IMU Integration

Graduate Course INTR-6000P

Week 8 - Lecture 15

Changhao Chen
Assistant Professor
HKUST (GZ)

Recap: The Mapping Problem

Input: sensor data (images, LiDAR, IMU)
Output: spatial representation of the environment
Challenges: dynamic objects, large-scale drift, scale ambiguity, real-time constraints

$$M=f(Z,X)$$

where Z are observations and X are poses.

2D vs. 3D Mapping

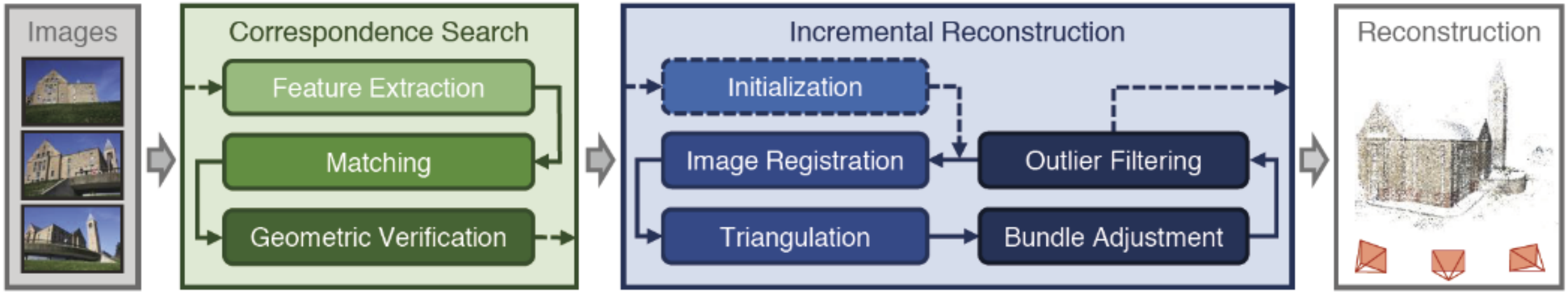
2D maps: for ground robots, indoor navigation
3D maps: essential for autonomous driving and drones

Recap: Structure-from-Motion

Colmap: Structure-from-Motion Revisited (2016)

SfM reconstructs 3D scene structure and camera poses from a set of 2D images.
The process can be divided into three main stages:

1.Correspondence Search
2.Feature Extraction & Matching
3.Incremental Reconstruction



Recap: Sparse vs. Dense Mapping

Sparse: Feature points or landmarks (e.g., ORB-SLAM)

Dense: Surface-level or volumetric reconstruction (e.g., DTAM, ElasticFusion)

Trade-offs: computation vs. detail, robustness vs. accuracy

Discussion:

How to enhance sparse mapping as dense mapping?

Recap: Scene Representation

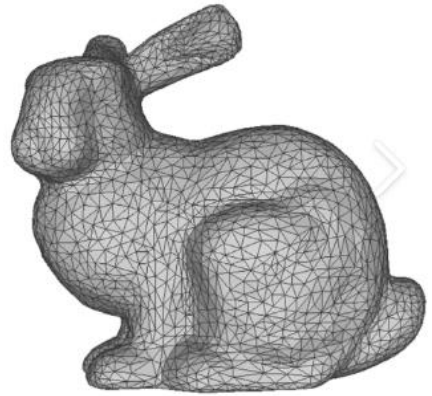
Geometric representations describe the **spatial structure** of the scene.

Common forms:
1.Point clouds
2.Meshes
3.Voxel grids
4.Signed distance fields (SDFs)
5.Neural Scene Representations

