AUTONOMOUS
INTELLIGENCE GROUP

LECTURE 9 — OBSERVATION MODELS (I): SIGNALS AND PROPRIOCEPTIVE SENSORS

INTRODUCTION TO AUTONOMOUS SYSTEMS · MODULE III

Topics: Data processing inequality · Sufficient statistics · Encoders & aliasing · IMU & Allan variance

AGENDA

§	TOPIC	KEY IDEAS
8	**The Observation Model, Revisited**	Raw signal → measurement · forward vs. inverse · data processing inequality · sufficiency
9	**Proprioceptive Sensors**	Quantisation · aliasing · IMU drift · Allan variance → Q, R

> *Lectures 5–8 built the estimator and took $p(o_t \mid x_t)$ as given. Lectures 9–10: where does it come from?*

LITERATURE

TOPIC	REFERENCE
Measurement models	Thrun, Burgard & Fox — *Probabilistic Robotics* (2005), Ch. 6
Information theory, DPI	Cover & Thomas — *Elements of Information Theory* (2006), Ch. 2
Sufficient statistics, factorisation	Kay — *Fundamentals of Statistical Signal Processing, Vol. I: Estimation Theory* (1993), Ch. 5
Sampling, aliasing, digital filters	Oppenheim & Schafer — *Discrete-Time Signal Processing* (3rd ed., 2010), Ch. 4, 7
IMU models	Groves — *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems* (2013)
Allan variance	El-Sheimy, Hou & Niu — "Analysis and Modeling of Inertial Sensors Using Allan Variance", *IEEE TIM* (2008)

RECAP — THE OBSERVATION MODEL SO FAR

- **Lec 2, §3.** h is almost never injective. Locally the blind directions are the **null space of** J_h; over time (f, h) together can recover what h alone loses (**observability**).
- **Lec 5–8.** Every filter's update step uses $p(o_t \mid x_t)$ — and every time it was **given**.
- **Lec 8, §6.** For $o = h(x) + v$ with noise density p_v: $p(o \mid x) = p_v\big(o - h(x)\big)$. A particle filter only needs to **evaluate** it.

Today: build h. Physically for proprioceptive and ranging sensors, geometrically and then by learning for cameras.

Notation: s = **raw signal** (waveform, pulse train, pixel array), $o = T(s)$ = **measurement**. Letters doing double duty: R noise covariance / rotation (IMU: body → world; camera: world → camera), K gain / intrinsics, f dynamics / focal length / frequency, B control matrix / bandwidth, τ delay / time constant / averaging time.

§8 · THE OBSERVATION MODEL, REVISITED

FROM H TO A MEASUREMENT PIPELINE

FROM h TO A MEASUREMENT PIPELINE

No sensor reports a state variable. Every sensor produces a **raw signal** s_t — a voltage, an echo waveform, a tick count, pixel intensities — and a processing chain T turns it into a measurement:

$$x_t \xrightarrow{\text{physics}} s_t \xrightarrow{\text{processing } T} o_t = T(s_t) \xrightarrow{\text{filter}} b_t.$$

The observation model of Lec 5–8 is the composition of the two left arrows, and it depends on more than the state:

$$o_t = h(x_t;\, m,\, \theta) + v_t.$$

- m — **environment**: the map a lidar's beams hit, the satellites a GNSS receiver sees.
- θ — **calibration**: intrinsics (focal length, scale factor), extrinsics (sensor mounting), biases.
- Unknown m or θ → **add them to the state** (Lec 5, §2.1). SLAM does this with m, online calibration with θ.

> *Extrinsics:* 1° of lidar mounting error shifts a wall at 20 m by ≈ 35 cm. *Time:* a measurement taken at $t - \tau$ measures $x_{t-\tau}$ — at 30 m/s, 50 ms of unmodelled latency is 1.5 m. The timestamp is part of the measurement.

FORWARD AND INVERSE MODELS

	FORWARD (GENERATIVE) MODEL	INVERSE (DISCRIMINATIVE) MODEL
Object	$p(o \mid x)$	$\hat{x} = g(o)$, or $p(x \mid o)$
Question answered	"If the state were x, how likely is this reading?"	"Given this reading, what is the state?"
Who needs it	every filter of Lec 5–8 (update step)	a controller acting on the measurement directly
Where it comes from	physics and a noise model	engineered pipeline or a trained network

Bayes connects them, $p(x \mid o) \propto p(o \mid x)\, p(x)$: the difference is the **prior**. An inverse model has a prior built in; a forward model does not, so the filter can supply its own, $\bar{b}_t$.

> **The question for each sensor:** can the forward model be written down, and can it be **evaluated**? Physical sensors (§9–§10): yes, in one line. Camera (§11): the forward model is rendering — simulable, not evaluable over pixels → inverse models, plugged back into a filter at the end of §11.

WHAT A GOOD PIPELINE PRESERVES — MUTUAL INFORMATION

T should compress s (10^4 samples, 10^6 pixels) to a small o **without discarding anything about** x.

Definition (Mutual information). For random variables x, y with joint density $p(x, y)$,

$$I(x; y) = D_{\mathrm{KL}}\big(p(x, y) \,\big\|\, p(x)\,p(y)\big) \;\geq 0,$$

zero iff $x \perp y$ (Gibbs, Lec 4, §2). The **conditional** $I(x; y \mid w)$ uses the densities given w, averaged over w; zero iff $x \perp y \mid w$.

Lemma (Chain rule). $I(x; s, z) = I(x; z) + I(x; s \mid z) = I(x; s) + I(x; z \mid s)$.

Proof. Since $\frac{p(x,s)}{p(x)p(s)} = \frac{p(x|s)}{p(x)}$: $I(x; s) = \mathbb{E}\big[\log \frac{p(x|s)}{p(x)}\big]$, $I(x; s \mid z) = \mathbb{E}\big[\log \frac{p(x|s,z)}{p(x|z)}\big]$. Then multiply and divide by $p(x \mid z)$ (or by $p(x \mid s)$) inside the log:

$$I(x; s, z) = \mathbb{E}\Big[\log \frac{p(x \mid s, z)}{p(x)}\Big] = \mathbb{E}\Big[\log \frac{p(x \mid z)}{p(x)}\Big] + \mathbb{E}\Big[\log \frac{p(x \mid s, z)}{p(x \mid z)}\Big]. \;\square$$

The log turns the two-stage update $p(x) \to p(x \mid z) \to p(x \mid s, z)$ into a sum; the order does not matter.

THE DATA PROCESSING INEQUALITY

Theorem (Data processing inequality, DPI). Let $z = T(s)$ be any function of the raw signal. Then

$$I(x; z) \;\le\; I(x; s),$$

with equality if and only if $x \perp s \mid z$.

Proof. Chain rule in both orders: $I(x; z) + I(x; s \mid z) = I(x; s) + I(x; z \mid s)$.
Given s, $z = T(s)$ is fixed, so $p(x \mid s, z) = p(x \mid s)$ and $I(x; z \mid s) = 0$. Hence

$$I(x; z) = I(x; s) - \underbrace{I(x; s \mid z)}_{\ge 0} \;\le\; I(x; s),$$

with equality exactly when $I(x; s \mid z) = 0$. $\square$

- Also holds for a **random** T (RANSAC, random beam subsampling, a network with dropout left on), as long as its randomness does not depend on x, i.e. z depends on x only through s.

Processing cannot create information about the state. It can only keep it or throw it away.

SUFFICIENT STATISTICS

> **Definition (Sufficient statistic).** $z = T(s)$ is **sufficient** for x if $p(x \mid s) = p(x \mid T(s))$ for every prior on x: the posterior depends on the raw signal only through z.

