Article

Adaptive On-line Lower Limb Locomotion Activity Recognition Using Semi-Markov Model and Single Wearable Inertial Sensor

Haoyu Li [1]*, Stéphane Derrode [1] and Wojciech Pieczynski [2]

[1] École Centrale de Lyon, Univ. Lyon, LIRIS, CNRS UMR 5205, France; haoyuli1990@gmail.com; stephane.derrode@ec-lyon.fr
[2] Telecom SudParis, SAMOVAR, CNRS UMR 5157, Institut Polytechnique de Paris, France; wojciech.pieczynski@telecom-sudparis.eu
* Correspondence: haoyuli1990@gmail.com

Version July 16, 2019 submitted to Sensors

Abstract: Lower limb locomotion activity is of great interest in the field of human activity recognition. In this work, a semi-Markov triplet model-based method is proposed to recognise the locomotion activities when lower limbs move periodically. In the proposed algorithm, the gait phases (or leg phases) are introduced into the hidden states, and Gaussian mixture density is introduced to represent the complex conditioned observation density. The introduced sojourn state forms the semi-Markov structure, which naturally replicates the real transition of activity and gait during motion. Then, batch mode and on-line Expectation-Maximization (EM) algorithms are proposed respectively for model training and adaptive on-line recognition. The algorithm is tested on two datasets collected from wearable inertial sensors. The batch mode recognition accuracy reaches up to 95.16%, whereas the adaptive on-line recognition gradually obtains high accuracy after the time required for model updating. Experimental results show an improvement of performance compared to the other competitive algorithms.

Keywords: Gait analysis; lower limb locomotion activity; triplet Markov model; semi-Markov model; on-line EM algorithm

1. Introduction

Locomotion activity has recently raised great research interest because of its significant potentials in many fields, *e.g.* rehabilitation for injured people [1], surveillance systems or health care for the elderly [2], daily activity management. . . Among these researches [3], many different types of sensors are used, such as camera, wireless beacon, electromyogram (EMG) sensors, electrocardiography (ECG) sensors, and inertial measurement units (IMUs). In a smart home, camera system or wireless beacon can help to understand the activity pattern of the host, and then provide suggestions for a healthy life or make decision when emergency is coming [4]. On the other hand, for the wearable sensors, EMGs can measure the electrical signal of muscles, while ECGs placed on specific body parts can monitor the heart rate. These kinds of signals can be used for evaluating the activity intensity. However, camera systems need to be pre-installed and calibrated, they are also sensitive to the light. While EMGs and ECGs have cables with the host, and they are sensitive to the moisture. By contrast, IMU sensors are small enough to be placed on the body and can be taken anywhere, providing information like 3D acceleration, angular rate, and magnetic field readings. In this work, given the advantages of using IMUs, we propose to use these sensors to collect the acceleration and angular rate of motion for the purpose of activity recognition.

Numerous single sensor-based and multiple sensors-based applications were developed under different scenarios. It seems that using multiple sensors is quite interesting and can help to recognise more complex activities. For example, Hsu *et al.* [5] utilized two IMU sensors placed on wrist and ankle to detect 10 daily activities and 11 sport activities. Xie *et al.* [6] used a hybrid system of inertial sensor and barometer to detect locomotion and static activities. While in this paper we are studying a generic model that can be applied to the recognition of lower limb locomotion activity. This kind of model can work for both single sensor-based and multiple sensors-based applications, the difference is that multiple sensors generate a higher observation dimension than single sensor. For simplification, the proposed model will be validated through only one IMU sensor placed on the lower limb.

The work proposed here is, to some extent, the continuation of our previous work [7], where a non-parametric triplet Markov chain (TMC-HIST) was designed to detect four lower limb locomotion activities: walking, running, stair ascent and stair descent. TMC [8,9] is an extension of hidden Markov chain model (HMC) that includes: the observation $\boldsymbol{Y}$ and hidden state $\boldsymbol{X}$ processes and a third auxiliary hidden state $\boldsymbol{U}$ process. While it keeps a similar parameter estimation and restoration algorithm as HMC. In the TMC-HIST, the hidden state process represented the considered activities, the auxiliary one modelized the gait cycle, and histograms were used to represent the non-Gaussian observation density conditioned on each hidden state. We also developed an adaptive on-line algorithm that based on TMC-HIST to recognise the targeted activities. Results showed that the combination of lower limb activity and gait cycle can significantly improve the recognition performance, and the adaptively parameter updating can gradually fit the motion pattern of people. However, the non-parametric histogram represented the marginal density of observation along one sensor axis, it does not involve the correlation among the three axes of sensor. As a consequence, this weakness may cause a failure when recognising the activity. In addition, the precision of histogram is highly dependent on the volume of data and the width of bins, which require large storage memory and will slow down the processing speed of on-line recognition.

