Lecture 9: Modeling and motion models

Whiteboard:

- Principles and some examples.

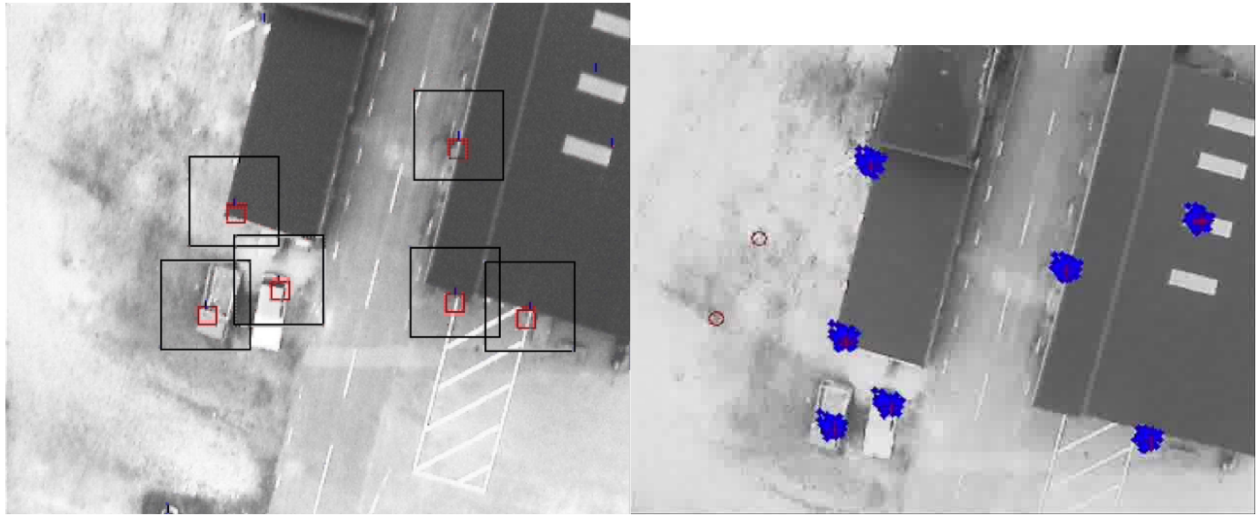
Slides:

- Sampling formulas.
- Noise models.
- Standard motion models.
  - Position as integrated velocity, acceleration,... in nD.
  - Orientation as integrated angular speed in 2D and 3D.
- Odometry
- (Extended) Kalman filter applications

Lecture 8: Summary

Simultaneous Localization And Mapping (SLAM)

- Joint estimation of trajectory $x_{1:k}$ and map parameters θ in sensor model $y_k = h(x_k; \theta) + e_k$.
- Algorithms:
  - NLS SLAM: iterate between filtering and mapping
  - EKF-SLAM: EKF (information form) on augmented state vector $z_k = (x_k^T, \theta^T)^T$.
  - FastSLAM: MPF on augmented state vector $z_k = (x_k^T, \theta^T)^T$.



http://youtu.be/VQlaCHl3Yc4 http://youtu.be/hA_NZeuoy9Y

Modeling

First problem: physics give continuous time model, filters require (linear or nonlinear) discrete time model:

Classification	Nonlinear	Linear
Continuous time	$\dot{x} = a(x,u) + v$	$\dot{x} = Ax + Bu + v$
	$y = c(x,u) + e$	$y = Cx + Du + e$
Discrete time	$x_{k+1} = f(x,u) + v$	$x_{k+1} = Fx + Gu + v$
	$y = h(x,u) + e$	$y = Hx + Ju + e$

But first, sampling overview.

Sampling formulas

LTI state space model

$$\dot{x} = Ax + Bu, \qquad x_{k+1} = Fx + Gu$$
$$y = Cx + Du, \qquad y_k = Hx + Ju.$$

u is either input or process noise (then J denotes cross-correlated noise!).
Zero order hold sampling assuming the input is piecewise constant:

$$x(t+T) = \underbrace{e^{AT}}_{F} x(t) + \underbrace{\int_0^T e^{A\tau} B \, u(t+T-\tau) d\tau}_{G}$$
$$= e^{AT} x(t) + \int_0^T e^{A\tau} d\tau B u(t).$$

First order hold sampling assuming the input is piecewise linear is another option.

