

Pedestrian Dead-Reckoning (PDR)

Cómo estimar la trayectoria de una persona con sensores Inerciales

Dr. Antonio Ramón Jiménez Ruiz
Centre for Automation and Robotics (CSIC-UPM)
Ctra. Campo Real km. 0,2 Arganda del Rey (Madrid)
e-mail: antonio.jimenez@csic.es

CENTRE FOR AUTOMATION AND ROBOTICS

CONSEJO SUPERIOR DE INVESTIGACIONES CIENTÍFICAS

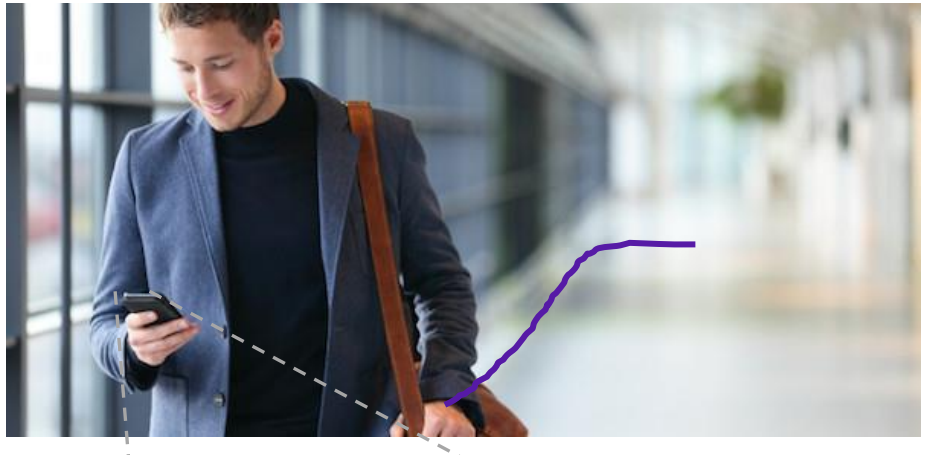
POLITÉCNICA

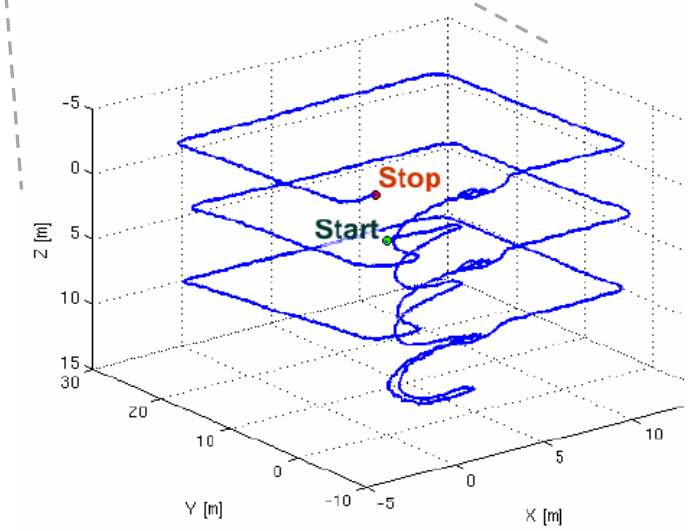
Outline

- Some theory:
  1. What is PDR?
  2. Inertial Navigation (INS)
  3. Implementation problems
  4. PDR algorithmic solutions
- Practice (Matlab):
  1. Introduction
  2. PDR with pre-recorded logfiles
  3. PDR with your own phone

What is PDR?

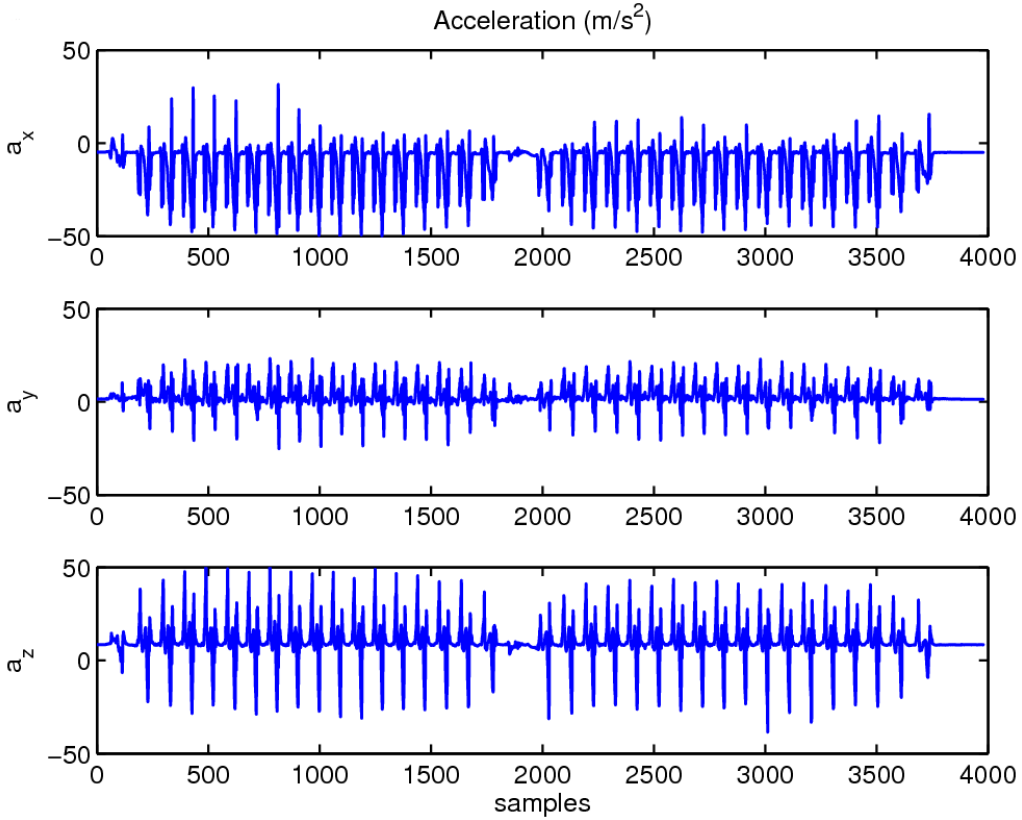
- A method to:
  - **estimate the user's trajectory** (Position & Heading)
  - by **integrating inertial measurements**
- No need of external beacons: GPS, Cell-positioning, LPS (WiFi, BLE, UWB, US, Light,…)
- Assumes known initial conditions:
  - position and orientation
- Uses Inertial data:
  - Acceleration (m/s^2)
  - Angular rate (rad/s)

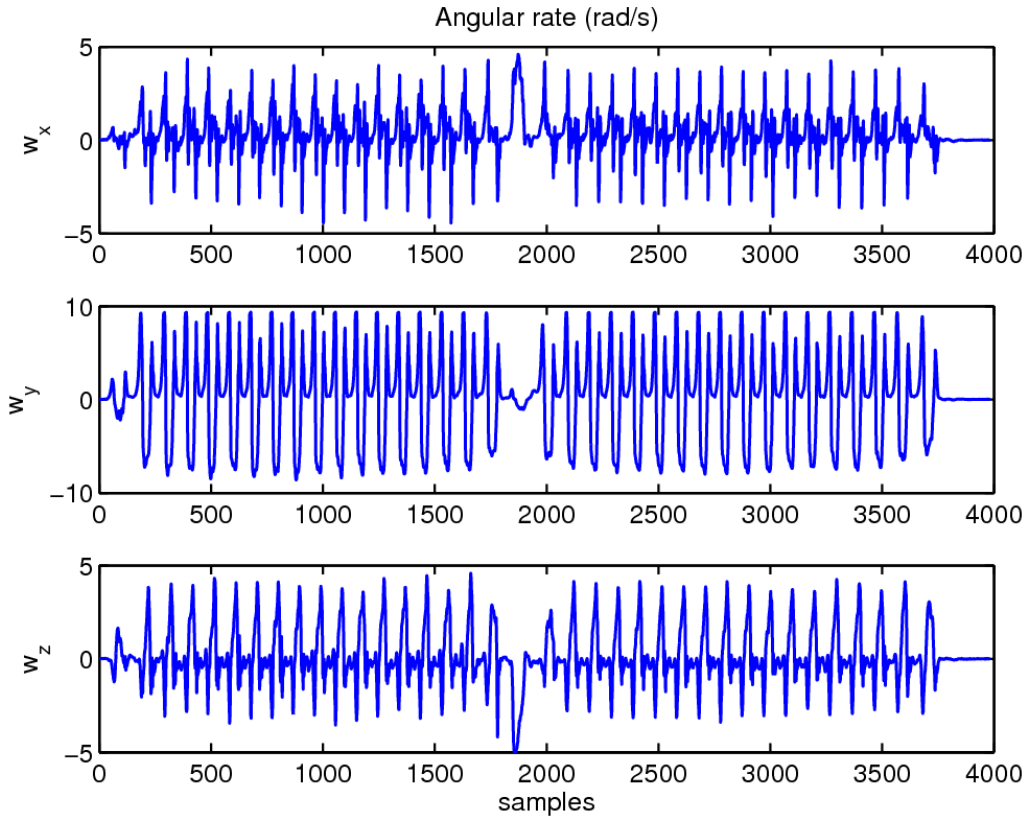




What is PDR?

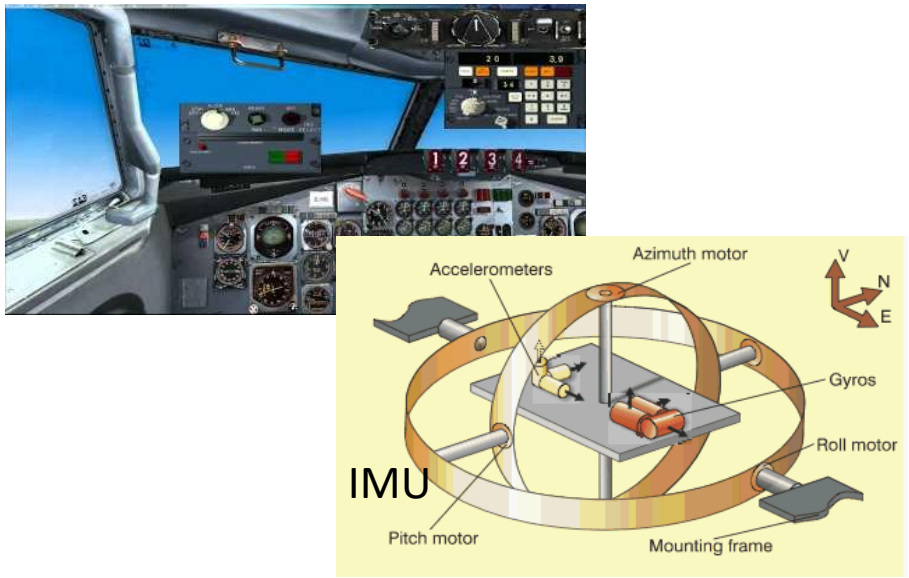
- «Acceleration & Angular rate» signals to integrate
  - Example: 60 meters walk
  - go (18 steps) + 180º turn (1 step) + return walk (18 steps)





What is PDR?

- Acceleration, Angular rate, trajectory integration ??…
  it sounds like «Inertial Navigation» or INS



INS



- Can I use Inertial Navigation System (INS) algorithms for humans (PDR)?

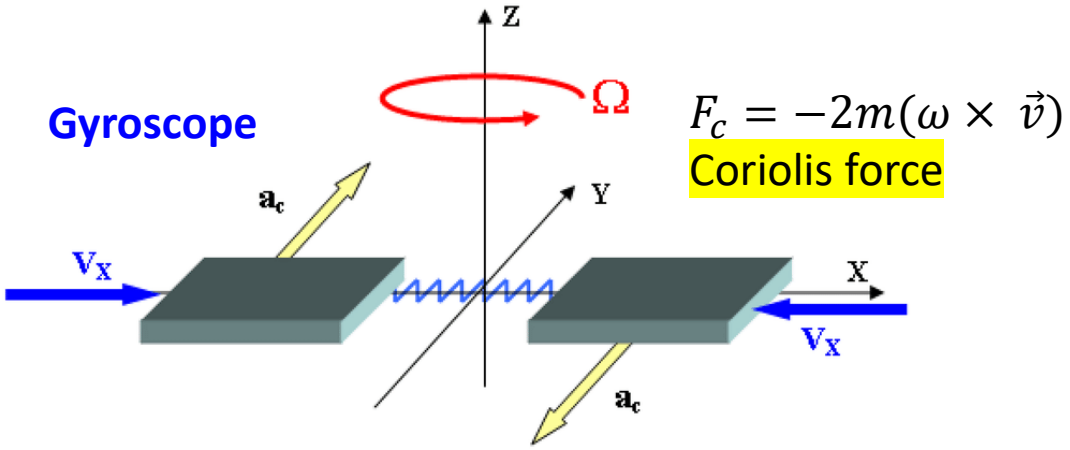
Outline

- Some theory:
  1. What is PDR?
  2. Inertial Navigation (INS)
  3. Implementation problems
  4. PDR algorithmic solutions
- Practice (Matlab):
  1. Introduction
  2. PDR with pre-recorded logfiles
  3. PDR with your own phone

2. Inertial Navigation (INS)

– Uses an IMU (Inertial Measurement Unit)

- 3 accelerometers (measuring “specific force” [m/s^2] caused by **motion** and also **gravity**)
- 3 giroscopes (measure “angular rate” [rad/s])

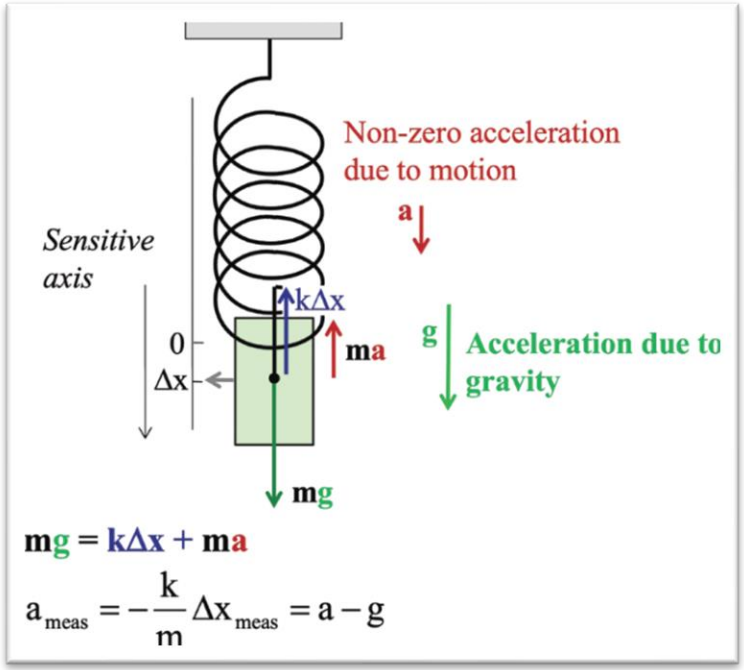


Gyroscope

$F_c = -2m(\omega \times \vec{v})$

Coriolis force

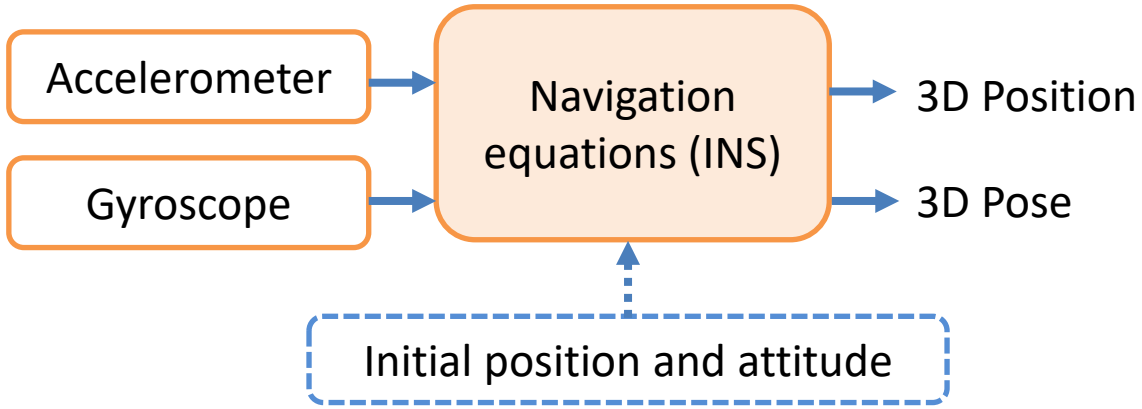
Oscillating masses. If rotation w => Measurable lateral force is generated

