State Estimation for Localization:

A Look at the Odometry State Estimator

Steve Macenski

Open-Source Robotics - Lead Engineering

Samsung Research America

WhoAmI

Open Source Robotics Lead @ Samsung Research

- Develop and Maintain 50+ ROS 1 and ROS 2 packages
- ROS2 Technical Steering Committee & Navigation Project Lead

Former Robotics Lead @ Simbe Robotics

I think about production navigation systems; perception, SLAM, planning, and sensor fusion

ROSCon 2018

ROSCon 2019

TechCrunch Sessions 2020

NASA Asteroid Redirect Mission

National Geographic

OVERVIEW

- Kalman Filters
- Non-Linear Filters
- Robot Localization
- Autoware Odometry

Kalman Filters (1/2)

Filters jobs are to track signals in presence of noise

Filters are smoothers based on data, error estimates, and a model

The Kalman Filter is used in linear systems: Predict and Correct

- Typically constant velocity or acceleration model **A**
- Observation model **H** relates the measurable data to the state
- Process covariance **Q** and measurement covariance **R**
- Measurement **z** taken to estimate the state $\hat{x}$

Predict

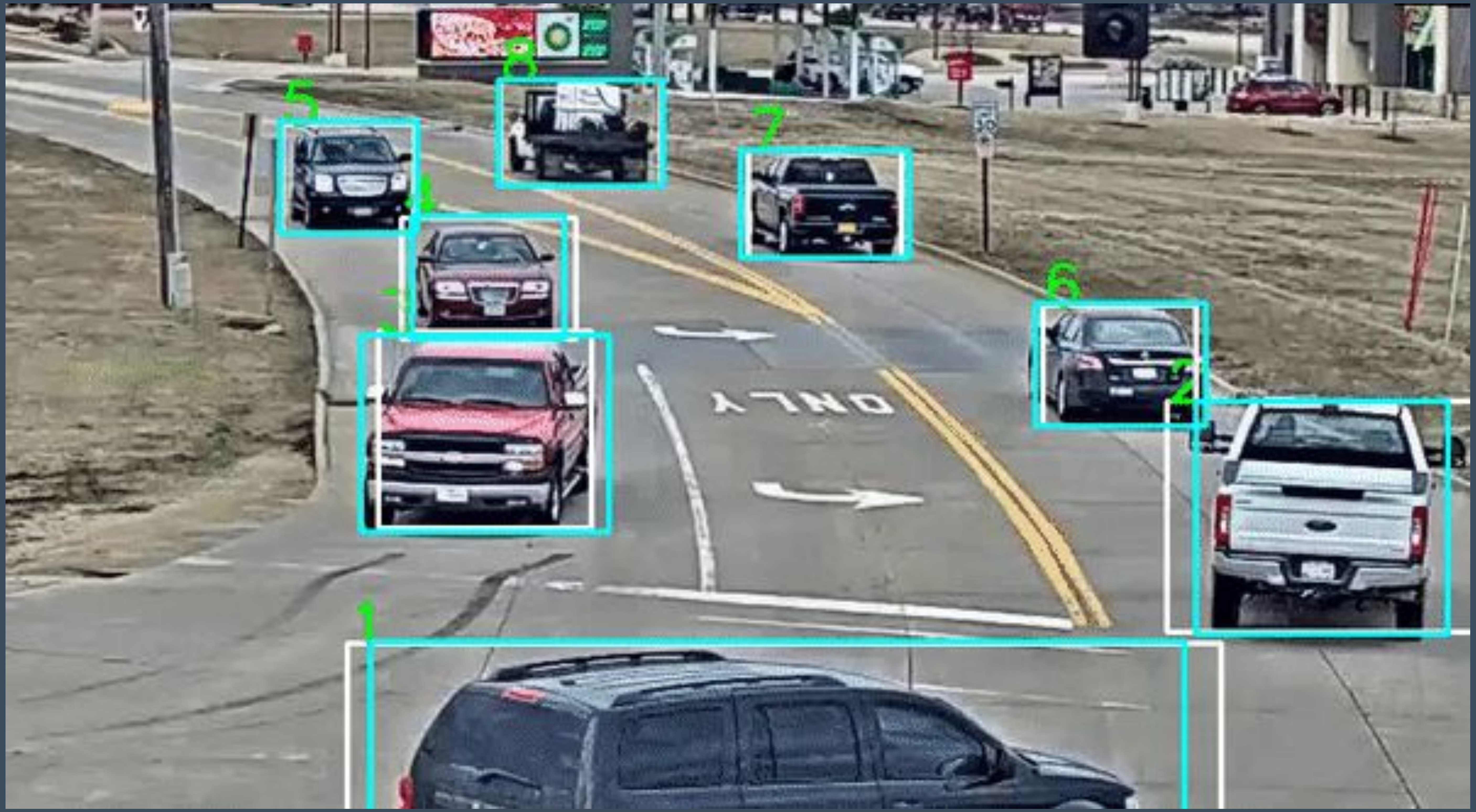
$$\hat{x}^-_{i+1} = A\,\hat{x}_i$$
$$P^-_{i+1} = A\,P_i\,A^T - Q$$