Sampling formulas

Bilinear transformation assumes band-limited input

$$\frac{2}{T}\frac{z-1}{z+1}x(t) \approx sx(t) = Ax + Bu$$

gives

$$\begin{aligned} M &= (I_{n_x} - T/2A)^{-1}, \\ F &= M(I_{n_x} + T/2A), \\ G &= T/2MB, \\ H &= CM, \\ J &= D + HG. \end{aligned}$$

Sampling of nonlinear models

There are two options:

- Discretized linearization (general):
  1. Linearization:

$$A = a'_x(x, u), \quad B = a'_u(x, u), \quad C = c'_x(x, u), \quad D = c'_u(x, u),$$

  2. Sampling: $F = e^{AT}$, $G = \int_0^T e^{At} dt B$, $H = C$ and $J = D$.
- Linearized discretization (best, when possible!):
  1. Nonlinear sampling:

$$x(t + T) = f(x(t), u(t)) = x(t) + \int_t^{t+T} a(x(\tau), u(\tau)) d\tau,$$

  2. Discretization: $F = f'_x(x_k, u_k)$ and $G = f'_u(x_k, u_k)$.

Sampling of state noise

a v_t is continuous white noise with variance Q.

$$\bar{Q}_a = \int_0^T e^{f'\tau} Q e^{(f')^T\tau} \, \mathrm{d}\tau$$

b $v_t = v_k$ is a stochastic variable which is constant in each sample interval and has variance Q/T.

$$\bar{Q}_b = \frac{1}{T} \int_0^T e^{f'\tau} \, \mathrm{d}\tau Q \int_0^T e^{(f')^T\tau} \, \mathrm{d}\tau$$

c v_t is a sequence of Dirac impulses active immediately after a sample is taken. Loosely speaking, we assume $\dot{x} = f(x) + \sum_k v_k \delta_{kT-t}$ where v_k is discrete white noise with variance TQ.

$$\bar{Q}_c = T e^{f'T} Q e^{(f')^T T}$$

d v_t is white noise such that its total influence during one sample interval is TQ.

$$\bar{Q}_d = TQ$$

e v_t is a discrete white noise sequence with variance TQ. That is, we assume that the noise enters immediately after a sample time, so $x(t+T) = f(x(t) + v(t))$.

$$\bar{Q}_e = Tg'(x)Qg'(x)^T$$

Recommendation: the simple solution in d works fine, but remember the scaling with T!

Models

Continuous time (physical) and discrete time counterparts

$$\dot{x}(t) = a(x(t), u(t), w(t); \theta),$$
$$x(t+T) = f(x(t), u(t), w(t); \theta, T).$$

- *Kinematic models* do not attempt to model forces. 'Black-box' multi-purpose.
  - *Translation kinematics* describes position, often based on $F = ma$
  - *Rotational kinematics* describes orientation.
  - *Rigid body kinematics* combines translational and rotational kinematics.
  - *Constrained kinematics.* Coordinated turns (circular path motion).
- *Application specific force models.*
- Gray-box models, where θ has to be calibrated (estimated, identified) from data.

Translational motion with n integrators

Translational kinematics models in nD, where $p(t)$ denotes the position X (1D), X, Y (2D) and X, Y, Z (3D), respectively. Another interpretation is $p(t) = \psi(t)$ or $p(t) = (\phi(t), \theta(t), \psi(t))^T$. The signal $w(t)$ is process noise for a pure kinematic model and a motion input signal in position, velocity and acceleration, respectively, for the case of using sensed motion as an input rather than as a measurement.

State	Continuous time $\dot{x} =$	Discrete time $x(t+T) =$
$x = p$	w	$x + Tw$
$x = \begin{pmatrix} p \\ v \end{pmatrix}$	$\begin{pmatrix} 0_n & I_n \\ 0_n & 0_n \end{pmatrix} x + \begin{pmatrix} 0_n \\ I_n \end{pmatrix} w$	$\begin{pmatrix} I_n & TI_n \\ 0_n & I_n \end{pmatrix} x + \begin{pmatrix} \frac{T^2}{2} I_n \\ TI_n \end{pmatrix} w$
$x = \begin{pmatrix} p \\ v \\ a \end{pmatrix}$	$\begin{pmatrix} 0_n & I_n & 0_n \\ 0_n & 0_n & I_n \\ 0_n & 0_n & 0_n \end{pmatrix} x + \begin{pmatrix} 0_n \\ 0_n \\ I_n \end{pmatrix} w$	$\begin{pmatrix} I_n & TI_n & \frac{T^2}{2} I_n \\ 0_n & I_n & TI_n \\ 0_n & 0_n & I_n \end{pmatrix} x + \begin{pmatrix} \frac{T^3}{6} I_n \\ \frac{T^2}{2} I_n \\ TI_n \end{pmatrix} w$

