# Module 5: Putting It together - An Autonomous Vehicle State Estimator

内容
- Sensor fusion
- Calibration
- Sensor failures
- Final project

## Lesson 1: State Estimation in Practice

内容
- Identify practical considerations that must be taken into account when implementing a state estimator on a real vehicle

例えば、sensor fusion and calibration, speed and accuracy requirements, localization failures, parts of the environment that are moving and changing around us.

Calibration: What do we need to know about sensors and vehicle to do sensor fusion?
- Sensor models, which may depend on car-specific parameters (e.g., wheel radius).
- 同じ座標系に変換：Relative poses between each sensor pair, so we can combine information in a common reference frame.
- 同期：Time offsets between sensor polling times, so we can combine only synchronized information.

Accuracy Requirements
- How accurate does the estimator need to be for safe self-driving?
- Typically less than a meter for highway lane keeping.
  - Less for driving in dense traffic.
- GPS accuracy is 1-5 meters in optimal conditions.
  - Need additional sensors.

Speed
- How fast do we need to update the vehicle state to ensure safe driving?
  - an update rate of 15Hz to 30Hz is a reasonable target for self-driving.
- How much computation power does the vehicle have on-board?
- How much power can the computing resources consume?

Localization Failures
- How can localization fail?
  - Sensors fail or provide bad data (e.g., GPS in a tunnel, GPS can have a difficult time coping with reflected signals in cities with a lot of tall buildings).
  - Estimation error (e.g., linearization error in the EKF).
  - Large state uncertainty (e.g., relying on IMU for too long).
    - The uncertainty in state grows as we propagate forward through the motion model and it only shrinks once we incorporate outside observations from LiDAR or GPS for example.

Our Dynamic World
- Many of the models we use in practice for sensors like LiDAR, Radar, cameras, etc. assume that the world is static and unchanging.
- Need to account for the moving and changing world in models, or find ways of ignoring objects that don't fit our assumptions.

# Lesson 2: Multisensor Fusion for State Estimation

内容

- Develop an error state EKF for estimating position, velocity and orientation using an IMU, GNSS sensor, and LiDAR.

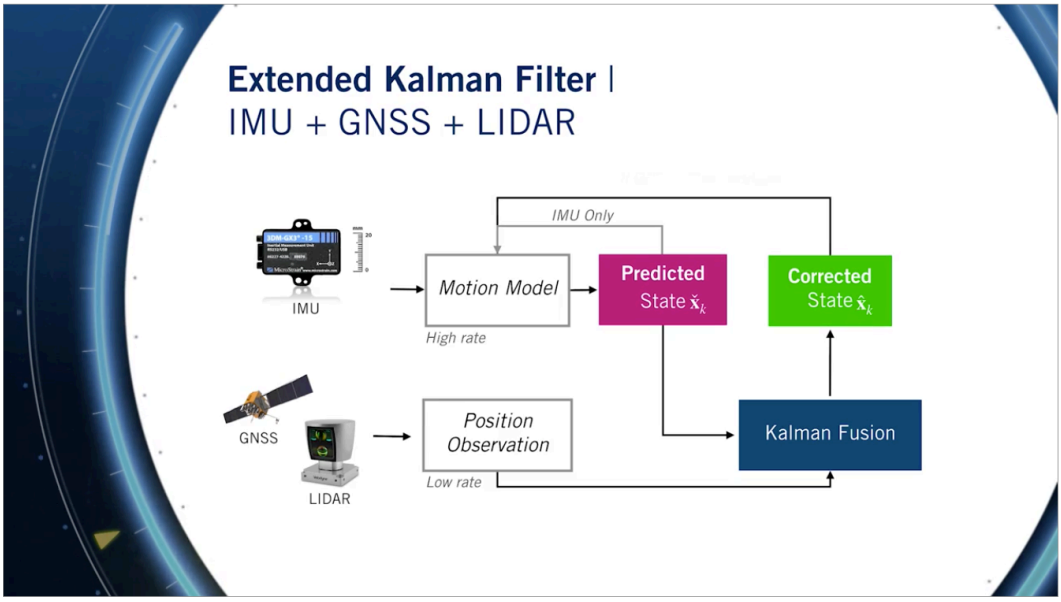
Why use GNSS with IMU & LiDAR?

