Northeastern University

EECE 5554 Mark Higger - Joseph Lynch - Chen Xu

Integration of the Ackermann Steering Model to Improve State Estimation for Autonomous Vehicles

Ackermann Model Overview

Ackermann Model:

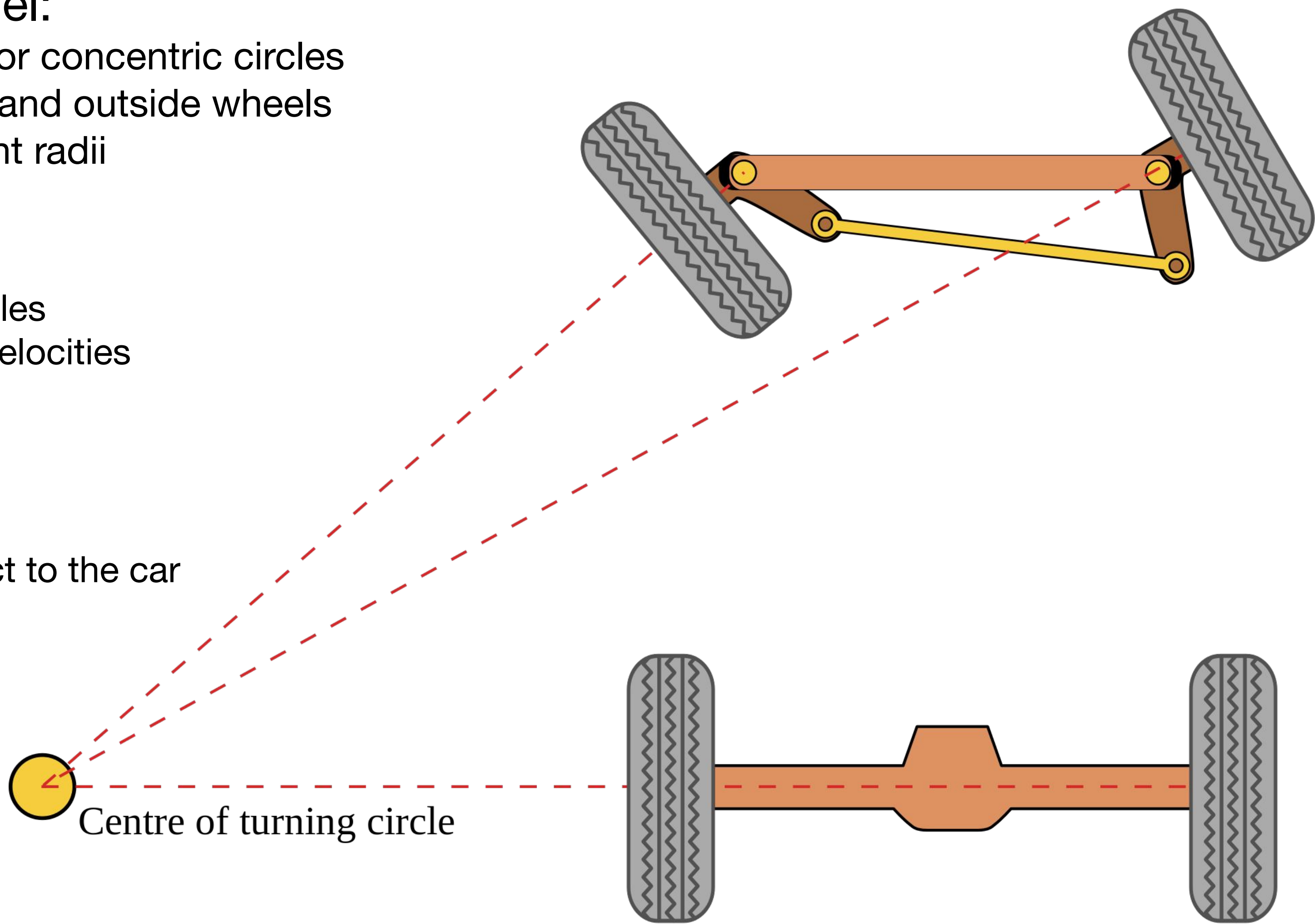
Geometric solution for concentric circles traced out by inside and outside wheels of a turn with different radii

ROS Node Inputs:

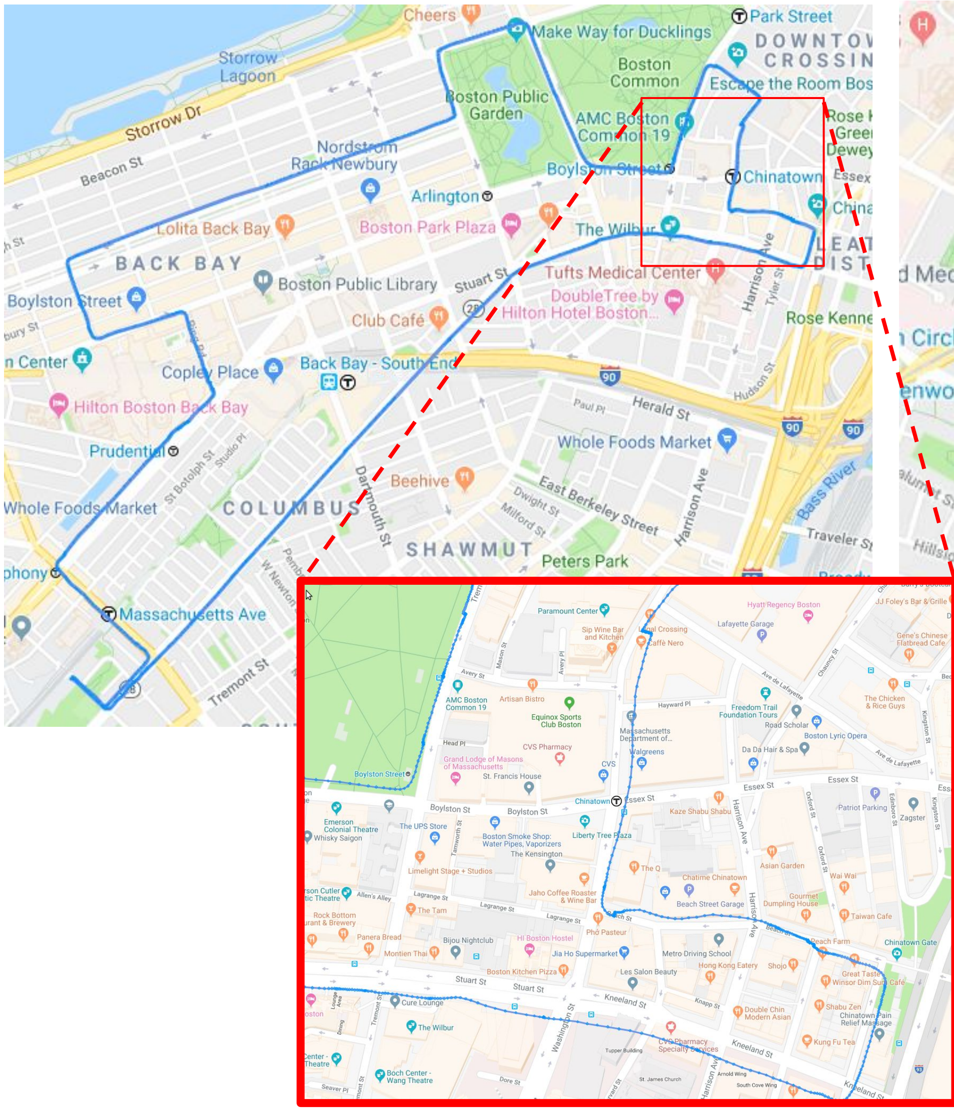
- Front Wheel Angles
- Wheel Angular Velocities