Point cloud

Voxel

Polygon mesh



Point Clouds

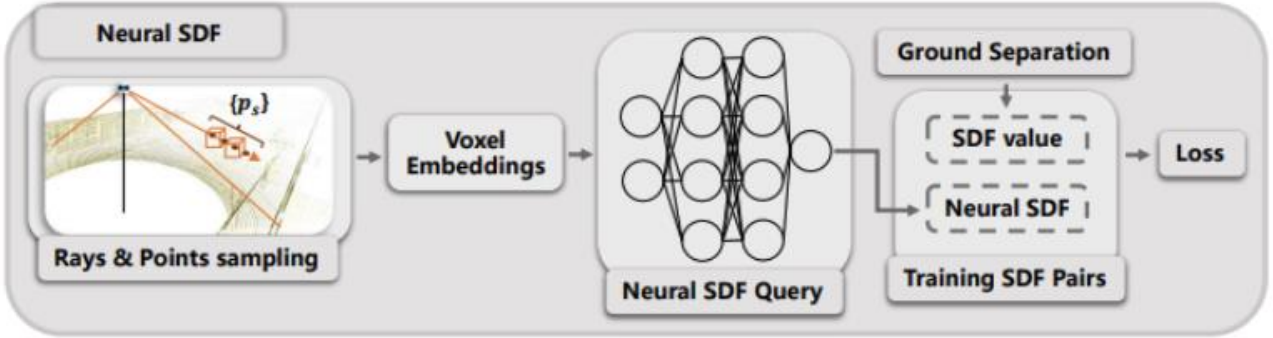
- Collection of 3D points sampled from scene surfaces.
- Directly obtained from LiDAR or stereo triangulation.
- Pros: Simple, flexible, accurate.
- Cons: Unstructured, memory-intensive, lacks topology.

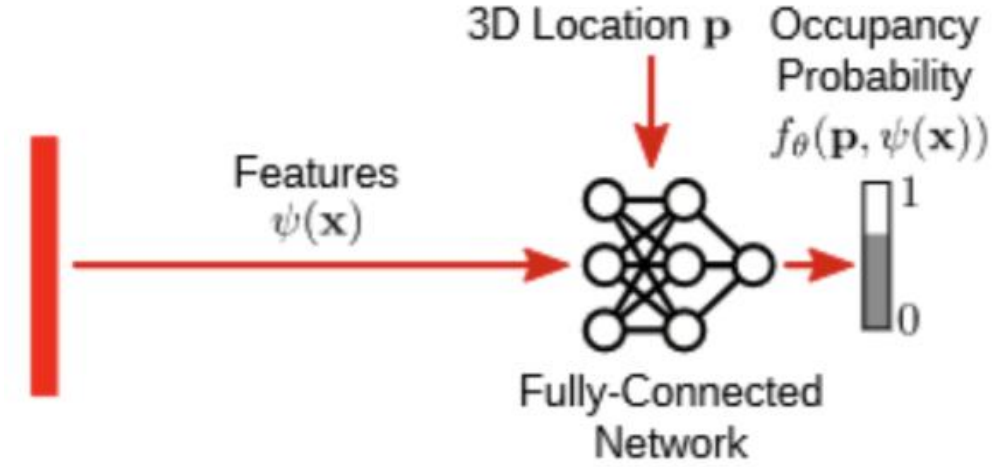
Recap: Neural Implicit Representations

Encode 3D scenes as continuous neural fields.
Implicit function

$$f_\theta(\mathbf{x}) \rightarrow$$ color, density, occupancy.

Smooth, memory-efficient, and differentiable.





Recap: Neural Rendering

1. Volume Rendering Principle

Each pixel's color is computed by integrating the emitted radiance and transmittance along the camera ray:

$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\,\sigma(\mathbf{r}(t))\,c(\mathbf{r}(t),\mathbf{d})\,dt$$

