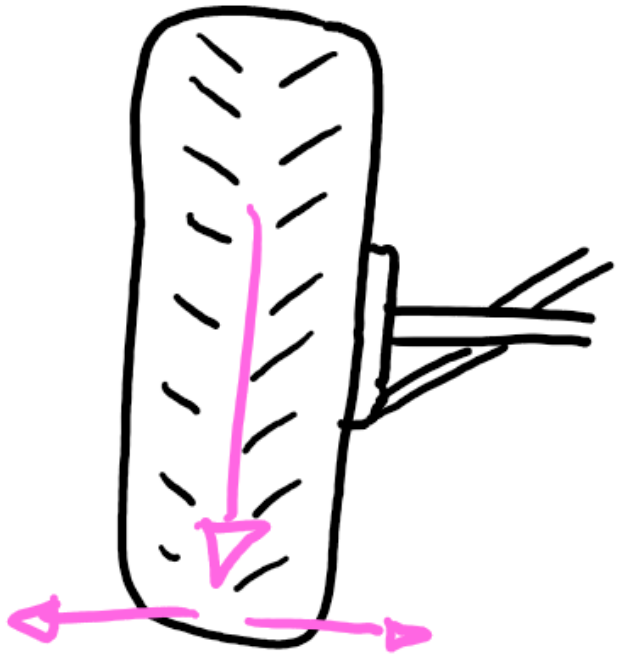


DOUBLE TRACK.



MORE?

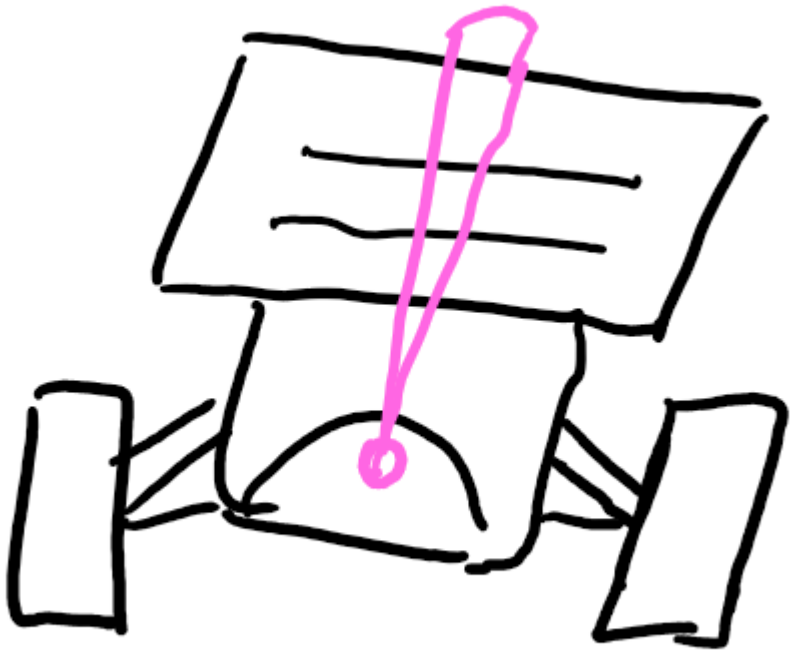
- Until now most basic concept
- Now: Driving at the limits
  - Slip in Corners
  - Slip in accel
  - steady to unsteady behavior
- Linear tires
- Pacejka tires
- But need normal forces for Grip calculations



STRESSTEST SKIDPAD

NORMAL FORCES.