For a prior with full support ($p(x) > 0$ for every x), sufficiency is exactly the **equality case** of the DPI, $I(x; T(s)) = I(x; s)$: a lossless pipeline.

Checking sufficiency without computing posteriors: look at the likelihood.

> **Theorem (Fisher–Neyman factorisation).** Let T map each raw signal s to a statistic $z = T(s)$. Then T is sufficient for x **if and only if** there are non-negative functions $g(z, x)$ and $k(s)$ such that, for all s and x,
>
> $$p(s \mid x) = g\big(T(s), x\big)\, k(s).$$
>
> Here g carries all the dependence on the state x but sees the signal only through $T(s)$; k may depend on s in any way but not on x.

PROOF OF THE FACTORISATION THEOREM

Proof. ($\Leftarrow$) Take any prior and any s with $p(s) > 0$; then $k(s) > 0$, and by Bayes

$$p(x \mid s) = \frac{g(T(s), x)\, k(s)\, p(x)}{\int g(T(s), x')\, k(s)\, p(x')\, dx'} = \frac{g(T(s), x)\, p(x)}{\int g(T(s), x')\, p(x')\, dx'},$$

which depends on s only through $T(s)$, for every prior.
($\Rightarrow$) Fix one prior with $p(x) > 0$ for every x. For $p(s) > 0$, Bayes and sufficiency give

$$p(s \mid x) = \frac{p(x \mid s)\, p(s)}{p(x)} = \underbrace{\frac{p(x \mid T(s))}{p(x)}}_{g(T(s),\, x)} \underbrace{p(s)}_{k(s)};$$

where $p(s) = 0$, take $k(s) = 0$. $\square$

To prove a pipeline lossless: factor its likelihood. (Fisher 1922, Neyman 1935)

EXAMPLE — AVERAGING IS LOSSLESS

n readings of the same x, $y_i = x + v_i$, with **independent** noise $v_i \sim \mathcal{N}(0, \sigma^2)$, i.e. $y_i \sim \mathcal{N}(x, \sigma^2)$; $s = (y_1, \dots, y_n)$. Given x the readings are independent, so the likelihood is a product of Gaussians:

$$p(s \mid x) \propto \exp\Big(-\tfrac{1}{2\sigma^2} \textstyle\sum_i (y_i - x)^2 \Big).$$

Expand $(y_i - x)^2 = y_i^2 - 2xy_i + x^2$ and separate what involves x:

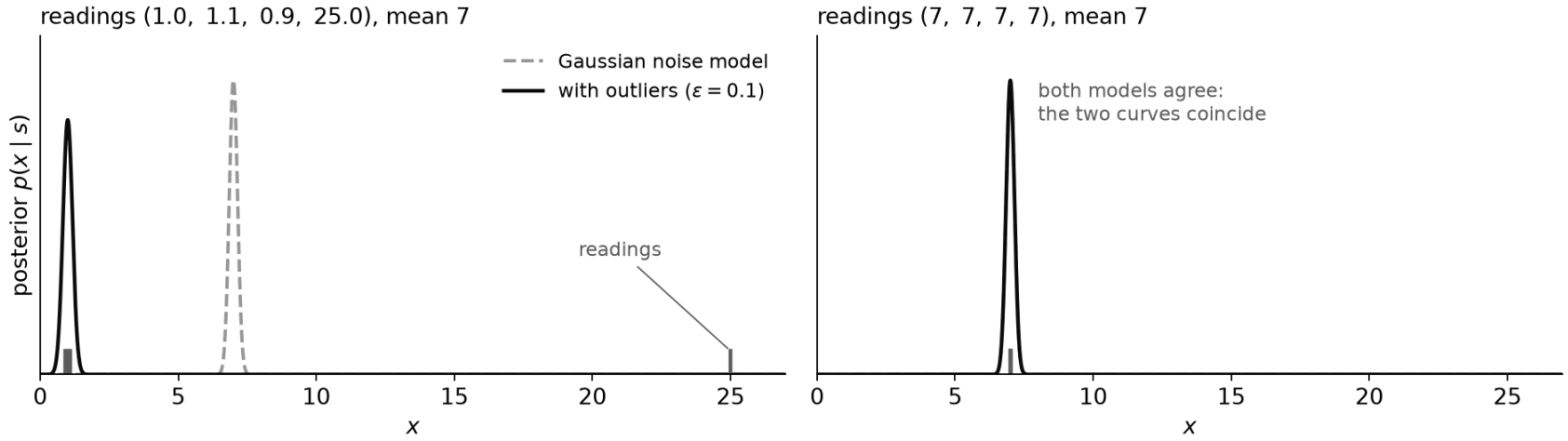
$$p(s \mid x) \propto \underbrace{\exp\Big(\tfrac{1}{\sigma^2} \big(x \textstyle\sum_i y_i - \tfrac{n}{2} x^2 \big) \Big)}_{g(\sum_i y_i,\, x)} \; \underbrace{\exp\Big(-\tfrac{1}{2\sigma^2} \textstyle\sum_i y_i^2 \Big)}_{k(s)}.$$

- The **sum** (equivalently the mean) is sufficient: averaging n samples into one number **loses nothing** — if the noise really is independent and Gaussian.
- **With outliers the mean is no longer sufficient.**

> **The thread of this lecture.** For each sensor: is the pipeline's output (close to) a **sufficient statistic** for x — and can we **derive** it, or must we **learn** it?

WHEN AVERAGING LOSES INFORMATION — OUTLIERS

Outliers are occasional gross errors: a lidar beam hits a passer-by instead of the wall, a GNSS or radar echo arrives via a reflection, an electrical glitch corrupts a sample. Under the Gaussian model a reading 10σ off has probability $\approx 10^{-23}$; real noise has **heavier tails**, e.g. $v_i \sim (1-\varepsilon)\,\mathcal{N}(0,\sigma^2) + \varepsilon \cdot \mathrm{uniform}$.



Same mean, different posteriors → the mean is not sufficient. The Gaussian model puts x at 7, where no reading is; with outliers the posterior depends on *which* readings disagree, so all readings must be kept. Robust likelihoods: beam model, §10.

§9 · PROPRIOCEPTIVE SENSORS

THE ROBOT'S OWN MOTION — FAST, CHEAP, ALWAYS ON, AND DRIFTING WHEN INTEGRATED

WHAT ARE PROPRIOCEPTIVE SENSORS?

Proprioceptive sensors measure the robot's **own** state — its configuration and motion — from inside the body.
Exteroceptive sensors (§10–§11: lidar, radar, GNSS, cameras) measure the **world** relative to the robot.

SENSOR	MEASURES	RATE	RELATION TO THE POSE
Joint encoder	joint angle q	1–10 kHz	**the configuration itself** (for a fixed-base arm)
Wheel encoder	wheel rotation	0.1–1 kHz	**change** in pose; odometry sums the changes and their errors → drift
Gyroscope	angular velocity	0.1–1 kHz	the **rate** of turning; integrating adds up errors → heading drifts
Accelerometer	specific force (incl. gravity)	0.1–1 kHz	acceleration, integrated twice → drifts; gravity gives roll, pitch
Joint torque sensor	joint torque	~ 1 kHz	effort, not pose — contact and collision detection

- **Strengths:** fast, cheap, always available, unaffected by lighting, weather or an unknown map.
- **Weakness:** none measures position or heading **relative to the world** (only gravity's direction is absolute) → integrating drifts; exteroceptive sensors bound it.