In this work, in order to overcome the weaknesses of TMC-HIST, we focus on developing a new parametric TMC model that can recognise lower limb locomotion activities using one single IMU sensor. Besides, the proposed algorithm should be adaptive and on-line applicable as well, *i.e.* it can adjust its parameters at run-time to suit for the user. By introducing a sojourn hidden state process to form semi-Markov structure, it allows the hidden states $\boldsymbol{X}$ and $\boldsymbol{U}$ keep the same for a while, which is consistent with the activity and gait transition during the motion. Semi-Markov structure is embedded into the TMC to better mimic the real state transition properties. Multi-dimensional Gaussian mixture model (GMM) is introduced to represent the non-Gaussian conditioned observation densities, at the mean time, it involves the observation correlation among the sensor axes. With the introduction of semi-Markov structure and Gaussian mixture density, the specific TMC model will be referred as SemiTMC-GMM in the remaining of this paper. Because of the parametric densities, an on-line parameter learning algorithm based on EM is applied. Therefore, our claimed contributions in this paper are:

- Semi-Markov structure is embedded into the TMC model to make the hidden state transition closer to the realistic motion.
- GMM is adopted to overcome the weakness of non-parametric density, while still allowing to model non-Gaussian data.
- EM-based on-line learning algorithm is adopted to SemiTMC-GMM for making the algorithm work on-line.

The remaining of the paper is organized as follows. Section 2 depicts the state-of-the-art works in the field of activity recognition using wearable sensors. Section 3 gives the definition of conventional TMC model, and gradually extends the model to SemiTMC-GMM. Then, how to apply the proposed model to recognise lower limb locomotion activities is presented in detail at the end of this Section. Section 4 depicts both batch mode and on-line mode parameter learning for the proposed model. In Section 5, the proposed recognition algorithm is tested on two datasets, one is the public dataset [10], another one is our own dataset. Also, the performance of the proposed algorithm are discussed compared to the competitive works. Finally, conclusions and future work are presented in the last Section.

2. Related works

Numerous works have investigated human activity recognition (HAR) in the last decade. The methodologies used recently can generally be classified into two dominant categories: (i), traditional classifiers; (ii), deep learning methods.

For the first category, numerous classiffers have been investigated. Parri *et al.* [11] proposed a fuzzy-logical classifier to identify lower limb locomotion mode, with the assistance of gait phases. The authors developed a lower limb wearable robot system that can help impaired people to perform locomotion activity. Chen *et al.* [12] proposed a robust activity recognition algorithm based on principal component analysis (PCA) and on-line support vector machine (OSVM), the algorithm obtained a robust recognition accuracy over a smartphone dataset collected in six different orientations. In the work [13], the authors compared the performances among the classifiers of SVM, Naive Bayes, k-Nearest Neighbour (kNN) and kStar. Results showed that kNN and kStar obtained the highest accuracy while Naive Bayes obtained the lowest. Zhao *et al.* [14] proposed a 2-layer model to detect six gait phases of walking, the algorithm used Neural Network (NN) to provide a pre-decision of gait phases to Hidden Markov Model (HMM), the final decision of gait phase from HMM obtained an accuracy of 98.11%. The limitation of this study is that only the activity of walking was considered, and the authors only tested their algorithm on straight forward walking, not free walking. In [15], hidden semi-Markov model (HSMM) and semi-Markov conditional random field (SMCRF) were applied to recognise human activity in smart home. The results showed that HSMM consistently outperformed HMM, while SMCRF obtained a similar result to CRF. However, because daily activities at home do not have stationary property, it is not practical to use a stationary transition matrix to represent the activity switches. Moreover, the authors only used Gaussian density to represent the conditioned observation density, which is quite limited for a complex scenario.