Different sampled models of a double integrator

State $x(t) = (p(t),\ v(t))$

Continuous time	$A = \begin{pmatrix} 0_n & I_n \\ 0_n & 0_n \end{pmatrix}$	$B = \begin{pmatrix} 0_n \\ I_n \end{pmatrix}$	$C = (I_n,\ 0_n)$	$D = 0_n$
ZOH	$F = \begin{pmatrix} I_n & TI_n \\ 0_n & I_n \end{pmatrix}$	$G = \begin{pmatrix} \frac{T^2}{2} I_n \\ TI_n \end{pmatrix}$	$H = (I_n,\ 0_n)$	$J = 0_n$
FOH	$F = \begin{pmatrix} I_n & TI_n \\ 0_n & I_n \end{pmatrix}$	$G = \begin{pmatrix} T^2 I_n \\ TI_n \end{pmatrix}$	$H = (I_n,\ 0_n)$	$J = \frac{T^2}{6} I_n$
BIL	$F = \begin{pmatrix} I_n & TI_n \\ 0_n & I_n \end{pmatrix}$	$G = \begin{pmatrix} \frac{T^2}{4} I_n \\ \frac{T}{2} I_n \end{pmatrix}$	$H = \left(I_n,\ \frac{T}{2} I_n\right)$	$J = \frac{T^2}{2} I_n$

Navigation models

- Navigation models have access to inertial information.
- 2D orientation (course, or yaw rate) much easier than 3D orientation

Rotational kinematics in 2D

The course, or yaw, in 2D can be modeled as integrated white noise

$$\dot{\psi}(t) = w(t),$$

or any higher order of integration. Compare to the tables for translational kinematics with $p(t) = \psi$ and $n = 1$.

Coordinated turns in 2D body coordinates

Basic motion equations

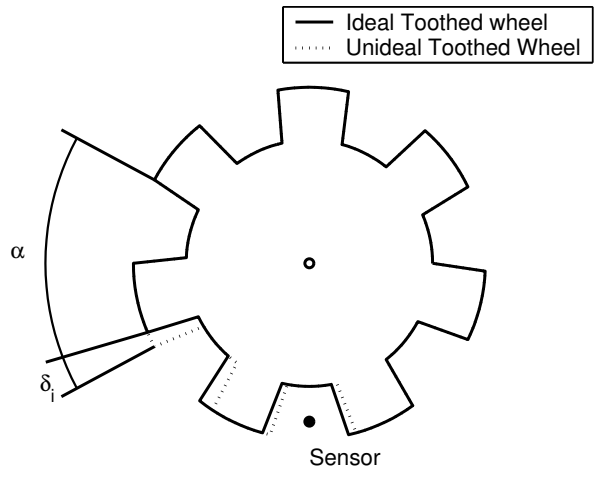
$$\dot{\psi} = \frac{v_x}{R} = v_x R^{-1},$$
$$a_y = \frac{v_x^2}{R} = v_x^2 R^{-1} = v_x \dot{\psi},$$
$$a_x = \dot{v}_x - v_y \frac{v_x}{R} = \dot{v}_x - v_y v_x R^{-1} = \dot{v}_x - v_y \dot{\psi}.$$

can be combined to a model suitable for the sensor configuration at hand. For instance,

$$x = \begin{pmatrix} \psi \\ R^{-1} \end{pmatrix}, \quad u = v_x, \quad y = R^{-1}$$
$$\dot{x} = f(x,u) + w = \begin{pmatrix} v_x R^{-1} \\ 0 \end{pmatrix} + w$$

is useful when speed is measured, and a vision system delivers a local estimate of (inverse) curve radius.

Wheel speed sensor



Each tooth passing the sensor (electromagnetic or Hall) gives a pulse. The number n of clock cycles between a number m of teeth are registered within each sample interval.

$$\omega(t_k) = \frac{2\pi}{N_{\text{cog}}(t_k - t_{k-1})} = \frac{2\pi}{N_{\text{cog}} T_c} \frac{m}{n}.$$

Problems: Angle and time quantization. Synchronization. Angle offsets δ in sensor teeth.