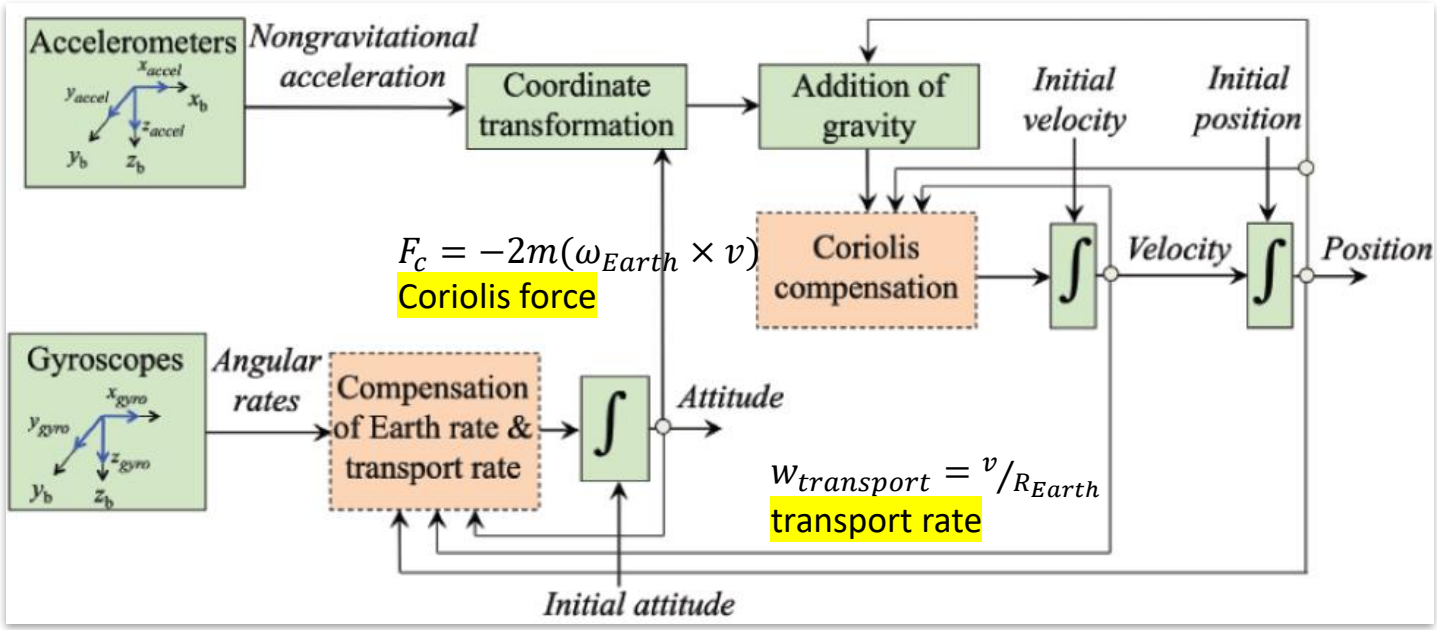


Accelerometer

2. Inertial Navigation (INS)

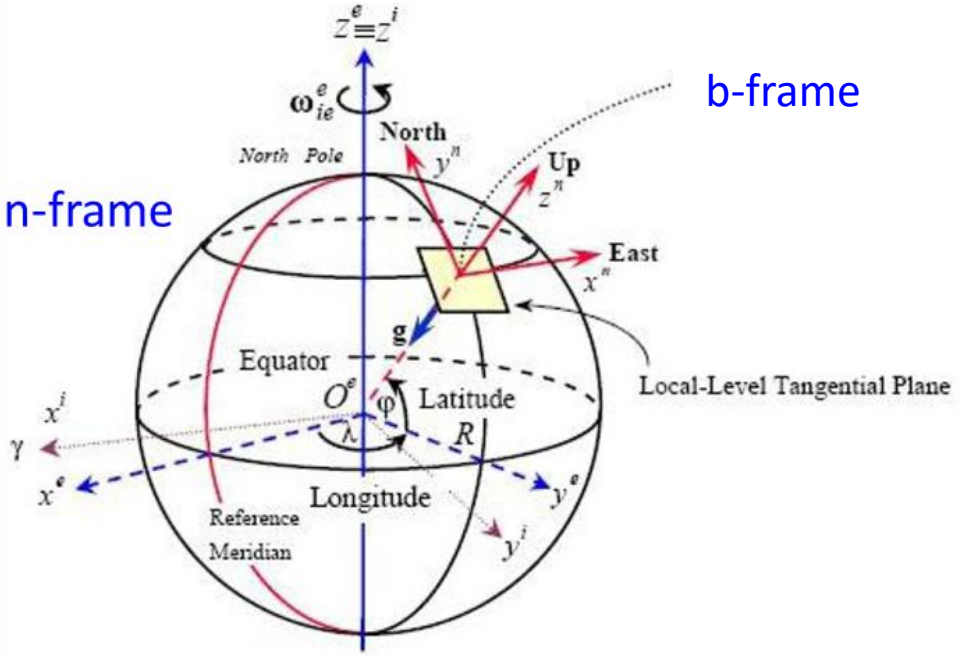
- INS Navigation equations:
  - Integrates Inertial data:
    - From aceleration -> velocity -> position
    - From angular rate -> angles (pose)
  - From a given initial position and pose

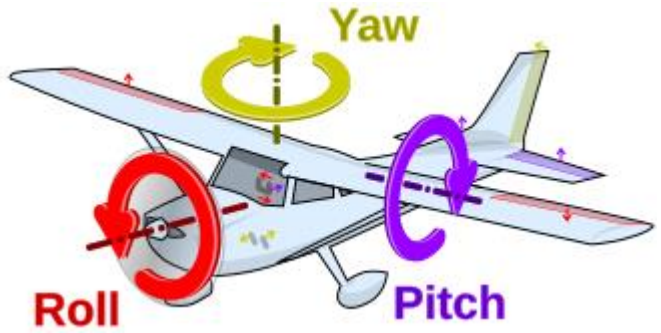




2. Inertial Navigation (INS)

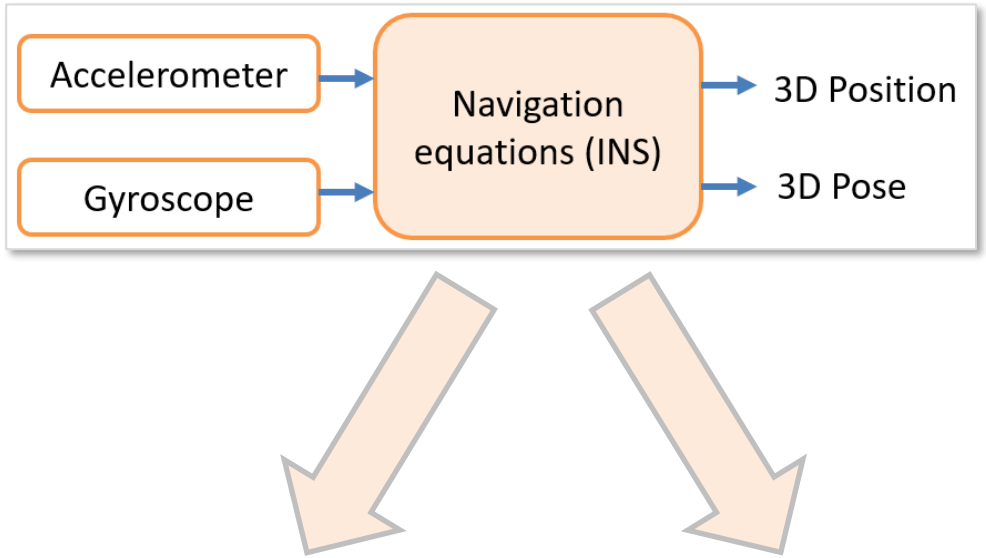
- Reference frames:
  - **Navigation** reference frame (n-frame):
    - Inertial (Earth Centered), ECEF, Local (Leveled)
  - **Body** reference frame (b-frame):
    - Attached to object to track (where IMU is placed)
- Attitude:
  - **3D orientation** IMU reference system (b-frame) respect to the global reference system (n-frame)
    - Euler: Roll, Pitch, Yaw (vector 1x3)
    - Direction Cosine Matrix (DCM): $\mathbf{R_b}^{\mathbf{n}}$, (matrix 9x9)
    - Quaternions, (vector 1x4)



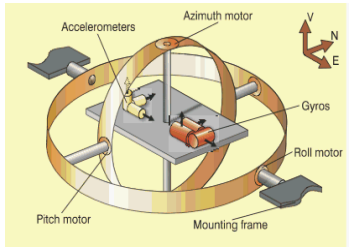


2. Inertial Navigation (INS)

2 types of INS:



1) INS with stabilized platform (Gimbal)



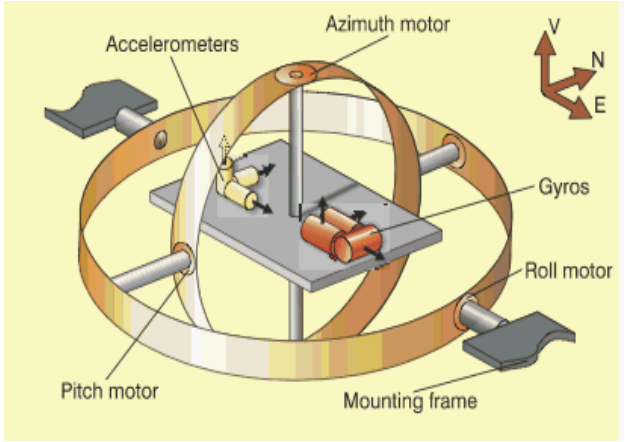
2) Strap-down INS





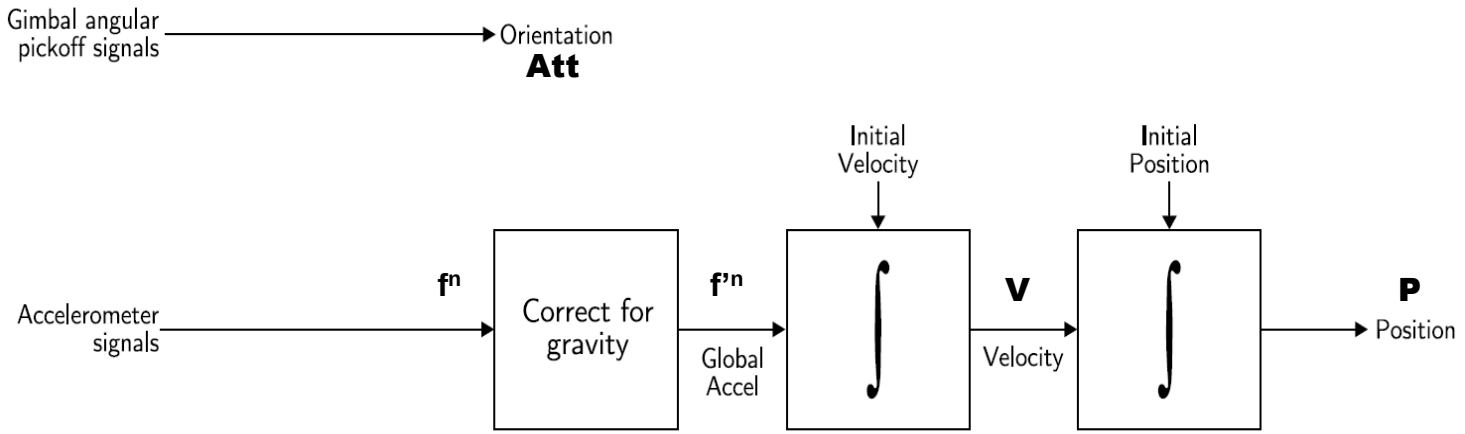
2. Inertial Navigation (INS)

- 1) INS with stabilized platform (Gimbal)
  - Stabilized with motors to keep gyro signals to zero.
  - Readings of axis: (Roll,Pitch,Yaw) directly gives the Attitude
  - Accelerometer signals are in n-frame => so INS only has to substract “gravity”, and double integration to get V and P



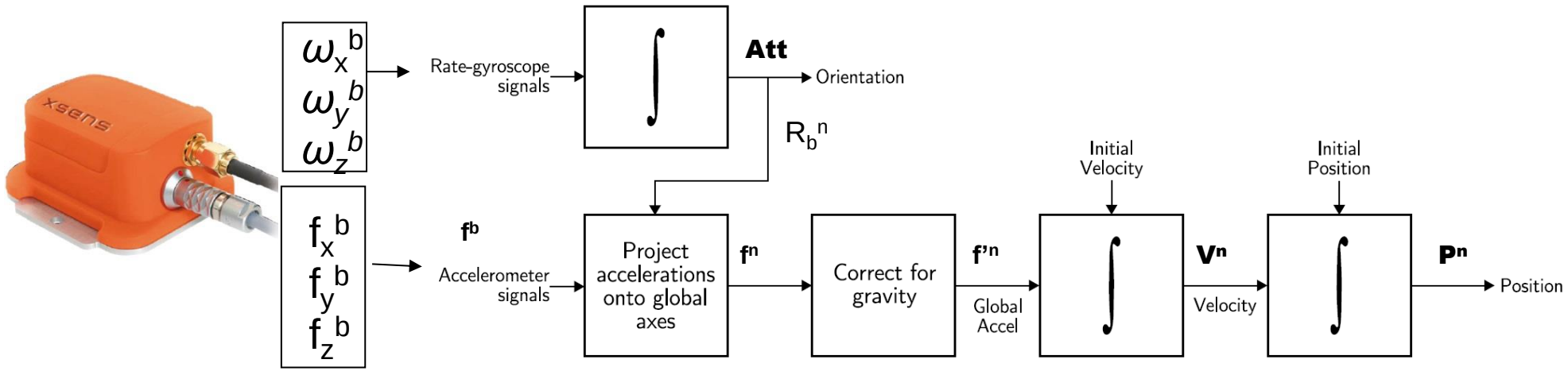
Gimbal frame





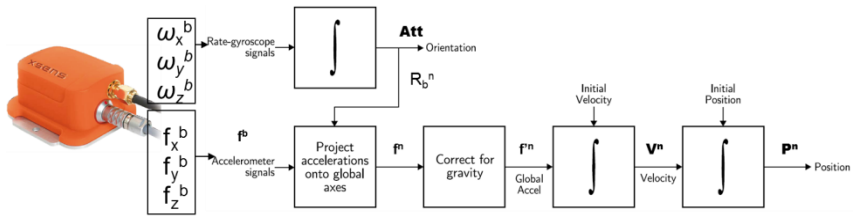
2. Inertial Navigation (INS)

- 2) Strap-down INS:
  - No stabilized platform. 3 Acc y 3Gyr fixed to the body.
  - Integrate angular rate (Gyro) to keep track of Attitude (R_b^n)
  - Transform especific force (f^b) measureed by acelerometers in b-frame to n-frame (f^n) using Attitude.
  - Eliminate gravity (g) to that acceleration (f^n), obtaing: f'^n
  - Integrate f'^n to obtain velocity (V^n), and again to get position (P^n)



2. Inertial Navigation (INS)

- Strap-down INS: Implementation in Matlab




```
%-------------------------------------------------------------------------
%  INS
%  Acc: IMU Acelerometer signal (3 axis) [m/s^2]
%  Gyr: IMU Angular Rate signal (3 axis) [rad/s]
%  fs:  Frecuencia muestreo [Hz]

% Actualizar matriz de orientación con valores de los "Gyros"
SkewGyros=[0 -Gyr(i,3)  Gyr(i,2);...
        Gyr(i,3)  0  -Gyr(i,1);...
        -Gyr(i,2) Gyr(i,1) 0]*pi/180; % rad/s
Rot_nb(:,:,i)=Rot_nb(:,:,i-1)*(2*eye(3)+SkewGyros/fs)*inv(2*eye(3)-SkewGyros/fs); % Pade(1,1)

%.........................................................
% Transformar medidas sensoriales de "b-frame" a "n-frame"
Acc_n(i,:)=(Rot_nb(:,:,i)*Acc(i,:)')';  %(en "n-frame": Noth-West-Up)

%.........................................................
% Eliminar gravedad terrestre (que se lee en acelerómetros)
Acc_n_sin_g(i,:)=Acc_n(i,:) - [0 0 gravedad];  %(en "n-frame": Noth-West-Up)

%.........................................................
% Integrar doblemente aceleraciones (ahora ya sin la gravedad)
Vel_n(i,:)=Vel_n(i-1,:) + 0.5*(Acc_n_sin_g(i,:)+Acc_n_sin_g(i-1,:))/fs;  % integracion trapezoidal
Pos_n(i,:)=Pos_n(i-1,:) + 0.5*(Vel_n(i,:)+Vel_n(i-1,:))/fs;  % integracion trapezoidal
```

2. Inertial Navigation (INS)

- INS concept is easy and we have the code



- So, PDR problem is solved:
  - Let's take an IMU and
  - implement PDR as INS !!!