In the second category, deep learning-based methodologies are very prevalent. Generally, this kind of method are more inclined for image processing, so it needs to convert sensor data to image discription to support extraction of discriminative features [16]. As reported in [17], convolutional neural network (CNN) is an important category of discriminative deep learning model for HAR. The work [18] proposed convolutional recurrent neural network to recognise daily activity; their algorithm gained an improvement of 6% compared to the state-of-the-art works. Recently, as reported in [19], transfer learning and semantic approach have raised great research interest. Bao [20] and Rokni [21] used transfer learning to automatically construct model for newly added wearable sensors; they obtained an accuracy enhancement between 9.3%-10%. However, the recognition accuracy highly depends on the performance of labeling from source devices, thus it still requires a reliable method for recognition on a single sensor.

Some other methods can also be applied to the dedicated applications and obtain good results. Schneider *et al.* [22] proposed an automatic extraction and selection method of highly relevant features, the method was tested on eight datasets and obtained a general accuracy over 90%. Rezaie *et al.* [23] proposed a feedback controller framework to adapt sampling rate for better efficiency and higher accuracy. Dao *et al.* [24] introduced a man-in-loop decision architecture and data sharing among users, and gradually obtained a high accuracy.

In fact, people perform lower limb locomotion activities everyday, such as moving from one place to another place, doing sports like running and cycling... There are a lot of methods that have been proposed for HAR, while to our best knowledge, very few methods can be found that are especially designed for lower limb locomotion activities, including but not limited to activities like walking and joging [25].

3. Model

In this section, the conventional TMC model is firstly introduced, then it is gradually equipped with more sophisticated structures, *i.e.* applying Gaussian mixture to TMC to obtain the TMC-GMM model and then applying semi-Markov structure to TMC-GMM to obtain the SemiTMC-GMM model. Afterwards, a detailed description of on-line EM algorithm suited for SemiTMC-GMM is given. As a matter of fact, these additional processes can be naturally added because of the high generality of the TMC model through the flexibility of the auxiliary processes.

3.1. Triplet Markov Chain

Consider two discrete stochastic processes $\boldsymbol{X} = (X_1, \cdots, X_N)$ and $\boldsymbol{U} = (U_1, \cdots, U_N)$ as hidden states, where $X_n \in \Lambda = \{1, \cdots, r\}$ and $U_n \in \Gamma = \{1, \cdots, \tau\}$, $n \in \{1, \ldots, N\}$. Let $\boldsymbol{Y} = (\boldsymbol{Y}_1, \cdots, \boldsymbol{Y}_N)$ be a real-valued process representing the observation of the model, each $\boldsymbol{Y}_n \in \mathbb{R}^w$, where w is the observation dimension. Then, the triplet $\boldsymbol{T} = (\boldsymbol{V}, \boldsymbol{Y})$, with $\boldsymbol{V} = (\boldsymbol{X}, \boldsymbol{U})$ is a TMC if $\boldsymbol{T}$ is Markovian. It should be noted here that, in classic TMC, none of processes $\boldsymbol{X}, \boldsymbol{U}, \boldsymbol{Y}, (\boldsymbol{X}, \boldsymbol{U}), (\boldsymbol{X}, \boldsymbol{Y}), (\boldsymbol{U}, \boldsymbol{Y})$ are necessarily Markovian.

Let the realizations of X_n, U_n and $\boldsymbol{Y}_n$ be denoted by their lower cases x_n, u_n and $\boldsymbol{y}_n$ respectively, so $\boldsymbol{v}_n = (x_n, u_n)$, $\boldsymbol{t}_n = (\boldsymbol{v}_n, \boldsymbol{y}_n)$. Also, for simplification, we will denote the probability $p(X_n = x_n, U_n = u_n | \boldsymbol{Y}_1 = \boldsymbol{y}_1, \cdots, \boldsymbol{Y}_N = \boldsymbol{y}_N)$ by $p(x_n, u_n | \boldsymbol{y}_1^N)$ for example. In a general TMC, the transition probability of $\boldsymbol{T}$, $p(\boldsymbol{t}_{n+1} | \boldsymbol{t}_n)$, is assumed to be of the following form:

$$p\left(\boldsymbol{t}_{n+1} | \boldsymbol{t}_n\right) = p\left(\boldsymbol{v}_{n+1} | \boldsymbol{v}_n, \boldsymbol{y}_n\right) p\left(\boldsymbol{y}_{n+1} | \boldsymbol{v}_{n+1}, \boldsymbol{v}_n, \boldsymbol{y}_n\right), \tag{1}$$