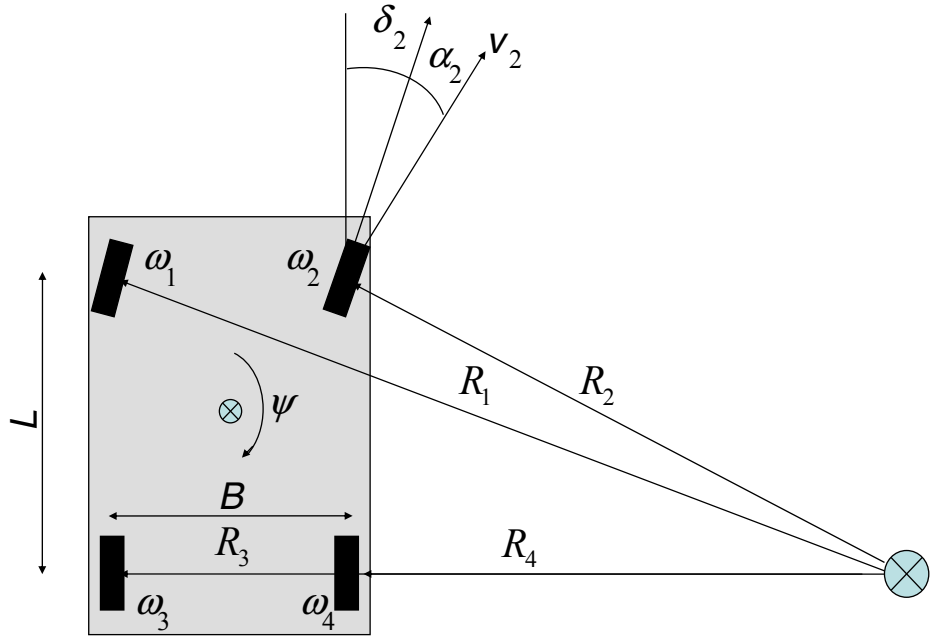
Virtual sensors

Longitudinal velocity, yaw rate and slip on left and right driven wheel (front wheel driven assumed) can be computed from wheel angular speeds **if** the radii are known:

$$
\begin{aligned}
v_x &= \frac{v_3 + v_4}{2} = \frac{\omega_3 r_3 + \omega_4 r_4}{2}, \\
\dot{\psi} &= \frac{\omega_3 r_3 - \omega_4 r_4}{B}, \\
s_1 &= \frac{\omega_1 r_1}{v_1} - 1, \quad s_2 = \frac{\omega_2 r_2}{v_2} - 1, \\
v_1 &= v_x \sqrt{\left(1 + \frac{R^{-1}B}{2}\right)^2 + (R^{-1}L)^2}, \\
v_2 &= v_x \sqrt{\left(1 - \frac{R^{-1}B}{2}\right)^2 + (R^{-1}L)^2}.
\end{aligned}
$$

Geometry

The formulas are based on geometry, the relation $\dot{\psi} = v_x R^{-1}$ and notation below.



Odometry

Odometry is based on the virtual sensors

$$v_x^m = \frac{\omega_3 r_3 + \omega_4 r_4}{2}$$
$$\dot{\psi}^m = v_x^m \frac{2}{B} \frac{\frac{\omega_3}{\omega_4}\frac{r_3}{r_4} - 1}{\frac{\omega_3}{\omega_4}\frac{r_3}{r_4} + 1}.$$

and the model

$$\psi_t = \psi_0 + \int_0^t \dot{\psi}_t dt,$$
$$X_t = X_0 + \int_0^t v_t^x \cos(\psi_t) dt,$$
$$Y_t = Y_0 + \int_0^t v_t^x \sin(\psi_t) dt.$$

to dead-reckon the wheel speeds to a relative position in the global frame.

The position $(X_t(r_3, r_4), Y_t(r_3, r_4))$ depends on the values of wheel radii r_3 and r_4. Further sources of error come from wheel slip in longitudinal and lateral direction. More sensors needed for navigation.

Rotational kinematics in 3D

Much more complicated in 3D than 2D! Could be a course in itself. Coordinate notation for rotations of a body in local coordinate system (x, y, z) relative to an earth fixed coordinate system:

Motion components	Rotation Euler angle	Angular speed
Longitudinal forward motion x	Roll ϕ	ω^x
Lateral motion y	Pitch θ	ω^y
Vertical motion z	Yaw ψ	ω^z

Euler orientation in 3D

An earth fixed vector g (for instance the gravitational force) is in the body system oriented as Qg, where

$$
\begin{aligned}
Q &= Q_\phi^x Q_\theta^y Q_\psi^z \\
&= \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{pmatrix} \begin{pmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{pmatrix} \begin{pmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \\
&= \begin{pmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \sin\phi\cos\theta \\ \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi & \cos\phi\cos\theta \end{pmatrix}.
\end{aligned}
$$

Note: this result depends on the order of rotations $Q_\phi^x Q_\theta^y Q_\psi^z$. Here, the *xyz* rule is used, but there are other options.

Euler rotation in 3D

When the body rotate with ω, the Euler angles change according to

$$\begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} = \begin{pmatrix} \dot{\phi} \\ 0 \\ 0 \end{pmatrix} + Q_\phi^x \begin{pmatrix} 0 \\ \dot{\theta} \\ 0 \end{pmatrix} + Q_\phi^x Q_\theta^y \begin{pmatrix} 0 \\ 0 \\ \dot{\psi} \end{pmatrix}$$

The dynamic equation for Euler angles can be derived from this as

$$\begin{pmatrix} \dot{\phi} \\ \dot{\psi} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi)\sec(\theta) & \cos(\phi)\sec(\theta) \end{pmatrix} \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix}$$

Singularities when $\theta = \pm\pi/2$, can cause numeric divergence!