- Any problem with that?
  - Yes, several….



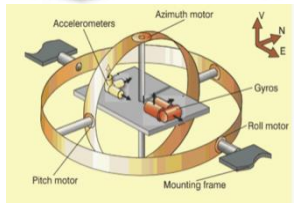
Outline

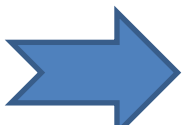
- Some theory:
  1. What is PDR?
  2. Inertial Navigation (INS)
  3. Implementation problems
  4. PDR algorithmic solutions
- Practice (Matlab):
  1. PDR code and tools
  2. PDR with pre-recorded logfiles
  3. PDR with your own phone

3. Implementation problems

- The **size** and **weight** of IMU:
  - Pure INS solutions require bulky sensors
  - Not good to be carried by a person in PDR





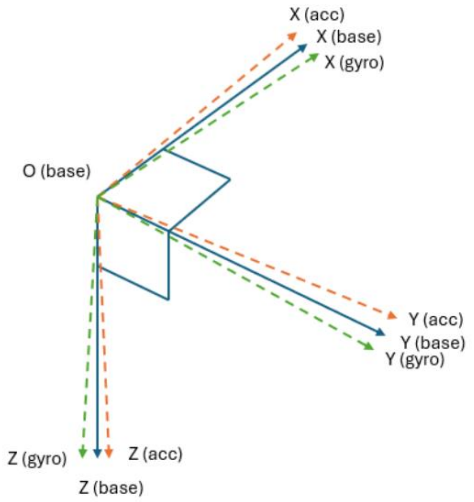


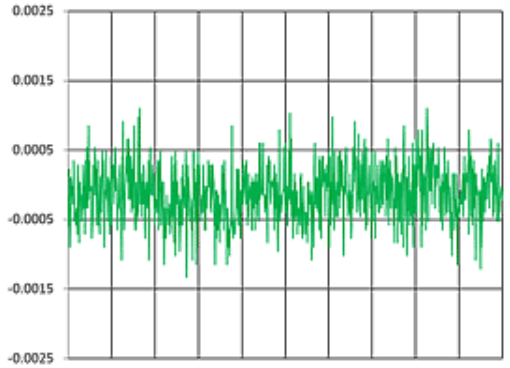


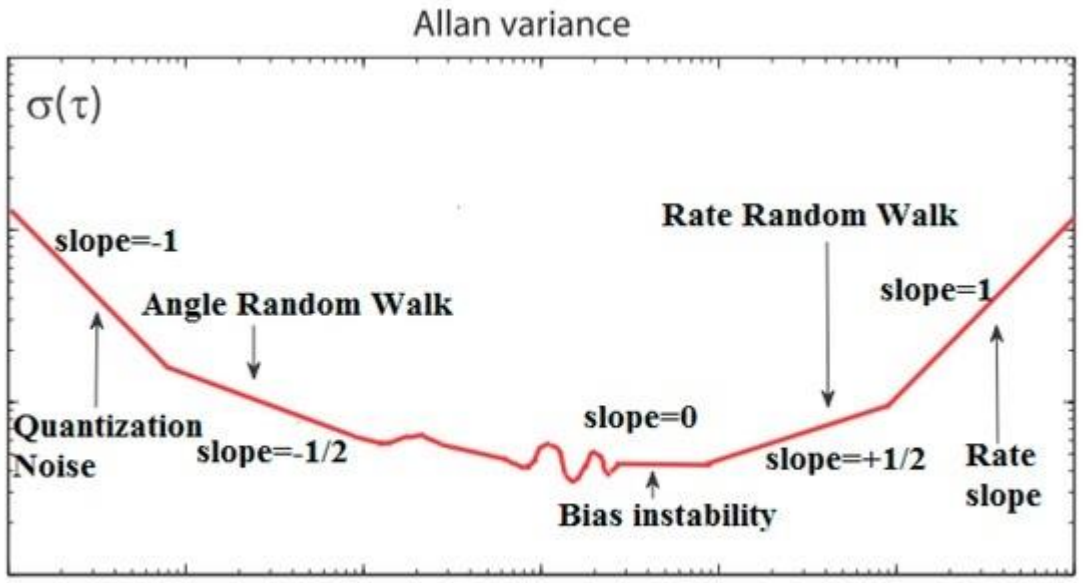
MEMS

3. Implementation problems

- The **noise** in the MEMS IMU:
  - Systematic (calibrable):
    - Ortogonality, scaling, thermal
  - Stocastic:
    - Additive noise in Acc & Gyr readings
    - Bias (systematic, unstability, turn on)







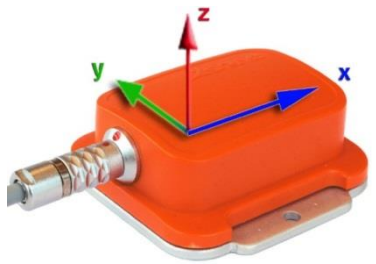
3. Implementation problems

- Ruido en señal en IMU según categoría (“sensor grade”)

	Consumer	**Industrial**	**Tactical**	**Navigation**
— Giróscopos —				
Bias repeatability (turn-on)	1 000 – 10 000 °/h	50 – 500 °/h	1 – 10 °/h	< 0.05 °/h
In-run bias stability (BRW)	10 – 100 °/h	1 – 10 °/h	0.1 – 1 °/h	< 0.01 °/h
ARW (Angular Random Walk)	0.5 – 2 °/√h	0.1 – 0.5 °/√h	0.03 – 0.1 °/√h	< 0.005 °/√h
— Acelerómetros —				
Bias repeatability (turn-on)	20 – 50 mg	1 – 5 mg	0.1 – 0.5 mg	< 0.05 mg
In-run bias stability (BRW)	0.05 – 1 mg	0.01 – 0.1 mg	0.005 – 0.05 mg	< 0.005 mg
VRW (Velocity Random Walk)	0.1 – 0.5 m/s/√h	0.02 – 0.1 m/s/√h	0.01 – 0.03 m/s/√h	< 0.005 m/s/√h
— Otros —				
Precio (MEMS)	1 – 20 €	100 – 2 000 €	2 000 – 20 000 €	> 20 000 €

3. Implementation problems

- Noise generates:
  - Integration errors (Velo, Pos)
  - Errors in Attitude (degress)



Xsens Mti:

Acc

Acceleration Range (g)	VRW ($\mu g/\sqrt{Hz}$)	BRW (mg)
±5/18	122	0.05

Gyr

Angular Rate Range (°/s)	ARW (°/$\sqrt{hr}$)	BRW (°/hr)
±300	3	50

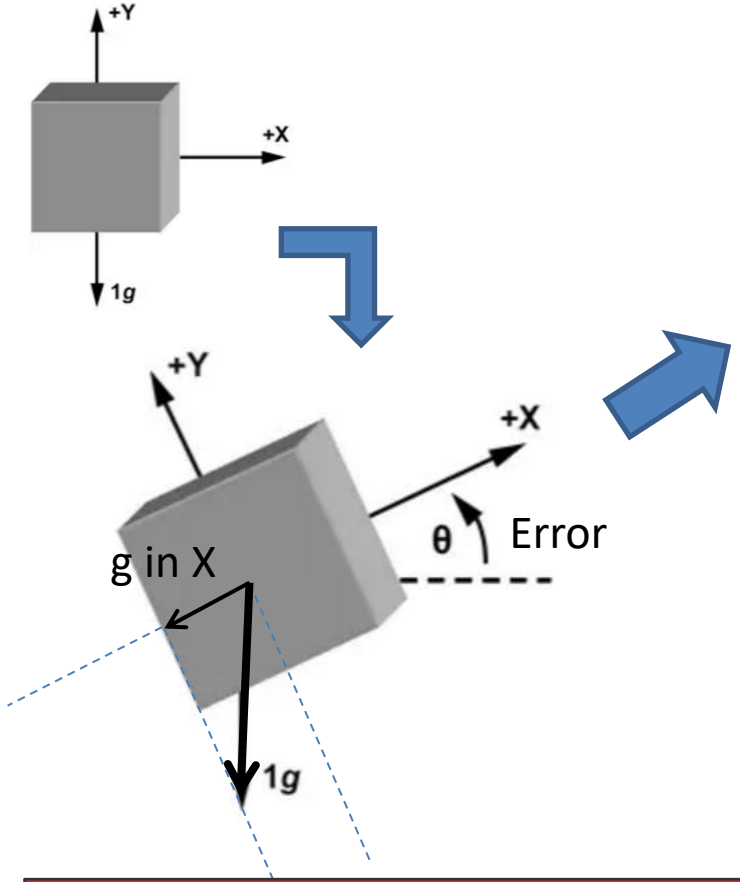
Static accuracy (roll/pitch)	<0.5 deg
Static accuracy (heading)[1]	<1 deg
Dynamic accuracy	2 deg RMS
Angular resolution	0.05 deg

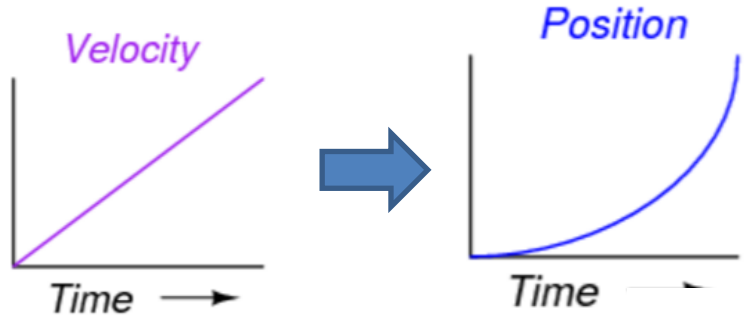
[1] in homogeneous magnetic environment

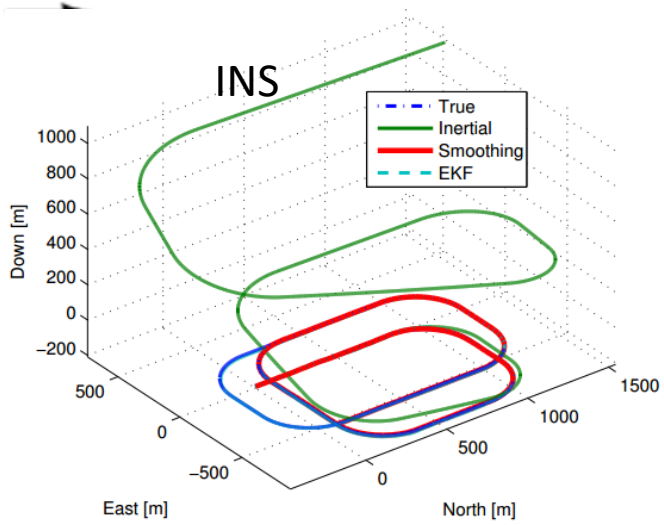
1 degree of inclination error → dramatic integration effects

3. Implementation problems

- Attitude error => gravity to leak as acceleration







1 degree error → metros de error en pocos segundos. Imposible usar el INS puro.

Outline

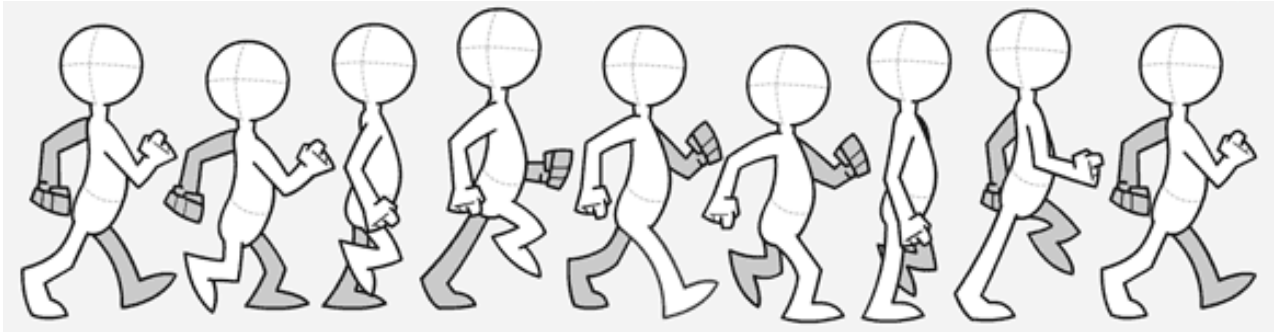
- Some theory:
  1. What is PDR?
  2. Inertial Navigation (INS)
  3. Implementation problems
  4. PDR algoritmic solutions
- Practice (Matlab):
  1. Introduction
  2. PDR with pre-recorded logfiles
  3. PDR with your own phone

4. PDR algorithmic solutions

- Ways to improve INS for Pedestrians:
  - 1) Split the INS problem into pieces: Step by step

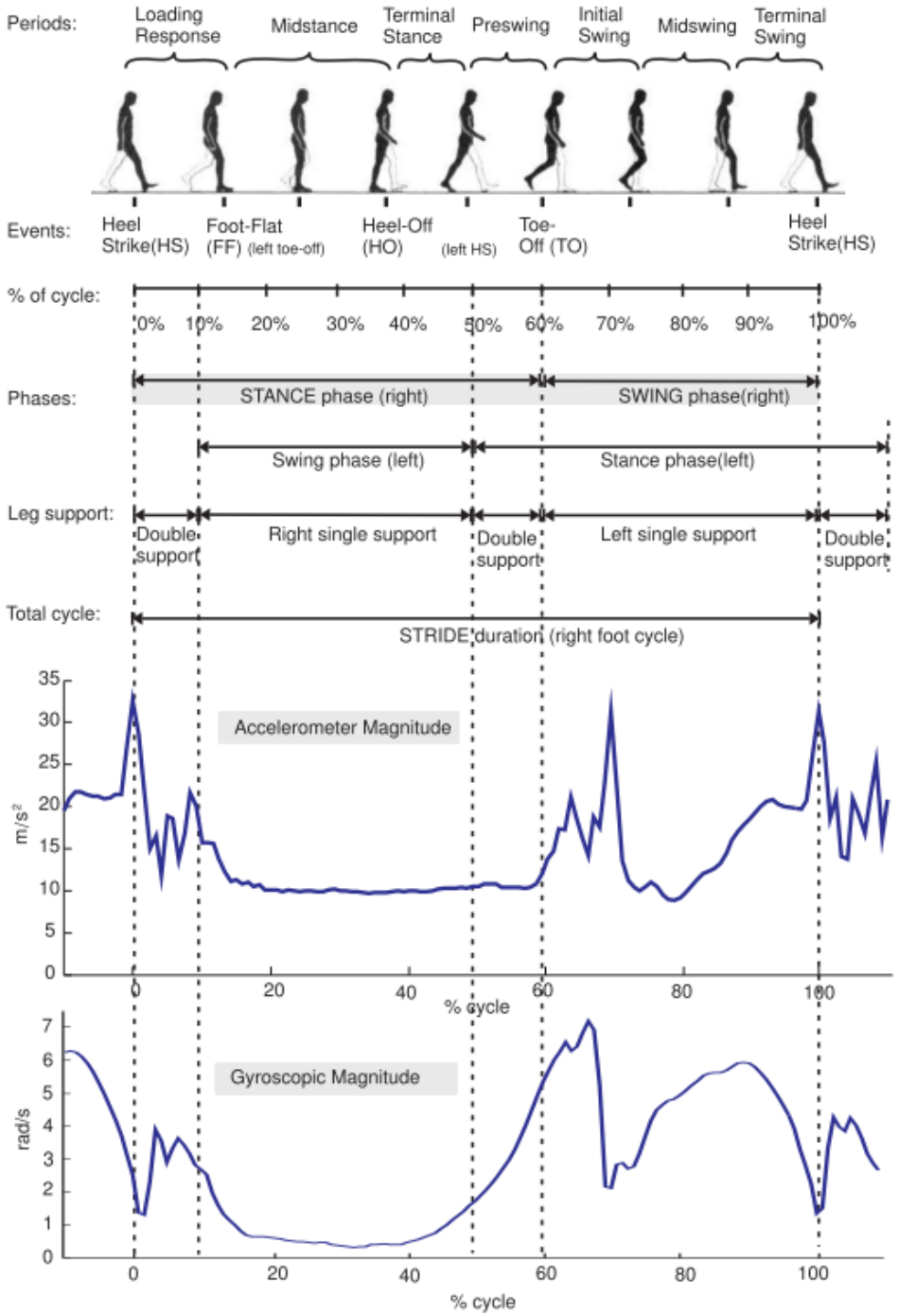


  - 2) Apply human motion models:
    - Step frequency: 1Hz (useful for signal filtering and stance length estimation),
    - Foot on floor: Acceleration peak (if IMU at foot => Velocity=0 at stance; ZUPT)