where hidden state $\boldsymbol{v}_{n+1}$ depends on $\boldsymbol{v}_n$ and $\boldsymbol{y}_n$, and observation $\boldsymbol{y}_{n+1}$ depends on $\boldsymbol{y}_n$, $\boldsymbol{v}_n$ and $\boldsymbol{v}_{n+1}$. However, in the applications of this paper, $\boldsymbol{y}_{n+1}$ has no links with $\boldsymbol{v}_n$ and $\boldsymbol{y}_n$. So the transition can be simplified in

$$p\left(\boldsymbol{t}_{n+1} | \boldsymbol{t}_n\right) = p\left(\boldsymbol{v}_{n+1} | \boldsymbol{v}_n\right) p\left(\boldsymbol{y}_{n+1} | \boldsymbol{v}_{n+1}\right), \tag{2}$$

which provides process $\boldsymbol{T}$ with the structure of a classical HMC. For simplification, this simplified TMC is referred as TMC in the remaining. The first term $p\left(\boldsymbol{v}_{n+1} | \boldsymbol{v}_n\right)$ in Equation (2) is the state transition probability, the dimension of the matrix is $(r \times \tau) \times (r \times \tau)$. The second term is the probability of observing $\boldsymbol{y}_n$ conditionally to each state. Most of the time, this kind of density is modeled by Gaussian distributions:

$$p\left(\boldsymbol{y}_n | \boldsymbol{v}_n = i\right) \sim \mathcal{N}\left(\boldsymbol{\mu}_i, \Sigma_i\right), i \in \Lambda \times \Gamma. \tag{3}$$

The dependency graph of this particular TMC is shown in Figure 1a, where the node $\boldsymbol{V}$ consists in $\boldsymbol{X}$ and $\boldsymbol{U}$. Regardless of the probabilistic links inside the node $\boldsymbol{V}$, the dependency of $\boldsymbol{Y}$ and $\boldsymbol{V}$ is just in the form of HMC.

For obtaining the probability of individual x_n and u_n conditioned on $\boldsymbol{y}_1^n$, $\boldsymbol{y}_1^N$, we only need to compute the marginal probability of $p(x_n, u_n | \boldsymbol{y}_1^n)$ and $p(x_n, u_n | \boldsymbol{y}_1^N)$ by

$$\begin{aligned} p(x_n | \boldsymbol{y}_1^n) &= \sum_{u_n} p(x_n, u_n | \boldsymbol{y}_1^n), \\ p(x_n | \boldsymbol{y}_1^N) &= \sum_{u_n} p(x_n, u_n | \boldsymbol{y}_1^N). \end{aligned} \tag{4}$$

Likewise, $p(u_n | \boldsymbol{y}_1^n)$ and $p(u_n | \boldsymbol{y}_1^N)$ can be obtained in a similar way. Commonly, the probability $p(x_n, u_n | \boldsymbol{y}_1^n)$ and $p(x_n, u_n | \boldsymbol{y}_1^N)$ are called filtering probability and smoothing probability, respectively.

3.2. TMC embedding a Gaussian Mixture Model

When extending TMC to TMC-GMM, it needs to introduce Gaussian mixture density into the conditioned observation probability. In fact, embedding GMM in TMC can be regarded as introducing a new statistic process $\boldsymbol{H} = (H_1, \cdots, H_N)$ into TMC, where H_n takes its value h_n in a finite set

$K = \{1, \cdots, \kappa\}$ and κ is the number of Gaussian components in the mixture. Let c_{ij} be the weight of jth Gaussian mixture component when $v_n = i$, with the constraint $\sum_{j=1}^{\kappa} c_{ij} = 1$. $\boldsymbol{\mu}_{ij}$ and Σ_{ij} are the mean value and co-variance of the Gaussian mixture component. Denote $\boldsymbol{Z} = (\boldsymbol{T}, \boldsymbol{H})$, and assuming that each H_n is independent from each other. Then $\boldsymbol{Z}$ is Markovian with transitions $p(z_{n+1}|z_n)$ given by

$$p(z_{n+1}|z_n) = p(v_{n+1}|v_n)p(h_{n+1}|v_{n+1})p(\boldsymbol{y}_{n+1}|v_{n+1}, h_{n+1}), \tag{5}$$