Unit quaternions

Vector representation: $q = (q^0, q^1, q^2, q^3)^T$.
Norm constraint of unit quaternion: $q^T q = 1$.
Then, the quaternion is defined from the Euler angles as

$$
\begin{aligned}
q^0 &= \cos(\phi/2)\cos(\theta/2)\cos(\psi/2) + \sin(\phi/2)\sin(\theta/2)\sin(\psi/2),\\
q^1 &= \sin(\phi/2)\cos(\theta/2)\cos(\psi/2) - \cos(\phi/2)\sin(\theta/2)\sin(\psi/2),\\
q^2 &= \cos(\phi/2)\sin(\theta/2)\cos(\psi/2) + \sin(\phi/2)\cos(\theta/2)\sin(\psi/2),\\
q^3 &= \cos(\phi/2)\cos(\theta/2)\sin(\psi/2) - \sin(\phi/2)\sin(\theta/2)\cos(\psi/2).
\end{aligned}
$$

Pros: no singularity, no 2π ambiguity.
Cons: more complex and non-intuitive algebra, the norm constraint is a separate challenge. Norm constraint can be handled by projection or as a virtual measurement with small noise.

Quaternion orientation in 3D

The orientation of the vector $\mathbf{g}$ in body system is $Q\mathbf{g}$, where

$$
\begin{aligned}
Q &= \begin{pmatrix} q_0^2+q_1^2-q_2^2-q_3^2 & 2q_1q_2-2q_0q_3 & 2q_0q_2+2q_1q_3 \\ 2q_0q_3+2q_1q_2 & q_0^2-q_1^2+q_2^2-q_3^2 & -2q_0q_1+2q_2q_3 \\ -2q_0q_2+2q_1q_3 & 2q_2q_3+2q_0q_1 & q_0^2-q_1^2-q_2^2+q_3^2 \end{pmatrix} \\
&= \begin{pmatrix} 2q_0^2+2q_1^2-1 & 2q_1q_2-2q_0q_3 & 2q_1q_3+2q_0q_2 \\ 2q_1q_2+2q_0q_3 & 2q_0^2+2q_2^2-1 & 2q_2q_3-2q_0q_1 \\ 2q_1q_3-2q_0q_2 & 2q_2q_3+2q_0q_1 & 2q_0^2+2q_3^2-1 \end{pmatrix}.
\end{aligned}
$$

Quaternion rotation in 3D

Rotation with ω gives a dynamic equation for q which can be written in two equivalent forms:

$$\dot{q} = \frac{1}{2} S(\omega) q = \frac{1}{2} \bar{S}(q) \omega,$$

$$S(\omega) = \begin{pmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & \omega_z & -\omega_y \\ \omega_y & -\omega_z & 0 & \omega_x \\ \omega_z & \omega_y & -\omega_x & 0 \end{pmatrix},$$

$$\bar{S}(q) = \begin{pmatrix} -q_1 & -q_2 & -q_3 \\ q_0 & -q_3 & q_2 \\ q_3 & q_0 & -q_1 \\ -q_2 & q_1 & q_0 \end{pmatrix}.$$

Sampled form of quaternion dynamics

The ZOH sampling formula

$$q(t+T) = e^{\frac{1}{2}S(\omega(t))T} q(t).$$

has actually a closed form solution

$$\begin{aligned} q(t+T) &= \left(\cos\left(\frac{\|\omega(t)T\|}{2}\right) I_4 + \frac{T}{2} \frac{\sin\left(\frac{\|\omega(t)\|T}{2}\right)}{\frac{\|\omega(t)\|T}{2}} S(\omega(t)) \right) q(t) \\ &\approx \left(I_4 + \frac{T}{2} S(\omega(t)) \right) q(t). \end{aligned}$$

The approximation coincides with Euler forward sampling approximation, and has to be used in more complex models where e.g. ω is part of the state vector.

Double integrated quaternion

$$\begin{pmatrix} \dot{q}(t) \\ \dot{\omega}(t) \end{pmatrix} = \begin{pmatrix} \frac{1}{2} S(\omega(t)) q(t) \\ w(t) \end{pmatrix}.$$

There is no known closed form discretized model. However, the approximate form can be discretized using the chain rule to

$$\begin{pmatrix} q(t+T) \\ \omega(t+T) \end{pmatrix} \approx \underbrace{\begin{pmatrix} \left(I_4 \frac{T}{2} S(\omega(t))\right) & \frac{T}{2} \bar{S}(q(t)) \\ 0_{3\times 4} & I_3 \end{pmatrix}}_{F(t)} \begin{pmatrix} q(t) \\ \omega(t) \end{pmatrix} + \begin{pmatrix} \frac{T^3}{4} S(\omega(t)) I_4 \\ T I_3 \end{pmatrix} v(t).$$

Rigid body kinematics

A "multi-purpose" model for all kind of navigation problems in 3D (22 states).

$$\begin{pmatrix} \dot{p} \\ \dot{v} \\ \dot{a} \\ \dot{q} \\ \dot{\omega} \\ \dot{b}^{\text{acc}} \\ \dot{b}^{\text{gyro}} \end{pmatrix} = \begin{pmatrix} 0 & I & 0 & 0 & 0 \\ 0 & 0 & I & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{2}S(\omega) & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} p \\ v \\ a \\ q \\ \omega \\ b^{\text{acc}} \\ b^{\text{gyro}} \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} v^a \\ v^\omega \\ v^{\text{acc}} \\ v^{\text{gyro}} \end{pmatrix}.$$

Bias states for the accelerometer and gyro have been added as well.