$$T(t) = \exp\left(-\int_{t_n}^{t}\sigma(\mathbf{r}(s))ds\right)$$

2. Training Objective

•Rendered color is compared with ground truth image pixel color

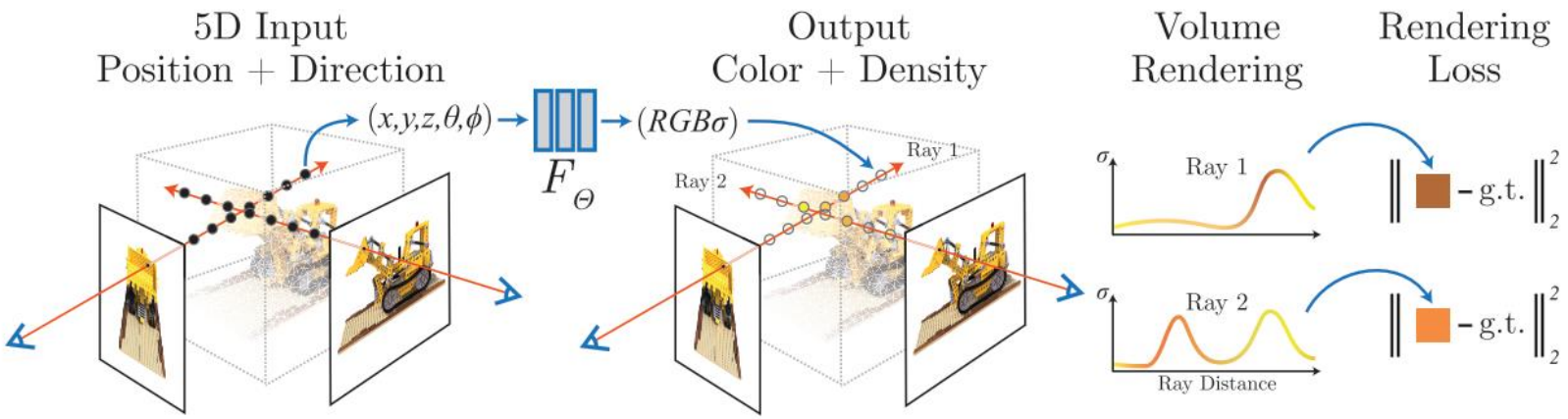
•**Loss:**

$$\mathcal{L} = \sum_{\mathbf{r}} \|C(\mathbf{r}) - C^*(\mathbf{r})\|^2$$

3. Optimization

•Train MLP parameters θ using multiple images with known camera poses.

•The network learns to encode scene geometry (via density) and appearance (via color).



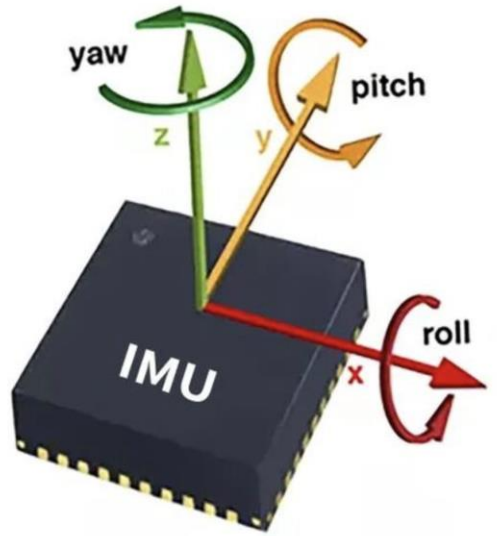
Inertial Sensor

IMU (Inertial Measurement Unit):

An IMU is a self-contained sensor that measures **specific force** and **rotation rate**.

Key Strengths:

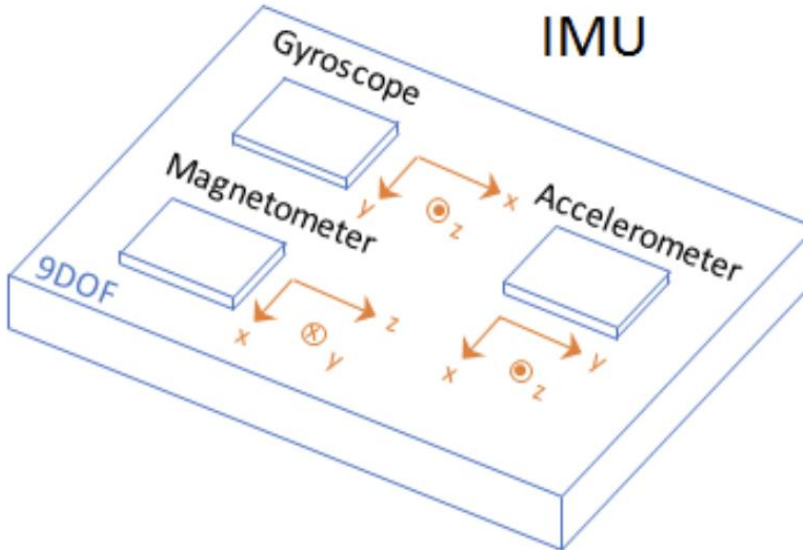
- **High Frequency:** Provides data at 100-1000 Hz, much faster than cameras (30 Hz) or LiDAR (10-20 Hz).
- **Complementary to Vision:** Insensitive to lighting, texture, and weather conditions that challenge cameras.
- **Metric Scale:** Inherently provides scale information, which monocular vision lacks.
- **Fill in the Gaps:** Covers for temporary vision/LiDAR failures (e.g., quick turns, tunnels, bright sun).



Critical Weakness: Its estimates drift over time. Any small error in measuring acceleration or rotation is integrated into the position estimate, leading to exponentially growing inaccuracies if not corrected by other sensors.