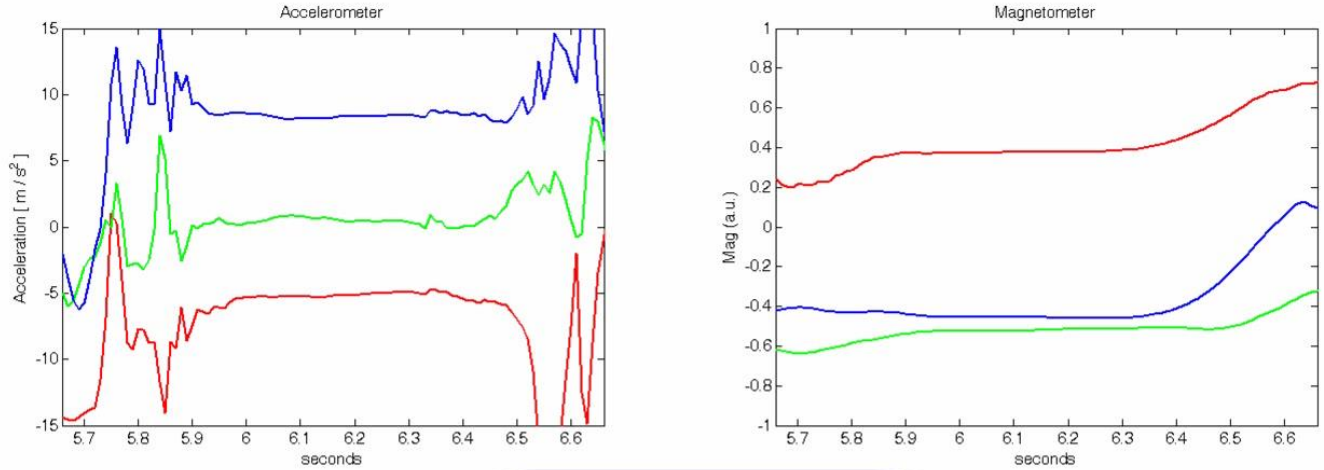
4. PDR algorithmic solutions

- Walking Phases:
  - 2 phases (stance and swing) and 7 different sub-phases
  - The stance (60%) and swing (40%) phases
  - For a foot-mounted IMU: Magnitude of Acc & Gyr are stable during Midstance (central stance phase)



4. PDR algorithmic solutions

- Foot-attached IMU signals at stance:

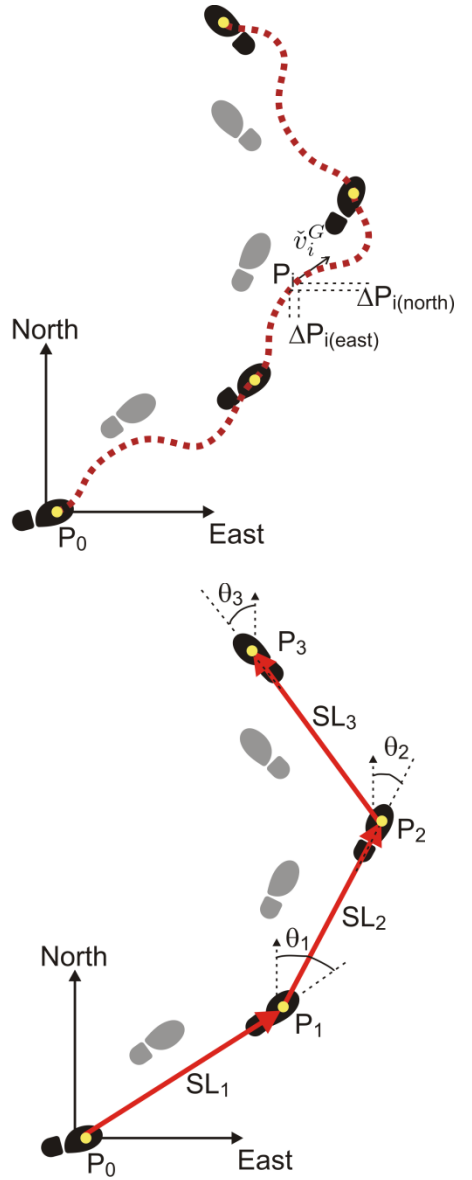


IMU signals at foot stance



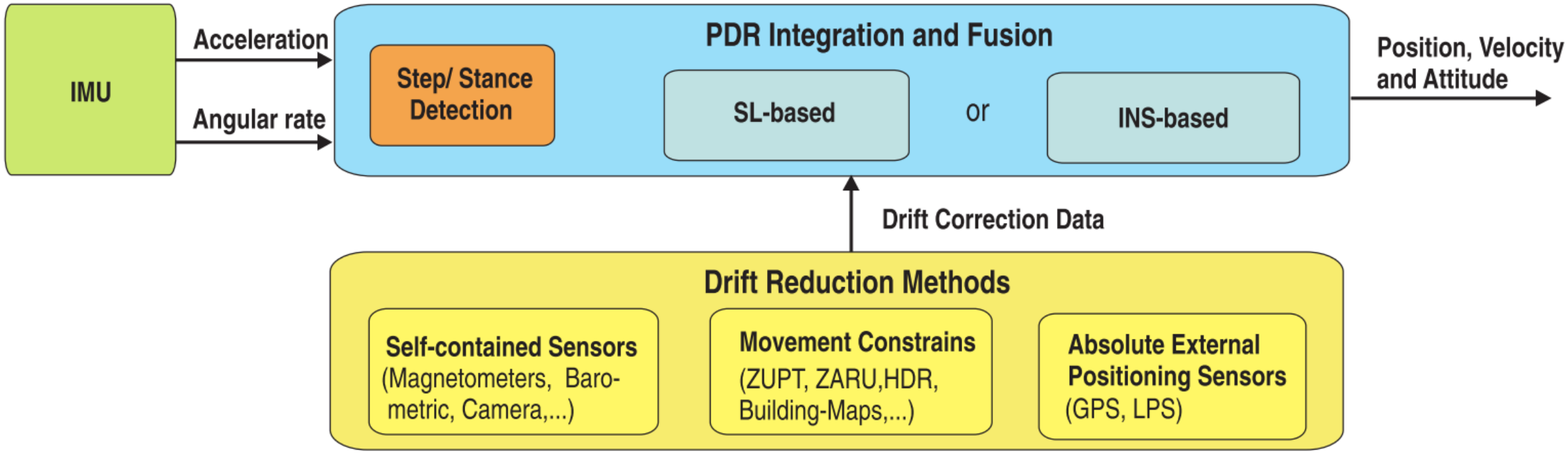
4. PDR algorithmic solutions

- There are **2 main types of PDR** algorithms:
  - "**INS-ZUPT**": Integrates accelerations (INS) and correct velocities with zero velocity updates (ZUPT) at stance. IMU must be on foot.
  - "**SL+θ**": Accumulates Stride Length (SL) estimations, along the Orientation angle (θ) at foot stance. General purpose (IMU anywhere)



4. PDR algorithmic solutions

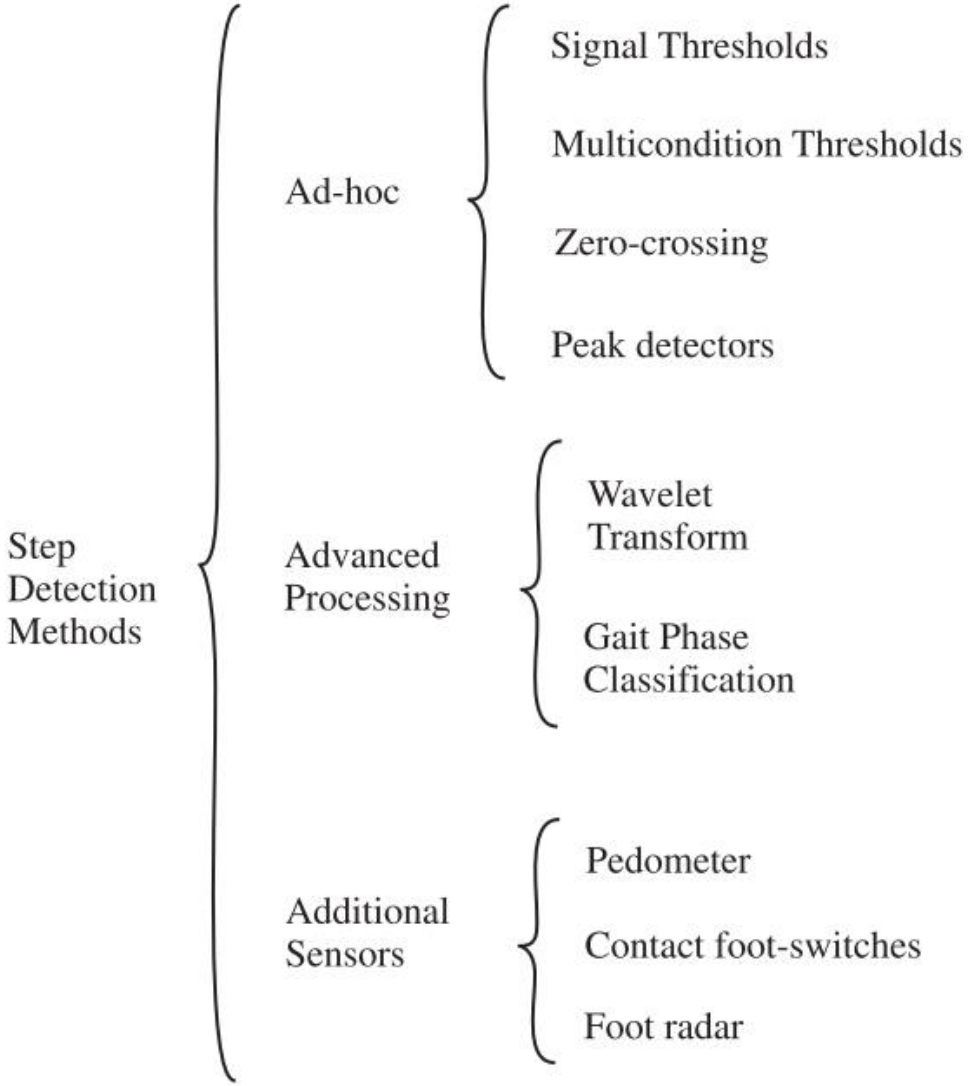
- General block diagram for an IMU-based Pedestrian Navigation System:



4. PDR algorithmic solutions

- Step detection





4. PDR algorithmic solutions

- **Step detection** using Accelerometers:

1 Magnitude of acceleration

$$a_i = \sqrt{a_{x_i}^2 + a_{y_i}^2 + a_{z_i}^2}$$

2 Local acceleration variance

$$\sigma_{a_i}^2 = \frac{1}{2w+1} \sum_{j=i-w}^{i+w} (a_j - \overline{a_j})^2$$

$$\overline{a_j} = \frac{1}{2w+1} \sum_{q=i-w}^{i+w} a_q$$

3 Thresholding

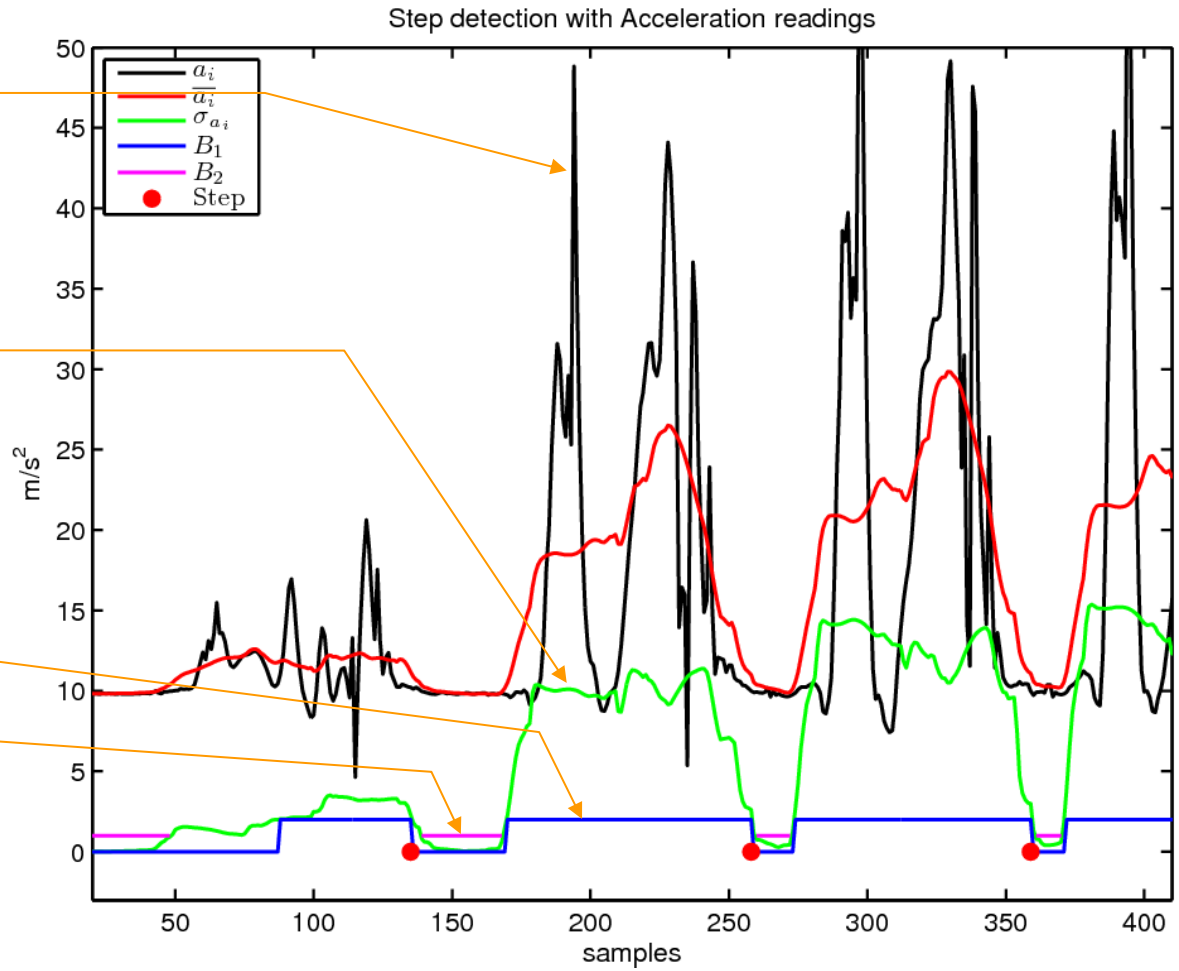
$$B_{1i} = \begin{cases} \text{T1} & \sigma_{a_i} > \text{T1} \\ 0 & \text{otherwise} \end{cases}$$

$$B_{2i} = \begin{cases} \text{T2} & \sigma_{a_i} < \text{T2} \\ 0 & \text{otherwise} \end{cases}$$

4 Conditions step detection (i sample)

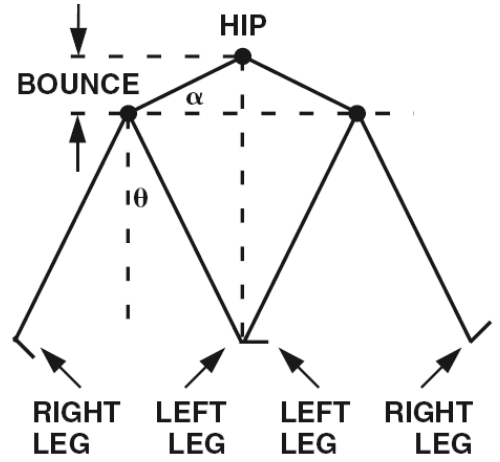
$$B_{1i-1} < B_{1i}$$

$$\max(B_{2i:i+w}) = \text{T2}$$



4. PDR algorithmic solutions

- **Step Length** (SL) Weinberg Algorith.
  - Assumes SL is prop. to BOUNCE (vertical movement of hip)
  - Bounce estimated from largest Acc.