§9.1 · ENCODERS, QUANTISATION AND SAMPLING

ROUNDING, DIFFERENCING, SAMPLING — THE NOISE MODEL IS DERIVED

AN ENCODER IS READ FASTER AND FASTER. DOES THE VELOCITY ESTIMATE FROM FINITE DIFFERENCES GET BETTER OR WORSE?

ENCODERS — QUANTISATION AND DIFFERENTIATION

Quantisation noise. N ticks per revolution, resolution $\Delta = 2\pi/N$: $o = \Delta \cdot \mathrm{round}(q/\Delta) = q + v$.
When q moves over many levels, $v \approx$ uniform on $[-\Delta/2, \Delta/2]$:

$$\mathrm{Var}(v) = \frac{1}{\Delta}\int_{-\Delta/2}^{\Delta/2} v^2\, dv = \frac{\Delta^2}{12}.$$

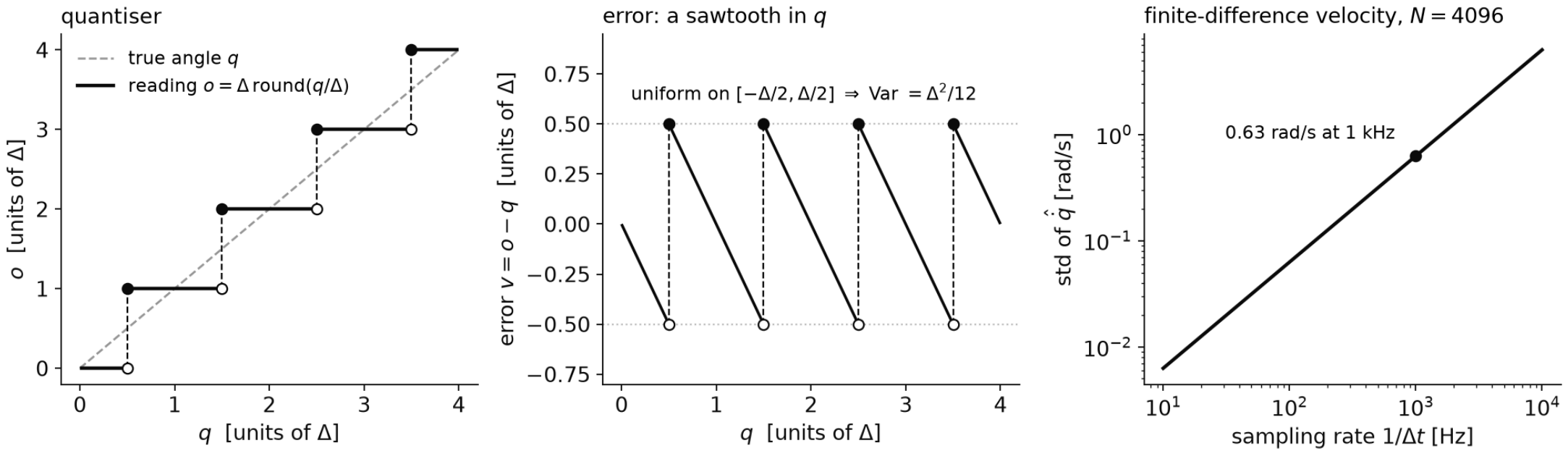
First time in the course that R is derived from the physics rather than assumed.
$N = 4096$: $\Delta = 1.53 \times 10^{-3}$ rad, $\sigma_v = 4.4 \times 10^{-4}$ rad. If q is nearly constant, v is the same every step: a **bias**, not noise.

Differentiation amplifies noise. $\hat{\dot{q}}_t = (o_t - o_{t-1})/\Delta t$ has noise variance

$$\frac{2\,\Delta^2/12}{\Delta t^2} \qquad \text{(independent errors)}.$$

Faster sampling makes it worse. Encoder above at 1 kHz: $\sigma_{\dot{q}} \approx 0.63$ rad/s.
Remedy: don't differentiate. Put $\dot{q}$ in the state; the KF estimates it through the dynamics (Lec 2, §3).

QUANTISATION AND DIFFERENTIATION, PICTURED



- **Left, middle:** the error $v = o - q$ is a *deterministic* sawtooth in q. It behaves like uniform noise, variance $\Delta^2/12$, only when q sweeps many levels; for a static joint it is a constant bias.
- **Right:** the finite-difference velocity noise $\sqrt{2\Delta^2/12}\,/\Delta t$ grows **linearly with the sampling rate** — the answer to the question: faster is worse.

ODOMETRY DRIFTS — AND NO UPDATE FIXES IT

What a wheel encoder reads: the rotation $\Delta\theta_t$ since the last reading, so with wheel radius r it gives a **velocity**, not a position: $o_t = r\,\Delta\theta_t/\Delta t = v_t + \text{noise}$.
Odometry adds up the distances $u_t = o_t\,\Delta t$, each with a small independent error w_t of variance q, and the errors add up too — a **random walk**:

$$p_t = p_{t-1} + u_t + w_t \quad \Longrightarrow \quad \mathrm{Var}(p_t) = \mathrm{Var}(p_0) + t\,q, \qquad \text{std} \propto \sqrt{t}.$$

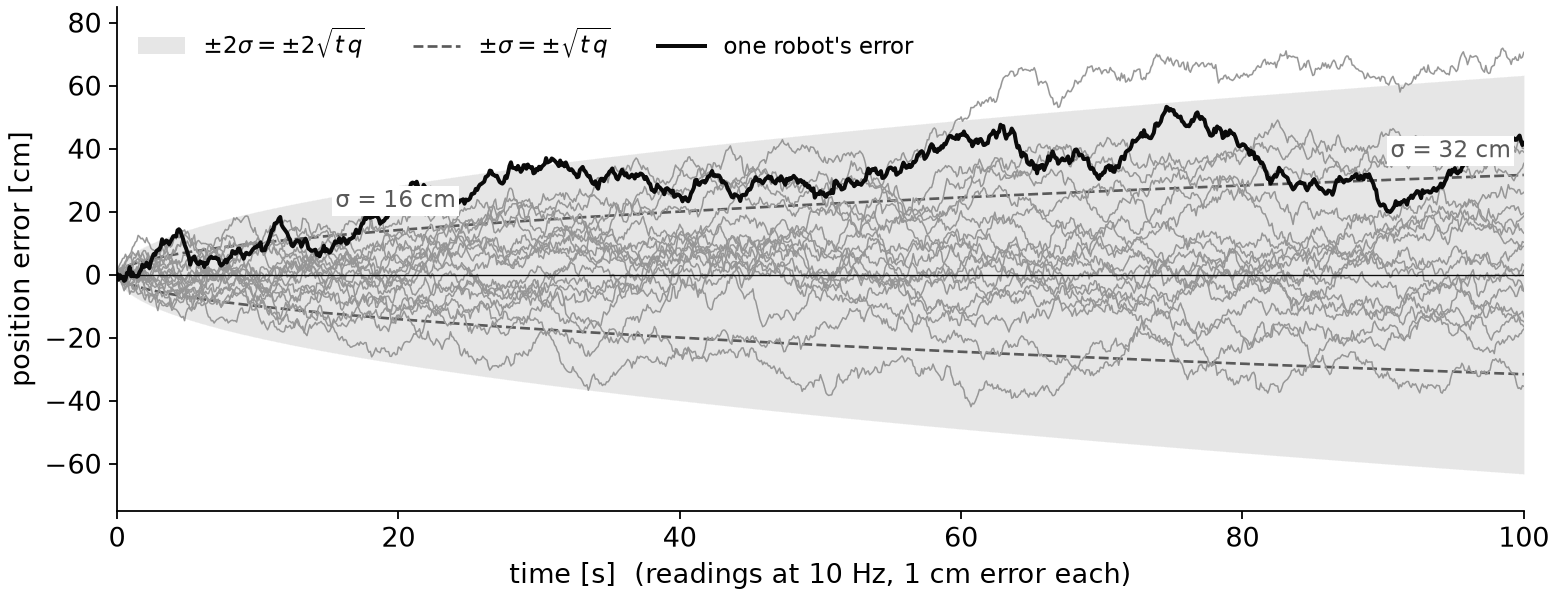
Heading errors are worse: a heading off by $\delta\psi$ tilts every later step, so the position error grows like (distance travelled) $\times\,\delta\psi$. This is the dead reckoning of Lec 7, §5.