Inertial Sensor

- **Gyroscope:**
  - **Measures:** Angular velocity (ω) in rad/s.
  - **What we get:** How fast the vehicle is turning (yaw, pitch, roll rates).
- **Accelerometer:**
  - **Measures:** Proper acceleration (a) in m/s^2.
  - **Crucial Point:** It measures *specific force*, which is the sum of:
    - **Linear acceleration** (what we want)
    - **Gravity** (a massive disturbance we must account for)



Role of IMU in Navigation

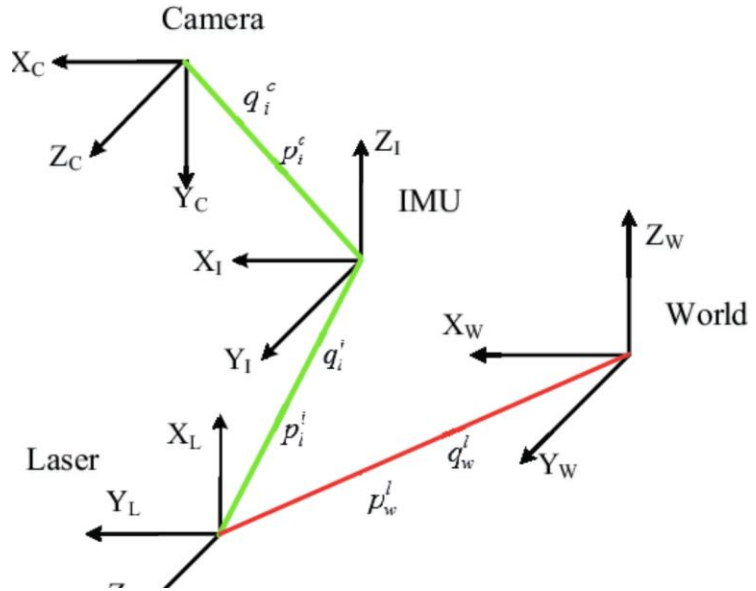
Content:

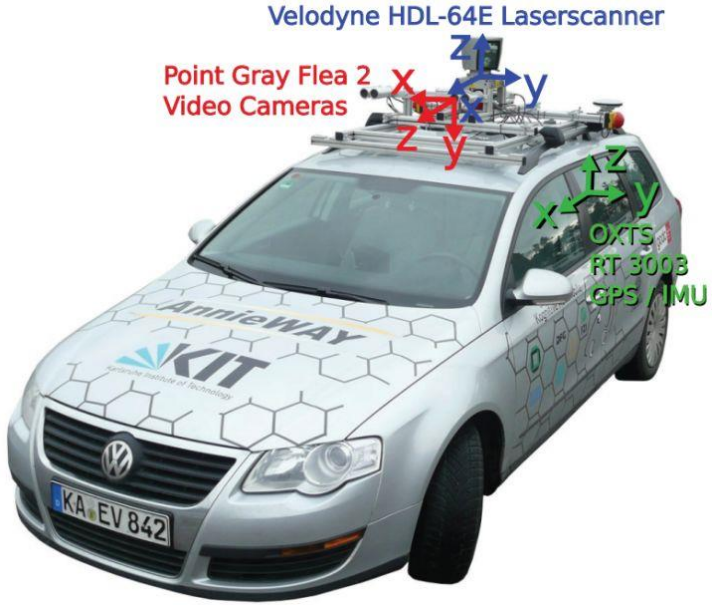
- Provides high-frequency motion cues (100–1000 Hz).
- Enables **short-term motion tracking** between visual updates.
- Critical for:
  - Fast maneuvers
  - Low-texture or low-light conditions
  - Reducing motion blur impact

IMU Coordinate Frames

World frame (W): global reference (gravity-aligned).
Body frame (B): attached to IMU (vehicle frame).
IMU provides motion in body frame; we estimate in world frame.

$$ {}^{W}\mathbf{p}_B,\ {}^{W}\mathbf{v}_B,\ {}^{W}\mathbf{R}_B $$





IMU Kinematic Equations

$$\dot{\mathbf{R}} = \mathbf{R}[\omega - b_g - n_g]_\times$$
$$\dot{\mathbf{v}} = \mathbf{R}(a - b_a - n_a) + g$$
$$\dot{\mathbf{p}} = \mathbf{v}$$

ω: angular velocity; a: linear acceleration.

b_g, b_a : biases;

n_g, n_a : Gaussian noise.