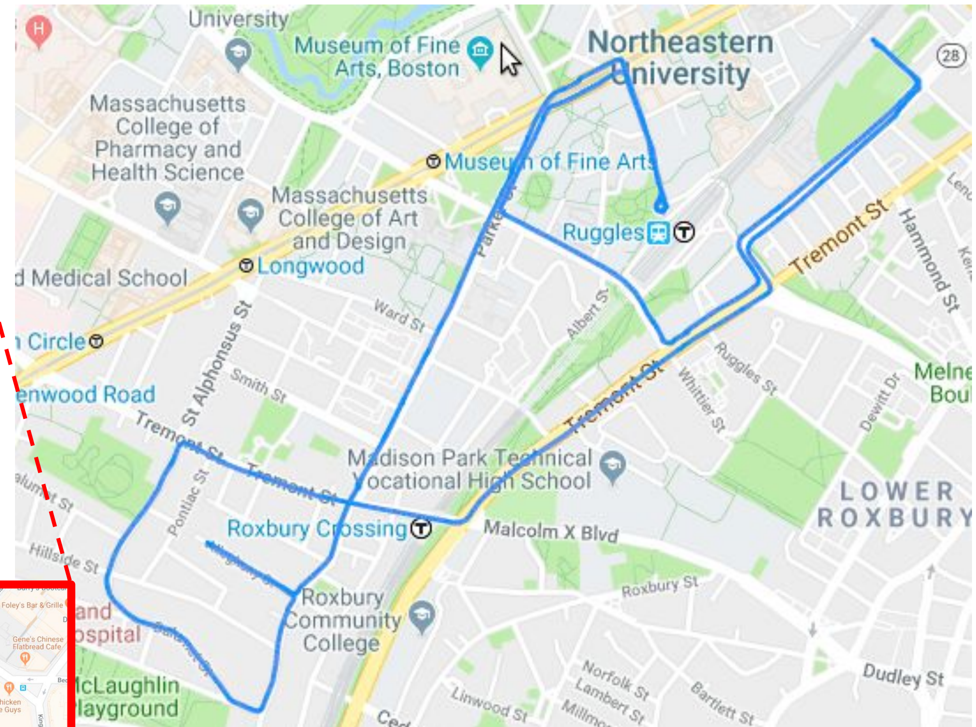
ROS Node Outputs:

- X + Y Velocity
- Angular Velocity
- Data with respect to the car base_link frame