Correct

$$K_{i+1} = P^-_{i+1}\,H^T\,(H\,P^-_{i+1}\,H^T + R)^{-1}$$
$$\hat{x}_{i+1} = \hat{x}^-_{i+1} + K_{i+1}\,(z_{i+1} - H\,\hat{x}^-_{i+1})$$
$$P_{i+1} = (I - K_{i+1}\,H)\,P^-_{k+1}$$

Kalman Filters (2/2)

Now we can take noisy measurements and track the signal (e.g. the car)



Detections
Tracks

Source: https://nanonets.com/blog/content/images/2019/07/object_tracker_gif.gif

Non-Linear Filters

What happens when your system isn't linear?

Nonlinear filters!

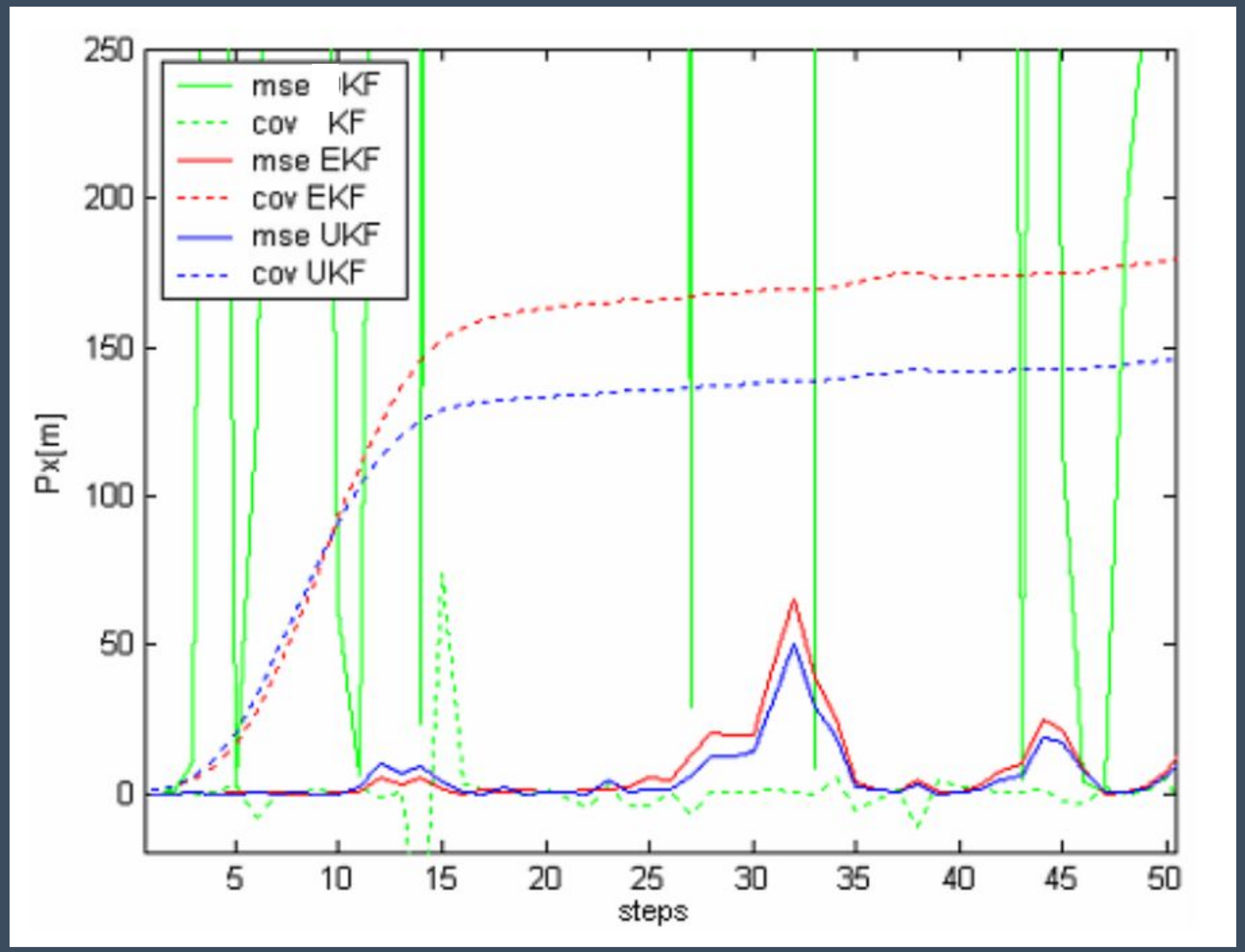
- Extended Kalman Filter (EKF)
- Unscented Kalman Filter (UKF)