1 Magnitude of acceleration

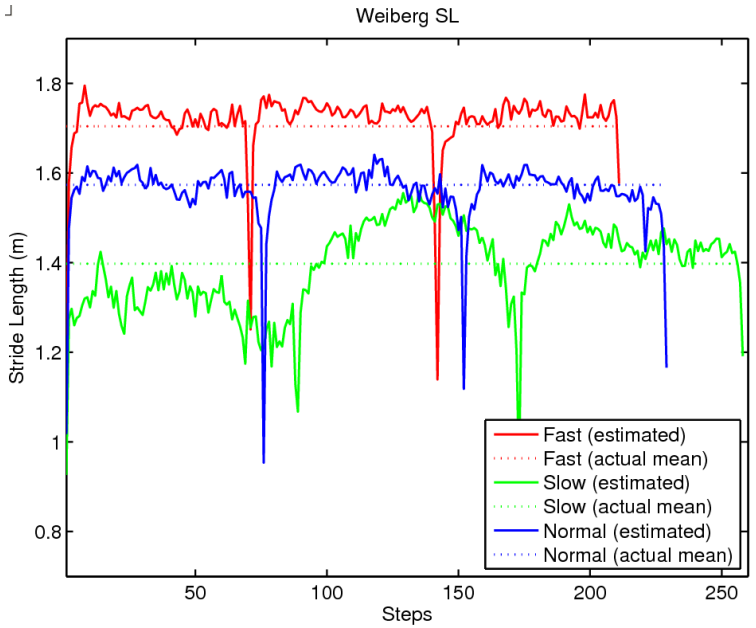
$$a_i = \sqrt{a_{x_i}^2 + a_{y_i}^2 + a_{z_i}^2}$$

2 Low-Pass Filter

$$\tilde{a}_i = LP(a_i)$$

3 Weiberg hip bounce model

$$\mathrm{SL}_{\mathrm{Weiberg}_k} = K \cdot \{\max_{j=[i_{(k)} \pm w]}^{\tilde{a}_j} - \min_{j=[i_{(k)} \pm w]}^{\tilde{a}_j}\}^{1/4}$$



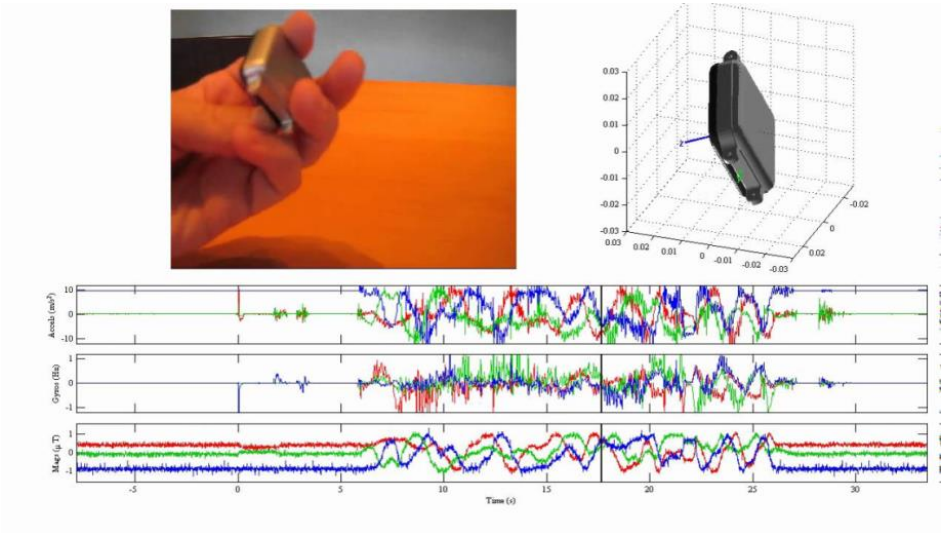
4. PDR algorithmic solutions

- **Attitude** estimation (AHRS):
  - Orientation from gyroscopes (relative) :

$$\omega^s = (\omega_x^s, \omega_y^s, \omega_z^s)$$ Gyro signals

$$C(t) = C(0) \cdot exp\left(\int_0^t \Omega(\tau) d\tau\right)$$

C (rotation matrix) computed integrating the skew simetric matrix from an initial orientation



  - Orientation from accelerometers and magnetometer (absolute references):

$$m^s = (m_x^s, m_y^s, m_z^s)$$ magnetometer signals

$$a^s = (a_x^s, a_y^s, a_z^s)$$ acceleration signals

$$\phi = \tan\left(\frac{a_y^s}{a_z^s}\right)^{-1}$$ Pitch

$$\theta = \tan\left(\frac{-a_x^s}{\sqrt{(a_y^s)^2 + (a_z^s)^2}}\right)^{-1}$$ Roll

$$\psi = \tan\left(\frac{-m_x^h}{m_y^h}\right)^{-1} \pm D$$ Yaw

  - Some Integrated AHRS algorithms (optimal weighting):
    - Madwick AHRS algorithm (Gradient descent optimization gyro vs. accel/magne).
    - Mahony AHRS algorithm (complementary filter)

$$q_{fused}(k) = \gamma \cdot q_g(k) + (1 - \gamma) \cdot q_{a/m}(k)$$

4. PDR algorithmic solutions

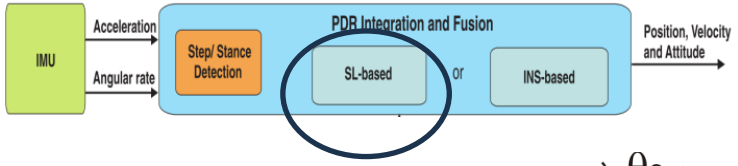
- **PDR Algorithm: "SL+θ"**:
  - The SL can be computed, e.g.:

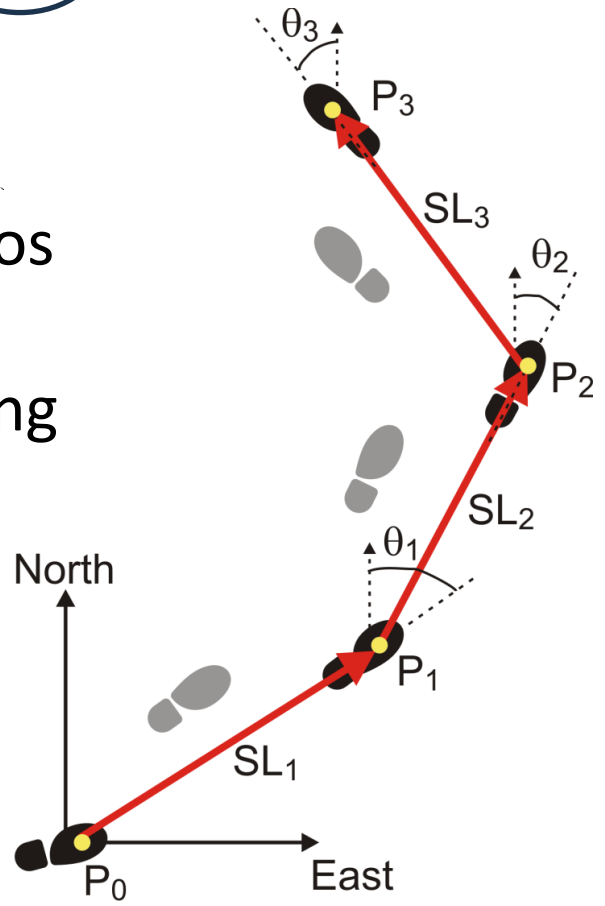
$$\mathrm{SL}_{\mathrm{Weiberg}_k} = K \cdot \{\max_{j=[i_{(k)} \pm w]}^{\tilde{a}_j} - \min_{j=[i_{(k)} \pm w]}^{\tilde{a}_j}\}^{1/4}$$

  - The θ angle at foot stance, computed using gyros (& opt. magnetometers).
  - Accumulates Stride Length (SL) estimations along the Orientation (θ)

$$\begin{cases} P_k(\mathrm{north}) = P_{k-1}(\mathrm{north}) + \mathrm{SL}_k \cdot cos(\theta_{\mathrm{stance}_k}) \\ P_k(\mathrm{east}) = P_{k-1}(\mathrm{east}) + \mathrm{SL}_k \cdot sin(\theta_{\mathrm{stance}_k}), \end{cases}$$

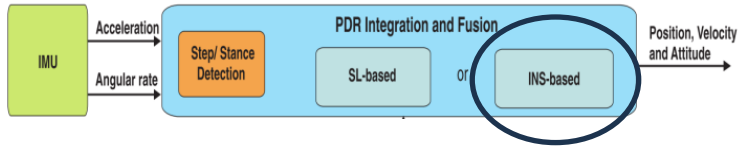
  - "K" number of steps
  - Method valid con "normal" walking (only forward walk).

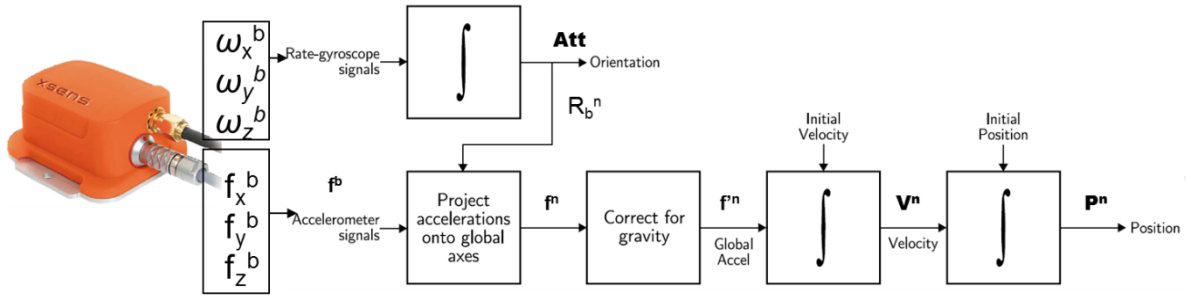




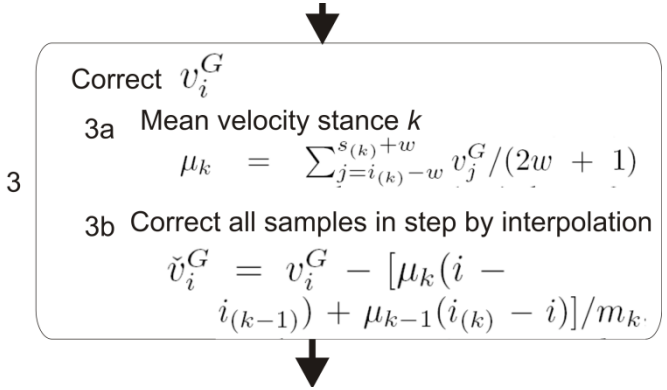
4. PDR algorithmic solutions

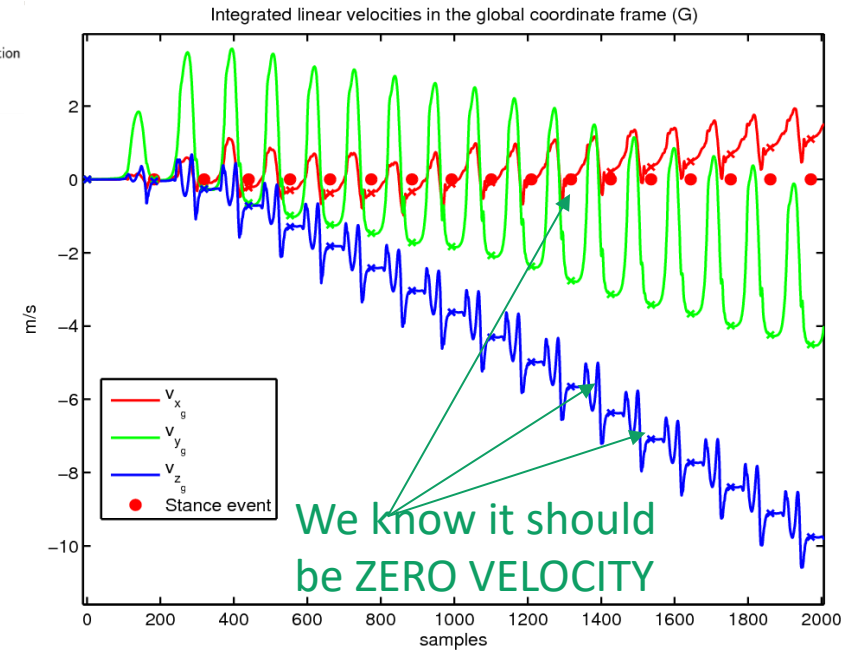
- **PDR Algorithm: "INS-ZUPT" :**
  - Integrate accelerations to obtain velocity (INS).





  - Correct velocity at foot stance (ZUPT).