- The body moves
  - Acceleration / curvature
- -> center of gravity shifts



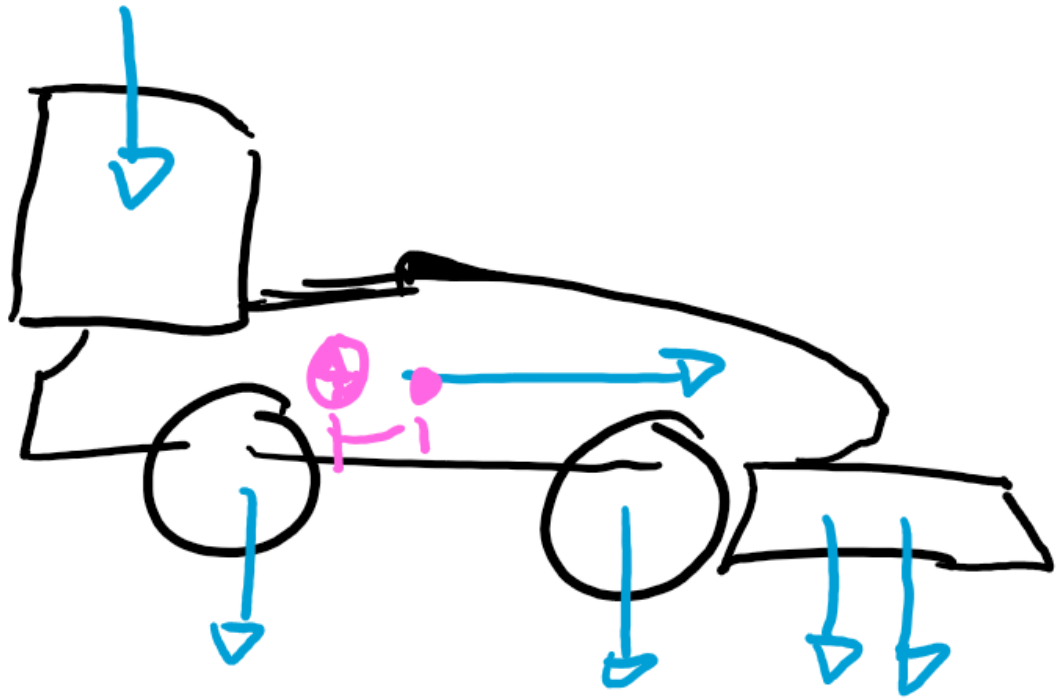
NORMAL FORCES.

- The body moves
  - Acceleration / curvature
- -> center of gravity shifts

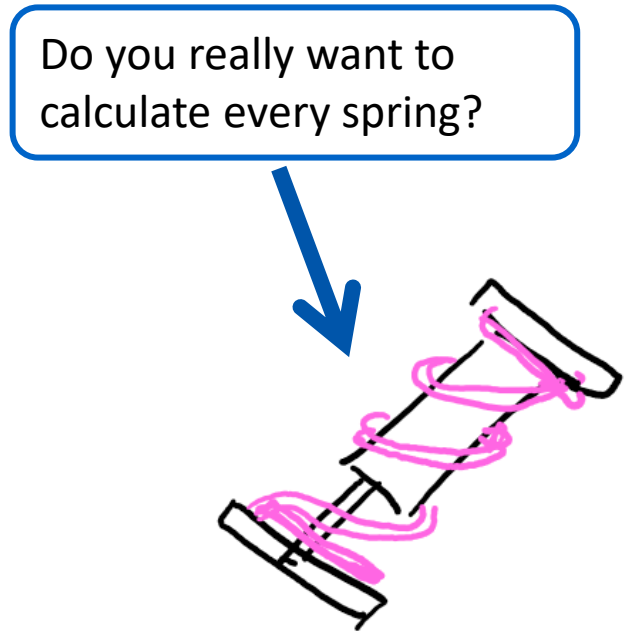
NORMAL FORCES.

- The body moves
  - Acceleration / curvature
- -> center of gravity shifts

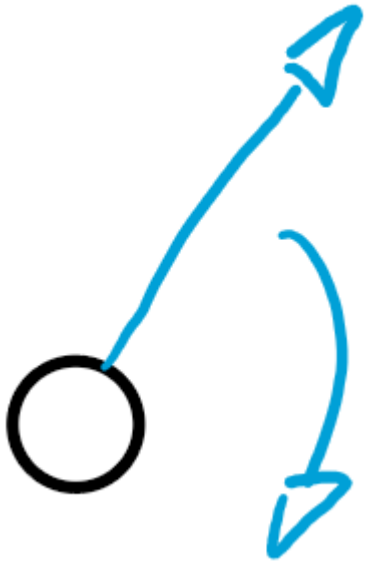
- Aero depending on velocity
  - (and rotation speed)