EKF:

- Linearizes about the single current estimate (mean, covariance)
- The math is similar, Jacobians to replace $\mathbf{A}\hat{\mathbf{x}}_i$ and $\mathbf{H}\hat{\mathbf{x}}_i$.

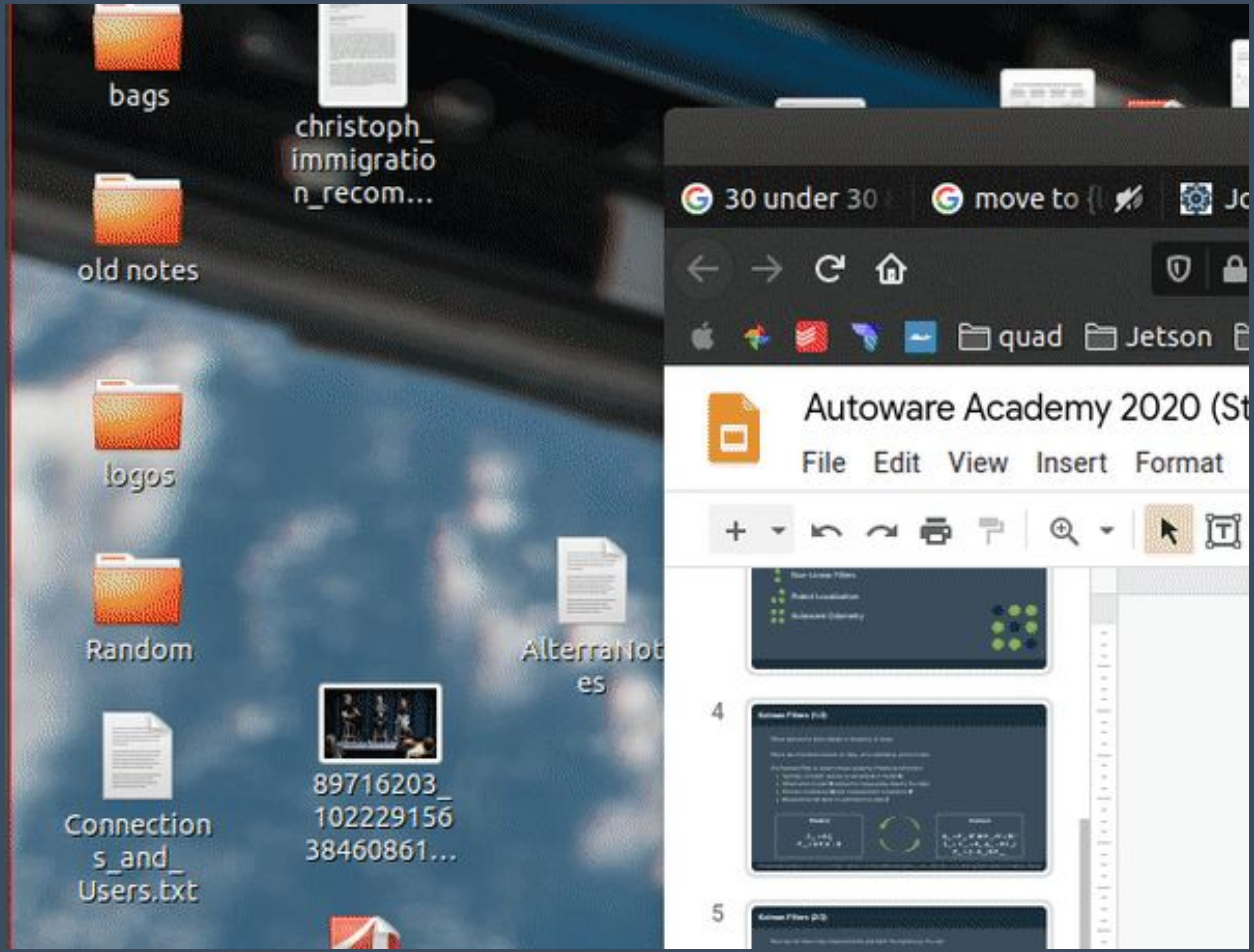
UKF:

- Uses the Unscented Transform for very non-linear systems
- Select "sigma points" in the distribution (many options)
- Transform the points using non-linear function
- New estimate derived from transformed distribution
- Additional accuracy derived from sets of means and covariances



Source: https://bit.ly/2LU10UG

EKF vs KF



https://gist.github.com/SteveMacenski/50cc1b4fe6e395f974697c501fede78a

Real Path
Extended Kalman Filter
Kalman Filter

Uses simulated wheel encoders and GPS updates

REP 105

Frame Conventions for ROS

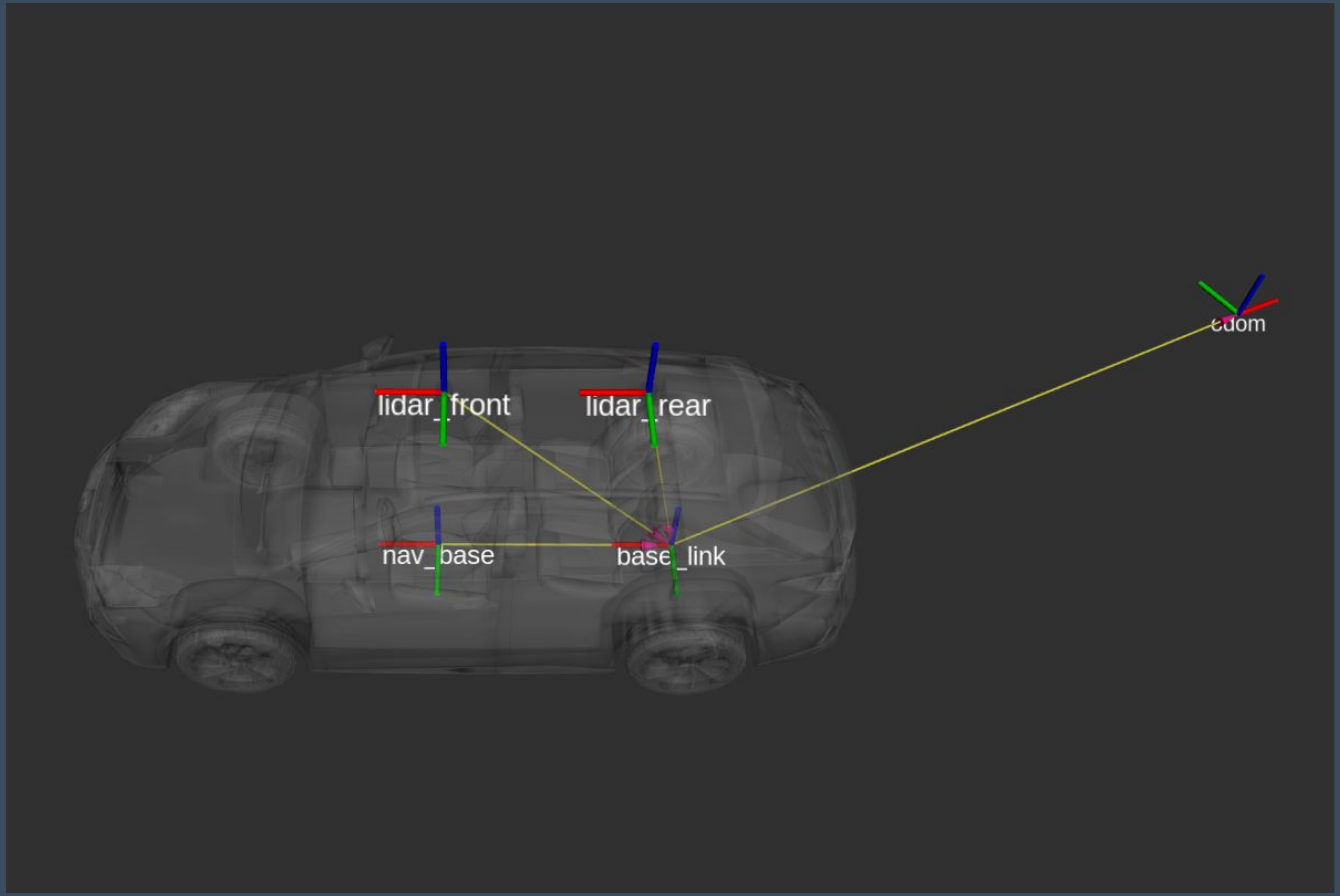
- base_link
- odom
- map

SLAM / Localization (map → odom)

- May be discontinuous or cusping
- Accurate globally in space and time
- Sensors: GPS, SLAM, NDT

Odometry (odom → base_link)

- Continuous and smooth
- Accurate locally in time / space
- Sensors: wheel encoders, IMUs, VIO



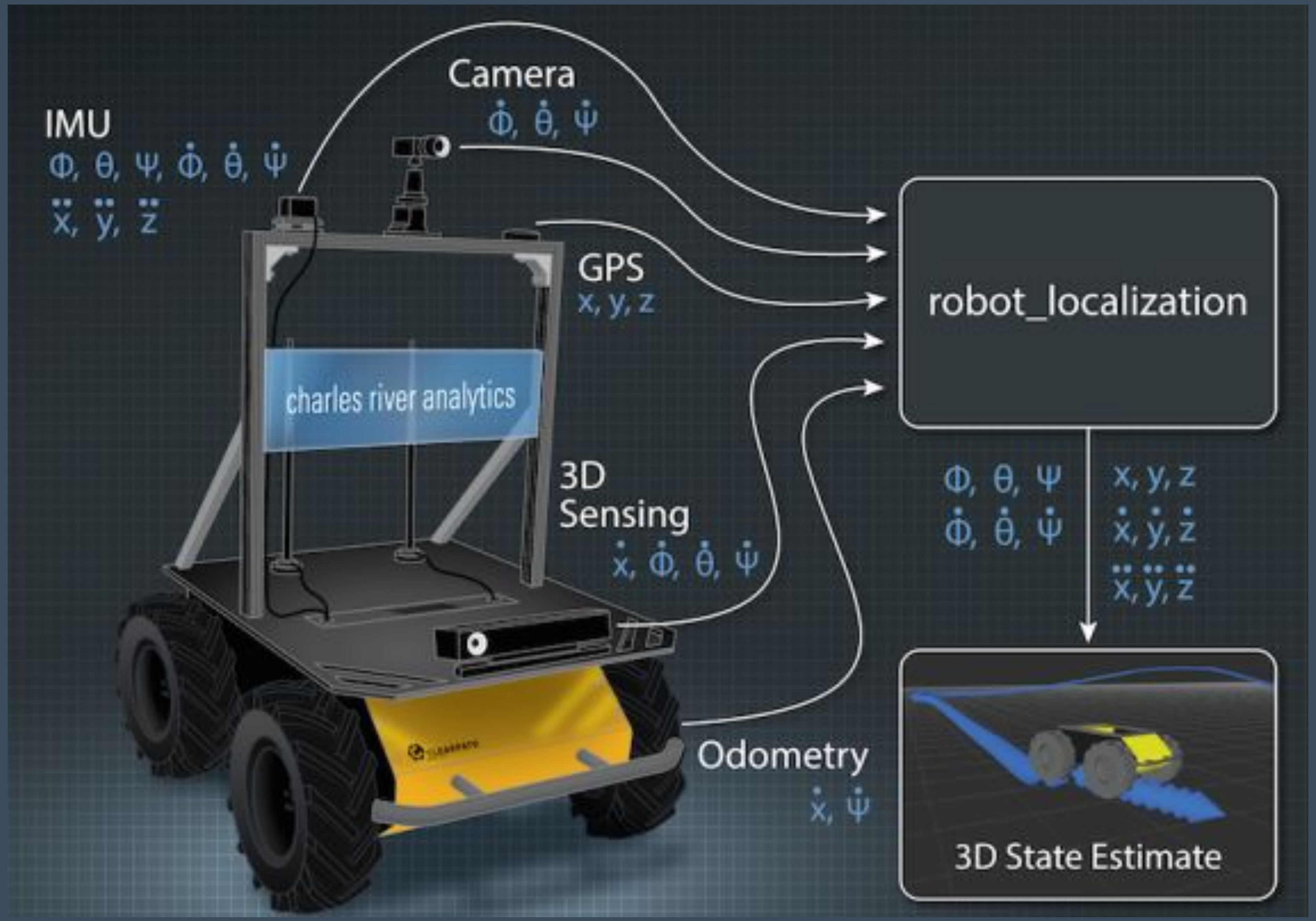
External Transformations Provided by Complete map → base_link Tree