where $p(\boldsymbol{y}_n|v_n)$ is

$$\begin{aligned} p(\boldsymbol{y}_n|v_n) &= \sum_{j=1}^{\kappa} c_{ij} \cdot p(\boldsymbol{y}_n|v_n = i, h_n = j), \\ p(\boldsymbol{y}_n|v_n = i, h_n = j) &\sim \mathcal{N}\left(\boldsymbol{\mu}_{ij}, \Sigma_{ij}\right), \quad i \in \Lambda \times \Gamma, j \in K, \end{aligned} \tag{6}$$

with $p(h_n = j|v_n = i) = c_{ij}$. We can see that Equations (5) and (6) are extensions of Equations (2) and (3), by introducing a new process H. The dependency graph of TMC-GMM is shown in Figure 1b.

One point should be noticed here is that we do not need to compute neither the probability of $p(h_n|\boldsymbol{y}_1^n)$ nor $p(h_n|\boldsymbol{y}_1^N)$, since the transition probability in Equation (5) is not conditioned on h_n , *i.e.* H_{n+1} does not have connection with $\boldsymbol{Z}_n$ at the previous time epoch. As a matter of fact, introducing $\boldsymbol{H}$ helps us to establish the model more intuitively, however, it does not change the infra structure of the transition of hidden state $\boldsymbol{V}_n$. Therefore, estimating the individual x_n and u_n in TMC-GMM follows the same as in TMC, by using Equation (4). The only difference between TMC and TMC-GMM is the way of computing the observation probability.

3.3. Semi TMC-GMM

Considering the stochastic process $\boldsymbol{V}$ is semi-Markov means that the hidden state has a remaining duration, which determines the time that the hidden state will keep the same. Generally, this kind of remaining duration is called as sojourn time. In a classic hidden semi-Markov model (HSMM) [26], there is a fixed sojourn time for each possible value of $\boldsymbol{V}$. When $\boldsymbol{V}$ switches to a new value, it will stay the same in a fixed length of remaining duration according to what is the new value. However, in most of the real practices, the sojourn time is not always the same. Then, a more commonly used semi-Markov model is that the sojourn time is distributed in a finite set, *i.e.* the remaining duration may probably be different when $\boldsymbol{V}$ switches to a value twice. Here, we utilize the latter one to establish our model. As described in [27], semi-Markov chain has two different ways of transition when the sojourn time becomes zero. The first one is that the probability $p(v_{n+1} = v_n) = 0$ when the sojourn time is 0 at time n, this guarantees that the hidden state will be switched to another value. While, the second one does not require the hidden state to be different when the sojourn time is 0; in fact, the transition at this exact time yields to a normal transition just like TMC and TMC-GMM. In this paper, we utilize the latter one to extend TMC-GMM into SemiTMC-GMM.

Let consider a new stochastic process $\boldsymbol{D} = (D_1, \cdots, D_N)$ that represents the sojourn state, and the realization of each D_n (denoted by d_n) takes its value in $L = \{0, 1, \cdots, \ell\}$. Then, we can extend TMC-GMM model into SemiTMC-GMM by using the couple $(\boldsymbol{Z}, \boldsymbol{D})$, and the transition probability $p(z_{n+1}, d_{n+1}|z_n, d_n)$ according to

$$p(z_{n+1}, d_{n+1}|z_n, d_n) = p(v_{n+1}|z_n, d_n)p(h_{n+1}|v_{n+1})p(d_{n+1}|v_{n+1}, d_n)p(\boldsymbol{y}_{n+1}|v_{n+1}, h_{n+1}), \tag{7}$$

$$p(v_{n+1}|z_n, d_n) = \begin{cases} \delta_{v_n}(v_{n+1}), & d_n > 0 \\ p^*(v_{n+1}|v_n), & d_n = 0 \end{cases}, \tag{8}$$

$$p(d_{n+1}|v_{n+1}, d_n) = \begin{cases} \delta_{d_n - 1}(d_{n+1}), & d_n > 0 \\ p(d_{n+1}|v_{n+1}), & d_n = 0 \end{cases}, \tag{9}$$