- Error dynamics are completely different and uncorrelated.
  - correlatedの意味：if one fails, is the other likely to fail as well.
- Choose sensors that are complimentary in nature.
  - IMU acts as a high-rate smoother of GPS or GNSS position estimates.
    - Wheel odometry is also possible (if only 2D position orientation is desired)
  - GNSS data can mitigate errors that are due to IMU drift.
  - LiDAR can compliment GNSS information, by providing very accurate position estimates within a known map and in sky obstructed locations. このknow mapは僕が開発したMapAPIのタスクだ。
    - Conversely, GNSS can tell LiDAR which map to use when localizing.

Tightly vs. Loosely Coupled

- Loosely coupled: assume the raw data have already been preprocessed to produce a noisy position estimate.
- Tightly couple: tedious to implement and requires a lot of tuning.

| **EKF Coupling** | | |
|---|---|---|
| | Tight | Loose |
| **GNSS/LIDAR Measurement** | Pseudo-ranges to satellites, LIDAR point clouds | Position |
| **Accuracy** | (Potentially) Higher | (Potentially) Lower |
| **Complexity** | **Higher** | **Lower** |




Extended Kalman Filter | IMU + GNSS + LIDAR

- Motion Modelの更新はcan happen hundreds of times a second.

- At a much slower rate, incorporate GNSS and LiDAR measurements whenever they become available, say once a second or slower.

Some preliminaries

- **Vehicle state** consists of position, velocity and parametrization of orientation using a unit quaternion: $x_k = \begin{bmatrix} p_k \\ v_k \\ q_k \end{bmatrix} \in R^{10}$
  - a 3D position, a 3D velocity, and a 4D unit quaternion representing the orientation of the vehicle with respect to a navigation frame.
- **Motion model input** will consist of specific force and rotational rates from IMI: $u_k = \begin{bmatrix} f_k \\ \omega_k \end{bmatrix} \in R^6$.
  - From accelerometer and gyroscope measurements.
  - For clarity, assume IMU measurements are unbiased.

Motion Model

- Position: $p_k = p_{k-1} + \Delta t\, v_{k-1} + \frac{\Delta t^2}{2}\left(C_{ns} f_{k-1} + g\right)$
  - 復習：Accelerometers measure acceleration relative to free-fall - this is also called the proper acceleration or specific force: $a_{meas} = f = \frac{F_{non-gravity}}{m}$.
  - 復習：computed using fundamental equation for accelerometers in a gravity field: $f + g = \ddot{r}_i$.
- Velocity: $v_k = v_{k-1} + \Delta t\left(C_{ns} f_{k-1} + g\right)$
  - つまりnavigation frameのdown方向とgは同じ方向? 路面と並行。
  - 復習：navigation frameはa frame that is fixed w.r.t. the ground. 2種類：NED frame & ENU frame.
- Orientation: $q_k = q_{k-1} \otimes q\left(\omega_{k-1}\Delta t\right) = \Omega\left(q\left(\omega_{k-1}\Delta t\right)\right) q_{k-1}$.
  - keep track of the orientation of sensor frame $s$, w.r.t. the navigation frame $n$.
  - the orientation change is measured in the local sensor frame.
- Where:
  - $C_{ns} = C_{ns}\left(q_{k-1}\right)$
    - 復習：unit quaternionsとrotation matrixの変換：

$$r_b = C(q_{ba}) r_a$$
$$C(q) = (q_w^2 - q_v^T q_v)1 + 2 q_v q_v^T + 2 q_w [q_v]_\times \text{, where } [a]_\times = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix}.$$

    - $C_{ns} = C\left(q_{k-1}\right)$の方が分かりやすいでしょう。
  - $\Omega\left(\begin{bmatrix} q_w \\ q_v \end{bmatrix}\right) = q_w 1 + \begin{bmatrix} 0 & -q_v^T \\ q_v & -[q_v]_\times \end{bmatrix}$.
  - $q(\theta) = \begin{bmatrix} cos\frac{|\theta|}{2} \\ \frac{\theta}{|\theta|} sin\frac{|\theta|}{2} \end{bmatrix}$.

