Unsupervised Pedestrian Trajectory Reconstruction from IMU Sensors
Stéphane Derrode, Haoyu Li, Lamia Benyoussef, Wojciech Pieczynski

To cite this version:
Stéphane Derrode, Haoyu Li, Lamia Benyoussef, Wojciech Pieczynski. Unsupervised Pedestrian Trajectory Reconstruction from IMU Sensors. Traitement et Analyse de l’Information Méthodes et Applications, Apr 2018, Hammamet, Tunisia. <hal-01786223>

HAL Id: hal-01786223
https://hal.archives-ouvertes.fr/hal-01786223
Submitted on 5 May 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Unsupervised Pedestrian Trajectory Reconstruction from IMU Sensors

Haoyu Li¹, Stéphane Derrode¹, Lamia Benyoussef², et Wojciech Pieczynski³

1 École Centrale de Lyon, Univ. de Lyon, LIRIS, CNRS UMR 5105, haoyuli1990@gmail.com
2 EPITA Lyon, lamia.derrode@epita.fr
3 Telecom SudParis, Univ. Paris-Saclay, SAMOVAR, CNRS UMR 5157, wojciech.pieczynski@telecom-sudparis.eu

Abstract. This paper presents a pedestrian navigation algorithm based on a foot-mounted 9DOF Inertial Measurement Unit, which provides accelerations, angular rates and magnetics along 3-axis during the motion. Most of algorithms used worldwide are based on stance detection to reduce the tremendous integration errors, from acceleration to displacement. As the crucial part is to detect stance phase precisely, we introduced a cyclic left-to-right style Hidden Markov Model that is able to appropriately model the periodic nature of signals. Stance detection is then made unsupervised by using a suited learning algorithm. Then, assisted by a simplified error-state Kalman filter, trajectory can be reconstructed. Experimental results show that the proposed algorithm can provide more accurate location, compared to competitive algorithms, w.r.t. ground-truth obtained from OpenStreet Map.

Key words Pedestrian Navigation, Stance Detection, Inertial Sensor, HMMs.

1 Introduction

In recent years, Pedestrian Navigation System (PNS) has gained a lot of attention and been investigated extensively with various kinds of sensors like Inertial Measurement Units (IMUs), camera-based systems and WIFI-based ones. Among these sensors, IMUs have great advantages as they are small and can be worn on the body. They also do not need to pre-install devices like cameras or WIFI systems, and can be used both indoor and outdoor. With the kinematics information acquired from IMUs, it is theoretically possible to transfer the signals from sensor frame to earth frame, or called global frame, based on the sensor orientation, and then to compute velocity and displacement of the motion. Therefore, the exactness of IMUs-based PNS algorithms highly depends on the accuracy of orientation estimation and displacement computation.

When a person is walking, his foot swings in the air and does not move when attached to the ground, alternately. Consequently, one step can be broken down into four phases [7]: stance, push-up, swing and step-down phases. Stance phase is also called zero-velocity state, as the foot is not moving. The most common way to detect stance phase is to set thresholds for both acceleration and angular rate. To avoid
manually setting thresholds, unsupervised learning methods can be used. J. Taborri
et al. proposed an HMM-based distributed classifier for rehabilitation application [2].
H. Pham et al. [8] introduced a LLE-HMM framework and use EMG signals for gait
recognition. A foot-mounted gyroscope for stance detection is implemented in [7],
however because of the weak initialization, parameter learning fails at some time.

In order to reconstruct the trajectory, an algorithm called Pedestrian Dead-
Reckoning (PDR) has been proposed, which computes the displacement by estimat-
ing and integrating each step length and heading. PDR is very easy to be imple-
mented and used in many applications, but its error particularly depends on the
sensor employed. E. Foxlin [4] firstly proposed a PNS algorithm that applies Ex-
tended Kalman Filter (EKF) to estimate the error and uses ZUPT approach to
reduce the large integration error from acceleration to velocity and then to displace-
ment. He called it INS-EKF-ZUPT (IEZ). Then, different strategies were proposed
according to different measurements during the stance phase. S. Rajagopal [9] sup-
poses the angular rate during stance phase to be zero and proposes a Zero Angular
Rate Update (ZARU) algorithm to compute trajectory. The orientation error, par-
ticularly the yaw error, can also be estimated and added into the EKF measurement
with the help of digital compass [1,5].

In this paper, we propose an adaptive stance detection algorithm that uses unsu-
pervised parameter learning algorithm, and employs a simplified Error-state Kalman
Filter for PNS trajectory reconstruction. The remaining of the paper is organized
as follows. First, a left-to-right HMM is presented to detect stance with a specific
initialization algorithm for unsupervised parameter learning. Then a simplified error-
state Kalman filter is developed to compensate for integration errors. An experiment
is conducted on true pedestrian data and the proposed algorithm is compared with
other algorithms.

2 Stance Detection with Left-to-Right HMM

Precise stance detection plays a critical role in ZUPT algorithm, since if the
detection result is wrong (i.e. the swing phase is regarded as a stance phase), then
the velocity will be wrongly compensated to zero while the foot is still moving.