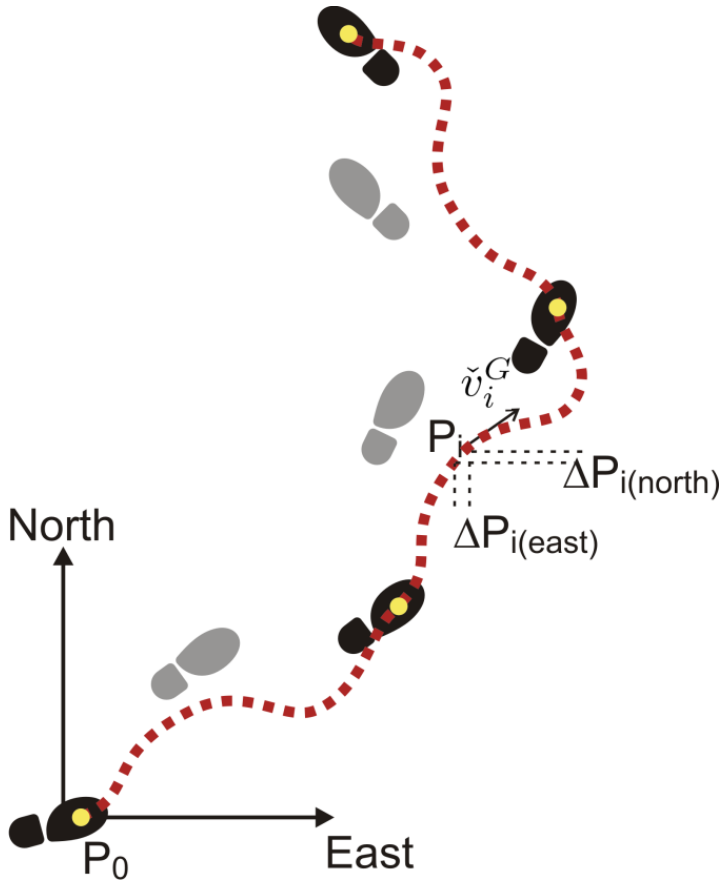
4. PDR algorithmic solutions

- **PDR Algorithm: "INS-ZUPT"** (cont):
  - Accumulates position increments:

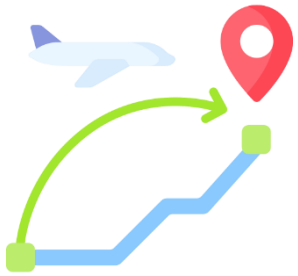
$$\Delta P_i = \check{v}_i^G \cdot \Delta T$$

$$\begin{cases} P_i(\text{north}) = P_{i-1}(\text{north}) + \Delta P_i(\text{north}) \\ P_i(\text{east}) = P_{i-1}(\text{east}) + \Delta P_i(\text{east}). \end{cases}$$

  - Method valid for "any" type of walk (forward/lateral/backwards walk, running, crawling, etc).
  - …but IMU must be on foot



Resumen Estrategias PDR viables



INS
Sistema de navegación inercial

Plataforma estable
IMU sobre cardán mecánico

Strapdown
IMU solidario al cuerpo

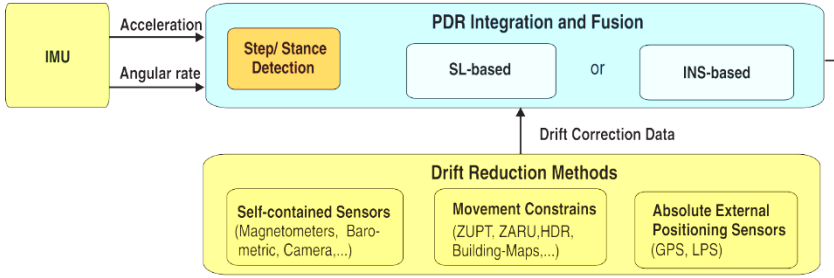
Sensor MEMS (pequeño) ->RUIDO



PDR
Pedestrian dead reckoning

SL + θ
Paso + rumbo
IMU en torso / cadera

INS-ZUPT
Strapdown + ZUPT
IMU en el pie

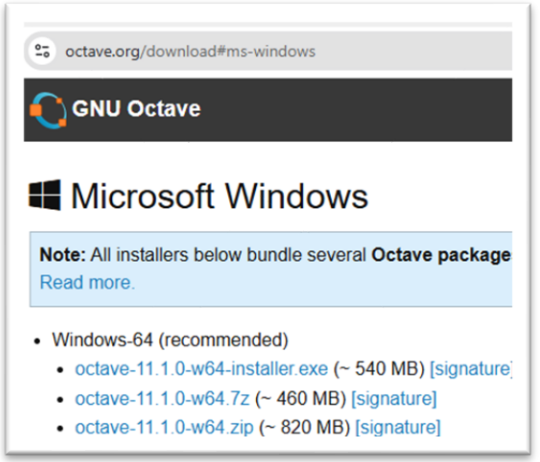


Outline

- Some theory:
  1. What is PDR?
  2. Inertial Navigation (INS)
  3. Implementation problems
  4. PDR algorithmic solutions
- Practice (Matlab):
  1. Introduction
  2. PDR with pre-recorded logfiles
  3. PDR with your own phone

Practice: Introduction

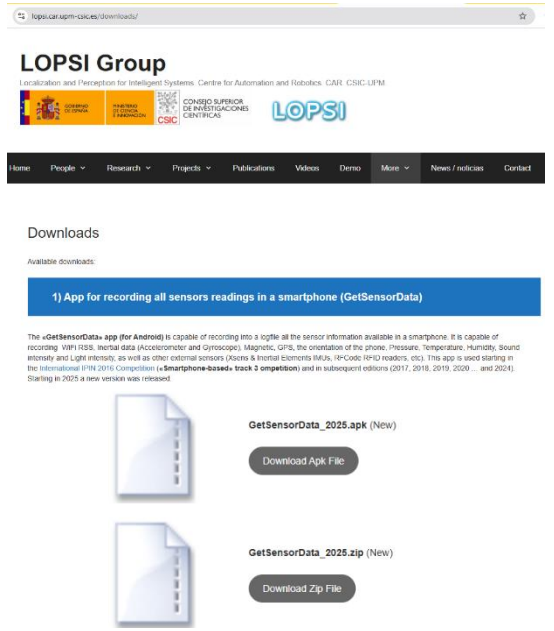
- **Tools needed** for practice:
  - **Laptop computer** (any O.S. valid: Mac / Linux / Windows).
  - **Octave** software (or **Matlab**) should be installed in your computer. Octave is freely available at: https://octave.org/download





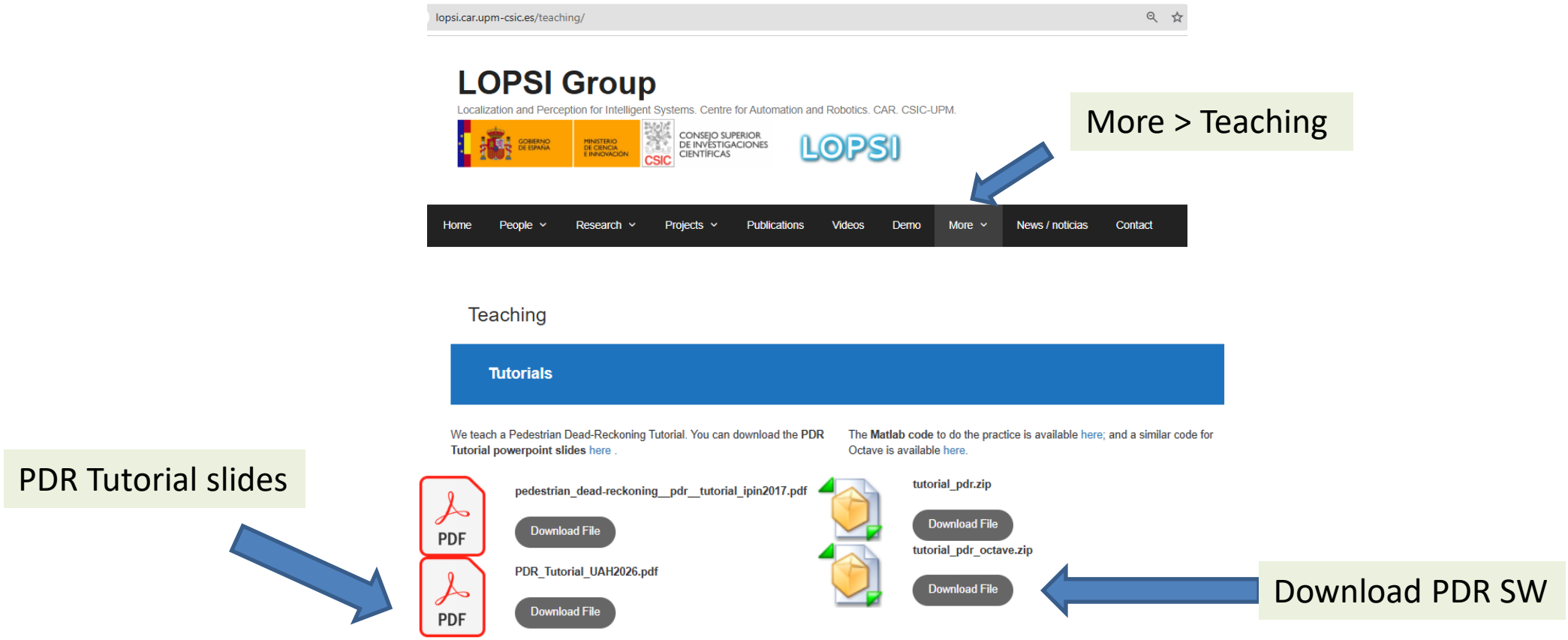
  - A **smartphone** with **Android** operating system is needed for one of the practices.
    The "**GetSensorData_2025.apk**" app should be installed.
    App available at “Lopsi > More>Downloads”:
    https://lopsi.car.upm-csic.es/downloads/

    App stores Log_files in your phone => transferred to your computer via USB cable, Bluetooth, e-mail,…



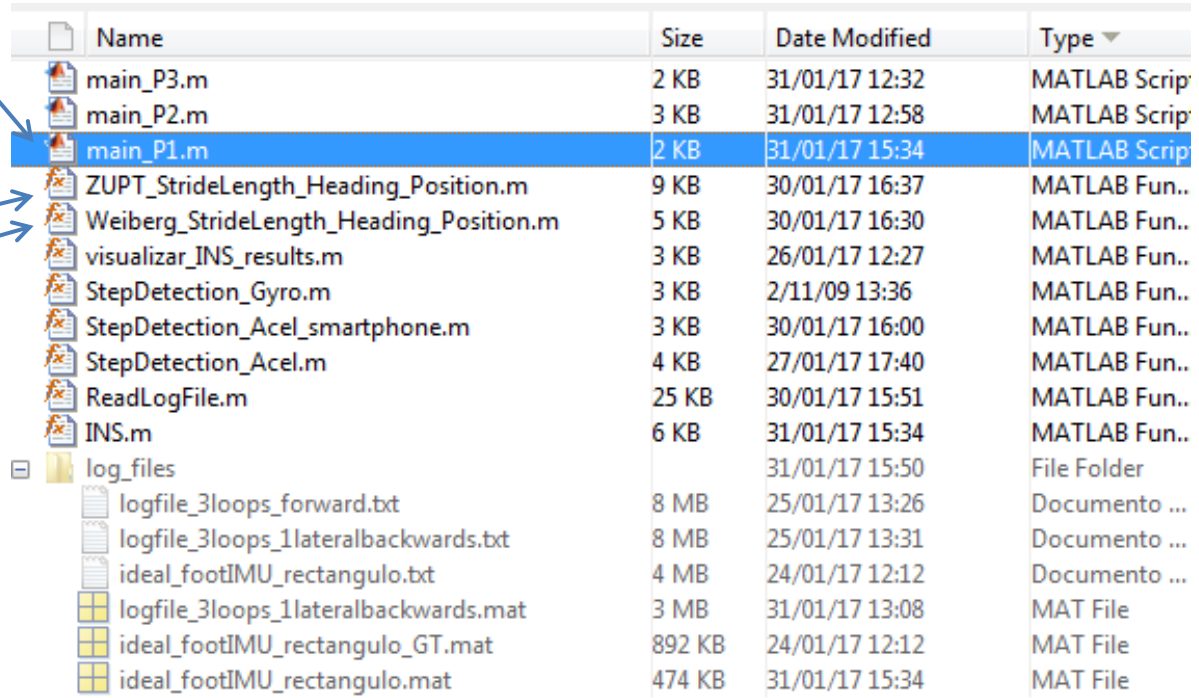
Practice: Introduction

- Download PDR scripts (Octave/Matlab) :



Practice: Introduction

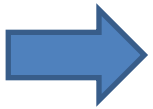
- **Matlab algorithms and logfiles**:
  - Three main files, for 3 practices
  - PDR algorithms:
    - INS for position & Attitude
    - Step detection
    - Step length estimation
    - Two PDR types:
      - INS-ZUPT,
      - SL+theta
  - Tools:
    - Visualization and
    - Log_file interpretation
  - Log_files:
    - #1 ideal (4 loops),
    - #2 rectangular (3 loops) with 2 different IMUs
    - You will create your own log_files



Name	Size	Date Modified	Type
main_P3.m	2 KB	31/01/17 12:32	MATLAB Script
main_P2.m	3 KB	31/01/17 12:58	MATLAB Script
main_P1.m	2 KB	31/01/17 15:34	MATLAB Script
ZUPT_StrideLength_Heading_Position.m	9 KB	30/01/17 16:37	MATLAB Fun..
Weiberg_StrideLength_Heading_Position.m	5 KB	30/01/17 16:30	MATLAB Fun..
visualizar_INS_results.m	3 KB	26/01/17 12:27	MATLAB Fun..
StepDetection_Gyro.m	3 KB	2/11/09 13:36	MATLAB Fun..
StepDetection_Acel_smartphone.m	3 KB	30/01/17 16:00	MATLAB Fun..
StepDetection_Acel.m	4 KB	27/01/17 17:40	MATLAB Fun..
ReadLogFile.m	25 KB	30/01/17 15:51	MATLAB Fun..
INS.m	6 KB	31/01/17 15:34	MATLAB Fun..
log_files		31/01/17 15:50	File Folder
logfile_3loops_forward.txt	8 MB	25/01/17 13:26	Documento ...
logfile_3loops_1lateralbackwards.txt	8 MB	25/01/17 13:31	Documento ...
ideal_footIMU_rectangulo.txt	4 MB	24/01/17 12:12	Documento ...
logfile_3loops_1lateralbackwards.mat	3 MB	31/01/17 13:08	MAT File
ideal_footIMU_rectangulo_GT.mat	892 KB	24/01/17 12:12	MAT File
ideal_footIMU_rectangulo.mat	474 KB	31/01/17 15:34	MAT File

Outline

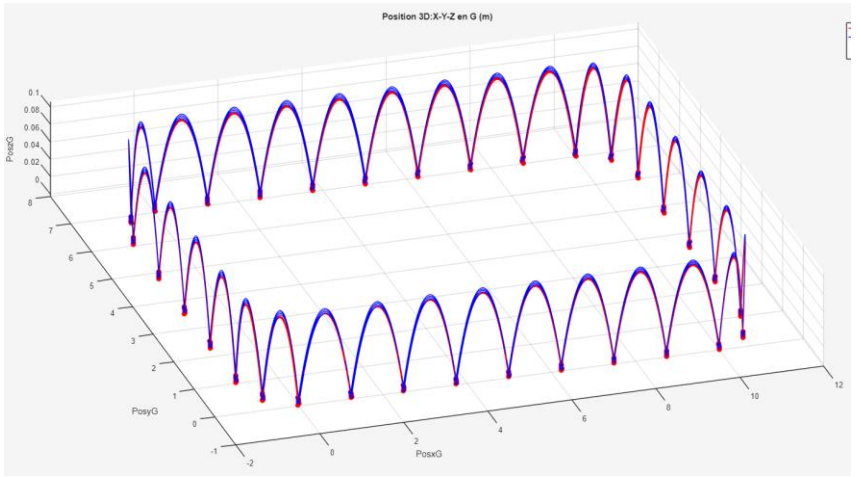
- Some theory:
  1. What is PDR?
  2. Inertial Navigation (INS)
  3. Implementation problems
  4. PDR algorithmic solutions
- Practice (Matlab):
  1. Introduction
  2. PDR with pre-recorded logfiles
  3. PDR with your own phone



PDR with pre-recorded logfiles

- PRACTICE 1: **Effects of noise on INS**
  - Using a synthetic logfile:
    - Simulated IMU:
      - acceleration (Acc), turn rates (Gyr) and magnetic field (Mag)
    - Ground-truth included (4 loops):
      - position (Pos), velocity (Vel) and orientation (Euler & DCM)
    - Units are in:
      - meters, seconds and radians.
    - Sampling frequency: 100 Hz
    - IMU rotated on foot as in picture.
    - All trajectories starts at point (0,0,0)
    - Gravity is 9.8 m/s2
  - Study INS results for noiseless and noise cases