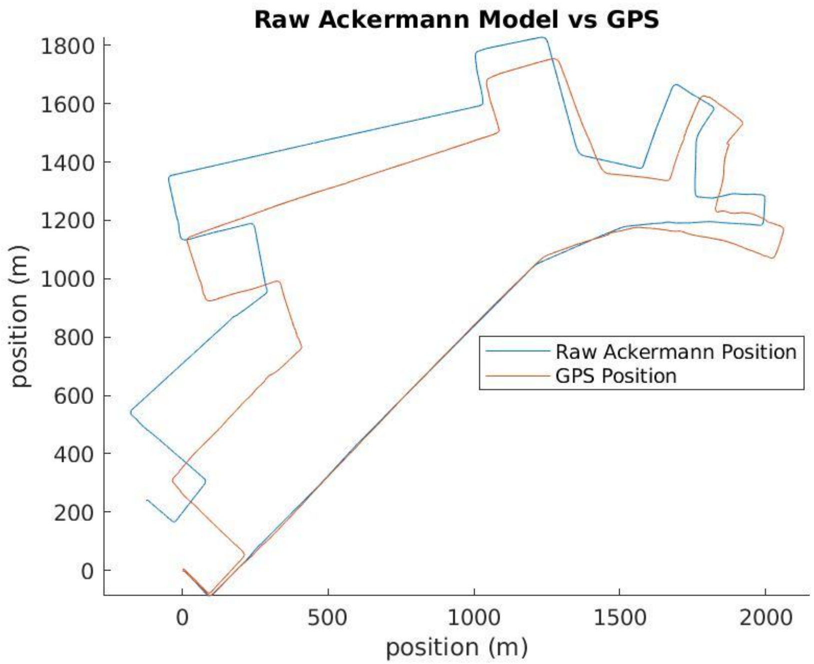
Position from GPS



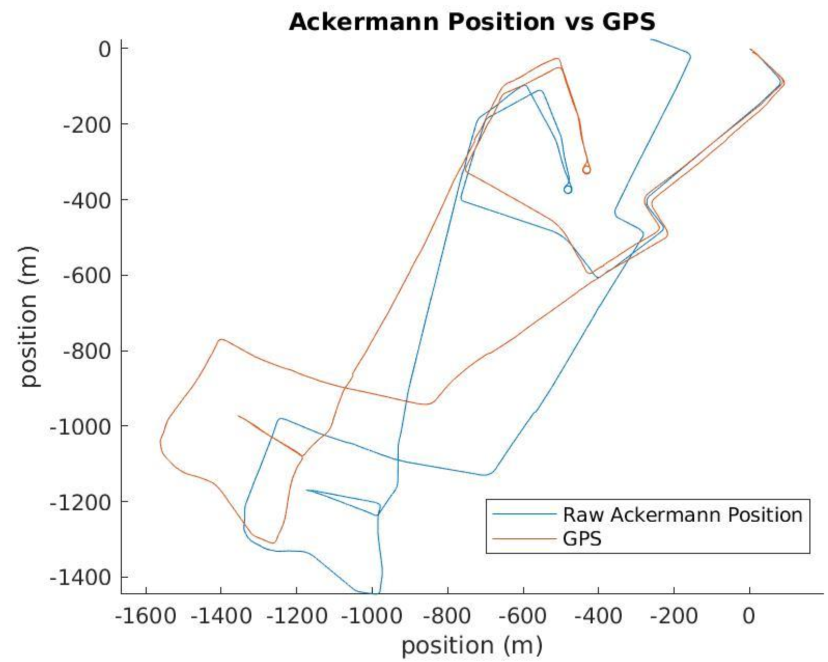


Note: The two paths driven for collecting data the path on the left was chosen to get bad GPS data by driving through narrow streets with tall buildings. The inset shows further detail on the bad GPS data. The path on the right was done to test the limits of the Ackermann model by driving around the Forsyth circle and uphill

Position from Ackermann Model



PATH I: Driving course through downtown Boston

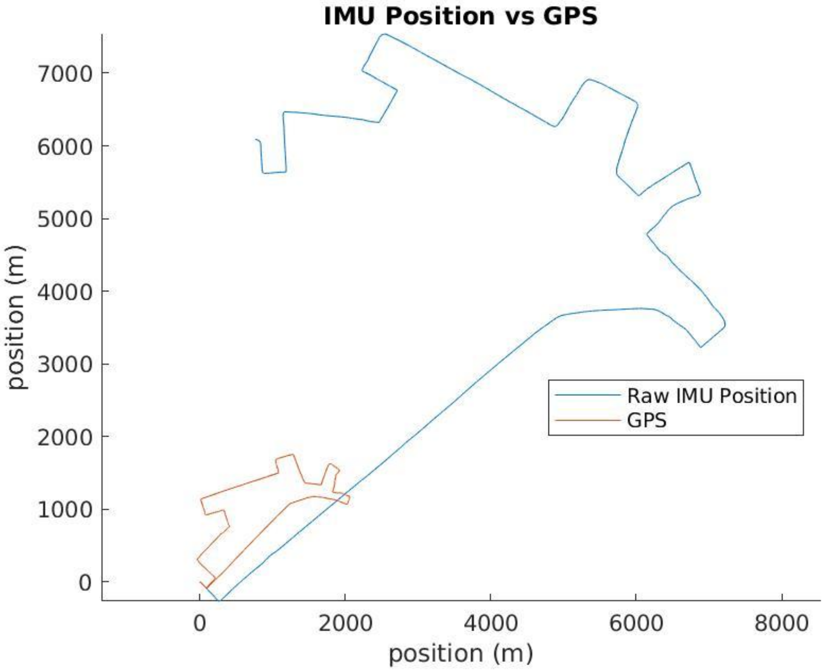


PATH II: Driving course through Mission Hill + the Forsyth circle

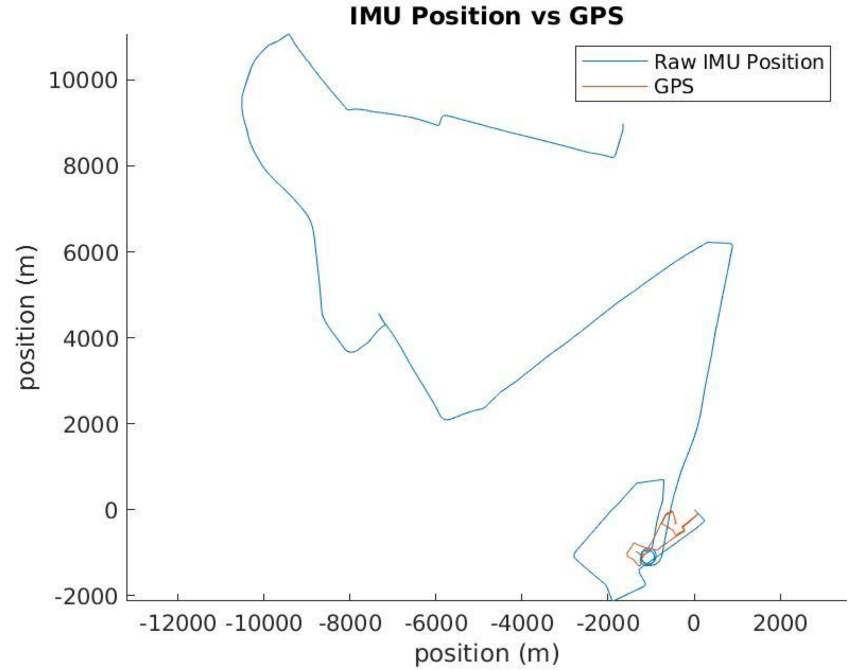
For PATH I and II, the initial ackermann model without kalman filter actually can record the distance and turn roughly. It behaves well on straight line but has a accumulated error in turning angle each time.