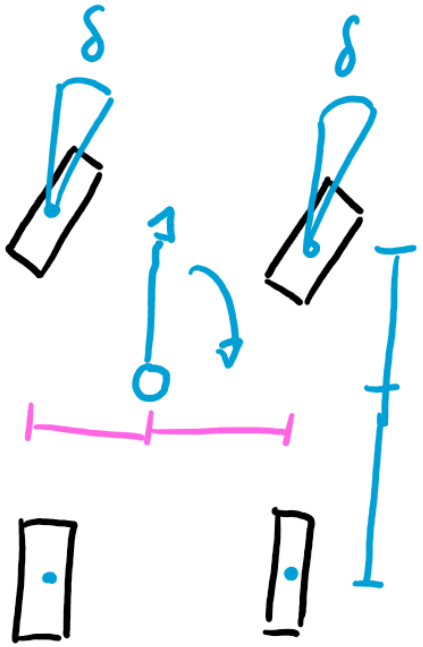


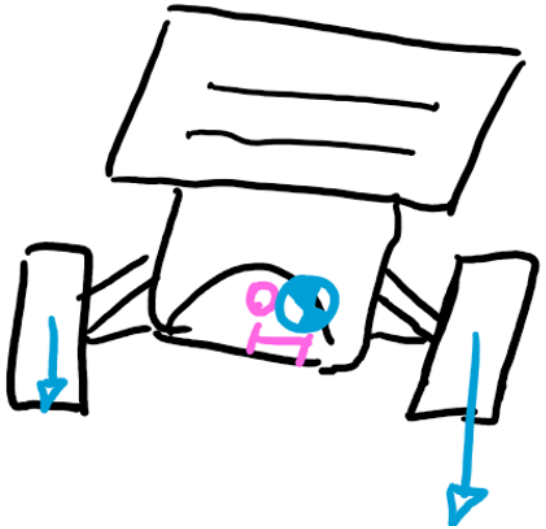
MY TIP.



- Start easy and look what you can use
- Add Things (you think help the most) one after another
  - A dynamic bicycle can be better than a static double track.
- Keep restrains in mind (accuracy, time, etc)







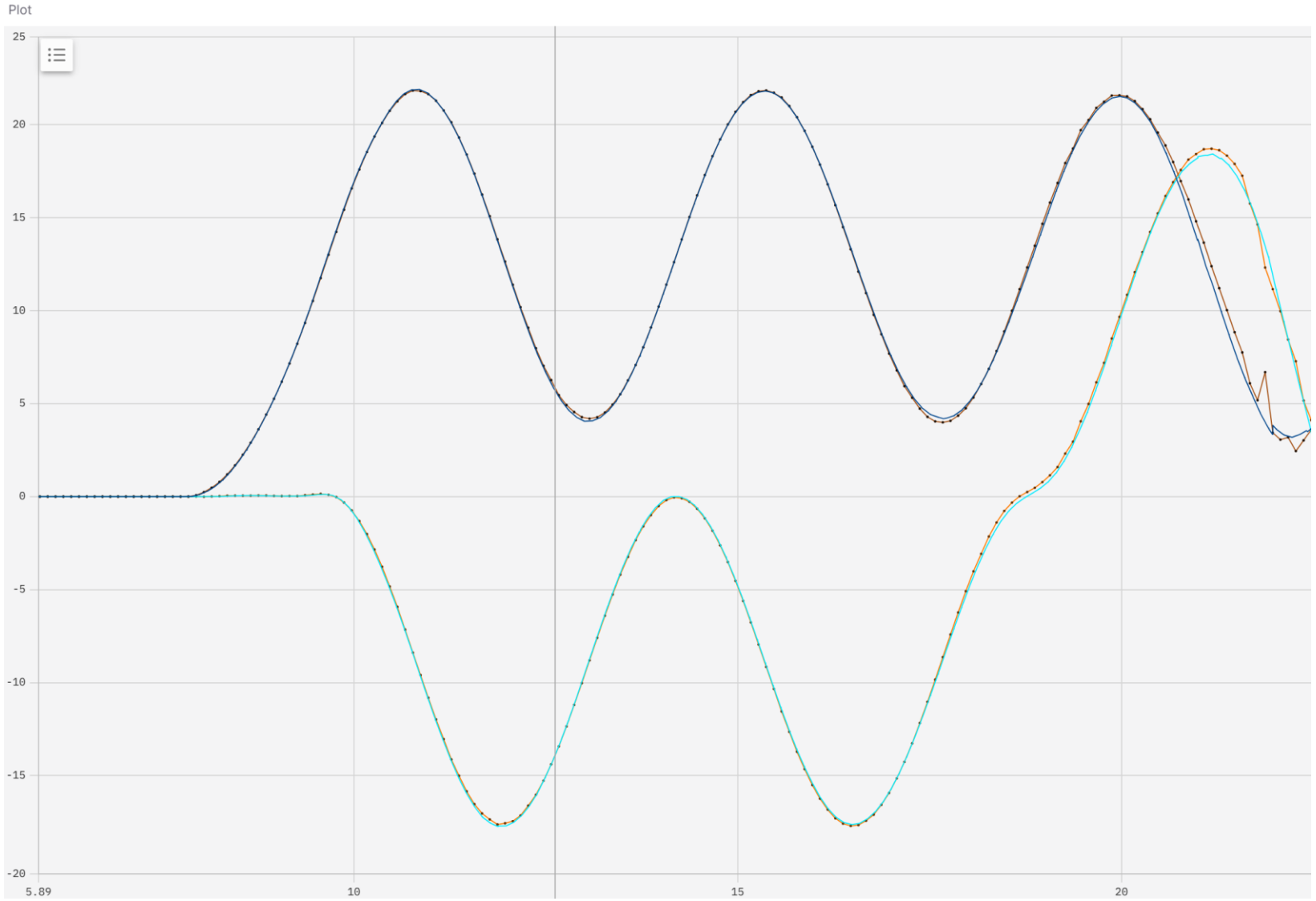
Overall Questions.