Linearized Motion Model

- Error State: $\delta x_k = \begin{bmatrix} \delta p_k \\ \delta v_k \\ \delta \phi_k \end{bmatrix} \in R^9$.
  - There are some advantages to representing the orientation error in the global or navigation frame, rather than in the local frame.
- Error Dynamics: $\delta x_k = F_{k-1}\delta x_{k-1} + L_{k-1} n_{k-1}$.
  - 復習：元のLinearized motion model:
    $x_k = f_{k-1}(\hat{x}_{k-1}, u_{k-1}, 0) + F_{k-1}(x_{k-1} - \hat{x}_{k-1}) + L_{k-1} w_{k-1}$.
    - つまりテイラー展開式はerror-state dynamicsの理論基礎だ。EKFじゃないと、error-state EKFもできない。EKFじゃないと、テイラー展開を使わないから。error-state EKFかどうかは、$F_{k-1}$と関係ない。$F_{k-1}$は常にMotion Model式のstate（error stateではなく）に対するJacobian行列。
  - 復習：Error state kinematicsに変換：
    $\underbrace{x_k - f_{k-1}(\hat{x}_{k-1}, u_{k-1}, 0)}_{\delta x_k} = F_{k-1} \underbrace{(x_{k-1} - \hat{x}_{k-1})}_{\delta x_{k-1}} + L_{k-1} w_{k-1}$.
- $F_{k-1} = \begin{bmatrix} 1 & 1\Delta t & 0 \\ 0 & 1 & -[C_{ns} f_{k-1}]_\times \Delta t \\ 0 & 0 & 1 \end{bmatrix}$. これの計算はよくわからない。特に３列目。つまりposition, velocity, orientationのそれぞれのMotion Modelのorientationに対する微分。
  - 直感的に言うと、positionはx, y, zなので、orientationとは関係なさそう、0は合理。もしくはpositionは間接的にorientationと関係する、velocity経由。
  - つまりpositionはpositionとvelocityと関係する。
  - velocityはvelocityとorientationと関係する。
  - orientationはorientationと関係する。
- 復習：$\frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial \mathbf{f}}{\partial x_1} & \cdots & \frac{\partial \mathbf{f}}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$.
- $L_{k-1} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$, $n_k \sim \mathcal{N}(0, Q_k) \sim \mathcal{N}\left(0, \Delta t^2 \begin{bmatrix} \sigma_{acc}^2 & \\ & \sigma_{gyro}^2 \end{bmatrix}\right)$.
  - 1は$3 \times 3$ identity matrix.

Measurement Model | GNSS

$$\begin{aligned} y_k &= h(x_k) + v_k \\ &= H_k x_k + v_k \\ &= [1\ 0\ 0] x_k + v_k \\ &= p_k + v_k \end{aligned}$$

$v_k \sim \mathcal{N}(0, R_{GNSS})$.

- use a very simple observation of the position, plus some additive Gaussian noise.

Measurement Model | LIDAR

$$y_k = h(x_k) + v_k$$
$$= H_k x_k + v_k$$
$$= [1\ 0\ 0]x_k + v_k$$
$$= p_k + v_k$$

. GNSSのmeasurement modelと一緒。

$v_k \sim \mathcal{N}(0, R_{LIDAR})$.

- We've assumed that LIDAR and GNSS will supply position measurements in the same coordinate frame.
- In a realistic implementation, the GNSS position estimates may serve as a way to select a known map against which the LIDAR data can then be compared.