Let start assuming a hidden Markov chain model with observations \( Y = \{Y_1, \ldots, Y_N\} \),
each \( Y_n \in \mathbb{R} \), and with unknown states \( X = \{X_1, \ldots, X_N\} \), each \( X_n = k \in \Omega = \{1, \ldots, 4\} \), where \( \Omega \) represents the stance, push-up, swing and step-down phases
respectively. Assuming a discrete time independent Markov process, \( X \) can be pa-
parameterized by an initial probabilities vector \( \pi = p(x_1) \) and a transition matrix
\( A = p(x_2|x_1) \). In a cyclic Left-to-Right HMM (LR-HMM), this transition matrix
has the following particular shape, only allowing to switch from one class to the
next:

\[ A = \begin{bmatrix}
1 - \Delta_2 & \Delta_2 & 0 & 0 \\
0 & 1 - \Delta_3 & \Delta_3 & 0 \\
0 & 0 & 1 - \Delta_4 & \Delta_4 \\
\Delta_1 & 0 & 0 & 1 - \Delta_1
\end{bmatrix}, \tag{1} \]

with \( \Delta_k = p(x_n = k | x_{n-1}) \) the transition probability from state \( k-1 \) to state \( x_n = k \). Here, the \( N \) observations represent gyroscope measurements \( y_n = [\omega_{xn} \omega_{yn} \omega_{zn}] \), which represent the angular rate of the sensor projected in sensor frame. The distributions of observations conditional to classes are assumed to be Gaussian

\[ p (y_n | x_n = k) \sim \mathcal{N}(\mu_k, \Sigma_k), \tag{2} \]

where \( \mu_k \) (3 \times 1 vector) and \( \Sigma_k \) (3 \times 3 matrix) are the mean and co-variance of observations corresponding to state \( k \). So that the LR-HMM model we deal with is parametrized by the following set of parameters \( \Theta = \{ \pi_k, \Delta_k, \mu_k, \Sigma_k \}_{k \in \Omega} \). All the parameters can be learned using the well-known Baum-Welch algorithm, which is based on the EM principle (Expectation-Maximization) for finding the maximum likelihood iteratively, starting from an initial guess \( \Theta^{(0)} \) of parameters and stopping after a criterion or a maximum number of iterations is reached.

Commonly, the initialization is performed using Kmeans algorithm. However, in LR-HMM the state transition has a specific transition order and structure (see eq. (1)) that Kmeans is not able to provide since it does not takes into account past observations. As an illustration, Fig. 1 shows the gyro observations. The observation values close to zero indicate the stance phase. It can be seen that angular rate goes across zero at transition between two states, then Kmeans wrongly classifies the state as stance phase.

To find the true transition order form Kmeans classification results, we propose the algorithm sketched in the diagram 2. Firstly, we filter the angular rate by using a low-pass Butterworth filter, and select the most significant axis, i.e. the axis whose signal has the largest magnitude, like the blue dashed line in Fig. 1. Secondly, we search for the movement durations between every stance phase that lasts longer than a threshold, e.g. 0.3s for example, and then we find the states corresponding to all the peaks and valleys in every movement duration, the peaks and valleys indicate the states of non-stance phases. Thirdly, we sort these states by time and only keep the first one if one state repeats twice or more. Thus, a list of non-stance states order in every movement duration can be acquired. At last, we count the most firstly appeared state in the order list, which means the push-up phase, the state of swing phase and step-down phase can be derived in the same way. In our example, after the disposal of Kmeans results the re-ordered state transition is 1 → 2 → 3 → 4 (Orange line in Fig. 1). This guess of states is then used to initialize EM for parameter learning.
Figure 1: Re-ordered initial state sequence: 1→2→3→4.

Figure 2: Diagram of state transition re-ordering from Kmeans classification results.

### 3 Error-state Kalman Filter

By knowing the stance phases, velocity is assumed to be zero and the velocity integration error can be acquired easily, but it is not able to obtain displacement integration error directly. Therefore the displacement integration error should be estimated appropriately, since it derives from velocity error and the correlation between velocity and displacement is determined by the integration function (5), thus it can be estimated by an appropriate way. Error-state Kalman filter is firstly introduced in [4] and gives a strategy to estimate the displacement integration error. In this work, We employ a simplified error-state Kalman filter (9 dimensional), since orientation estimation is performed independently by a gradient descent algorithm based quaternion method [6]. The error-state only takes into account the acceleration.
Unsupervised Pedestrian Trajectory Reconstruction from IMU Sensors

The error-state transition equation writes

\[ \delta \eta_{n|n-1} = \Phi_n \delta \eta_{n-1|n-1} + w_{n-1}, \]  

where the superscripts \( e \) and \( s \) represent the earth frame and sensor frame respectively. \( w_n \) represents the process noise with covariance matrix \( Q_n = E(w_n w_n^\top) \) and where the error-state transition matrix \( \Phi_n \) is a 9 × 9 matrix given by

\[ \Phi_n = \begin{bmatrix} I_{3 \times 3} & \Delta t \cdot I_{3 \times 3} & 0_{3 \times 3} \\ 0_{3 \times 3} & I_{3 \times 3} & \Delta t \cdot \hat{C}_n \\ 0_{3 \times 3} & 0_{3 \times 3} & I_{3 \times 3} \end{bmatrix}. \]

Matrix \( \Phi_n \) is time-variant and depends on the value of \( \hat{C}_n \), which represents the rotation matrix required to convert vectors from sensor frame to earth frame at time \( n \). It is derived from the estimated quaternion\[11\].