Or more details?

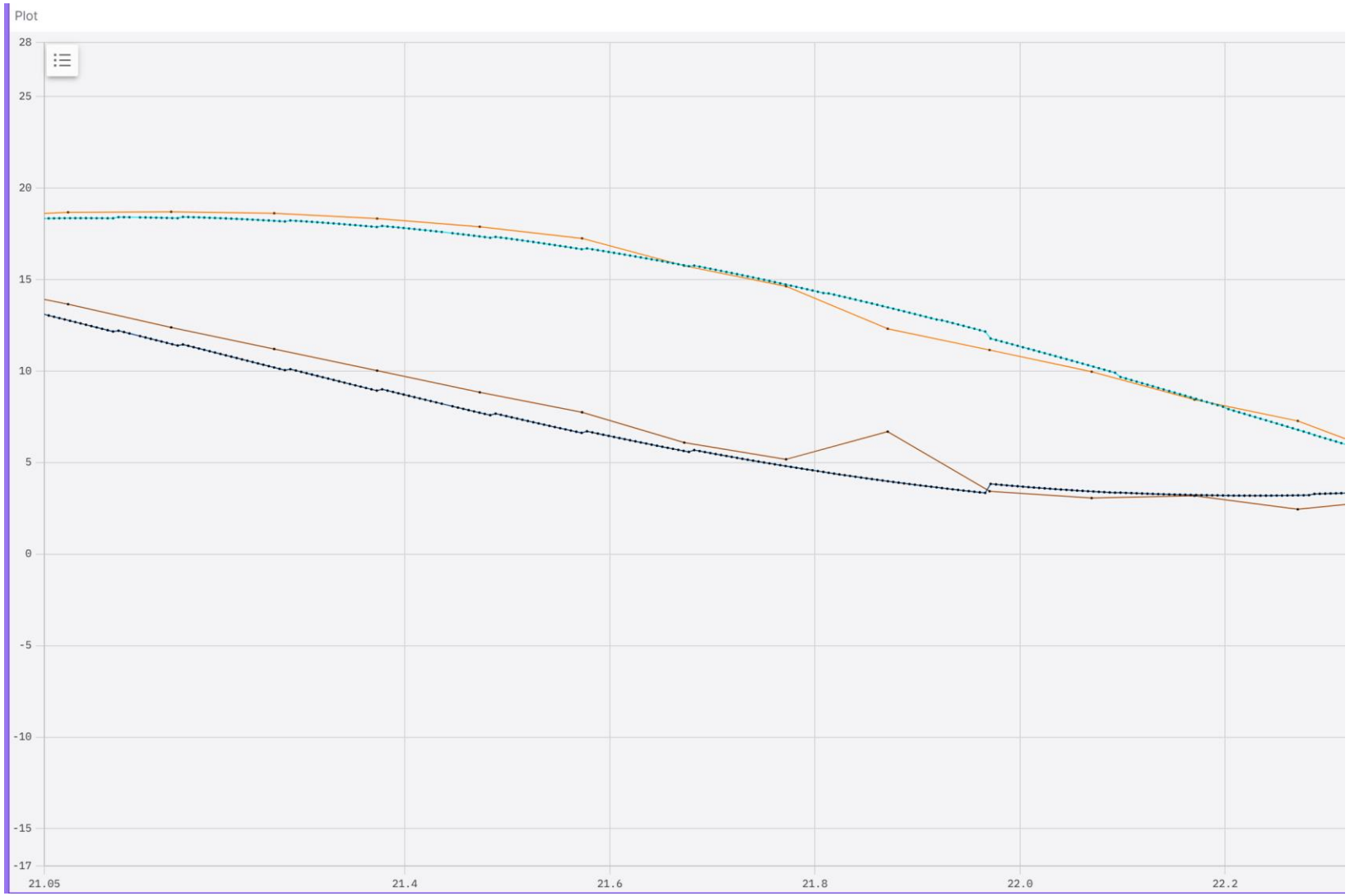
Deep Dive example.