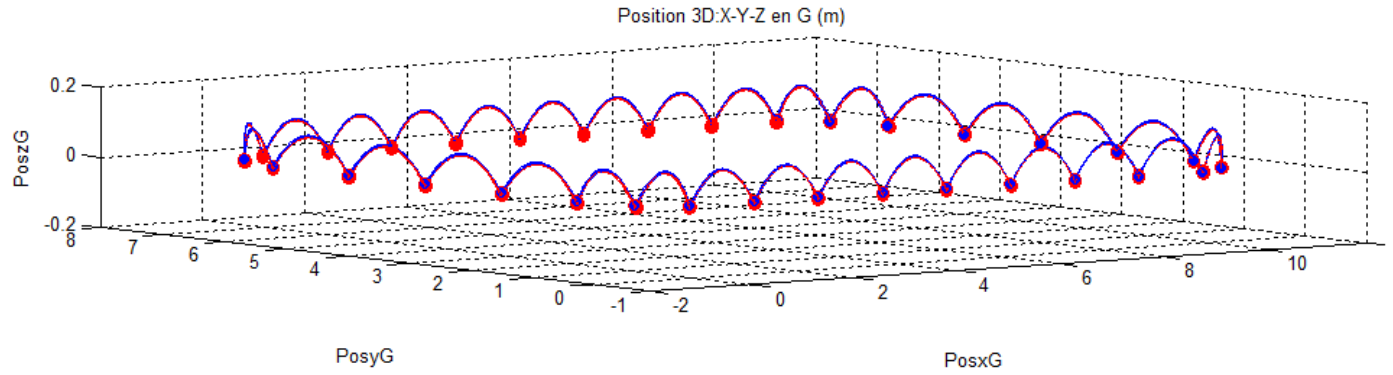




PDR with pre-recorded logfiles

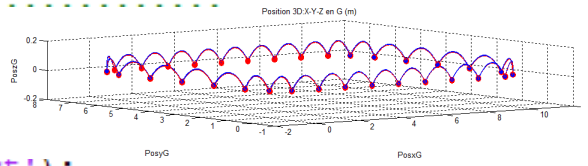
- PRACTICE 1: **Effects of noise on INS**

```
% ----------------------------------------
%        PDR Tutorial: PRACTICE 1 (main_P1.m)
%       Lopsi Group. CAR-CSIC/UPM
%                   2017
%----------------------------------------
clc; clear all; close all; disp('PRACTICE 1: Effects of noise on INS');
%                PRACTICE 1: Effects of noise on INS
% 1) Load ideal noise-free data (ideal_footIMU_rectangulo.txt) and inspect
% 2) Apply INS algorithm (INS.m) to ideal IMU signals and check correct integration
% 3) Add noise (rand or bias) to IMU signals and inspect drift in velocity and position generated
```



PDR with pre-recorded logfiles

- PRACTICE 1: **Effects of noise on INS**




```
%.....................................................................................
% 1) Load ideal noise-free data (ideal_footIMU_rectangulo.txt) and inspect

% Read log_file
disp('1) Load ideal noise-free data (ideal_footIMU_rectangulo.txt) and inspect');
disp('Reading Logfile...');
[~,~,Acc,Gyr]=ReadLogFile('.\log_files\ideal_footIMU_rectangulo.txt','Xsens');  % load IMU simulated data: A
load('.\log_files\ideal_footIMU_rectangulo_GT.mat'); % load Ground_truth data: 'Pos_G','Vel_G','Att_G','Rot_
disp('Logfile Read...');
disp('-> TO DO: Inspect IMU signals  (press enter to continue)');
pause;

%.....................................................................................
% 2) Apply INS algorithm (INS.m) to ideal IMU signals and check correct integration
disp(sprintf('\n2) Apply INS algorithm (INS.m) to ideal IMU signals and check correct integration'));
% Apply INS to obtain Pos,Vel y Att:
disp('Applying INS...');
[Pos_G_rec,Vel_G_rec,Att_G_rec]=INS(Acc,Gyr);
disp('INS ended. Showing results...');
visualizar_INS_results(Pos_G_rec,Vel_G_rec,Att_G_rec,Pos_G,Vel_G,Att_G,Rot_GS,Stance,StepDectSample,10);
disp('-> TO DO: Check correct integration (press enter to continue)');
pause;
```

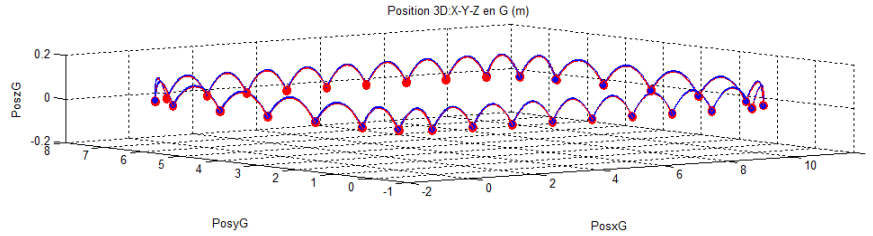
PDR with pre-recorded logfiles

- PRACTICE 1: **Effects of noise on INS**

```
%.....................................................................................
% 3) Add noise (rand or bias) to IMU signals and inspect drift
disp(sprintf('\n3) Add noise (rand or bias) to IMU signals and inspect drift'));
Amplitud=0.0;  % m/s^2
Bias=0.0;       % m/s^2
samples=length(Acc);
Acc(:,1:3)=Acc(:,1:3)+Amplitud*randn(samples,3);
Acc(:,3)=Acc(:,3)+Bias*ones(samples,1);

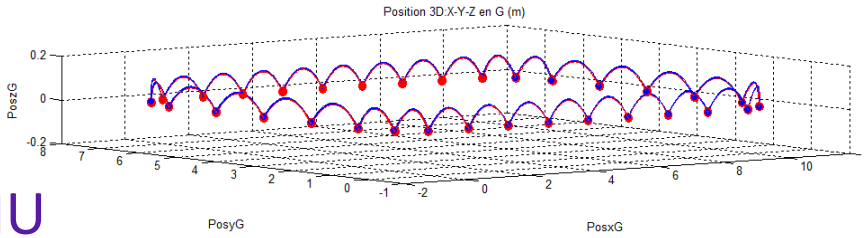
Amplitud=0.0;  % rad/s
Bias=0.001;       % rad/s
samples=length(Gyr);
Gyr(:,1:3)=Gyr(:,1:3)+Amplitud*randn(samples,3);
Gyr(:,3)=Gyr(:,3)+Bias*ones(samples,1);

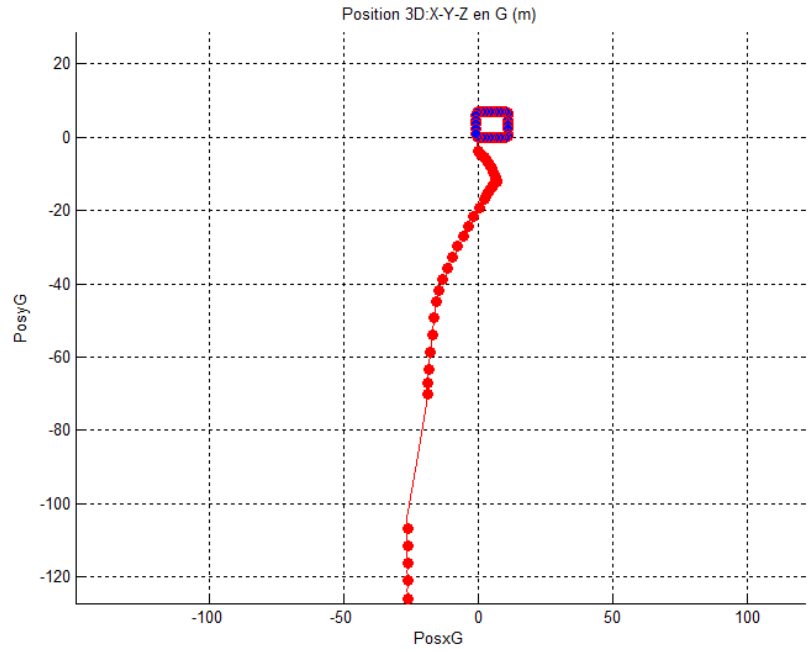
% Apply INS to obtain Pos,Vel y Att:
disp('Applying INS with noise...');
[Pos_G_rec,Vel_G_rec,Att_G_rec]=INS(Acc,Gyr);
disp('INS with noise ended. Showing results...');
visualizar_INS_results(Pos_G_rec,Vel_G_rec,Att_G_rec,Pos_G,Vel_G,Att_G,Rot_GS,Stance,StepDectSample,20);
disp('-> TO DO: Check correct integration (press enter to continue)');
```



PDR with pre-recorded logfiles

- PRACTICE 1: **Effects of noise on INS**
  - **Conclusions:**
    - INS works perfectly on ideal or noiseless IMU data
    - INS does not work if sensor data has:
      - Bias, Noise,
      - Low sampling frequency
      - Quantization, saturated
      - Axis misalignements

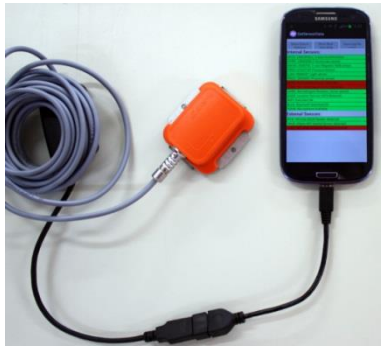




PDR with pre-recorded logfiles

- PRACTICE 2: **PDR with foot-mounted real Xsens-MEMS IMU**
  - One square 3 times (76 steps: 22+23+31, forward walk, but last 3 sides lateral/backwards walk)







PDR with pre-recorded logfiles

- PRACTICE 2: **PDR with foot-mounted real Xsens-MEMS IMU**

```
% ----------------------------------------
%          PDR Tutorial: PRACTICE 2 (main_P2.m)
%       Lopsi Group. CAR-CSIC/UPM
%                   2017
%----------------------------------------

clc; clear all; close all; disp('PRACTICE 2: PDR with FOOT-MOUNTED real XSENS-MEMS IMU');
%
%         PRACTICE 2: PDR with FOOT-MOUNTED real XSENS-MEMS IMU
% 1) Load real data with foot-mounted IMU (logfile_3loops_1lateralbackwards.txt)
%       -One square 3 times (76: 22+23+31 steps, forward walk, but last 3 sides lateral/backwards walk)
% 2) Apply INS PDR algorithm and analyse drifts in position
% 3) Apply INS-ZUPT and analyse processing:
%       -Detection of steps and stance
%       -Correction of Velocities to zero (ZUPT)
%       -Walking direction
```

PDR with pre-recorded logfiles

- PRACTICE 2: **PDR with foot-mounted real Xsens-MEMS IMU**

```
%.....................................................................................
% 1) Load real data with foot-mounted IMU (logfile_3loops_1lateralbackwards.txt)
%       -One square 3 times (76: 22+23+31 steps, forward walk, but last 3 sides lateral/backwards walk)

% Read log_file
disp('1) Load real data with foot-mounted test and inspect');
disp('Reading Logfile...');
% load IMU read data: Acc,Gyr de Xsens (3 loops)
[~,~,Acc,Gyr]=ReadLogFile('.\log_files\logfile_3loops_1lateralbackwards.txt','Xsens',1);  % 76 steps (2 loops
%[~,~,Acc,Gyr]=ReadLogFile('.\log_files\logfile_3loops_forward.txt','Xsens',1);  % 66 steps (3 loops)
disp('Logfile Read...');
disp('-> TO DO: Inspect IMU signals and bias (press enter to continue)');
pause;
```

PDR with pre-recorded logfiles

- PRACTICE 2: **PDR with foot-mounted real Xsens-MEMS IMU**

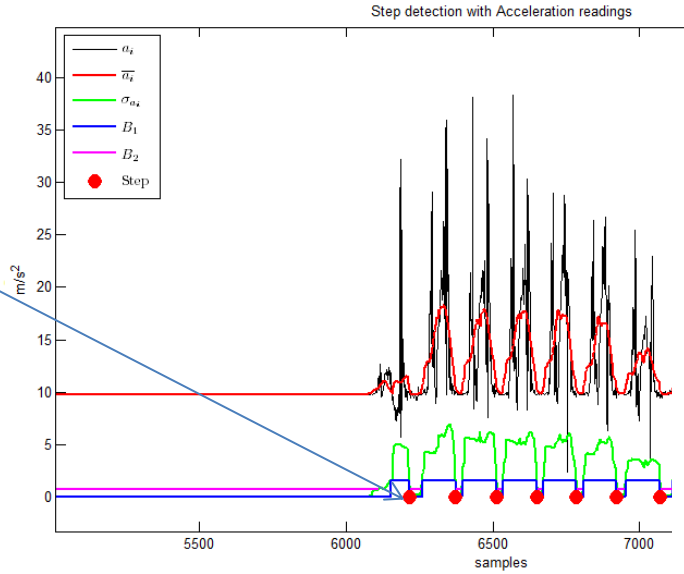
```
%.....................................................................................
% 2) Apply INS PDR algorithm and analyse drifts in position (remove bias)
disp(sprintf('\n2) Apply INS PDR algorithm and analyse drifts in position (remove bias)'));
% Remove bias Gyro
samples=5000;  % asumo 50 segundos parado (y fs=100 Hz)
bias_Gyr=[mean(Gyr(1:samples,1)), mean(Gyr(1:samples,2)), mean(Gyr(1:samples,3))];
Gyr_unbiased=Gyr;  %[nx4]
Gyr_unbiased(:,1:3)=[Gyr(:,1)-bias_Gyr(1), Gyr(:,2)-bias_Gyr(2), Gyr(:,3)-bias_Gyr(3)];

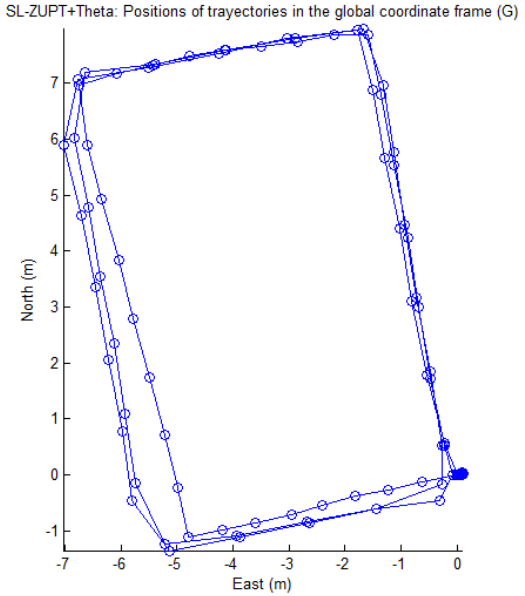
% Apply INS to obtain Pos,Vel y Att:
disp('Applying INS...');
[Pos_G_rec,Vel_G_rec,Att_G_rec]=INS(Acc,Gyr);
%[Pos_G_rec,Vel_G_rec,Att_G_rec]=INS(Acc,Gyr_unbiased);
disp('INS ended. Showing results...');
idx_fig=10;
visualizar_INS_results(Pos_G_rec,Vel_G_rec,Att_G_rec,idx_fig);
disp('-> TO DO: Check uncorrect INS integration/ remove bias (press enter to continue)');
pause;
```

PDR with pre-recorded logfiles

- PRACTICE 2: **PDR with foot-mounted real Xsens-MEMS IMU**

76 steps should be detected






```
%.....................................................................
% 3) Apply INS-ZUPT and analyse processing:
%        -Detection of steps and stance
%        -Correction of Velocities to zero (ZUPT)
%        -Walking direction

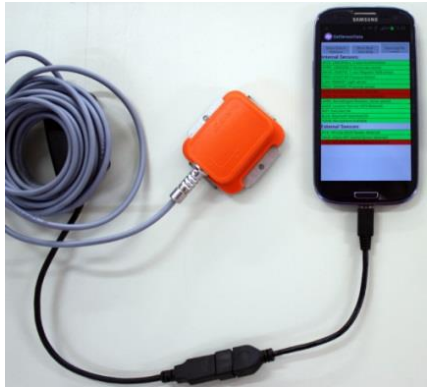
disp(sprintf('\n3) Apply INS-ZUPT and analyse processing'));
disp('Applying INS-ZUPT...');
%-----Step detection------
idx_fig=20;
[Num_steps,Step_events,StancePhase,idx_fig]=StepDetection_Acel(Acc,1,idx_fig);
%-----INS-ZUPT-----------
[StrideLengths, Thetas, Positions,idx_fig]=ZUPT_StrideLength_Heading_Position(Acc,Gyr,Step_events,StancePhase,1,idx_fig);
%[StrideLengths, Thetas, Positions,idx_fig]=ZUPT_StrideLength_Heading_Position(Acc,Gyr_unbiased,Step_events,StancePhase,1,idx_fig);

disp('INS-ZUPT ended. Showing results...');
disp('-> TO DO: Check correct integration INS-ZUPT');
```