EKF | IMU + GNSS + LIDAR
Loop:

1. Update state with IMU inputs: $\check{x}_k = \begin{bmatrix} \check{p}_k \\ \check{v}_k \\ \check{q}_k \end{bmatrix}$, $\check{p}_k = p_{k-1} + \Delta t\, v_{k-1} + \frac{\Delta t^2}{2}\left(C_{ns} f_{k-1} + g_n\right)$, $\check{v}_k = v_{k-1} + \Delta t\left(C_{ns} f_{k-1} + g_n\right)$, $\check{q}_k = \Omega\left(q\left(\omega_{k-1}\Delta t\right)\right) q_{k-1}$
   - loop every time an IMU measurement occurs.
   - The last state may be a fully corrected state, or one that was also propagated using the motion model only.
     - Depending on whether or not we received the GNSS or LIDAR measurement at the last time step.
2. Propagate uncertainty: $\check{P}_k = F_{k-1} P_{k-1} F_{k-1}^T + L_{k-1} Q_{k-1} L_{k-1}^T$.
   - propagate the state uncertainty through the linearized motion model.
   - accounting for whether or not the previous state was corrected or uncorrected.
   - At this point, if don't have any GNSS or LIDAR measurements available, can repeat step one and two.
3. If GNSS or LIDAR position available:
   1. Compute Kalman gain: $K_k = \check{P}_k H_k^T (H_k \check{P}_k H_k^T + R)^{-1}$.
      - $R$は$R_{GNSS}$もしくは$R_{LIDAR}$.
   2. Compute error state: $\delta x_k = K_k(y_k - \check{p}_k)$.
      - しかしvelocityやorientationはcorrectできないでしょう。

      $$\hat{p}_k = \check{p}_k + \delta p_k$$

   3. Correct predicted state: $\hat{v}_k = \check{v}_k + \delta v_k$ .

      $$\hat{q}_k = q(\delta\phi) \otimes \check{q}_k$$

      - quaternionの更新式がこうなっている理由は、quaternion is a constrained quantity that is not a simple vector.
      - $\delta\phi$はglobal orientation error (??). だから⊗の左側にある。motion modelにあるorientation change（incremental quaternion）はsensor frameに対するので、⊗の右側にある。まだ分かっていない。
   4. Computed corrected covariance: $\hat{P}_k = (1 - K_k H_k)\check{P}_k$.

Summary | EKF for Vehicular State Estimation
- We used a loosely coupled EKF to fuse GNSS with IMU and LIDAR measurements.
- Assumptions:
  - LIDAR provides positions in the same reference frame as GNSS (possible).
  - IMU has no biases (not realistic).
  - State initialization is provided (realistic).
  - Sensors are spatially and temporally aligned (somewhat realistic).

[2017] Quaternion kinematics for the error-state Kalman filter: https://arxiv.org/pdf/1711.02508.pdf

[2017] Advanced Vehicle State Estimation: A Tutorial and Comparative Study: https://www.sciencedirect.com/science/article/pii/S2405896317323674

# Lesson 3: Sensor Calibration - A Necessary Evil

単語
- flat: a flat is a tyre that does not have enough air in it.

Calibration: A Necessary Evil
- Intrinsic Calibration (single sensor)
  - deal with sensor specific parameters.
  - 例えば、$v = \omega R$の$R$.
- Extrinsic Calibration (multiple sensors)
  - deal with how the sensors are positioned and oriented on the vehicle.
- Temporal Calibration
  - deal with time offset between different sensor measurements.

Intrinsic Sensor Calibration
- Another example of an intrinsic sensor parameter is the elevation angle of a scan line in a LiDAR sensor like the Velodyne.
- How can we determine the parameters of sensor models?
  - Manufacturer specifications.
  - Measure by hand.
  - Estimate as part of the state.
    - either on the fly or more commonly as a special calibration step before putting the sensors into operation.
    - メリット： this approach has the advantage of producing an accurate calibration that's specific to the particular sensor and can also be formulated in a way that can handle the parameters varying slowly over time.
    - For example, if you continually estimate the radius of your tires, this could be a good way of detecting when you have a flat.