Skidpad (nearly) stress test.
And easy things to see.

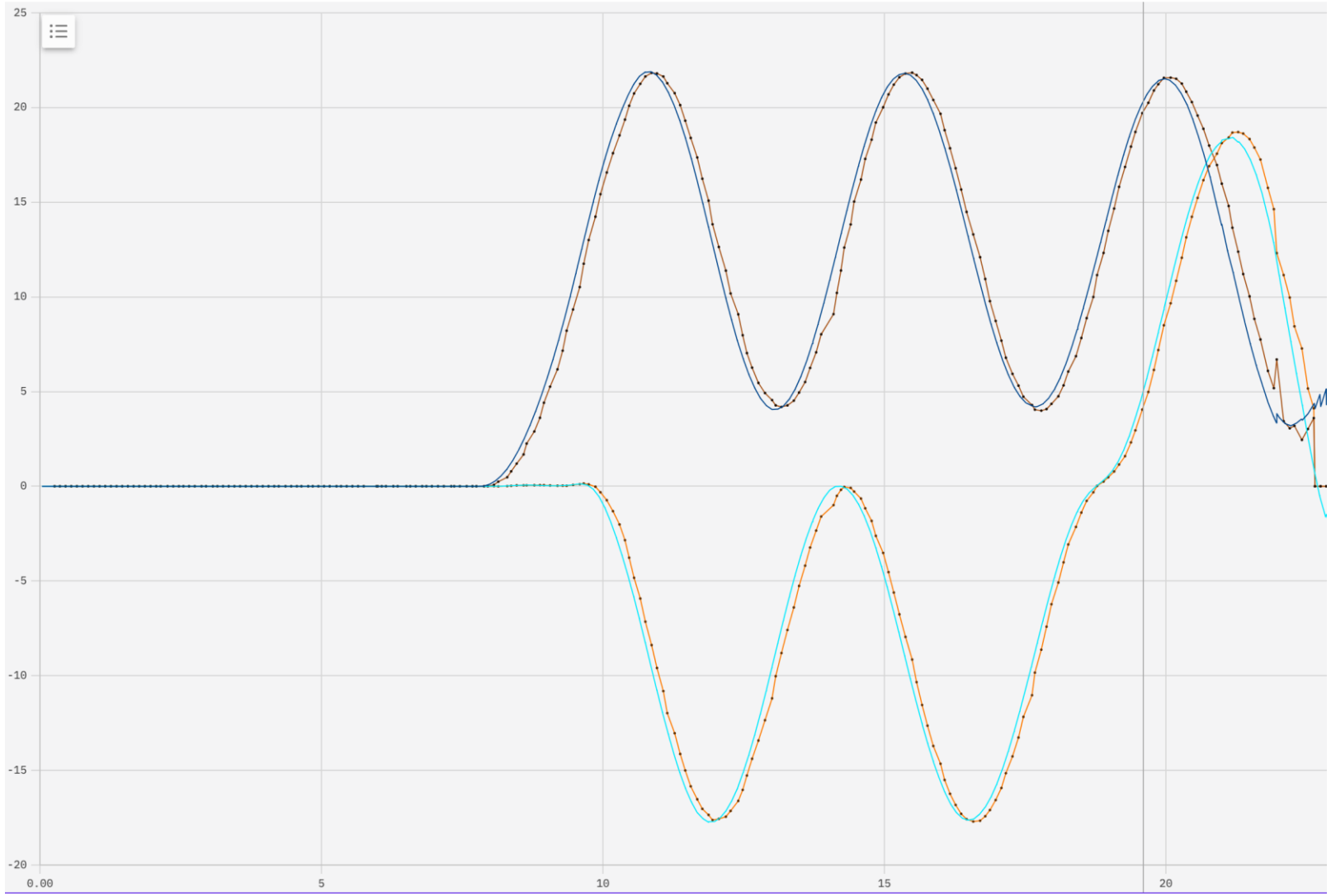