Northeastern University

Position from IMU



PATH I: Driving course through downtown Boston

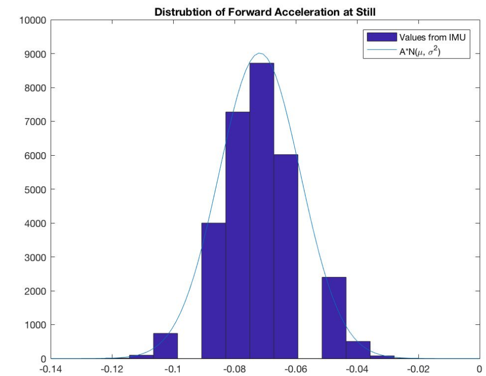


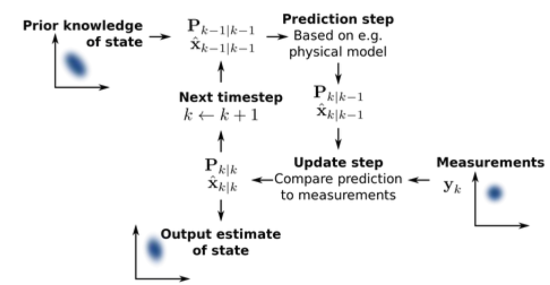
PATH II: Driving course through Mission Hill + the Forsyth circle

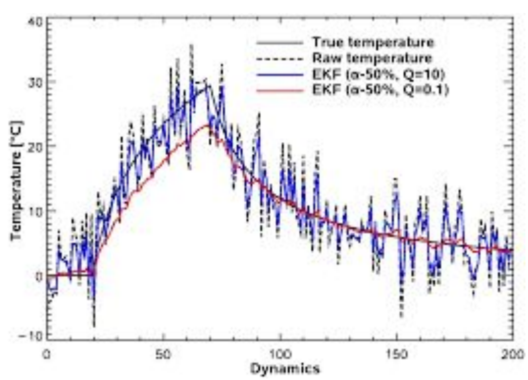
For PATH I and II, the IMU data has bad error on distance and turning angle. The error is mainly due to double integration from linear and angular accelerations. But PATH I record each turn one by one. PATH II creates a total mess for the position estimate.

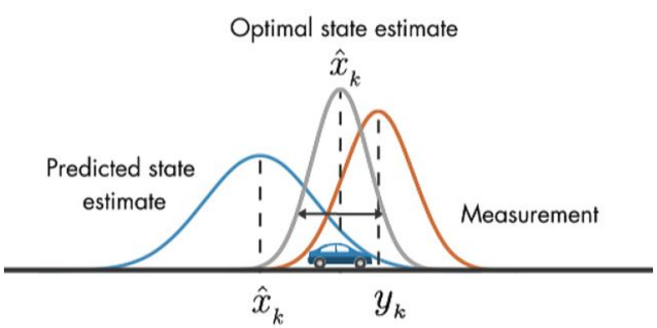
The Kalman Filter

- Measurements are not accurate
  - Can be modeled as Gaussian
    - Normal Distribution $\sim \mathcal{N}(\mu, \sigma^2)$
- Kalman Filter
  - Estimates States based off Previous State and External Measurements



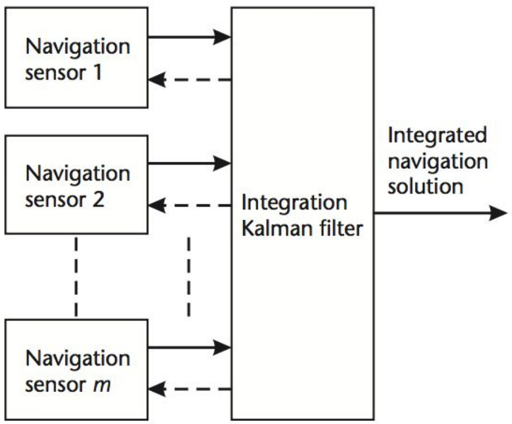






Centralized Sensor Integration

Total State Integration (ROS Robot_Localization Package)



$$\hat{x} = \begin{bmatrix} \hat{x}_{imu} \\ \hat{x}_{gps} \\ \hat{x}_{Ack} \end{bmatrix}$$

$$\hat{x}_i = \begin{bmatrix} \hat{\Psi}_i \\ \hat{\omega}_i \\ \hat{a}_i \\ \hat{v}_i \\ \hat{p}_i \end{bmatrix}$$

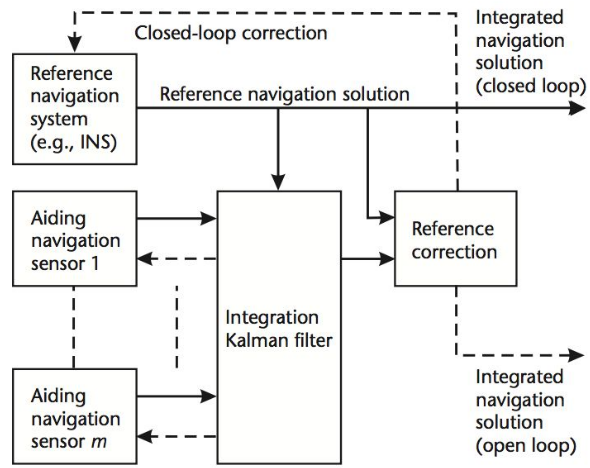
Ψ = Attitude, $\omega = \frac{d\Psi}{dt}$

p = position, $v = \frac{dp}{dt}, a = \frac{d^2p}{dt^2}$

b_a = Accelerometer bias

b_g = Gyro bias

State Error Integration (Matlab)