Sensor model for kinematic model

Inertial sensors (gyro, accelerometer, magnetometer) are used as sensors.

$$
\begin{aligned}
y_t^{\text{acc}} &= R(q_t)(a_t - \mathbf{g}) + b_t^{\text{acc}} + e_t^{\text{acc}}, \qquad e_t^{\text{acc}} \sim \mathcal{N}\left(0, R_t^{\text{acc}}\right), \\
y_t^{\text{mag}} &= R(q_t)\mathbf{m} + b_t^{\text{mag}} + e_t^{\text{mag}}, \qquad e_t^{\text{mag}} \sim \mathcal{N}\left(0, R_t^{\text{mag}}\right), \\
y_t^{\text{gyro}} &= \omega_t + b_t^{\text{gyro}} + e_t^{\text{gyro}}, \qquad e_t^{\text{gyro}} \sim \mathcal{N}\left(0, R_t^{\text{gyro}}\right),
\end{aligned}
$$

Bias observable, but special calibration routines are recommended:

- Stand-still detection. When $\|y_t^{\text{acc}}\| \approx \mathbf{g}$ and/or $\|y_t^{\text{gyro}}\| \approx 0$, the gyro and acc bias is readily read off. Can decrease drift in dead-reckoning from cubic to linear.
- Ellipse fitting. When "waving the sensor" over short time intervals, the gyro can be integrated to give accurate orientation, and acc and magnetometer measurements can be transformed to a sphere from an ellipse.

Tracking models

- Navigation models have access to inertial information, tracking models have not.
- Orientation mainly the direction of the velocity vector.
- Course (yaw rate) critical parameter.
- Not so big difference of 2D and 3D cases.

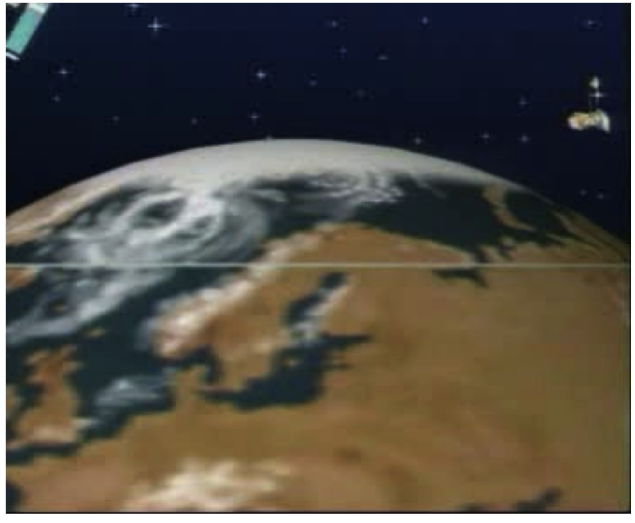
Coordinated turns in 2D world coordinates

Cartesian velocity	Polar velocity
$\dot{X} = v^X$ $\dot{Y} = v^Y$ $\dot{v}^X = -\omega v^Y$ $\dot{v}^Y = \omega v^X$ $\dot{\omega} = 0$	$\dot{X} = v\cos(h)$ $\dot{Y} = v\sin(h)$ $\dot{v} = 0$ $\dot{h} = \omega$ $\dot{\omega} = 0$
$\begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -\omega & -v^Y \\ 0 & 0 & \omega & 0 & v^X \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & \cos(h) & -v\sin(h) & 0 \\ 0 & 0 & \sin(h) & v\cos(h) & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$
$X_{t+T} = X + \frac{v^X}{\omega}\sin(\omega T) - \frac{v^Y}{\omega}(1-\cos(\omega T))$ $Y_{t+T} = Y + \frac{v^X}{\omega}(1-\cos(\omega T)) + \frac{v^Y}{\omega}\sin(\omega T)$ $v^X_{t+T} = v^X\cos(\omega T) - v^Y\sin(\omega T)$ $v^Y_{t+T} = v^X\sin(\omega T) + v^Y\cos(\omega T)$ $\omega_{t+T} = \omega$	$X_{t+T} = X + \frac{2v}{\omega}\sin(\frac{\omega T}{2})\cos(h + \frac{\omega T}{2})$ $Y_{t+T} = Y - \frac{2v}{\omega}\sin(\frac{\omega T}{2})\sin(h + \frac{\omega T}{2})$ $v_{t+T} = v$ $h_{t+T} = h + \omega T$ $\omega_{t+T} = \omega$

Sounding rocket

Navigation grade IMU gives accurate dead-reckoning, but drift may cause return at bad places.
GPS is restricted for high speeds and high accelerations.
Fusion of IMU and GPS when pseudo-ranges are available, with IMU support to tracking loops inside GPS.