- **As control input** (the usual way): the equation above is the predict step; no update → variance grows without bound.
- **As a measurement** o_t — does not help. With state (p, v): $J_h = (0 \;\; 1)$, position is in the **null space**, and $p_{t+1} = p_t + v_t\Delta t$ never feeds p into any measurement → p **unobservable** (Lec 2, §3).

> **What matters is which parts of the state the sensor can see (alone or through the dynamics), not how little noise it has.** Only an absolute measurement (GNSS, lidar on a map, a landmark) bounds the pose covariance.

WHY "RANDOM WALK"?

Subtract the true motion: the error $e_t = p_t - p_t^{\text{true}}$ obeys $e_t = e_{t-1} + w_t$ — each reading adds one **random step**, like a walker stepping in a random direction (Pearson, 1905).



- **Mean 0, spread grows:** $\mathrm{Var}(e_t) = t\,q$, so $\sigma \propto \sqrt{t}$; $4\times$ longer means $2\times$ the error (16 cm at 25 s, 32 cm at 100 s).
- **No pull back to zero:** one robot's error (black) wanders off and stays off. That is **drift**.

SAMPLING AND ALIASING

A digital sensor samples at f_s, $t_n = n/f_s$. **Sampling is an observation map** — and it is not injective:

$$\cos\big(2\pi(f + k f_s)\, t_n\big) = \cos\big(2\pi f\, t_n + 2\pi k n\big) = \cos\big(2\pi f\, t_n\big), \qquad k \in \mathbb{Z}.$$

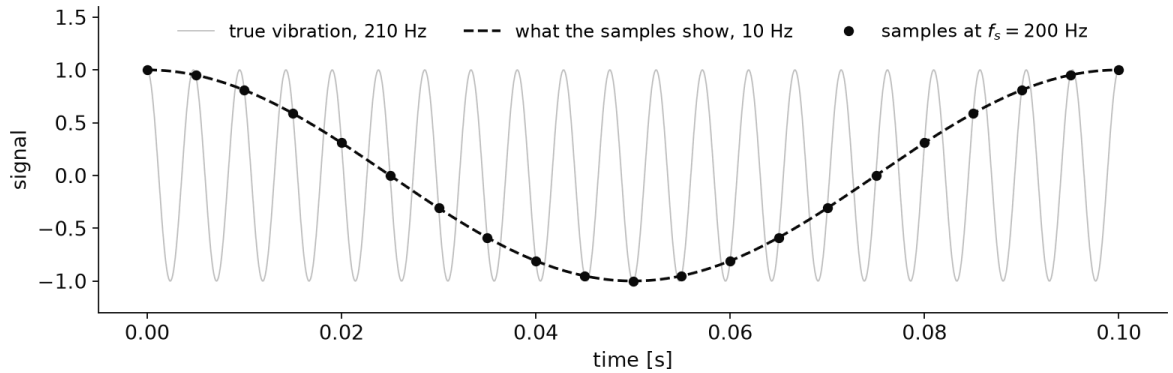
Frequencies f and $f + k f_s$ give **identical samples**: a high frequency masquerades as a low one. This is **aliasing**.

> **Theorem (Nyquist–Shannon sampling).** A signal containing no frequencies above B is uniquely determined by its samples if $f_s > 2B$.

In Lec 2's language: the sampling map is **injective on band-limited signals**, not on all signals.

ALIASING — THE DRONE EXAMPLE

Setup. A drone's IMU is sampled at $f_s = 200$ Hz for the attitude filter. The propellers shake the frame at 210 Hz — a tiny vibration the filter should ignore.



The problem. Its samples are **identical** to those of a 10 Hz rocking ($|210 - 200| = 10$), a plausible body motion. The filter reports rocking that is not there, the controller fights it, and the drone shakes.

- **Not fixable afterwards:** the merged signals come from **different states**, so the samples are not sufficient and, by the DPI, the information is gone. (Non-injective alone is harmless: the mean is sufficient.)
- **Fix before sampling**.

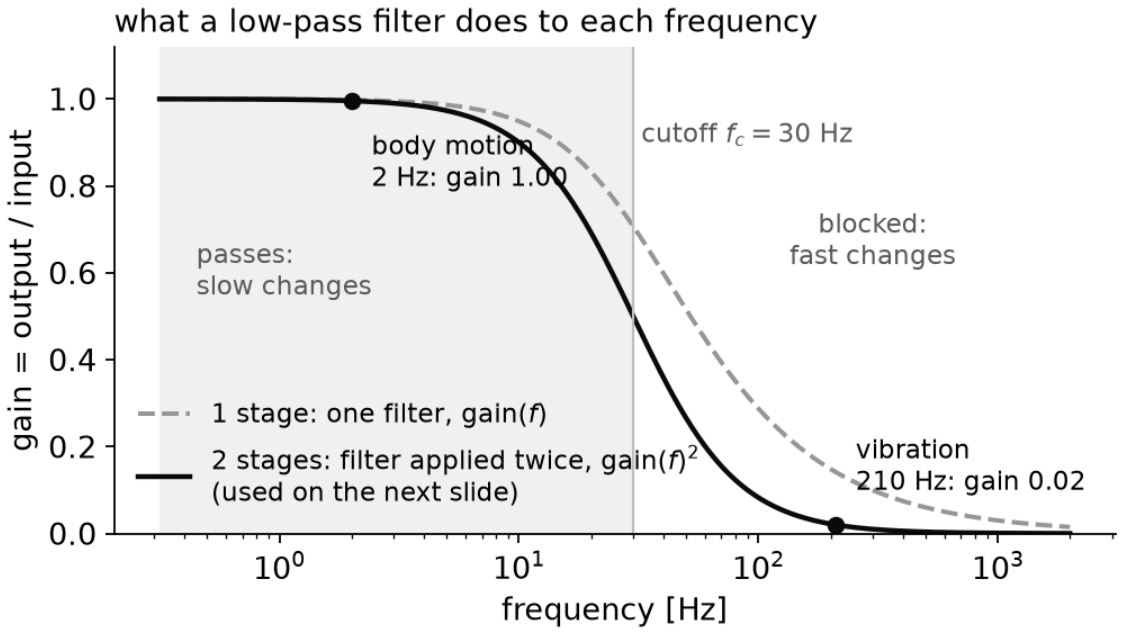
WHAT IS A LOW-PASS FILTER?