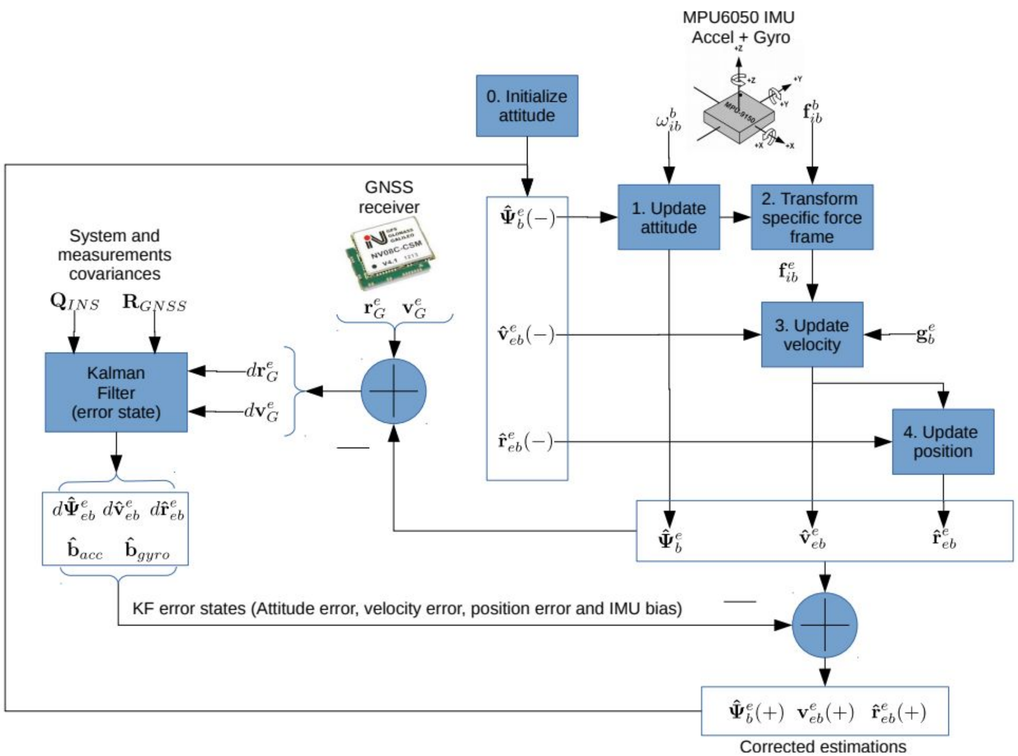


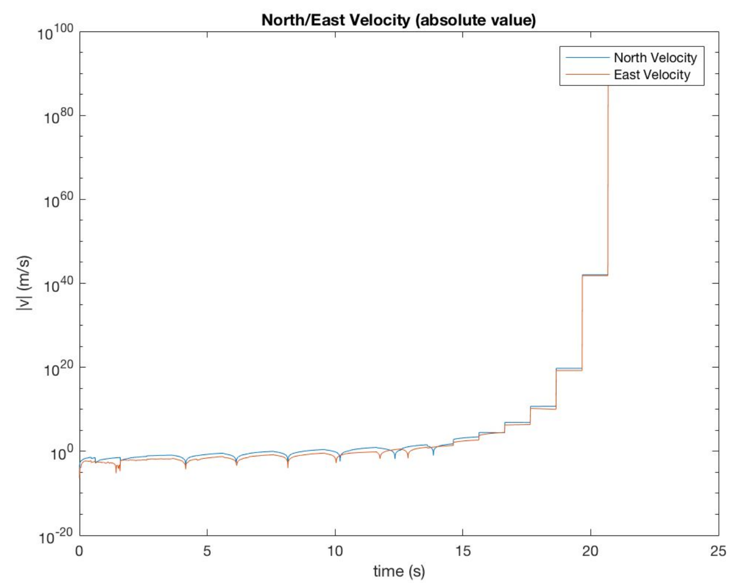
$$\hat{x} = \begin{bmatrix} \delta\hat{\Psi} \\ \delta\hat{v} \\ \delta\hat{p} \\ \hat{b}_a \\ \hat{b}_g \end{bmatrix}$$

Loose IMU/GNSS Integration

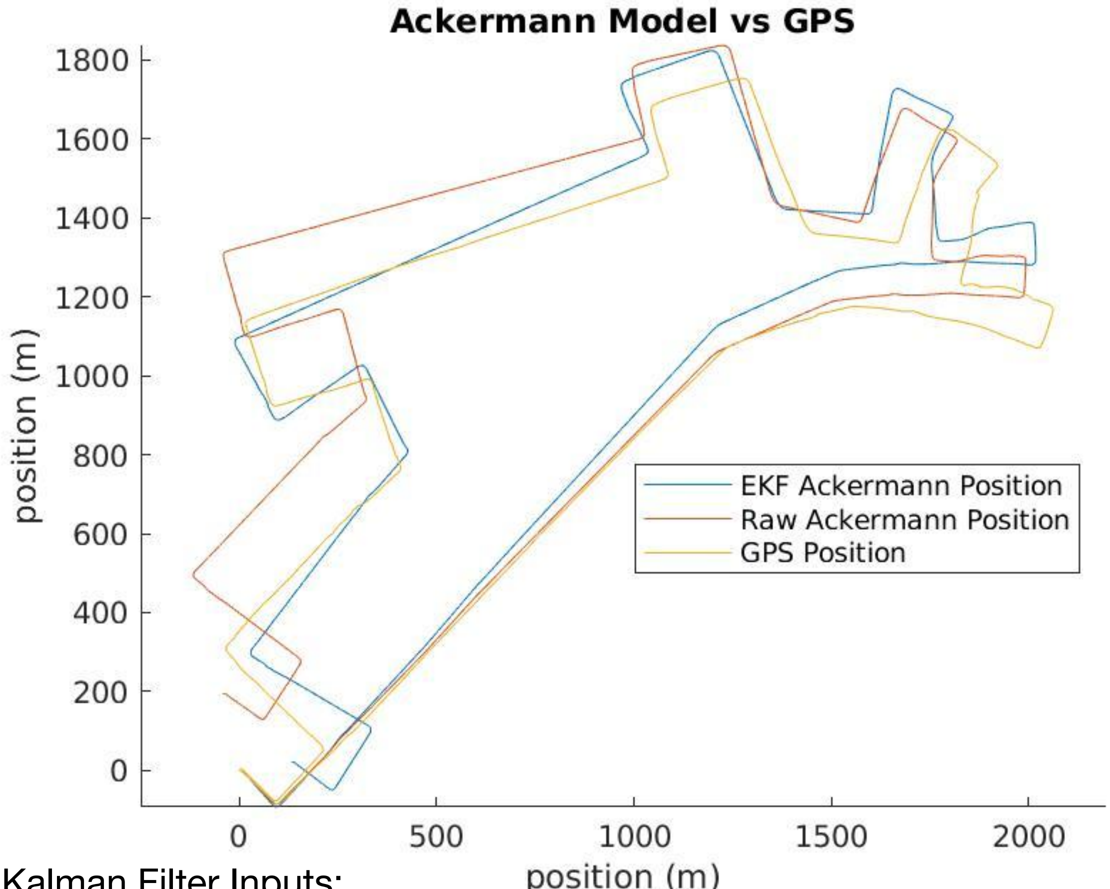
- Attempt to use KF integration

$$
\mathbf{F}_{INS}^{n} = \begin{pmatrix} \mathbf{F}_{11}^{n} & \mathbf{F}_{12}^{n} & \mathbf{F}_{13}^{n} & \mathbf{0}_3 & \hat{\mathbf{C}}_b^n \\ \mathbf{F}_{21}^{n} & \mathbf{F}_{22}^{n} & \mathbf{F}_{23}^{n} & \hat{\mathbf{C}}_b^n & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{F}_{32}^{n} & \mathbf{F}_{33}^{n} & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \end{pmatrix}
$$