Calibration by Estimation
タイヤの半径を例として。
- State: $x = \begin{bmatrix} p \\ \dot{p} \\ R \end{bmatrix}, u = \ddot{p}, \dot{p} = v = \omega R.$
- Motion Model: $x_k = \begin{bmatrix} 1 & \Delta t & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} x_{k-1} + \begin{bmatrix} 0 \\ \Delta t \end{bmatrix} u_{k-1} + w_{k-1}, w_k \sim \mathcal{N}(0, Q_k).$
- Position / Wheel Rate Observations: $y_k = \begin{bmatrix} p_k \\ \frac{\dot{p}_k}{R_k} \end{bmatrix} + v_k, v_k \sim \mathcal{N}(0, R_k).$
  - also observe the wheel rotation rate through the encoder.

Extrinsic Sensor Calibration
- Determine the relative poses of sensors usually with respect to the vehicle frame.
- For example, the relative pose of IMU and LiDAR, so that the rates reported by the IMU are expressed in the same coordinate system as the LiDAR point clouds.

- CAD model: all of the sensor frames have been nicely laid out.
- Measure by hand: this is often difficult or impossible to do accurately since many sensors have the origin of their coordinate system inside the sensor itself.
- Estimate as part of the state.

Temporal Calibration

- For example, if we get LiDAR scans at 15Hz and IMU readings at 200Hz, might want to pair each LiDAR scan with the IMU reading whose timestamp is the closest match.
- But in reality, there's an unknown delay between when the LiDAR or IMU actually records an observation and when it arrives at the computer.

• Assume zero.
• Hardware synchronization.
  • use hardware timing signals to synchronize the sensors, but this is often an option only for more expensive sensor setups.
• Estimate as part of the state.

PCD Viewer

# Lesson 4: Loss of One or More Sensors

単語

- firmware: in computer systems, firmware is a set of commands which are stored on a chip rather than as part of a program, because the computer uses them very often.

内容

- Examine the importance of sensor redundancy for robust localization.
- Explore several examples of sensor failures in localization.

Sensor Measurement Ranges

- sensor layoutの理由
- 例えば2016 Tesla Model S Autopilot Sensors
  - Rearward looking side cameras: 100m
  - Wide forward camera: 60m
  - Main forward camera: 150m
  - Narrow forward camera: 250m
  - Rear view camera: 50m
  - Forward looking side cameras: 80m
  - Ultrasonics: 8m
  - Radar: 160m
- Most cars have long, medium, and short range sensing.
  - short range sensors for parking and to ensure not to collide with nearby vehicles.
  - medium range sensors help in pedestrian and cyclist detection and in lane keeping.
  - long range sensors help detect and predict the motion of approaching distant obstacles.
- If one of these fails, have to ensure that the safety of the car occupants or the people around us is not compromised.

Redundant Systems: 777 PFC

- The 777 operates on a triple redundancy principle.
- All major systems, including the flight computer, have two backups, each with independent power and separate programming and specifications.
  - separate programmingはちょっとすごい。同じバグを防げるのは分かっているが。どう言う意味？下の説明によると、hardwareのみseparateされるらしい。つまりsoftwareは同じ？ではsoftwareのバグが発生したら、ヤバイ。多分これはそもそも許されないこと。
- Each of the flight computers has a different processor to ensure that a bug in the architecture itself doesn't affect all three computers.
  - 例えばprocessorはそれぞれmicro-processor AMD 29050, MOTOROLA 68040, INTEL 80486.
- If one of the computers fails or malfunctions, the 777 uses a consensus algorithm to seamlessly switch to another one.

Redundancy is Crucial

- Carefully consider and characterize the different failure modes of the different sensing paradigms.

The Challenges of State Estimation

- Data association
  - associate some elements, an attribute that you see in the data to an entity in the world.
  - when it becomes hard to relate what you are seeing with respect to what you wish you could infer, that becomes hard.
- Ecosystem of hardware, software, and applications moving forward.

Final Project Overview
- observe the inner workings and interactions of various components found in a full state estimation pipeline.
- The sensor array consists of an IMU, a GNSS receiver, and a LiDAR, all of which provide measurements of varying reliability and at different rates.
- You'll be implementing the error state extended Kalman filter, since it provides a high degree of robustness, and is a staple in state estimation.