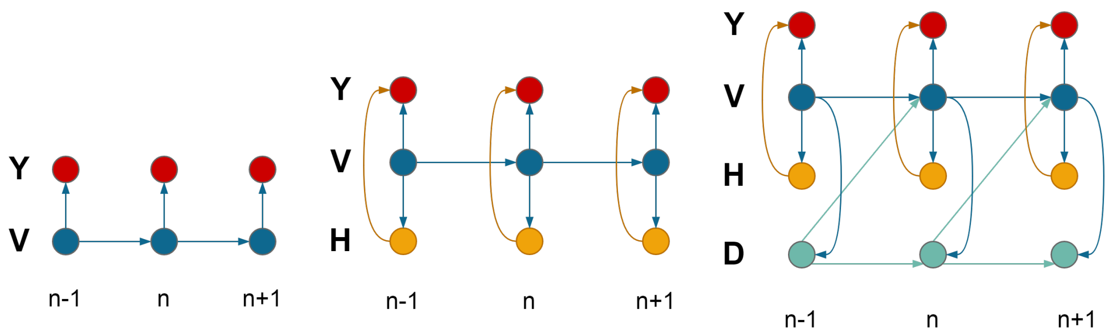
where δ is the Kronecker function ($\delta_a(b) = 1$ for $a = b$ and $\delta_a(b) = 0$ for $a \neq b$).

The properties of the four terms on the right side of Equation (7) are clarified in following:

1. $p(v_{n+1}|z_n, d_n)$ is the transition probability of v_{n+1} conditioned on (z_n, d_n). In Equation (8), p^* is introduced for representing the transition probability when $d_n = 0$. We can see that v_{n+1} is only probably be different from v_n when $d_n = 0$, otherwise v_{n+1} will be exactly the same as v_n. When $d_n = 0$, the transition $p^*(v_{n+1}|v_n)$ behaves the same as the state transition of TMC and TMC-GMM, which means that v_{n+1} can be different from or same as v_n, depending on the distribution of $p^*(v_{n+1}|v_n)$.
2. $p(d_{n+1}|v_{n+1}, d_n)$ is the sojourn state transition probability conditioned on z_n and d_n. In Equation (9), the function $\delta_{d_n - 1}(d_{n+1})$ makes sure that the sojourn time is decreasing, and $p(d_{n+1}|v_{n+1})$ is the distribution of sojourn time conditioned on v_{n+1}.
3. $p(h_{n+1}|v_{n+1})$ and $p(\boldsymbol{y}_{n+1}|v_{n+1}, h_{n+1})$ are same as the ones in TMC-GMM, shown in Equation (6).

Now, the Equations (8) and (9) together describe how the hidden states, V_n and D_n, transfer in SemiTMC-GMM.

The dependency graphs of the three models, *i.e.* TMC, TMC-GMM and SemiTMC-GMM, are shown in Figure 1. The couple $\boldsymbol{V} = (\boldsymbol{X}, \boldsymbol{U})$ is regarded as one hidden state for reducing the complexity of the graphs. Also remind that the total number of processes involved in the three models are 3, 4 and 5 respectively.



(a) TMC. **(b)** TMC-GMM. **(c)** SemiTMC-GMM.

Figure 1. Dependency graphs.

Estimating the individual x_n and u_n is different from both TMC and TMC-GMM, for the sense of introducing the sojourn state D_n. The probabilities of x_n can be obtained by

$$\begin{aligned} p(x_n|\boldsymbol{y}_1^n) &= \sum_{u_n}\sum_{d_n} p(x_n, u_n, d_n|\boldsymbol{y}_1^n), \\ p(x_n|\boldsymbol{y}_1^N) &= \sum_{u_n}\sum_{d_n} p(x_n, u_n, d_n|\boldsymbol{y}_1^N). \end{aligned} \tag{10}$$

The probabilities $p(u_n|\boldsymbol{y}_1^n)$ and $p(u_n|\boldsymbol{y}_1^N)$ are obtained in a similar way.

3.4. Application of SemiTMC-GMM

The question is now how to apply the proposed model to recognise lower limb locomotion activities. In our previous work [7], gait cycle was introduced into the estimation of four locomotion activities, and the results show that it can improve the accuracy. As introduced in [28], one gait cycle can be divided into four gait phases, *i.e.* stance, push-up, swing and step down. In this work, we are pursuing a method that does not require the sensor to be placed on the feet only. On contrary, it can be placed on different places of the lower limb, such as thigh, shank, and foot. The segmentation of gait cycle is based on the motion of foot, so similarly we can define 'leg cycle' based on the motion of leg. One leg cycle can be segmented into four leg phases, which are low position, lifting, high position and dropping.