ROS EKF - Ackermann Model Only



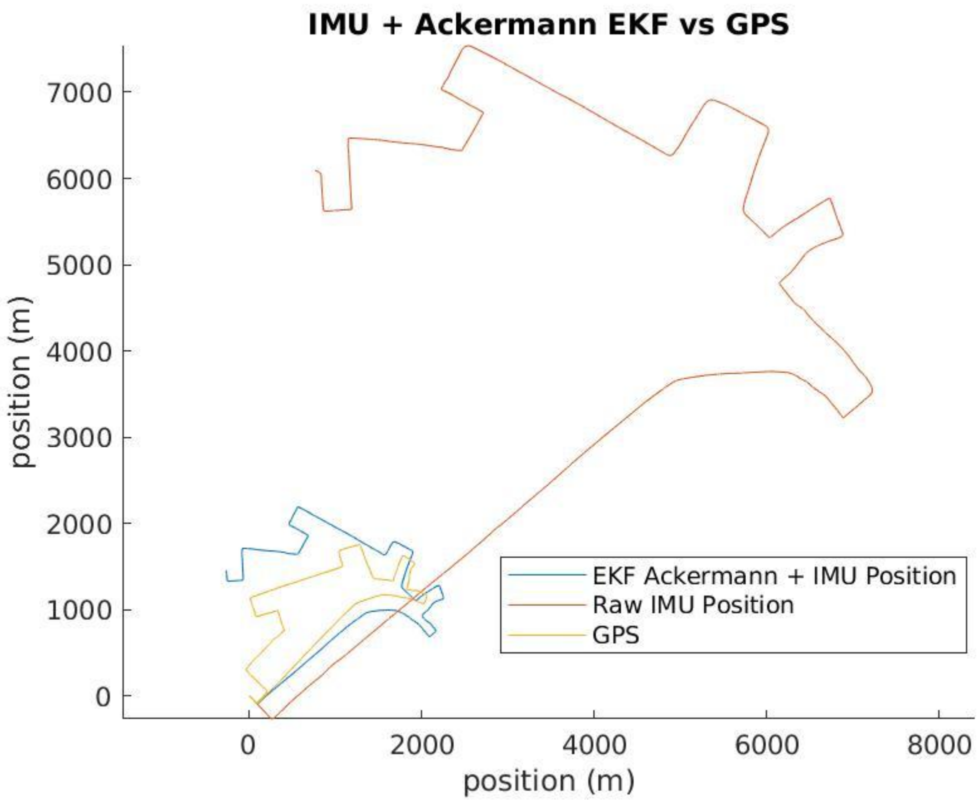
	End Position Drift
GPS	4.07 m
Raw Ackermann Position	198.47 m
EKF Ackermann Position	134.02 m

- The Ackermann kalman filtered position shows an improvement over the raw position as expected
- There is still some error due to sensor covariances not being tuned properly and due to this kalman filter only taking input from one sensor model

Kalman Filter Inputs:

	X	Y	Z	Roll	Pitch	Yaw	vX	vY	vZ	vRoll	vPitch	vYaw	aX	aY	aZ
Ackermann							✓	✓				✓			

ROS EKF - IMU + Ackermann Model



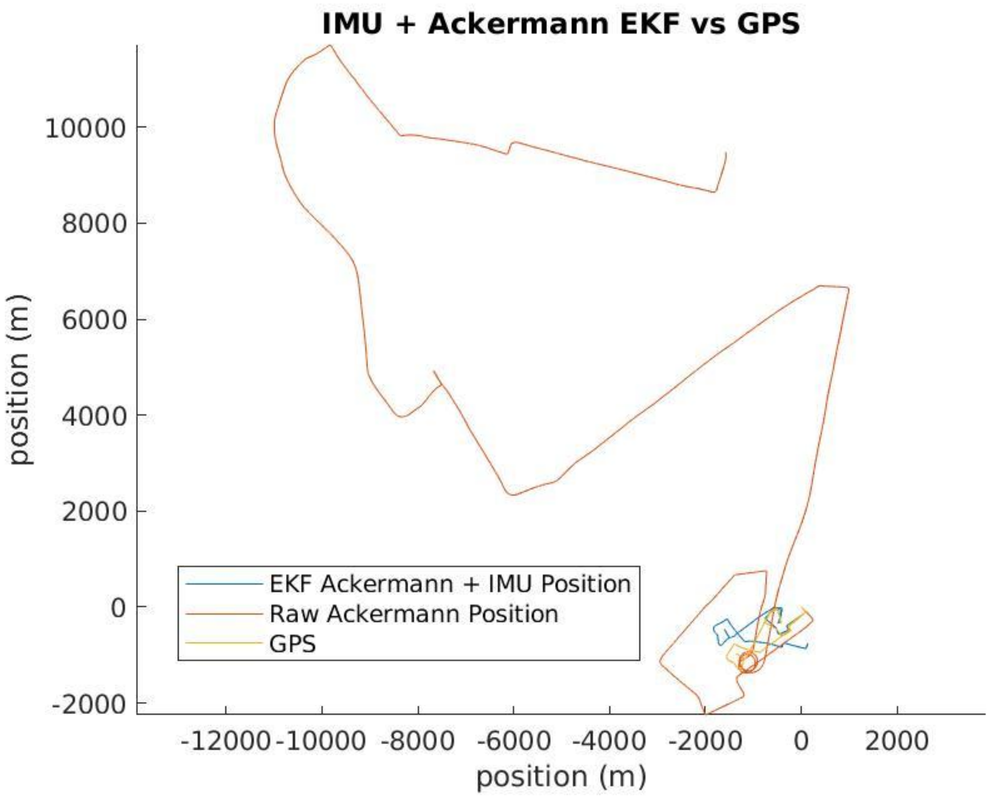
	End Position Drift
GPS	4.07 m
Raw Ackermann Position	198.47 m
Raw IMU Position	6139.4 m
EKF Ackermann + IMU Position	1481.8 m

- The Kalman filtered position shows an improvement over the raw IMU position
- The x + y velocities from the kalman filter get rid of the IMU position drift error
- There is still a large amount of error in the heading of the car due to sensor covariances not being tuned properly