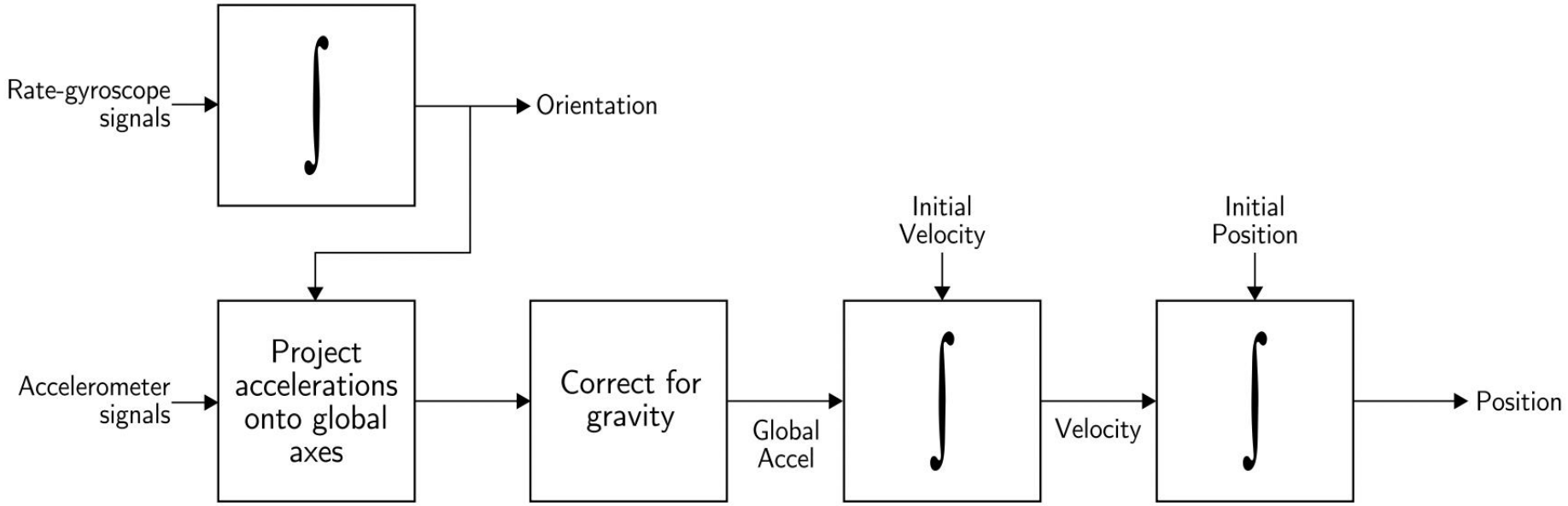
IMU State Vector

$$\mathbf{x} = [\mathbf{R}, \mathbf{v}, \mathbf{p}, b_g, b_a]$$

Orientation, velocity, position.

Biases evolve as random walks:

$$\dot{b}_a = n_{ba}, \quad \dot{b}_g = n_{bg}$$



Discrete-Time Propagation

IMU data sampled at discrete times (Δt).

Approximate Propagation Equations:

$$\mathbf{R}_{k+1} = \mathbf{R}_k \exp([\omega_k - b_g]_\times \Delta t)$$
$$\mathbf{v}_{k+1} = \mathbf{v}_k + (\mathbf{R}_k(a_k - b_a) + g)\Delta t$$
$$\mathbf{p}_{k+1} = \mathbf{p}_k + \mathbf{v}_k \Delta t + \tfrac{1}{2}(\mathbf{R}_k(a_k - b_a) + g)\Delta t^2$$

Orientation Integration

Integrate angular velocity to get orientation.

Representations:

- Euler angles (simple but singular).
- Rotation matrix (orthogonal constraint).
- Quaternion (common in robotics).

$$\mathbf{q}_{t+\Delta t} = \mathbf{q}_t \otimes \exp_q((\omega - b_g)\Delta t)$$

Velocity and Position Integration

Integrate acceleration twice → velocity → position.
IMU measures specific force = acceleration − gravity.

$$\mathbf{a}_W = \mathbf{R}_B^W (a_B - b_a) + g$$

What the accelerometer measures:

$$\mathbf{f}_B = \mathbf{a}_B - \mathbf{R}_W^B \mathbf{g}$$

where

- $\mathbf{f}_B$: *specific force* (measured by IMU in body frame)
- $\mathbf{a}_B$: *true linear acceleration of the IMU center*
- $\mathbf{g}$: *gravity vector* in world frame (e.g., [0, 0, −9.81] m/s²)
- $\mathbf{R}_W^B$: rotation from world to body frame

Velocity and Position Integration

The accelerometer **does not directly measure gravity** — it measures **everything except gravity** (i.e., *specific force*).

