Pedestrian Dead-Reckoning (PDR) Tutorial

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
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

• Evaluation (Kahoot)
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...

Can I use inertial navigation syst. (INS) for 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

• Evaluation (Kahoot)
2. Inertial Navigation (INS)

- **INS**
  - Uses an **IMU** (Inertial Measurement Unit)
    - 3 accelerometers (measuring “specific force” [m/s²] caused by **motion** and also **gravity**)
    - 3 gyroscopes (measure “angular rate” [rad/s])
  - Applies navigation equations integrating Inertial data
    - Starting from an initial position and pose, estimates the final trajectory of a moving object
2. Inertial Navigation (INS)

- Global Reference frame (n-frame o navigation):
  - Inertial (Earth Centered), ECEF, Local (Leveled)
- Attitude:
  - 3D orientation of body IMU reference system (b-frame) respect to the global reference system (n-frame)
    - Euler: Roll, Pitch, Yaw,
    - Direction Cosine Matrix (DCM): $R_{b}^{n}$,
    - Quaternions, etc
2. Inertial Navigation (INS)

• Attitude example (Euler):
  – 3 independent rotations around:
    • $X$(Roll) $\rightarrow Y$(Pitch) $\rightarrow Z$ (Yaw)
2. Inertial Navigation (INS)

- 2 types of 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 subtract "gravity", and double integration to get V and P.
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 ($R_{bn}^{n}$)
  – Transform specific force ($f^b$) measured by accelerometers in b-frame to n-frame ($f^n$) using Attitude.
  – Eliminate gravity ($g$) to that acceleration ($f^n$), obtaining: $f'^n$
  – 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

```matlab
% INS
% Acc: IMU Accelerometer 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)
```
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

• Evaluation (Kahoot)
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
3. Implementation problems

- The noise in the MEMS IMU:
  - Additive noise in Acc & Gyr readings
  - Bias (systematic, unstability, turn on)
  - Orthogonality, scaling, thermal 

- Noise generates:
  - Integration errors
  - Errors in Attitude

<table>
<thead>
<tr>
<th></th>
<th>Acc Range (g)</th>
<th>VRW (µg/√Hz)</th>
<th>BRW (mg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>±5/18</td>
<td>122</td>
<td>0.05</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Angular Rate Range (°/s)</th>
<th>ARW (°/√hr)</th>
<th>BRW (°/hr)</th>
</tr>
</thead>
<tbody>
<tr>
<td>±300</td>
<td>3</td>
<td>50</td>
<td></td>
</tr>
</tbody>
</table>

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
3. Implementation problems

- Attitude error => gravity to leak as acceleration
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

• Evaluation (Kahoot)
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 (if IMU at foot => Velocity=0 at stance; ZUPT)
4. PDR algorithmic solutions

• Walking Phases:
  – 2 states (stance and swing) and 7 different 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:
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

  - Ad-hoc
    - Signal Thresholds
    - Multicondition Thresholds
    - Zero-crossing
    - Peak detectors

  - Advanced Processing
    - Wavelet Transform
    - Gait Phase Classification

  - Additional Sensors
    - Pedometer
    - Contact foot-switches
    - Foot radar
4. PDR algorithmic solutions

- **Step detection** using Accelerometers:

\[ a_i = \sqrt{a_{x_i}^2 + a_{y_i}^2 + a_{z_i}^2} \]
4. PDR algorithmic solutions

• **Step Length** (SL) Weinberg Algorithm.
  – 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. **Weinberg hip bounce model**
   
   \[ SL_{\text{Weinberg}} = K \cdot \left\{ \max_{j=[i(k)\pm w]} \tilde{a}_j - \min_{j=[i(k)\pm w]} \tilde{a}_j \right\}^{1/4} \]
4. PDR algorithmic solutions

- **Attitude** estimation (AHRS):
  - Orientation from gyroscopes (relative):
    \[
    \omega^s = (\omega^s_x, \omega^s_y, \omega^s_z) \quad \text{Gyro signals}
    \]
    \[
    C(t) = C(0) \cdot \exp \left( \int_0^t \Omega(\tau) d\tau \right)
    \]
    C (rotation matrix) computed integrating the skew symmetric matrix from an initial orientation.