Kalman Filter Inputs:

	X	Y	Z	Roll	Pitch	Yaw	vX	vY	vZ	vRoll	vPitch	vYaw	aX	aY	aZ
Ackermann							✓	✓							
IMU				✓	✓	✓				✓	✓	✓	✓	✓	✓

ROS EKF - IMU + Ackermann Model Bad Data



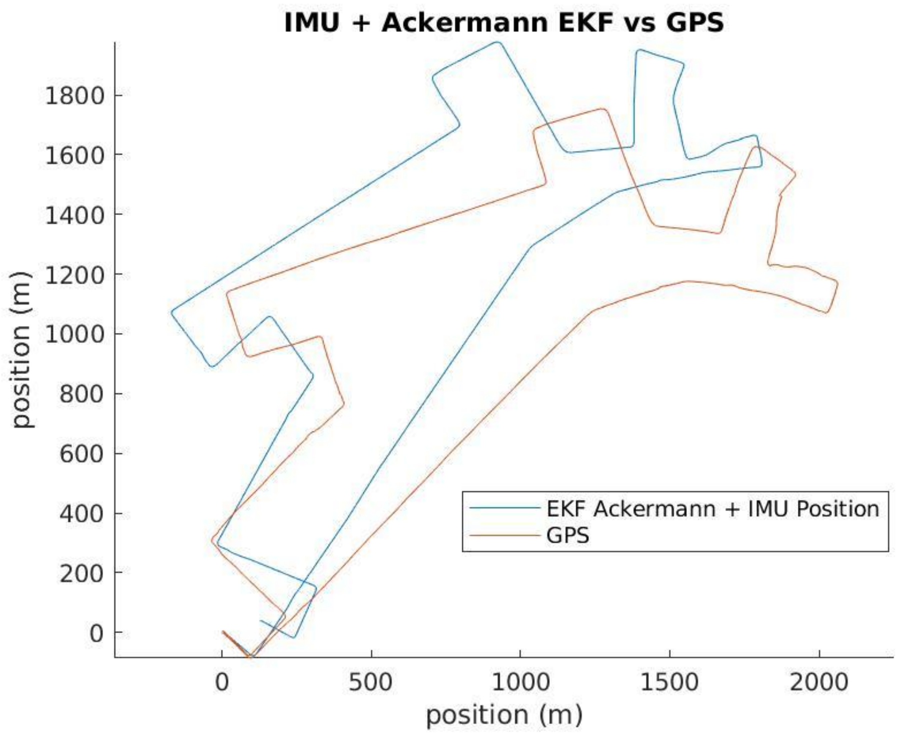
	End Position Drift
GPS	11.6 m
Raw Ackermann Position	264.7 m
Raw IMU Position	9609.9 m
EKF Ackermann + IMU Position	763.9 m

- The Kalman filtered position shows an improvement over the raw IMU position
- This analysis was ran on the course with the forsyth circle and hill data. It is promising to see that these bad data points don't have much effect on the kalman filtered position
- The remaining error in the heading of the car is due to sensor covariances not being tuned properly

Kalman Filter Inputs:

	X	Y	Z	Roll	Pitch	Yaw	vX	vY	vZ	vRoll	vPitch	vYaw	aX	aY	aZ
Ackermann							✓	✓							
IMU				✓	✓	✓				✓	✓	✓	✓	✓	✓

ROS EKF - IMU + Ackermann Model - Improved



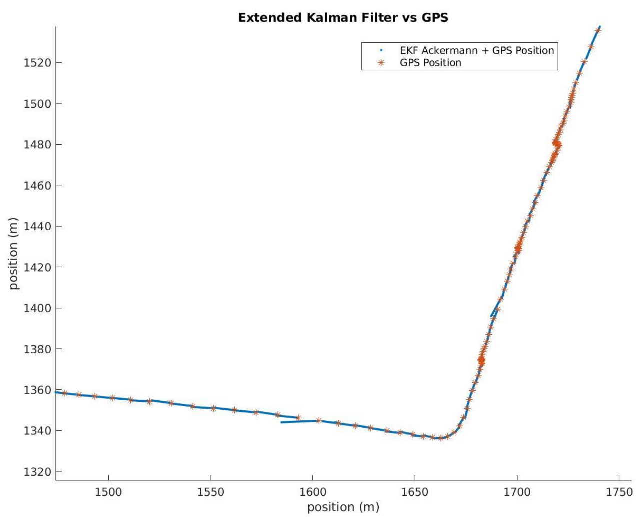
	End Position Drift
GPS	4.07 m
Raw Ackermann Position	198.47 m
Raw IMU Position	6139.4 m
EKF Ackermann + IMU Position Original	1481.8 m
EKF Ackermann + IMU Position Improved	133.43 m

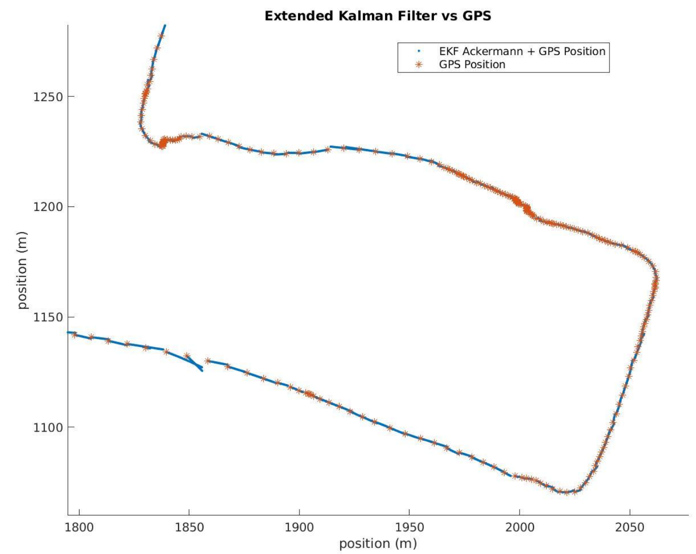
- This Kalman filtered position calculating orientation from just the Ackermann model shows a large improvement (1348.37 m) over the kalman filtered position when calculating orientation from the IMU
- There is still a large bit of error in the final position of the car due to sensor covariances not being tuned properly

Kalman Filter Inputs:

	X	Y	Z	Roll	Pitch	Yaw	vX	vY	vZ	vRoll	vPitch	vYaw	aX	aY	aZ
Ackermann							✓	✓				✓			
IMU													✓	✓	✓

ROS EKF - GPS + Ackermann Model



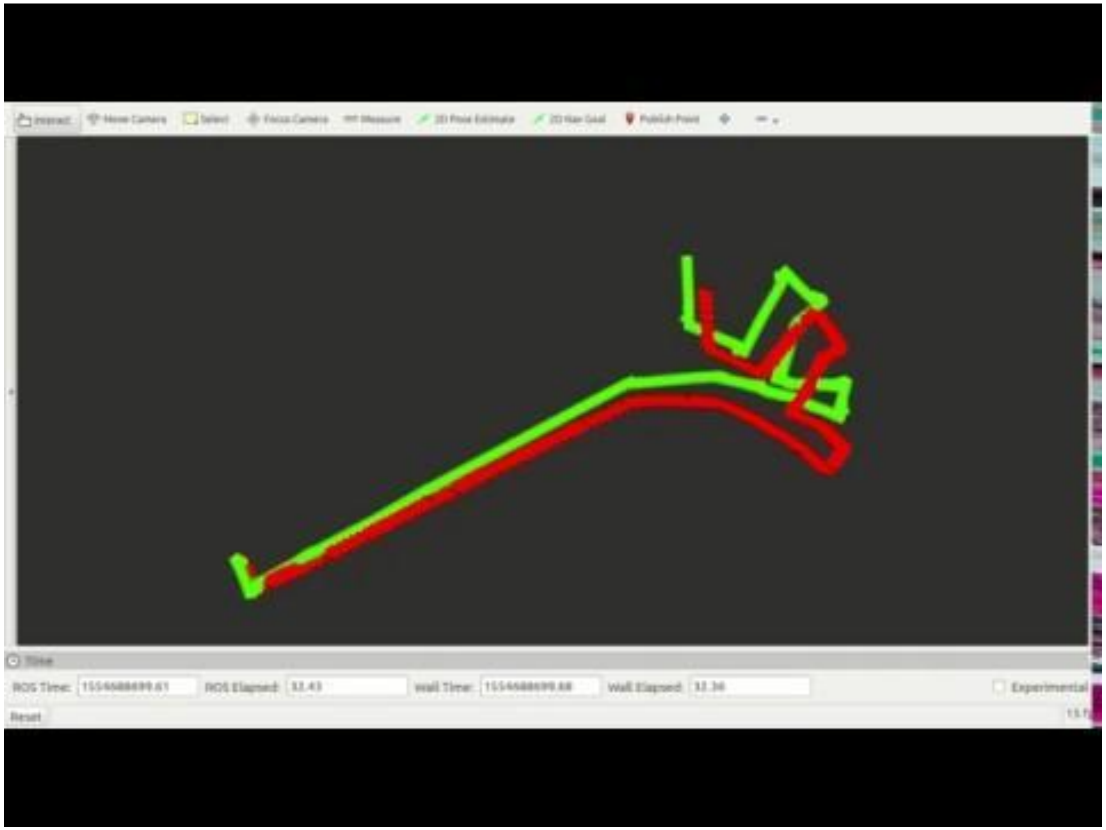


- These plots show zoomed in sections of the GPS + Ackermann data kalman filtered.
- This is an improvement over GPS data only as we now have continuous state estimates.
- However the covariances still need some tuning as this kalman filter implementation places too much trust in the GPS position data

Kalman Filter Inputs:

	X	Y	Z	Roll	Pitch	Yaw	vX	vY	vZ	vRoll	vPitch	vYaw	aX	aY	aZ
Ackermann							✓	✓				✓			
GPS	✓	✓													

Real-time Data Visualization



- The benefits of the kalman filter is that it requires relatively low processing power and can be run in real time
- This is a sped-up video demonstration of our kalman filtering running on ROS and plotting the GPS position (red) and estimated kalman filter position (green) in RVIZ

Future Work

- Tune Kalman Filter
  - Improve covariance estimates for sensors
  - Previous work is a proof of concept for level of accuracy that can be achieved

- Other ways to integrate sensors
  - Zero-Update kalman Filter
  - Tight GPS integration

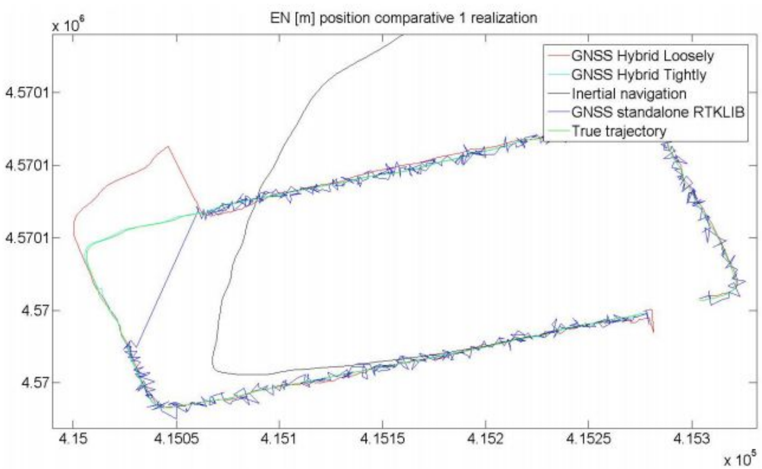


Fig. 7: Output of *ekf_localization_node* (green) when fusing data from odometry, two IMUs, and two GPS units.

References

- https://en.wikipedia.org/wiki/Ackermann_steering_geometry
- “Low-cost GNSS/INS/Odometric sensor fusion platform for ground intelligent transportation systems”, Arribas, Moragrega, Fernandez–Prade, Closas
  - Loose Integration Block Diagram
  - Future result possibilities
- “Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems”, Paul D. Groves
  - Centralized Sensor Integration Block Diagrams
- https://docs.ros.org/api/robot_localization/html/_downloads/robot_localization_ias13_revised.pdf
- “Robust Adaptive Extended Kalman Filtering for Real Time MR-Thermometry Guided HIFU Interventions”, Roujal et. al.
  - EKF example
- http://docs.ros.org/melodic/api/robot_localization/html/index.html
- Wikipedia “Kalman Filter”
  - Kalman Filter Diagram