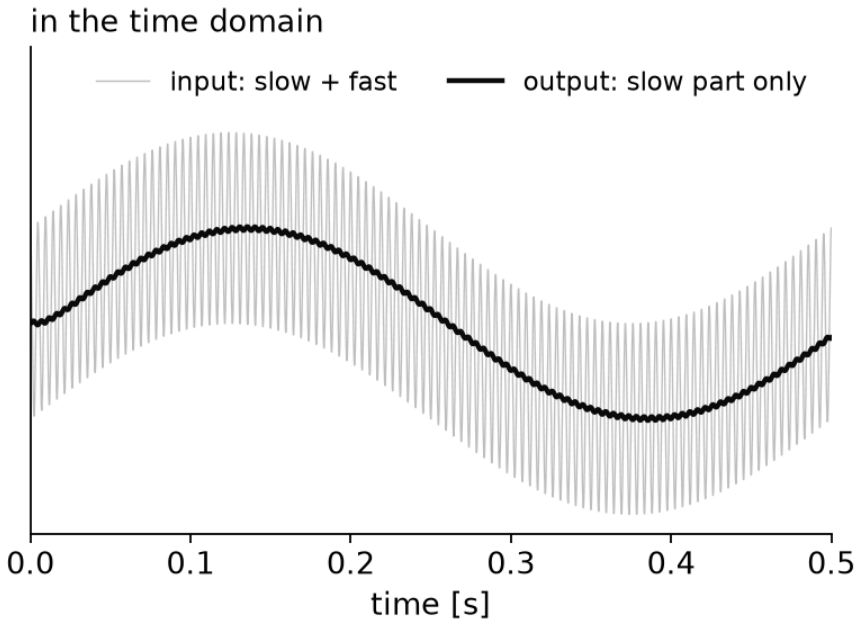
A **low-pass filter** lets **slow** changes through and suppresses **fast** ones; the **cutoff** f_c separates the two. The simplest one smooths each new input into a running value:

$$y_k = y_{k-1} + a\,(x_k - y_{k-1}), \qquad a = 1 - e^{-2\pi f_c \Delta t} \approx 2\pi f_c\,\Delta t, \qquad \mathrm{gain}(f) = \frac{1}{\sqrt{1 + (f/f_c)^2}}.$$

$\mathrm{gain}(f)$: a sinusoid of frequency f comes out scaled by this factor (one stage; plotted on the next slide). a sets f_c only together with the step Δt (at 1 kHz, $f_c = 30$ Hz needs $a \approx 0.17$). **Against aliasing it must act before the sampler**: an RC circuit in hardware. Its opposite, a **high-pass**, keeps the fast part $x - y$ (§9.3).

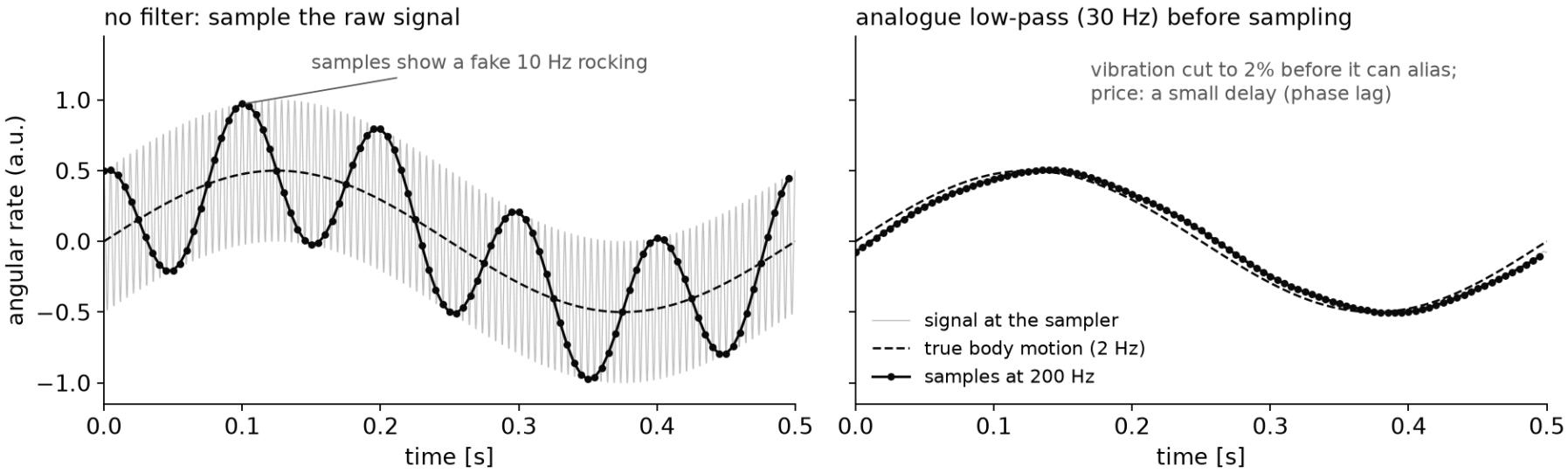
LOW-PASS FILTER — PICTURED





Stages: a 2-stage filter feeds the output of one filter into a second, identical one (two RC circuits in series). Gains multiply, $\text{gain}(f)^2$: slow signals barely change, fast ones fall much faster — at 210 Hz, 0.14 with one stage, 0.02 with two.