- Loose integration: direct fusion approach $y_k = p_k + e_k$.
- Tight integration: TDOA fusion approach $y_k^i = \|p_k - p_k^i\|/c + t_k + e_k$.



http://youtu.be/zRHFXfZLQ64

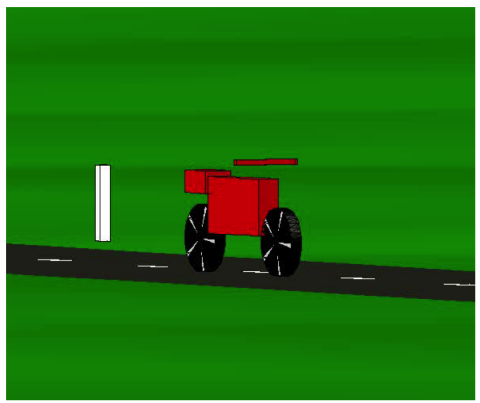
MC leaning angle

- Headlight steering, ABS and anti-spin systems require leaning angle.
- Gyro very expensive for this application.
- Combination of accelerometers investigated, lateral and downward acc worked fine in EKF.

Model, where z_y, z_z, a_1, a_2, J are constants relating to geometry and inertias of the motorcycle, $u = v_x$

$$x = (\varphi, \dot{\varphi}, \ddot{\varphi}, \dot{\psi}, \ddot{\psi}, \delta_{ay}, \delta_{az}, \delta_{\dot{\varphi}})^T,$$

$$y = h(x) = \begin{pmatrix} a_y \\ a_z \\ \dot{\varphi} \end{pmatrix} = \begin{pmatrix} ux_4 - z_y x_3 + z_y x_4^2 \tan(x_1) + g \sin(x_1) + x_6 \\ -ux_4 \tan(x_1) - z_z \left(x_2^2 + x_4^2 \tan^2(x_1)\right) + g \cos(x_1) + x_7 \\ -a_1 x_3 + a_2 x_4^2 \tan(x_1) - ux_4 J + x_6 \end{pmatrix}$$



http://youtu.be/hT6S1FgHxOc0%

Virtual yaw rate sensor

- Yaw rate subject to bias, orientation error increases linearly in time.
- Wheel speeds also give a gyro, where the orientation error grows linearly in distance,

Model, with state vector $x_k = \left(\dot{\psi}_k, \ddot{\psi}_k, b_k, \frac{r_{k,3}}{r_{k,4}}\right)$,

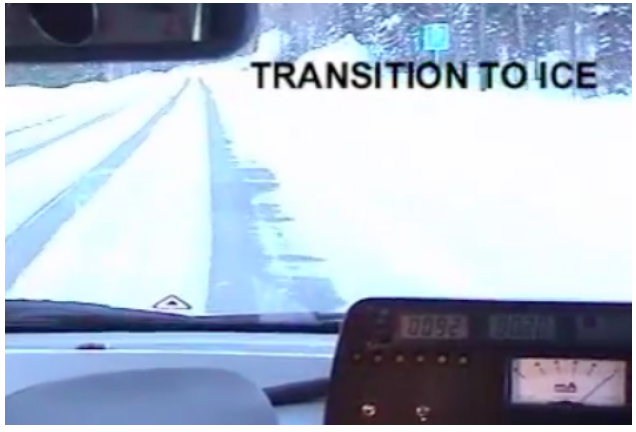
$$y_k^1 = \dot{\psi}_k + b_k + e_k^1,$$

$$y_k^2 = \frac{\omega_3 r_{nom} + \omega_4 r_{nom}}{2} \frac{2}{B} \frac{\frac{\omega_3}{\omega_4}\frac{r_{k,3}}{r_{k,4}} - 1}{\frac{\omega_3}{\omega_4}\frac{r_{k,3}}{r_{k,4}} + 1} + e_k^2.$$