Therefore, to obtain true acceleration in world frame:

$$\mathbf{a}_W = \mathbf{R}_B^W(\mathbf{f}_B) + \mathbf{g}$$

Integration Steps

$$\mathbf{v}_{k+1} = \mathbf{v}_k + (\mathbf{R}_B^W(\mathbf{f}_B) + \mathbf{g})\Delta t$$
$$\mathbf{p}_{k+1} = \mathbf{p}_k + \mathbf{v}_k\Delta t + \tfrac{1}{2}(\mathbf{R}_B^W(\mathbf{f}_B) + \mathbf{g})\Delta t^2$$

IMU Error Sources

Deterministic:
- Bias (offset)
- Scale factor
- Misalignment

Stochastic:
- White noise
- Bias instability / random walk

Allan Variance and Noise Modeling

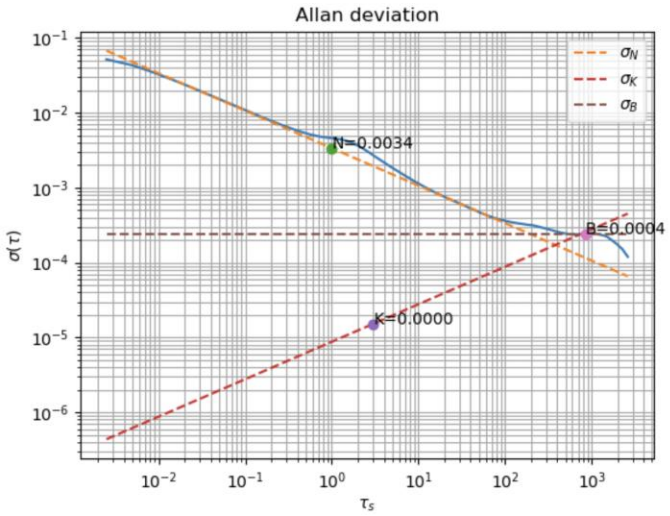
$$\Omega(t) = \Omega_{Ideal}(t) + Bias_N(t) + Bias_B(t) + Bias_K(t)$$

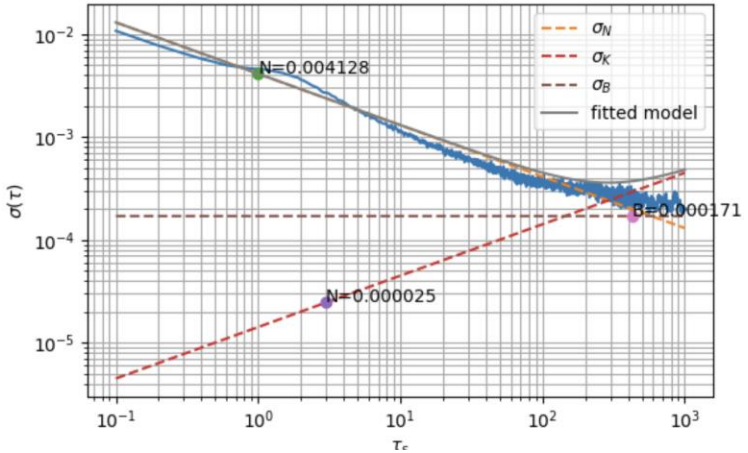
Used to characterize IMU noise.
Log-log plot: deviation σ vs. averaging time τ.

```
#Accelerometer
accelerometer_noise_density: 0.006308226052016165
accelerometer_random_walk: 0.00011673723527962174

#Gyroscope
gyroscope_noise_density: 0.00015198973532354657
gyroscope_random_walk: 2.664506559330434e-06

rostopic: '/sensors/imu' #Make sure this is correct
update_rate: 400.0 #Make sure this is correct
```