Let assume the hidden state X represents the activity, and U be the gait cycle or leg cycle. Thus, the dimension of Λ (r) depends on the number of activities; while for Γ, τ is equal to 4. The transition of X and U follows a specific order, because the feet move from attaching on the ground to swinging in the air alternately, or the legs switch between lifting to dropping. Therefore, we define a specific transition graph for X and U. As shown in Figure 2, the numbers 1-4 represent the hidden state U, the four gait and leg phases. We can see that U transfers from phase 1 to phase 4 and back to phase 1 again cyclically if the activity does not change. While when the activity is switching, U transfers from phase 1 of the previous activity to phase 2 of the current activity.

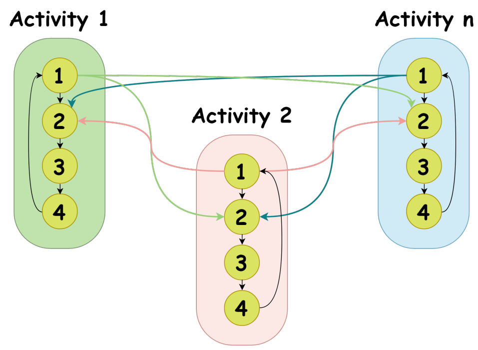


Figure 2. Hidden state transition graph. The activities represent X, the numbers 1-4 represent U and stand for the four gait phases or leg phases.

The hidden states H and D are not the final goal of the recognition, and they have no physical meaning neither. For simplification, the dimension of L (ℓ) is set to 9. This value was determined by our experience, a too small value will make the results of SemiTMC-GMM no difference from that of TMC-GMM, while a too large value will cost too much time for running the code. The performance of different GMM components number (κ) is evaluated on two datasets, as depicted in Section 5.

The observation is obtained by the feature extraction from the sensor readings. The utilized features are the sliding mean value and standard deviation. Since IMUs measure 3-dimensional acceleration and angular rate, then the dimension of the observation $\boldsymbol{Y}$ (w) equals to 12. The initialization of the hidden states is the same as the one in our previous work [7], so it will not be repeated here. Afterwards, based on the initial hidden states and features, the initial GMM density can be easily obtained. When the initialization is done, batch mode EM algorithm can be applied to train the model. Then, the trained model can be used for the batch mode testing, or, as the initial model of on-line EM algorithm.

4. Parameter estimation

From previous section, it is now clear how the hidden state transfers and how to compute the observation probability. In this section, we focus on how to obtain the filtering and smoothing probabilities, and to apply parameter updating based on the on-line EM algorithm.

Before starting the explanation, we need to introduce the parameter set first. As described in the previous Section, the parameter set can be defined as $\theta = \{\zeta_k, a_{lk}, c_{ij}, \boldsymbol{\mu}_{ij}, \Sigma_{ij}\}$, in which ζ_k is the initial probability of hidden state, and a_{lk} is the l-th row and k-th column element in the transition matrix A. Because GMM density only depends $\boldsymbol{v}_n$, then $i \in \Lambda \times \Gamma$, $j \in K$. While in SemiTMC-GMM, the entire hidden state is $(\boldsymbol{V}, \boldsymbol{D})$, then $l, k \in \Lambda \times \Gamma \times L$, and l, k equal to the couple of (i, d_n). Therefore, the initial probability becomes $\zeta_k = p((v_1, d_1) = k)$, and $a_{lk} = p((\boldsymbol{v}_{n+1}, d_{n+1}) = k | (\boldsymbol{v}_n, d_n) = l)$. For simplification, the indices i, j, l, k will keep the same meaning and will no longer be specified in the remaining.