Check: Step detection thresholds

PDR with pre-recorded logfiles

- PRACTICE 2: **PDR with foot-mounted real Xsen-MEMS IMU**
  - **Conclusions:**
    - Bias&noise in Gyros cause attitude errors growing linear with time => big problems
    - INS does not work with MEMS IMUs, even with bias removed
    - INS-ZUPT makes PDR, with bias removed, to work pretty well (for foot-mounted IMU)





PDR with pre-recorded logfiles

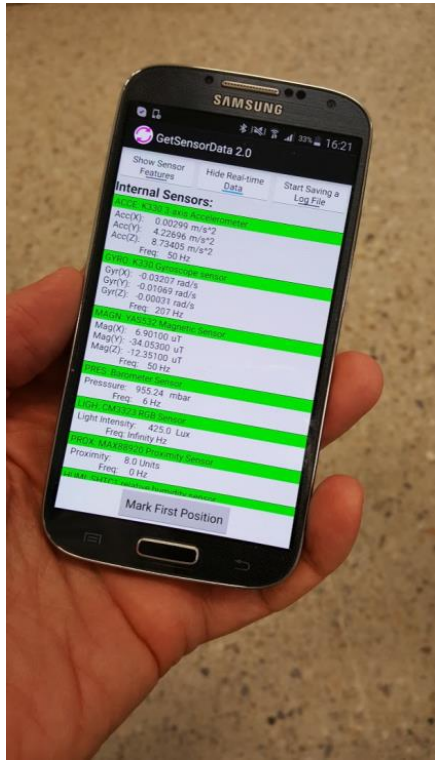
- PRACTICE 3: **PDR with hand-held real smartphone IMU (Samsung S4)**

```
% ----------------------------------------
%        PDR Tutorial: PRACTICE 3 (mai_P3.m)
%       Lopsi Group. CAR-CSIC/UPM
%                   2017
%----------------------------------------

clc; clear all; close all; disp('PRACTICE 3: PDR with hand-held real smartphone IMU (Samsung S4) ');
%
%        PRACTICE 3: PDR with hand-held real smartphone IMU (Samsung S4)
%  1) Load real data with smart-phone IMU (logfile_3loops_1lateralbackwards.txt)
%          -One square 3x (76 steps, forward walk, but last 3 sides lateral/backwards walk)
%  2) Apply SL+theta PDR algorithm and analyse results:
%        -Check bias remove effect
%        -Step detection & Stride Length estimation
%        -Position estimation while walking lateral/backwards

%.....................................................................................
% 1) Load real data with smart-phone IMU (logfile_3loops_1lateralbackwards.txt)
%       -One square 3 times (76 steps, forward walk, but last 3 sides lateral/backwards walk)

% Read log_file
disp('1) Load real data with smart-phone IMU and inspect');
disp('Reading Logfile...');
% load IMU read data: Acc,Gyr de Xsens (3 loops)
%[~,~,Acc,Gyr]=ReadLogFile('.\log_files\logfile_3loops_1lateralbackwards.txt','Xsens',1); %ON FOOT %(2 loops + 1 l
[Acc,Gyr,~,~]=ReadLogFile('.\log_files\logfile_3loops_1lateralbackwards.txt','smartphone',1); %ON HAND %(2 loops +
disp('Logfile Read...');
```



PDR with pre-recorded logfiles

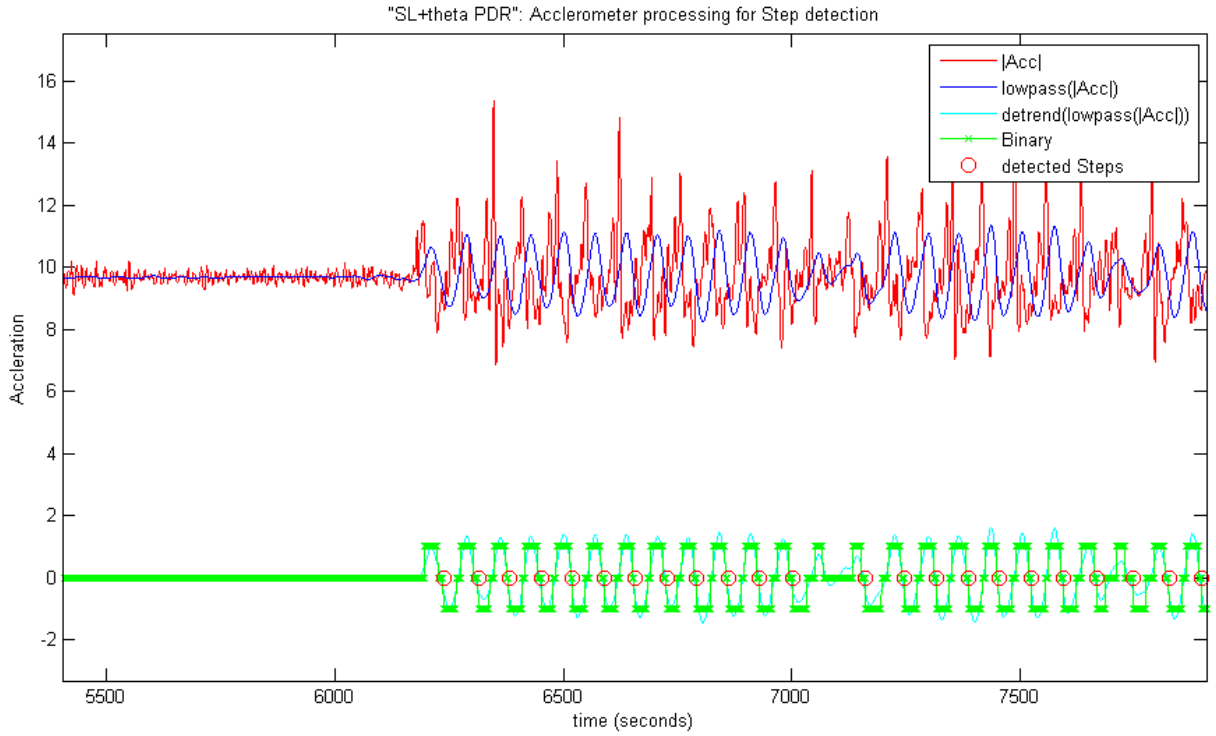
- PRACTICE 3: **PDR with hand-held real smartphone IMU (Samsung S4)**

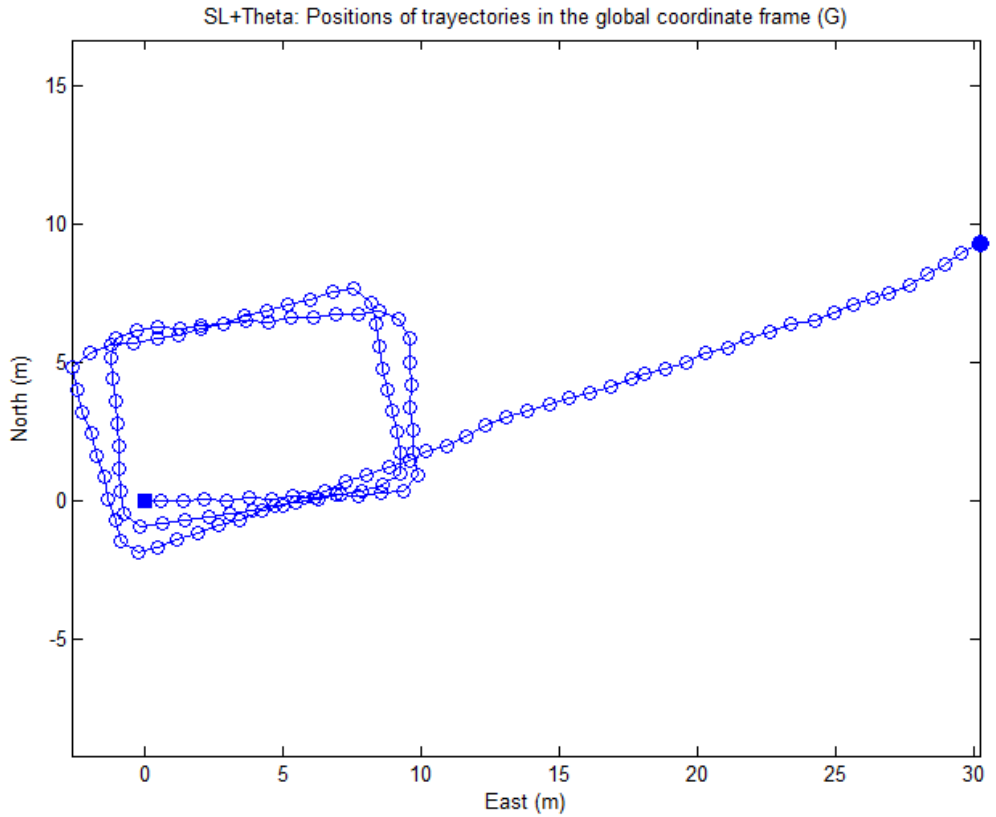
```
%.....................................................................................
% 2) Apply SL+theta PDR algorithm and analyse results
%        -Check bias remove effect
%        -Step detection & Stride Length estimation
%        -Position estimation while walking lateral/backwards
disp(sprintf('\n2) Apply SL+theta PDR algorithm and analyse rsults'));
% Remove bias Gyro
samples=5000;  % asumo 50 segundos parado (y fs=100 Hz)
bias_Gyr=[mean(Gyr(1:samples,1)), mean(Gyr(1:samples,2)), mean(Gyr(1:samples,3))];
Gyr_unbiased=Gyr;  %[nx4]
Gyr_unbiased(:,1:3)=[Gyr(:,1)-bias_Gyr(1), Gyr(:,2)-bias_Gyr(2), Gyr(:,3)-bias_Gyr(3)];

% Apply INS to obtain Pos,Vel y Att:
disp('Apply SL+theta PDR...');
%-----Step detection------
idx_fig=20;
%[Num_steps,Step_events,StancePhase,idx_fig]=StepDetection_Acel(Acc,1,idx_fig);
[Num_steps,Step_events,StancePhase,idx_fig]=StepDetection_Acel_smartphone(Acc,1,idx_fig);
%-----SL-theta-----------
%[StrideLengths, Thetas, Positions,idx_fig]=Weiberg_StrideLength_Heading_Position(Acc,Gyr,Step_events,StancePhase,
[StrideLengths, Thetas, Positions,idx_fig]=Weiberg_StrideLength_Heading_Position(Acc,Gyr_unbiased,Step_events,Stanc
```

PDR with pre-recorded logfiles

- PRACTICE 3: **PDR with hand-held real smartphone IMU (Samsung S4)**





PDR with pre-recorded logfiles

- PRACTICE 3: **PDR with hand-held real smartphone IMU (Samsung S4)**
  - Conclusions:
    - Step detection is more challenging
    - SL+theta PDR is good for forward walking, but can be cheated if direction of motion and direction of IMU/phone is not the same
    - Still to be done: create a robust PDR method for free phone position vs person's motion



Outline

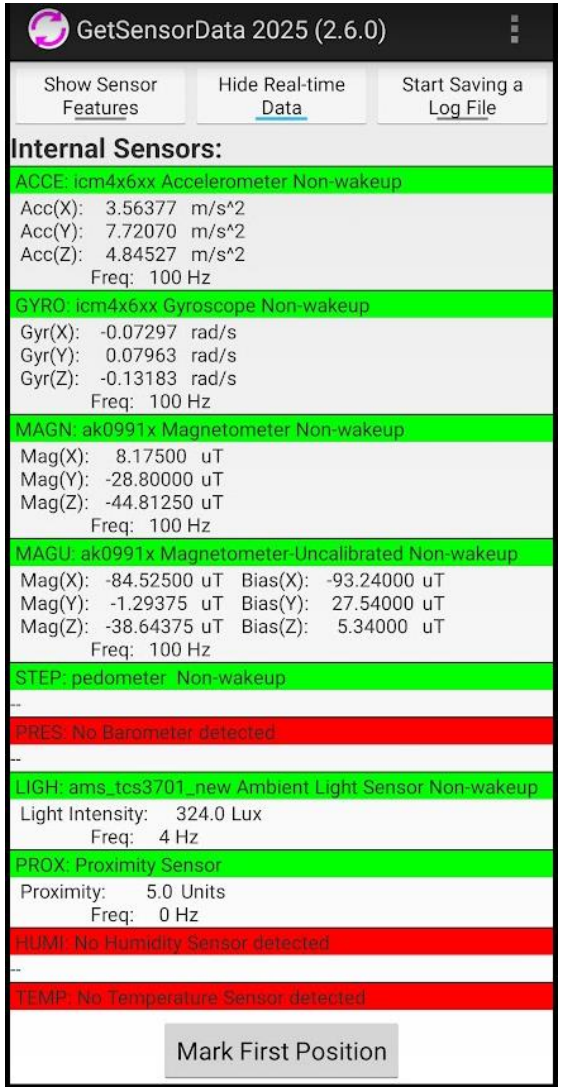
- Some theory:
  1. What is PDR?
  2. Inertial Navigation (INS)
  3. Implementation problems
  4. PDR algorithmic solutions
- Practice (Matlab):
  1. Introduction
  2. PDR with pre-recorded logfiles
  3. PDR with your own phone

PDR with your own phone

- PRACTICE 4: **SL+θ PDR with your phone**
  - Now is when you need:
    1. Your Android phone
    2. App «**GetSensorData**».
       Download App from:
       https://lopsi.car.upm-csic.es/downloads/



  - If not such a phone &App => Team

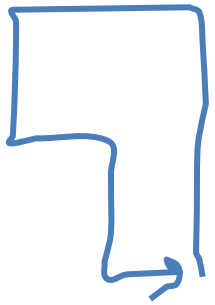


PDR with your own phone

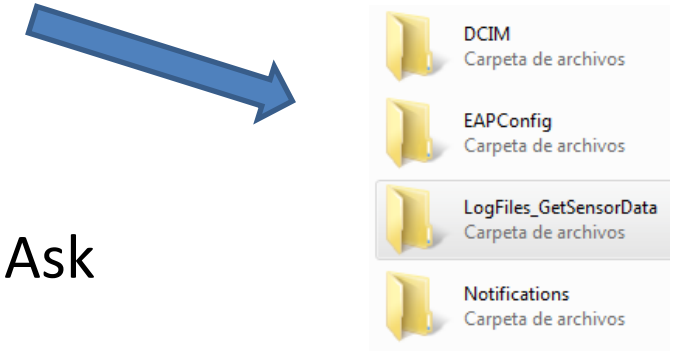
Start Button



- PRACTICE 4: **SL+θ PDR with your phone**
  - **Tasks to be done:**



    - **Design a trajectory** of your wish (control number of steps, turns, ending position,..)
    - Remember to keep **phone in front of you**
    - **Start recording** with the App (Start button)
    - Initial **30 sec still** (easy Gyro bias removal)
    - Move along planned trajectory. Pres Stop button.
    - Transfer recorded logfile to your PC
    - Apply the **SL+theta PDR** algorithm (main_P3.m)
    - Analize results by teams (repeat test if necessary). Ask any question…



Some PDR references

- Oliver J. Woodman, “**An introduction to inertial navigation**”, UCAM-CL-TR-696, 2007.
- R. Harle. “**A survey of indoor inertial positioning systems for pedestrians**”. IEEE Communications Surveys & Tutorials, 15(3), pp. 1281-1293, 2013.
- Skog et al, “**Zero-velocity detection in pedestrian navigation systems - an algorithm evaluation**,” Biomedical Engineering, vol. 57, no. 11, pp. 2657–2666, 2010.
- A.R. Jiménez et al., «**A Comparison of Pedestrian Dead-Reckoning Algorithms using a Low-Cost MEMS IMU**», WISP, pp. 37-42, 2009.

Thank you



Dr. Antonio Ramón Jiménez Ruiz
Centre for Automation and Robotics (CSIC-UPM)
Ctra. Campo Real km. 0,2 Arganda del Rey (Madrid)
e-mail: antonio.jimenez@csic.es

CAR
CENTRE FOR AUTOMATION AND ROBOTICS

CSIC
CONSEJO SUPERIOR DE INVESTIGACIONES CIENTÍFICAS

POLITÉCNICA