Source: https://github.com/ori-drs/allan_variance_ros

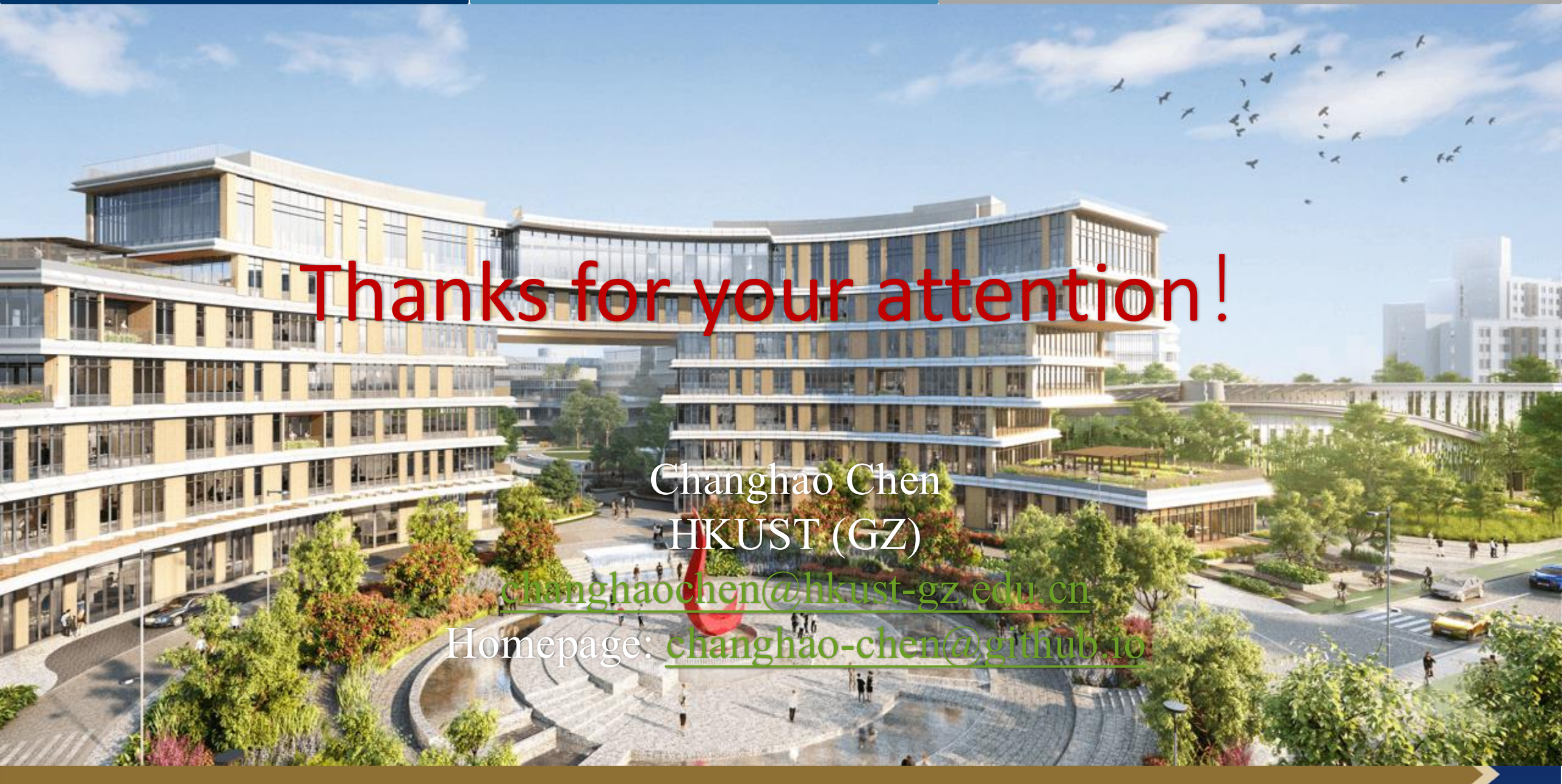
Error Growth and Drift

Gyroscope bias drift → orientation error.
Accelerometer bias → position drift.

IMU integration drift accumulates:

$$\text{Orientation drift} \propto t, \quad \text{Velocity drift} \propto t^2, \quad \text{Position drift} \propto t^3$$

Static calibration: measure offsets while stationary.
Dynamic calibration: perform known rotations/motions.
Use toolkits (Kalibr, AllanTools).



Thanks for your attention!

Changhao Chen
HKUST (GZ)
changhaochen@hkust-gz.edu.cn
Homepage: changhao-chen@github.io