Now, the measurement equation writes

\[ z_n = H \delta \eta_{n|n} + \nu_n, \]

where \( z_n \) is the measurement from sensor, \( H = [0_{3 \times 3}, I_{3 \times 3}, 0_{3 \times 3}] \) is the measurement matrix, and \( \nu_n \) is the measurement noise, assumed Gaussian with covariance \( R_n \).

Because error-state measurement is only available when \( \hat{x}_n \) is detected as stance phase, the error-state is only updated during this period. The error-state measurement in stance phase is \( z_n = e \nu_n - [0, 0, 0]^\top \) (zero represents the real velocity), thus by the prediction and estimation in Kalman Filter \[10\], the velocity and displacement can be compensated by \( e \nu_n - \delta^e \nu_n \) and \( e \rho_n - \delta^e \rho_n \) respectively, \( \delta^e \rho_n \) and \( \delta^e \nu_n \) in error-state \( \delta \eta_n \) should be reset to zero after the compensation.

4 Experimental Results

An experiment was conducted on a road nearby the campus of École Centrale de Lyon, Ecully, France. The ground truth is obtained from Openstreet Map, the total travel distance is 1075m, the data was stored in the IMU embedded SD card. The sampling rate was set to 100 Hz, the range of accelerometers, gyroscopes and magnetometers were set to 8g, 1000deg/s and 2.5Ga respectively\[1\].

\[1\] more details can be found at manufacturer’s site http://www.shimmersensing.com/images/uploads/docs/ConsensysPRO_Spec_Sheet_v1.1.0.pdf
Before starting to walk, a short standing time period without motion is necessary for initializing the quaternion corresponding to the earth frame (North-West-Up coordinate system), the magnetic declination at Lyon is 1.2°. The LR-HMM method we propose is tested and compared to another threshold based stance detection method detailed in [3]. Parameters of both algorithms are learned or tuned for getting the best result.

![Figure 3: Shimmer3 sensor and placement on the shoe.](image)

![Figure 4: Stance detection, step #1 represents the stance.](image)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Steps Number</th>
<th>Missing Number</th>
<th>False Negative (rate in %)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR-HMM</td>
<td>735</td>
<td>0</td>
<td>6 (0.823)</td>
</tr>
<tr>
<td>Threshold</td>
<td>724</td>
<td>17</td>
<td>14 (1.920)</td>
</tr>
</tbody>
</table>

The total steps number in experiment is 1458, so the steps number of one foot is 729. Compared with the threshold based stance detection method, LR-HMM obtains a more regular stance pattern, rarely makes a false negative detection or misses one step (Fig. 4). Table 1 reports the missing number and false negative detection number of both algorithms.

Trajectory reconstruction is done by different algorithms, including the proposed algorithm in this paper and a commonly used IEZ algorithm (15 dimensional error-
state Kalman Filter). In Fig. 5, the proposed algorithm (HMM+GDA) makes a travel distance of $1077.7m$, the relative travel distance error is 0.25%, the End-to-End error is $23.3m$. From Tab. 2, in spite of the fact that the End-to-End error of HMM+IEZ is a little smaller than the proposed algorithm, the Dynamic Time Warping (DTW) distance of the proposed algorithm is smaller than HMM+IEZ. This interesting point means that the trajectory derived from the proposed algorithm is closer to the ground truth. And furthermore, when tuning parameters for all algorithms, HMM+GDA has the most robust performance for trajectory reconstruction.

![Figure 5: Trajectory of different algorithms compared with the ground truth.](image)

<table>
<thead>
<tr>
<th>Table 2: PNS Trajectory Error.</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMM+GDA</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td>End-to-End Error(m)</td>
</tr>
<tr>
<td>End-to-End Positioning Accuracy(%)</td>
</tr>
<tr>
<td>Travel Distance(m)</td>
</tr>
<tr>
<td>Relative Error of Travel Distance(%)</td>
</tr>
<tr>
<td>DTW Distance</td>
</tr>
</tbody>
</table>
5 Conclusion

We present an algorithm for trajectory reconstruction from a foot-mounted IMU sensor. The basic difficulty of such an algorithm mainly relies on minimizing the double integration required to calculate the displacement (earth frame) from the observed kinematic signals (sensor frame). We propose an algorithm that is mainly unsupervised and relies on a cyclic Left-Right HMM to mimic the periodicity of the step phases during a walk.

It seems that our algorithm performs better than the state-of-the-art methods, but we still need to experiment further the algorithm, to study large-scale motion, and also to study the performance with respect to the elevation when the terrain is not plane. We also plan to investigate if using two sensors together on both two feet can improve trajectory reconstruction results. In that case we have to study possible interference between the two sensors, and their influence on the reconstructed trajectory.

References