http://youtu.be/d9rzCCIBS9I

Friction estimation

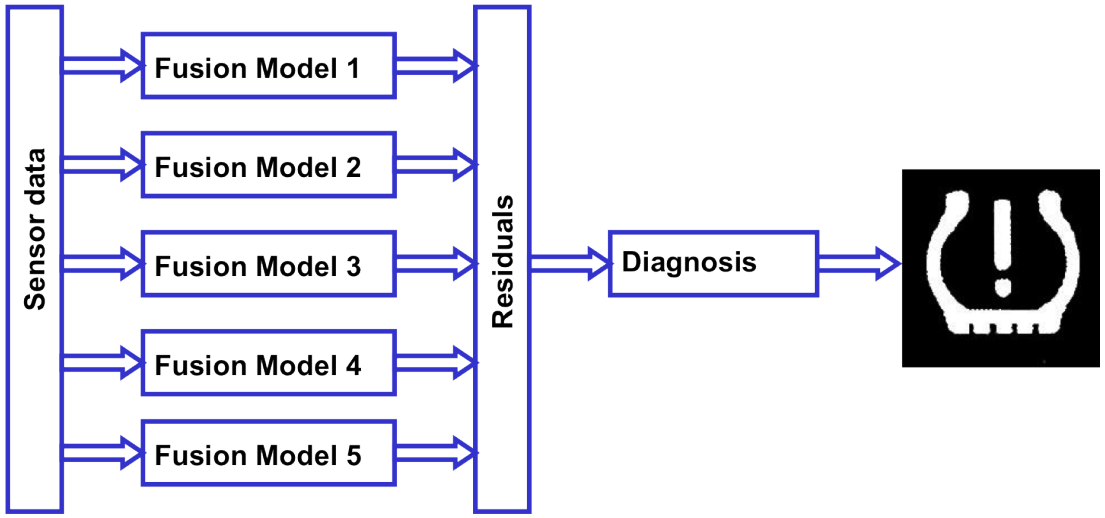
- Friction model $y_k = 1/s\mu_k + \delta_k + e_k$, where y_k is the slip, s is the friction dependent slip slope, μ_k is the normalized traction force from the engine, δ_k a tire radius dependent offset and e_k is measurement noise.
- Kalman filter estimated s adaptively.
- When s decreases under a threshold, a friction alarm is issued.



http://youtu.be/savmzo-xonk

Tire Pressure Monitoring System

- TPMS required in US since 2007, and in EU from 2013.
- Direct systems (with pressure sensors inside tire) most common, but indirect systems based on wheel speeds (ABS sensors) also on the market.
- The TPMS by NIRA Dynamics is installed in 4.75 (may 2013) million cars (Audi, VW, Skoda, Seat).
- Based on many different physical principles and EKF's, including the two ones above!



Shooter localization

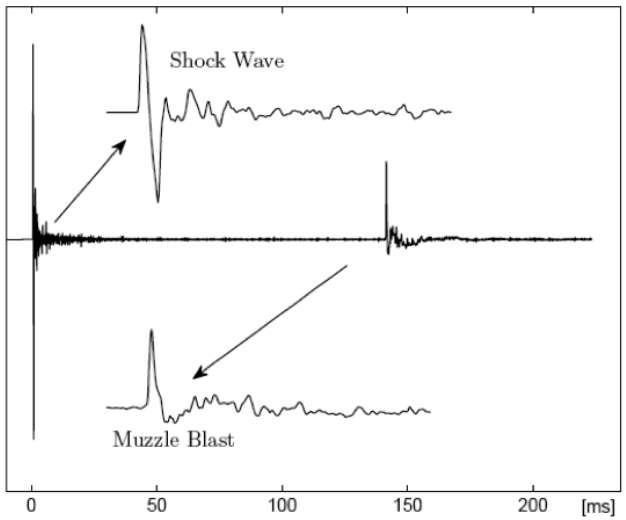
Network with 10 microphones around a 'camp'. Shooter aiming inside the camp. Supersonic bullet injects a shock wave followed by an acoustic muzzle blast. Fusion of time differences give shooter position and aiming point.

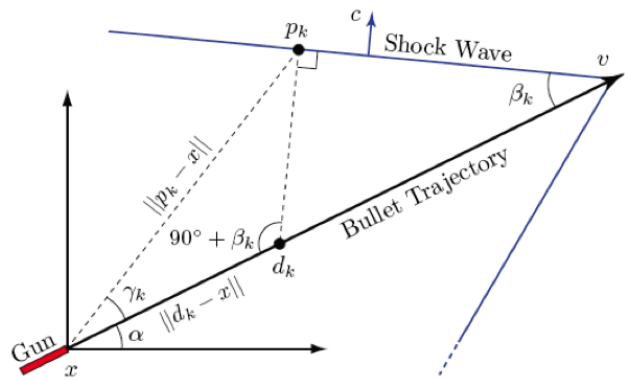
Model, with x being the position of the shooter,

$$y_k^{\mathrm{MB}} = t_0 + b_k + \tfrac{1}{c}||p_k - x|| + e_k^{\mathrm{MB}},$$

$$y_k^{\mathrm{SW}} = t_0 + b_k + \tfrac{1}{r}\log\frac{v_0}{v_0 - r||d_k - x||} + \tfrac{1}{c}||d_k - p_k|| + e_k^{\mathrm{SW}}.$$

where d_k is an implicit function of the state.





http://youtu.be/q-tU6YyJ5gU

Shooter localization results

WLS loss function illustrates the information in $y_k^{\mathrm{MB}} - y_k^{\mathrm{SW}}$, $k = 1, 2, \ldots, N = 10$.

Both shooter position and aiming direction α well estimated for each shot.

Also bullet's muzzle speed and ammunition length can be estimated!