SKIDPAD EXAMPLE



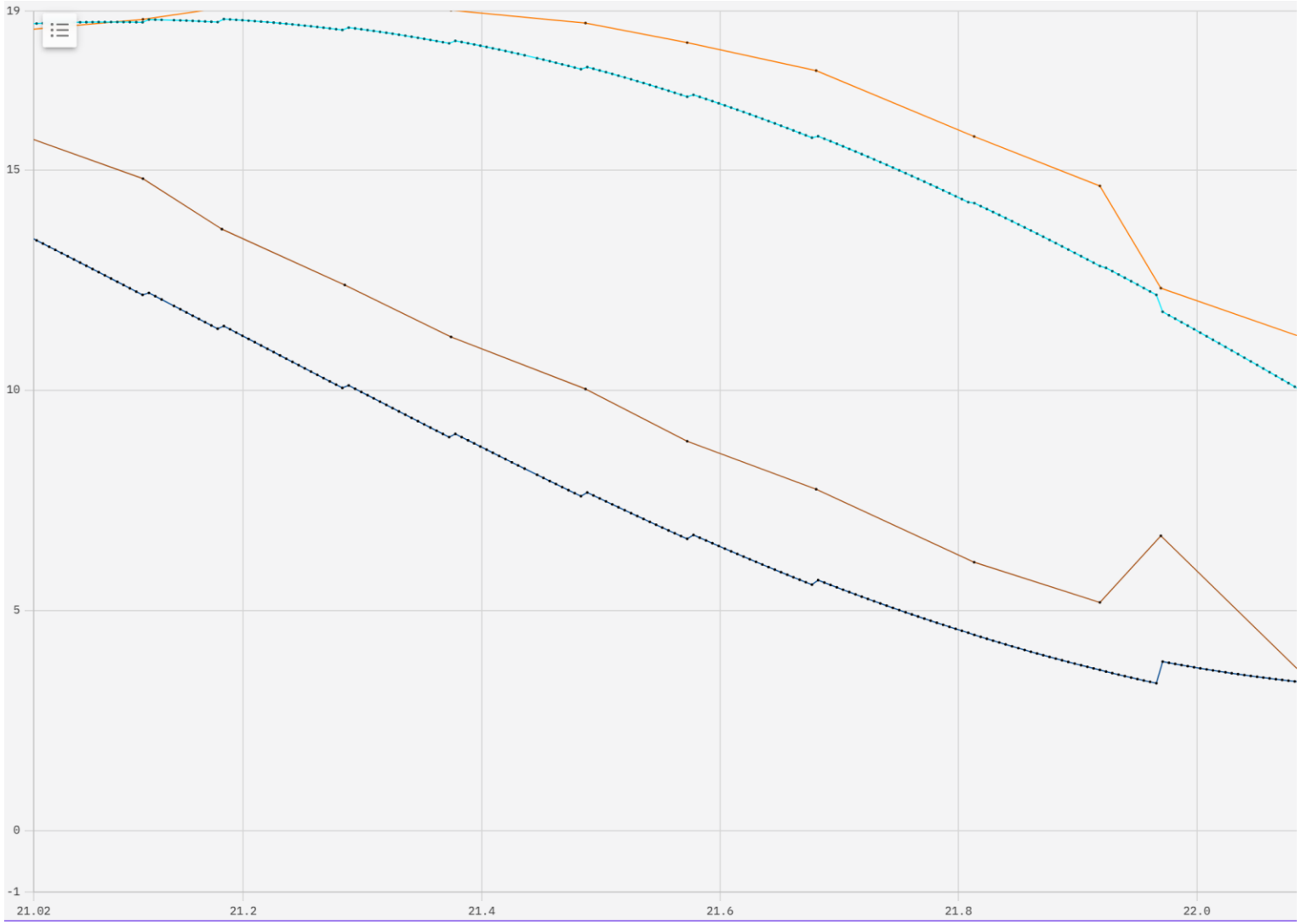
SKIDPAD EXAMPLE