Robot Localization (1/2)

ROS package from Tom Moore while at Charles River Analytics

Robot Localization Non-Linear Filters:

- EKF and UKF
- Many sensors
- Variable rates
- Subset of sensor data



Why is that important?

- Multiple source of the same data coming in
- Sensors are asynchronous
- Data comes in at different rates and qualities
- Some sensor data might be bad
- Turn off bad data better than outlandish covariances

Robot Localization (2/2)

Robot Localization Also Enables GPS Navigation

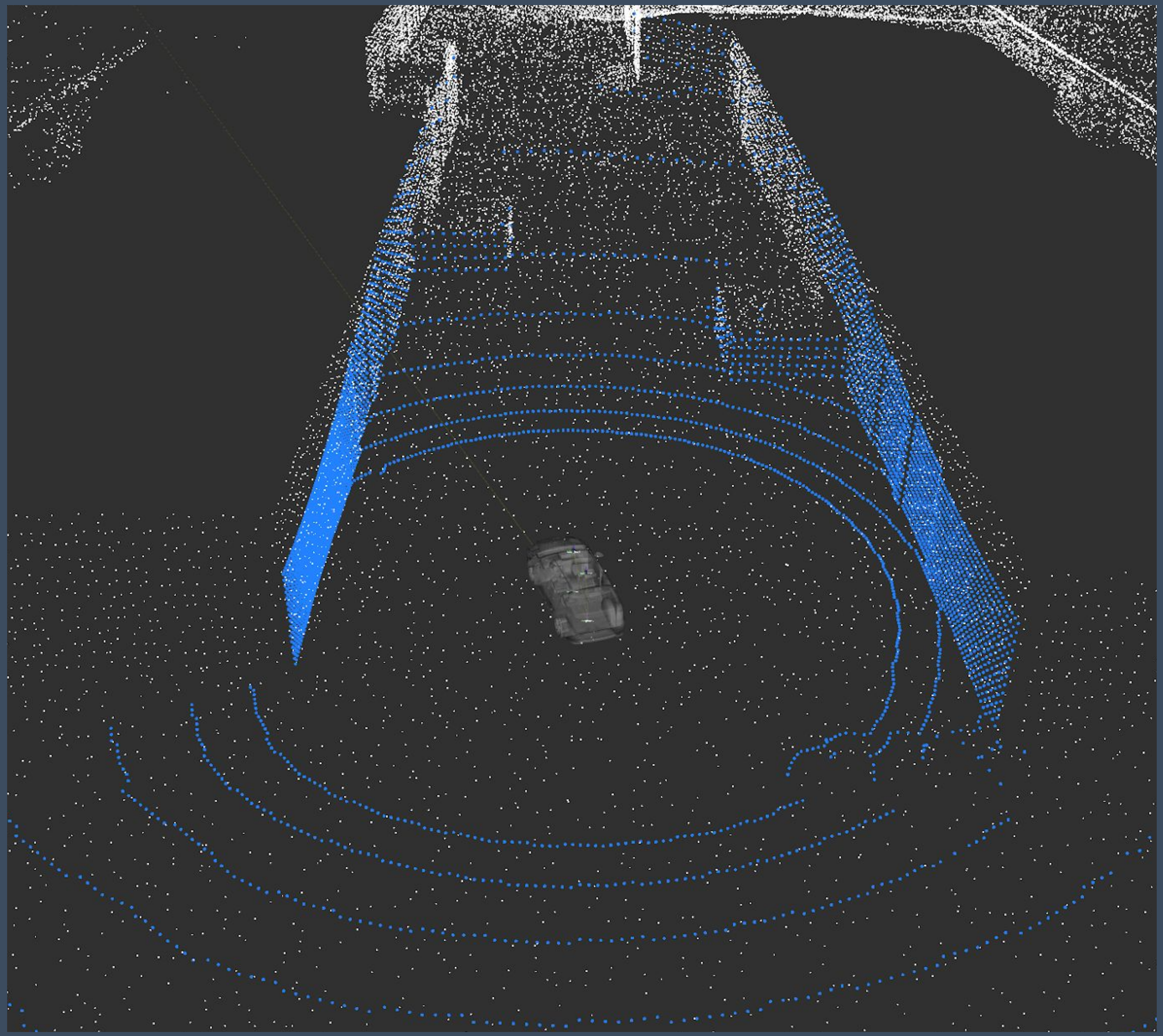
- Nav Sat Transform

It may be used for both local and global filtering

- Filter noisy single global pose sources (NDT)
- Filter multiple global pose sources together
- Filter multiple local sources for odometry

Covariances

- Sensor: measure of confidence in a data source
- Process: measurement of confidence in the model/filter
- Lower is “more confident” - less variance in inputs
- Often implemented as a main diagonal matrix of variances
  - e.g. no “co”-variances (independent)