FIXING ALIASING: FILTER BEFORE YOU SAMPLE

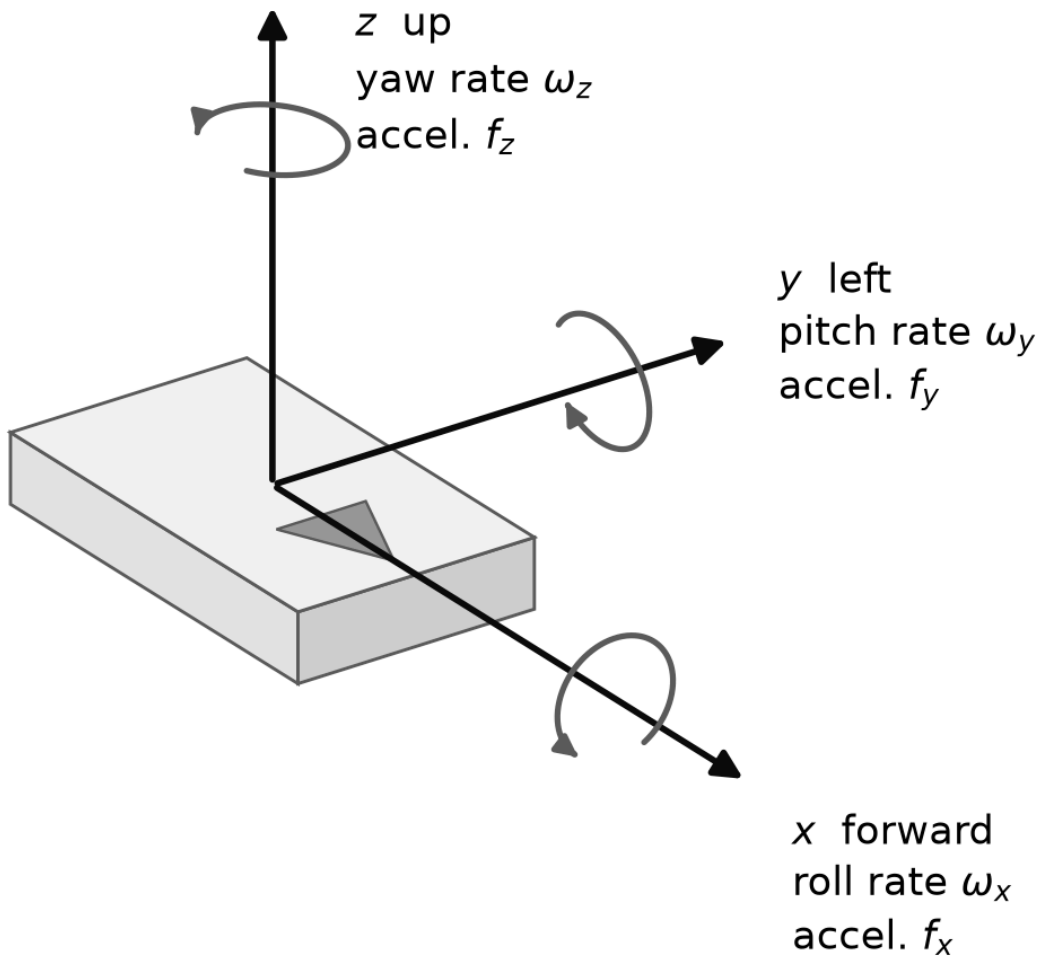


- **Anti-aliasing filter:** an **analogue** low-pass in front of the sampler removes everything above $f_s/2$ while it is still a separate frequency. Here a 2-stage RC filter at 30 Hz cuts the 210 Hz vibration to 2%; the 2 Hz body motion passes, with a small delay.
- **Alternative:** sample much faster (e.g. 8 kHz), low-pass **digitally**, then downsample to 200 Hz — most IMU chips (the tiny silicon sensors in phones and drones) do this internally. **Why:** a digital filter can be far sharper, and is exact and adjustable; a mild analogue filter is still needed, but only above the new $f_s/2 = 4$ kHz.

§9.2 · THE IMU

SPECIFIC FORCE, WANDERING BIASES, AND DRIFT

THE IMU — WHAT IT MEASURES



IMU = 3-axis **gyroscope** + 3-axis **accelerometer**, fixed to the robot. Each sample is **six numbers** in the **body frame**:

- **Gyroscope:** angular velocity $\omega = (\omega_x, \omega_y, \omega_z)$, the rates of **roll** (about x, banking), **pitch** (about y, nose up/down) and **yaw** (about z, heading).
- **Accelerometer:** specific force $f = (f_x, f_y, f_z)$ along the axes — not quite acceleration (see the accelerometer slide).

Integrating ω gives the **orientation**: the roll, pitch and yaw **angles**, or equivalently the rotation matrix $R \in SO(3)$ (body → world: any vector u has $u_{\text{world}} = R\,u_{\text{body}}$; the columns of R are the robot's axes in world coordinates).

THE IMU — GYROSCOPE

Gyroscope — angular velocity in the body frame, plus bias and noise:

$$\omega_m = \omega + b_g + n_g, \qquad \dot{b}_g = n_{b_g}.$$

ω_m: the **m**easured reading (the observation o_t); ω: the true angular velocity; b_g: bias; n_g: white noise (subscript g = **g**yroscope; the accelerometer's are b_a, n_a — not gravity g).

- The bias **wanders slowly** (random walk driven by n_{b_g}).
- A slowly varying bias is **not white noise** → the Markov observation assumption of Lec 5 fails unless b_g is **in the state**.

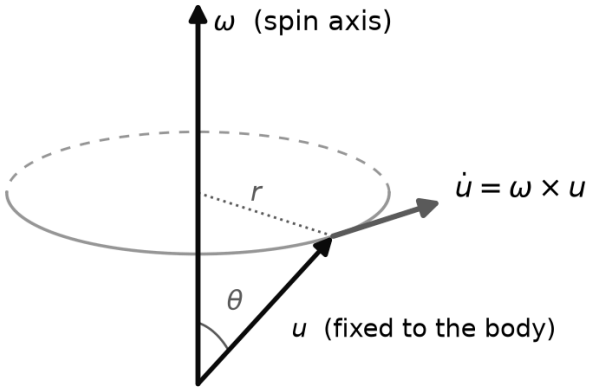
Finding R: integrate the measured rate from a known start R_0:

$$\dot{R} = R\,[\omega]_\times, \qquad R_{t+1} = R_t \exp\big([\omega_t]_\times \Delta t\big), \qquad [\omega]_\times = \begin{pmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{pmatrix}$$

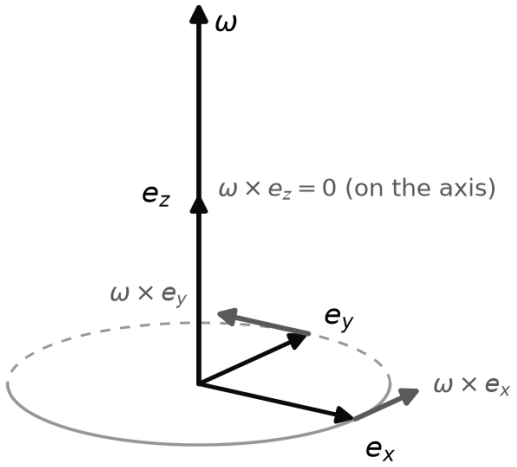
($[\omega]_\times u = \omega \times u$; the $\exp$ step keeps $R \in SO(3)$). Every error in ω is integrated too → R **drifts**, by $\approx b_g\, t$.

WHY $\dot{R} = R\,[\omega]_\times$?

one vector on a spinning body



all three body axes: $\dot{R}$, column by column



the tip runs around the circle:
direction $\perp \omega$ and $\perp u$, speed $= |\omega|\, r$, $r = |u| \sin\theta$

columns of R = body axes, each moves by $\omega \times e_i$
$\Rightarrow \dot{R} = [\omega_{\text{world}}]_\times R = R\,[\omega]_\times$ (gyro: ω in the body frame)

Algebraically:

$$R^\top R = I \;\Rightarrow\; \dot{R}^\top R + R^\top \dot{R} = 0 \;\Rightarrow\; R^\top \dot{R} \text{ is skew-symmetric} = [\omega]_\times \;\Rightarrow\; \dot{R} = R\,[\omega]_\times .$$

So what: only **3** of the 9 entries of $\dot{R}$ are free — one vector ω, exactly what a 3-axis gyro measures.

WHAT DOES AN ACCELEROMETER LYING ON A TABLE READ? AND IN FREE FALL?

THE IMU — ACCELEROMETER

It does **not** measure acceleration. It measures **specific force** (non-gravitational force per unit mass), body frame:

$$f_m = R^\top (a - g) + b_a + n_a, \qquad \dot{b}_a = n_{b_a}, \qquad g = (0, 0, -9.81)\ \mathrm{m/s^2},$$

with f_m the **m**easured reading and a the true acceleration in the world frame. As for the gyro, the bias b_a **wanders** (mostly with temperature): a random walk driven by n_{b_a}, so it too belongs in the state.

- **At rest** ($a = 0$): reads 9.81 m/s² pointing **up** — it feels the table pushing. **Free fall** ($a = g$): reads zero.