  - Orientation from accelerometers and magnetometer (absolute references):
    \[
    m^s = (m^s_x, m^s_y, m^s_z) \quad \text{magnetometer signals}
    \]
    \[
    a^s = (a^s_x, a^s_y, a^s_z) \quad \text{acceleration signals}
    \]
    \[
    \begin{align*}
    \phi &= \tan \left( \frac{a^s_y}{a^s_z} \right)^{-1} \\
    \theta &= \tan \left( \frac{-a^s_x}{\sqrt{(a^s_y)^2 + (a^s_z)^2}} \right)^{-1} \\
    \psi &= \tan \left( \frac{-m^h_x}{m^h_y} \right)^{-1} \pm D
    \end{align*}
    \]
    Pitch, Roll, 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.:
    
    $$ SL_{\text{Weiberg}} = K \cdot \left\{ \max_{j=[i(k)\pm w]} \tilde{a}_j - \min_{j=[i(k)\pm w]} \tilde{a}_j \right\}^{1/4} $$
  
  - The $\theta$ angle at foot stance, computed using gyros (& opt. magnetometers).
  
  - Accumulates Stride Length (SL) estimations along the Orientation ($\theta$)
    
    $$\begin{cases}
    P_k(\text{north}) = P_{k-1}(\text{north}) + SL_k \cdot \cos(\theta_{\text{stance}_k}) \\
    P_k(\text{east}) = P_{k-1}(\text{east}) + SL_k \cdot \sin(\theta_{\text{stance}_k})
    \end{cases}$$
  
  - “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).

---

**Correct velocity at foot stance (ZUPT).**

3a. Mean velocity stance $k$

$$\mu_k = \sum_{j=i(k)-w}^{i(k)+w} \dot{v}_j^G / (2w + 1)$$

3b. Correct all samples in step by interpolation

$$\tilde{v}_i^G = v_i^G - [\mu_k (i - i_{(k-1)}) + \mu_{k-1}(i_{(k)} - i)]/m_k$$

*We know it should be ZERO VELOCITY*
4. PDR algorithmic solutions

- **PDR Algorithm:** “**INS-ZUPT**” (cont):
  - Accumulates position increments:
    \[ \Delta P_i = \dot{v}_i^G \cdot \Delta T \]
    \[
    \begin{align*}
    P_i(\text{north}) &= P_{i-1}(\text{north}) + \Delta P_i(\text{north}) \\
    P_i(\text{east}) &= P_{i-1}(\text{east}) + \Delta P_i(\text{east}).
    \end{align*}
    \]
  - Method valid for “any” type of walk (forward/lateral/backwards walk, running, crawling, etc).
  - ...but IMU must be on foot
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

• Evaluation (Kahoot)
Practice: Introduction

• Requirements 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://www.gnu.org/software/octave/download.html](https://www.gnu.org/software/octave/download.html)
    - A **smartphone** with **Android** operating system is needed for one of the practices. The "**GetSensorData.apk**" app should be installed. App available at “Lopsi > More>Downloads”: [http://lopsi.weebly.com/downloads.html](http://lopsi.weebly.com/downloads.html)
      - Log_files in your phone => transferred to your computer via USB cable, Bluetooth, e-mail,…
  – If not => you must integrate into a team
Practice: Introduction

• Download PDR Octave/Matlab software:
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
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

• **Evaluation (Kahoot)**
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 and 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**

```matlab
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**

```matlab
% 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:
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('
2) 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**

```matlab
% 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 misalignments
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**

```matlab
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**

```matlab
%% % 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**

```matlab
% 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

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

disp('3) 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)**

```matlab
clc; clear all; close all; disp('PRACTICE 3: PDR with hand-held real smartphone IMU (Samsung S4) ');

1) Load real data with smart-phone IMU (logfile_3loops_lateralbackwards.txt)
   - One square 3x (76 steps, forward walk, but last 3 sides lateral/backwards)

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_lateralbackwards.txt)
   - One square 3 times (76 steps, forward walk, but last 3 sides lateral/backwards)

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_lateralbackwards.txt','Xsens',1);

[Acc,Gyr,~]=ReadLogFile('.\log_files\logfile_3loops_lateralbackwards.txt','smartphone',1);

disp('Logfile Read...');
```
PDR with pre-recorded logfiles

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

```matlab
% Exercise 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 results'));

% 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)];

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,Stane
```
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

• Evaluation (Kahoot)
PDR with your own phone

- **PRACTICE 4: SL+$\theta$ PDR with your phone**
  - Now is when you need:
    1. Your Android phone
    2. App «*GetSensorData*».


- If not such a phone &App => Team
PDR with your own phone

**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 50 sec still (easy Gyro bias removal)
  - Move along planned trajectory. Press Stop button.
  - Transfer recorded logfile to your PC
  - Apply the **SL+theta PDR** algorithm (main_P3.m)
  - Analyze results by teams (repeat test if necessary). Ask any question...
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

Evaluation (Kahoot)
Testing PDR with Kahoot

- **Play PDR evaluation game:**

You must visit this web page: [https://kahoot.it/](https://kahoot.it/)
Some PDR references

ありがとうございました
Arigatō
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