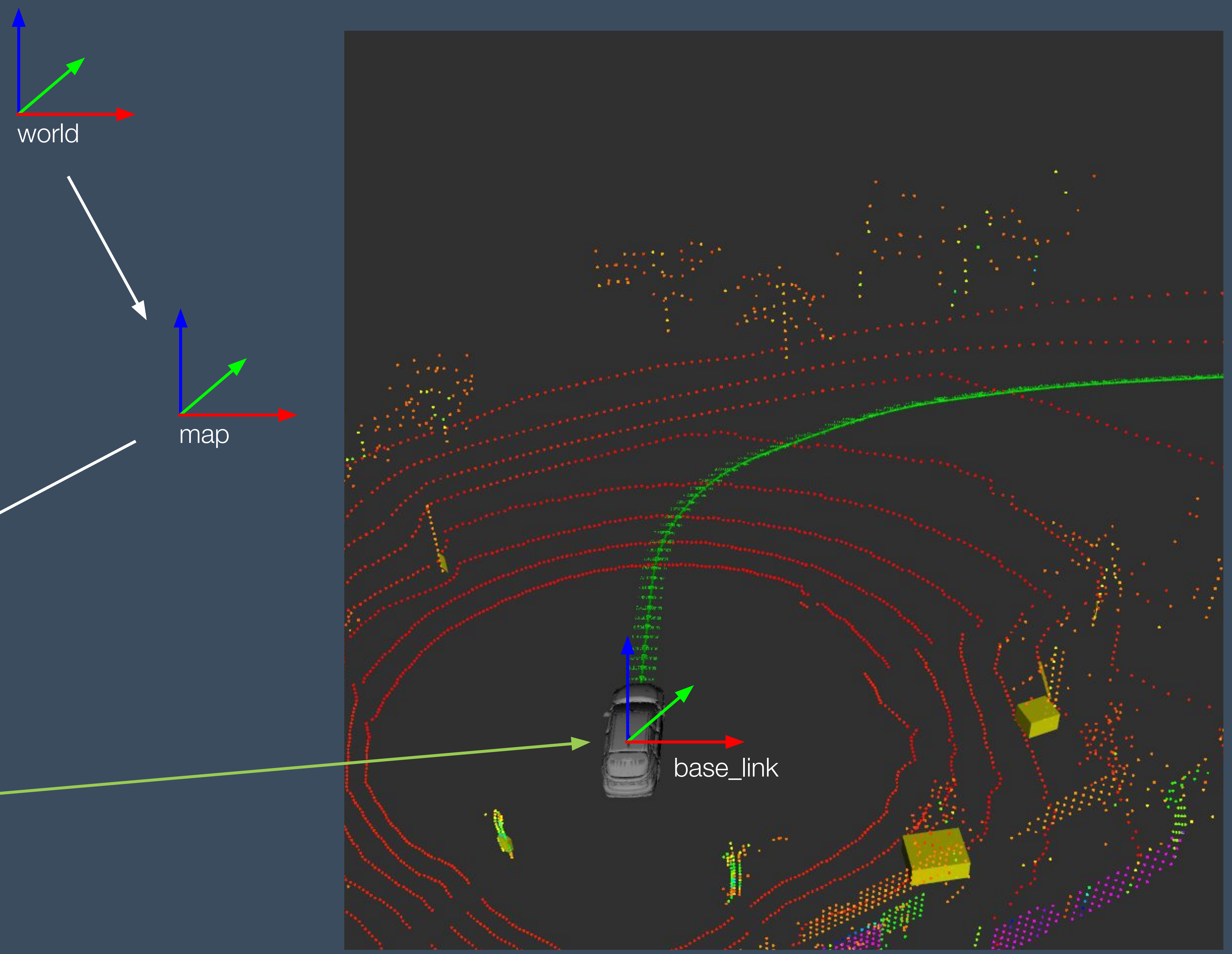
Autoware Odometry

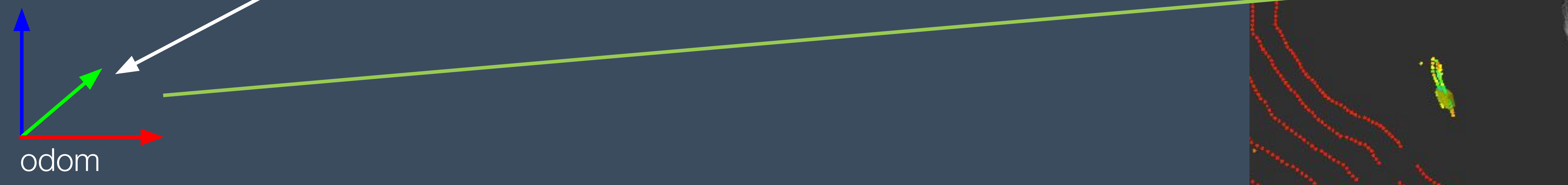
Robot Localization

- Using EKF Node
- Why not UKF?
  - May be slightly more accurate
  - Also more compute for similar results

Sensors

- Simulation: IMU, “GNSS” odometry *
- Hardware: IMU, wheel odometry
- What else?
  - RADAR
  - LIO
  - VIO





* /gnss/pose is smoothed odometry, LGSVL doesn’t properly represent odometry, GPS is fused for global positioning **only**

Autoware Odometry Configuration

- 50 Hz
- map_frame: map
- odom_frame: odom
- base_link_frame: base_link
- world_frame: odom

- Fusing 2 devices: a Pose and an IMU
  - Pose
    - Odometry source from simulator or hardware wheel encoders
    - Fusing XYZ and RPY, no velocities or accelerations
  - IMU
    - IMU source from simulator and hardware
    - Fusing RPY, angular velocities, and accelerations
    - Generally not recommended: Only RPY and no accelerations

- Default process and initial covariances

```yaml
pose0: /gnss/pose
pose0_config: [true,  true,  true,
               true,  true,  true,
               false, false, false,
               false, false, false,
               false, false, false]
pose0_queue_size: 10
pose0_nodelay: false
pose0_differential: false
pose0_relative: true
```

```yaml
imu0: /imu/imu_raw
imu0_config: [false, false, false,
              true, true, true,
              false, false, false,
              true, true, true,
              true, true, true]
imu0_nodelay: false
imu0_differential: false
imu0_relative: false
imu0_queue_size: 10
imu0_remove_gravitational_acceleration: true
```

RECAP

- Kalman Filters
- Non-Linear Filters
- Robot Localization
- Autoware Odometry

Demo

GitLab: *https://gitlab.com/autowarefoundation/autoware.auto*

Documentation: *https://autoware.readthedocs.io/*

Steve Macenski
Open Source Robotics Engineering Lead
s.macenski@samsung.com
stevenmacenski@gmail.com