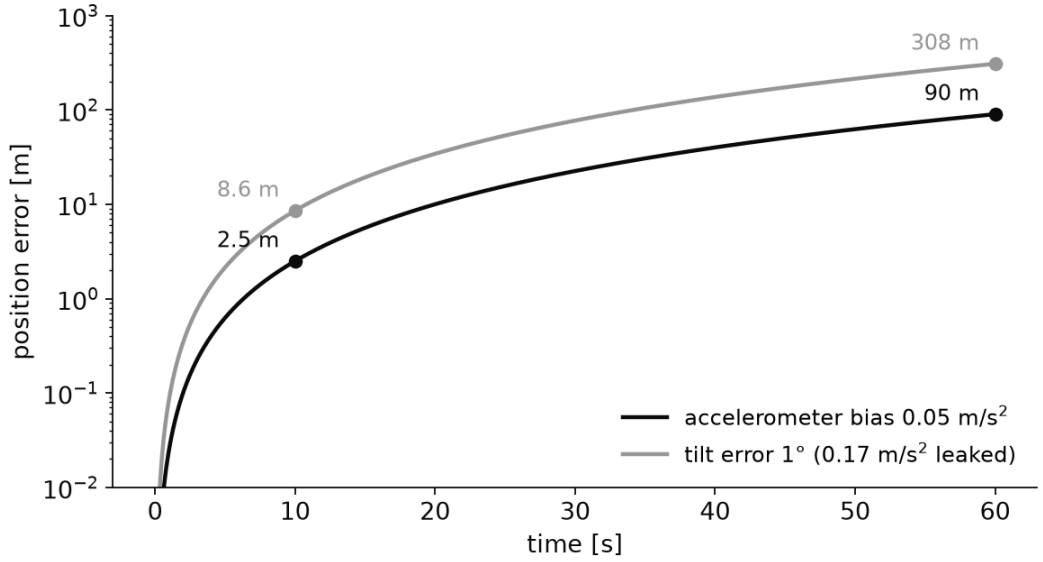
Consequences:

- **At rest, the accelerometer is a tilt sensor.** The direction of f_m is gravity in the body frame → roll and pitch. **No yaw**: rotating about the gravity vector leaves $R^\top g$ unchanged — yaw is a **null direction** of this h.
- **To get acceleration, gravity must be subtracted** — which needs R. An orientation error leaks gravity into the acceleration estimate.
- **Correcting R:** the tilt reading pulls the gyro-integrated R back in roll and pitch (the complementary filter, §9.3 in Lecture 10); yaw needs a compass or GNSS.

INERTIAL NAVIGATION DRIFTS

Integrate the gyro → orientation; double-integrate the gravity-compensated accelerometer → position. **Every error is integrated too.**

ERROR SOURCE	GROWTH
Gyro bias b_g → orientation	$b_g\, t$
Gyro white noise → orientation	$N_g\sqrt{t}$
Accel. bias b_a → position	$\frac{1}{2} b_a t^2$
Tilt error $\delta\theta$ → position	$\frac{1}{2} g\, \delta\theta\, t^2$



Useless alone beyond seconds — superb short-term. How it is used → next slide.

THE IMU IN A FILTER

The IMU reading drives the **predict step** as the known input, in place of the action a_{t-1} in $x_t = f(x_{t-1}, a_{t-1}) + w_t$ — like odometry in Lec 7, §5. Nobody models *why* the robot accelerated; the measured rates are plugged straight into the kinematics.

With the biases in the state, $x = (p, v, R, b_g, b_a)$:

$$R_t = R_{t-1}\exp\big([\omega_m - b_g]_\times \Delta t\big), \qquad v_t = v_{t-1} + \big(R_{t-1}(f_m - b_a) + g\big)\Delta t, \qquad p_t = p_{t-1} + v_{t-1}\Delta t.$$

- **Predict at IMU rate** (100 Hz – 1 kHz): the IMU noise n_g, n_a enters as process noise Q.
- **Update at the absolute sensor's rate** (GNSS 1–10 Hz, lidar, camera): bounds the drift and, through the correlations, **estimates the biases** b_g, b_a.

Not to be confused: a_{t-1} is the course's action, a (accelerometer slide) the acceleration; $f(\cdot)$ is the dynamics, f_m the accelerometer's specific-force reading.

WHERE Q AND R COME FROM — ALLAN VARIANCE

Lec 6–8 **assumed** the noise covariances. For an IMU they are **measured**: record it **at rest** for hours. The true rate is then 0, so the record $y_1, y_2, \ldots$ is pure sensor error.

> **Definition (Allan variance).** Cut the record into consecutive windows of length τ and let $\bar{y}_k$ be the average of window k. The **Allan variance** is half the mean squared change between neighbouring averages,
>
> $$\sigma^2(\tau) = \tfrac{1}{2}\,\mathbb{E}\big[(\bar{y}_{k+1} - \bar{y}_k)^2\big];$$
>
> its square root $\sigma(\tau)$ is the **Allan deviation**. Computed for every τ, it is a curve.

- **Why differences, not the plain variance of $\bar{y}_k$:** no mean has to be estimated (a constant bias cancels in $\bar{y}_{k+1} - \bar{y}_k$), and a wandering bias cannot inflate it with the record length. It measures only what changes **over** τ.
- **Why $\frac{1}{2}$:** so that for white noise $\sigma^2(\tau)$ is **exactly** the variance of one τ-average (next slide).
- **How each noise scales:** averaging white noise over τ shrinks it, $\sigma \propto \tau^{-1/2}$; a bias random walk drifts further between longer windows, $\sigma \propto \tau^{+1/2}$.

ALLAN VARIANCE = VARIANCE OF A τ-AVERAGE (WHITE NOISE)

Let the readings be white: $y_i = \mu + \varepsilon_i$, ε_i independent, variance s^2, sampled at f_s; a window of length τ holds $n = f_s\tau$ samples.

1. **One average:** $\bar{y}_k = \frac{1}{n}\sum_{i\in k} y_i$ has mean μ and variance $\mathrm{Var}(\bar{y}_k) = s^2/n$.
2. **Two neighbours:** the windows share no samples, so $\bar{y}_k$, $\bar{y}_{k+1}$ are independent and μ cancels:

$$\mathbb{E}\left[(\bar{y}_{k+1} - \bar{y}_k)^2\right] = \mathrm{Var}(\bar{y}_{k+1}) + \mathrm{Var}(\bar{y}_k) = \frac{2s^2}{n}.$$

3. **Allan variance:** $\sigma^2(\tau) = \frac{1}{2}\cdot\dfrac{2s^2}{n} = \dfrac{s^2}{n} = \mathrm{Var}(\bar{y}_k)$ — the factor $\frac{1}{2}$ undoes the doubling.
4. **So** $\sigma(\tau) = N/\sqrt{\tau}$ with **noise density** $N = s/\sqrt{f_s}$ $(= \sigma(1\,\mathrm{s}))$. A filter taking one reading every Δt needs exactly $R = \sigma^2(\Delta t) = N^2/\Delta t$.

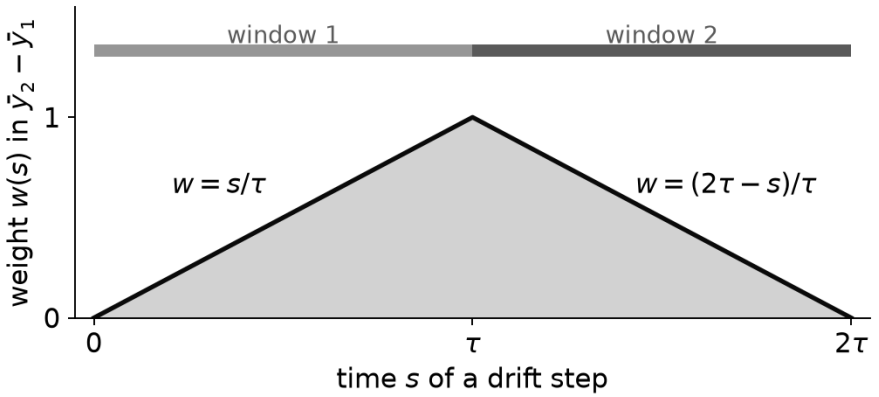
When they differ: with a bias random walk the plain variance of $\bar{y}_k$ grows with the record length and has no fixed value, while $\sigma^2(\tau)$ stays finite and grows with τ ($\sigma^2 \propto \tau$). The Allan variance agrees with the sample variance where that is meaningful and stays defined where it is not.