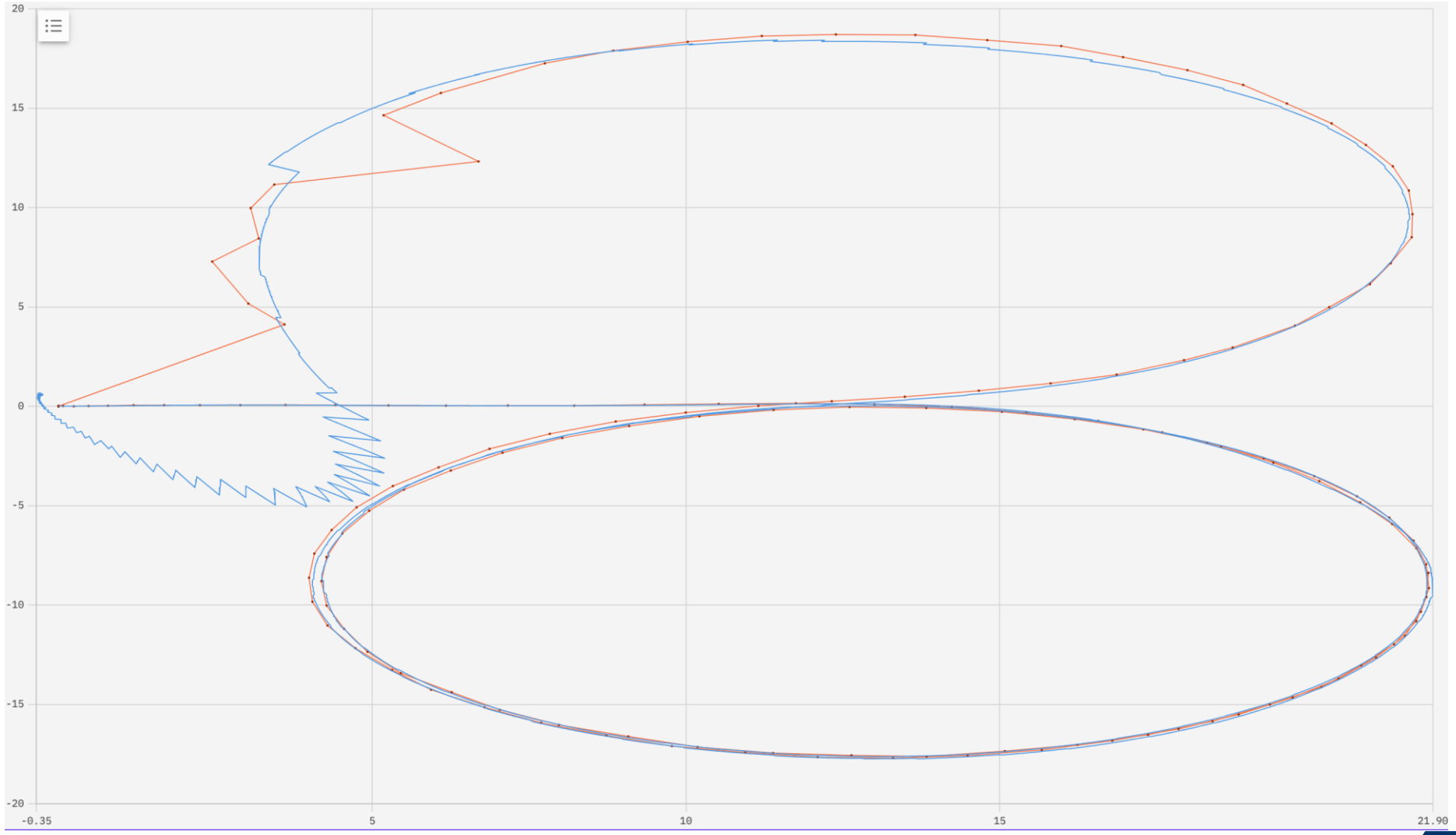
SKIDPAD EXAMPLE



SKIDPAD EXAMPLE



SKIDPAD EXAMPLE



SKIDPAD EXAMPLE