ALLAN VARIANCE OF A BIAS RANDOM WALK

Error = drifting bias $y = b, \dot{b} = n_b$: $\mathrm{Var}[b(t+h) - b(t)] = K^2 h$. K = how fast the bias wanders, $\approx K\sqrt{h}$ after time h (datasheet: *rate random walk*).

1. **Constant part cancels:** $b(0)$ appears in both averages, so only the drift steps $db(s)$ matter.
2. **Each step's weight:** a step at time s raises b for all later times, so it raises each average by the fraction of that window after s. In $\bar{y}_2 - \bar{y}_1$ this leaves the triangle $w(s)$:

$$\bar{y}_2 - \bar{y}_1 = \int_0^{2\tau} w(s)\, db(s).$$



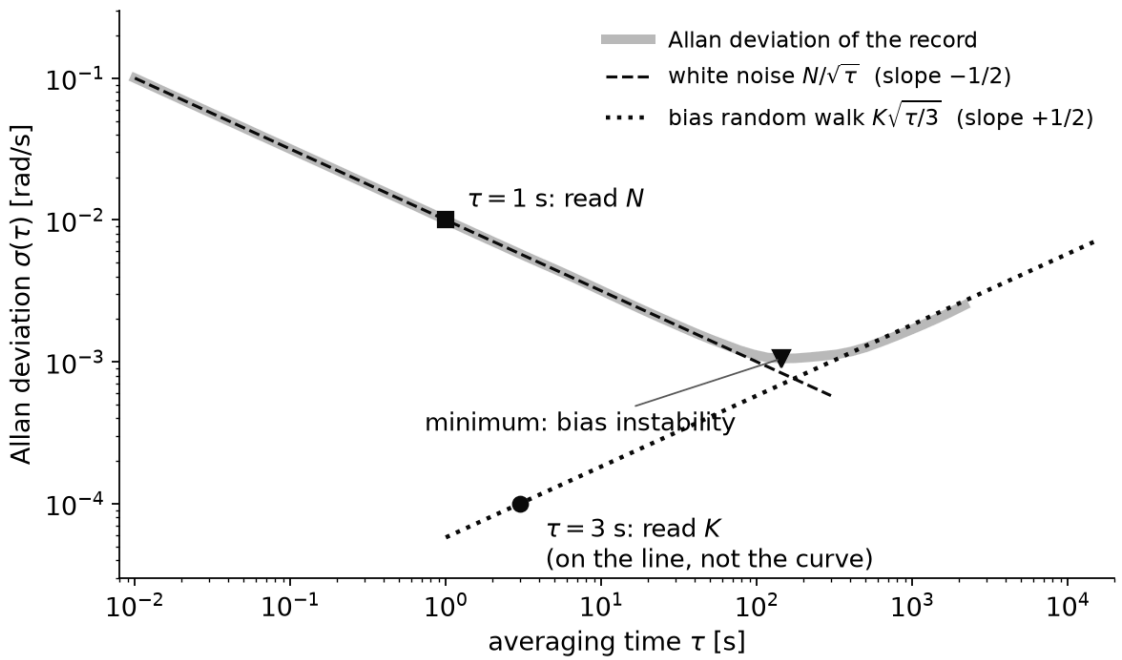
3. **Independent steps add variances:** $\mathrm{Var}(\bar{y}_2 - \bar{y}_1) = K^2 \int_0^{2\tau} w^2\, ds = K^2\left(\frac{\tau}{3} + \frac{\tau}{3}\right) = \frac{2}{3}K^2\tau$.
4. **Allan variance:** $\sigma^2(\tau) = \frac{1}{3}K^2\tau$, so $\log\sigma = \log K + \frac{1}{2}\log\tau - \frac{1}{2}\log 3$: **slope** $+\frac{1}{2}$, and $\sigma(3\,\mathrm{s}) = K$. The filter needs the bias **change** per step, $\mathrm{Var} = K^2\Delta t$ by definition: $Q_b = K^2\Delta t$.

READING THE ALLAN DEVIATION PLOT

On log–log axes, each noise has its own slope of $\sigma(\tau)$:

- **white noise:** $\log\sigma = \log N - \frac{1}{2}\log\tau$, a line of slope $-\frac{1}{2}$ (averaging helps); its value at $\tau = 1$ s is the **noise density** $N \to R = N^2/\Delta t$ or $Q = N^2\Delta t$ (next slide);
- **bias random walk:** $\sigma = K\sqrt{\tau/3}$, slope $+\frac{1}{2}$ (averaging hurts); $K = \sigma(3\,\mathrm{s})$ → bias process noise $Q_b = K^2\Delta t$;
- **minimum** = **bias instability**: the best averaging can do.

Datasheets report exactly these numbers.



Simulated gyro at rest, 24 h at 100 Hz: $N = 0.01$, $K = 10^{-4}$. The dashed and dotted lines are the two formulas, not fits.

FROM THE ALLAN PLOT TO Q AND R

From the plot: gyro N_g, K_g, accelerometer N_a, K_a. **IMU drives the predict step** (the usual case) → **all IMU noise is process noise.** Error state $(\delta p, \delta v, \delta\theta, \delta b_g, \delta b_a) \in \mathbb{R}^{15}$ ($\delta\theta$: the 3 DoF of R); per step Δt:

$$Q = \Delta t \begin{pmatrix} 0_3 & & & & \\ & N_a^2 I_3 & & & \\ & & N_g^2 I_3 & & \\ & & & K_g^2 I_3 & \\ & & & & K_a^2 I_3 \end{pmatrix} \begin{matrix} \leftarrow \delta p \\ \leftarrow \delta v: \text{ velocity random walk} \\ \leftarrow \delta\theta: \text{ angle random walk} \\ \leftarrow \delta b_g: \text{ gyro bias random walk} \\ \leftarrow \delta b_a: \text{ accel. bias random walk} \end{matrix} \qquad (15 \times 15).$$

R belongs to the updating sensor only, e.g. GNSS position: $R = \sigma_p^2 I_3$ (3×3). Example (figure, $\Delta t = 10$ ms): $N_g^2 \Delta t = 10^{-6}$ rad², $K_g^2 \Delta t = 10^{-10}$ (rad/s)².

δp gets no direct noise: position errors arise only by integrating δv. Different axes may have different N, K: replace $N^2 I_3$ by $\mathrm{diag}(N_x^2, N_y^2, N_z^2)$.

SUMMARY

§	QUESTION	ANSWER
8	What makes a pipeline good?	$s \to T(s) = o$ can only **lose** information (DPI); a lossless pipeline outputs a **sufficient statistic** — prove it by factorising the likelihood.
9	What do proprioceptive sensors give?	Derived noise: quantisation $\Delta^2/12$; aliasing is a non-injective map. The IMU measures specific force and angular rate; integration **drifts**; the Allan plot gives Q and R. Gravity makes the accelerometer absolute in roll and pitch.

> **The thread:** is the output sufficient for x, and was it derived or learned? So far every pipeline was **derived**.

WHAT'S NEXT

Lecture 10 — Observation Models (II): Ranging Sensors and Vision

The IMU's two sensors fail in opposite ways: the gyro drifts, the accelerometer is absolute in roll and pitch but noisy. Lecture 10 first fuses them — the **complementary filter** — and then turns to the sensors that bound the drift by measuring the world: ranging sensors (time of flight, the matched filter, lidar, GNSS, radar), then cameras, where the derivation runs out and the pipeline has to be learned.

> *Central question:* How does the agent measure the world — and what happens when the observation model can no longer be derived?