235 *4.1. Batch mode EM algorithm*

The batch mode parameter restoration using EM algorithm is quite simple and has been utilized in many researches. A dominated way to do this is using the well-known Baum-Welch algorithm. This is an algorithm that make the expectation step and maximization step recursively. Here we simply describe how to extend the expectation and maximization steps to SemiTMC-GMM model, within one iteration of the EM algorithm. It is assumed that the forward result $\alpha_n(k)$ and backward result $\beta_n(k)$ have already been obtained according to [7]. Then, the algorithm requires the following probabilities:

$$\gamma_n(k) = p((v_n, d_n) = k|\boldsymbol{y}_1^N) = \frac{\alpha_n(k)\beta_n(k)}{\sum\limits_{k' \in \Lambda \times \Gamma \times L} \alpha_n(k')\beta_n(k')}, \tag{11}$$

$$\tilde{\gamma}_n(i) = \sum_{d_n} \gamma_n((i, d_n)) = \sum_{d_n} p(v_n = i, d_n|\boldsymbol{y}_1^N), \tag{12}$$

$$\tilde{\gamma}_n(i,j) = \tilde{\gamma}(i) \cdot \frac{c_{ij} p(\boldsymbol{y}_n|v_n = i, h_n = j)}{\sum\limits_{j' \in K} c_{ij'} p(\boldsymbol{y}_n|v_n = i, h_n = j')}, \tag{13}$$

$$\xi_n(l,k) = \frac{\alpha_n(l) \cdot p\left(\boldsymbol{y}_{n+1}, h_{n+1}, (v_{n+1}, d_{n+1}) = k \mid \boldsymbol{y}_n, h_n, (v_n, d_n) = l\right) \cdot \beta_{n+1}(k)}{\sum\limits_{l',k' \in \Lambda \times \Gamma \times L} \left\{ \alpha_n(l') \cdot p\left(\boldsymbol{y}_{n+1}, h_{n+1}, (v_{n+1}, d_{n+1}) = k' \mid \boldsymbol{y}_n, h_n, (v_n, d_n) = l'\right) \cdot \beta_{n+1}(k') \right\}}. \tag{14}$$

$\gamma_n(k)$ is the probability of (v_n, d_n) conditioned on all observed data $\boldsymbol{y}_1^N$. $\tilde{\gamma}_n(k)$ is the marginal probability of $\gamma_n(k)$ over d_n, this probability is the one that we are looking for to estimate the concerning hidden state v_n. $\tilde{\gamma}_n(i,j)$ is the probability of each Gaussian component *w.r.t.* $\tilde{\gamma}_n(k)$; this probability helps to compute the parameters related to Gaussian mixture, *i.e.* c_{kj}, $\boldsymbol{\mu}_{kj}$, Σ_{kj}. $\xi_n(l,k)$ is the joint probability of $(v_n, d_n) = l$ and $(v_{n+1}, d_{n+1}) = k$ conditioned on $\boldsymbol{y}_1^N$. Here we give the formula of parameter update by using Equations (11)-(14):

$$\zeta_k = \gamma_1(k), \tag{15}$$

$$a_{lk} = \frac{\sum\limits_{n=1}^{N-1} \xi_n(l,k)}{\sum\limits_{n=1}^{N-1} \gamma_n(l)}, \tag{16}$$

$$c_{ij} = \frac{\sum\limits_{n=1}^{N} \tilde{\gamma}_n(i,j)}{\sum\limits_{n=1}^{N} \tilde{\gamma}_n(i)}, \tag{17}$$

$$\boldsymbol{\mu}_{ij} = \frac{\sum\limits_{n=1}^{N} \tilde{\gamma}_n(i,j)\boldsymbol{y}_n}{\sum\limits_{n=1}^{N} \tilde{\gamma}_n(i,j)}, \tag{18}$$

$$\Sigma_{ij} = \frac{\sum\limits_{n=1}^{N} \tilde{\gamma}_n(i,j)(\boldsymbol{y}_n - \boldsymbol{\mu}_{ij})^{\mathsf{T}}(\boldsymbol{y}_n - \boldsymbol{\mu}_{ij})}{\sum\limits_{n=1}^{N} \tilde{\gamma}_n(i,j)}. \tag{19}$$

In fact, Equations (11)-(14) are the expectation step in one iteration of EM algorithm, while Equations (15)-(19) are the maximization step. Then, the parameter can be learned by recursively performing the two steps until the iteration number exceeds a pre-defined value, 100 maximum iterations for example.

4.2. Sufficient data statistics

Since Gaussian Markov models belong to the exponential family, the likelihood function of SemiTMC-GMM can be written in the form of [29]

$$p_\theta(z_n, d_n) = f(z_n, d_n) \exp\left(\langle s(z_n, d_n), \psi(\theta)\rangle - J(\theta)\right), \tag{20}$$

where $s(z_n, d_n)$ is a vector of complete-data sufficient statistics belonging to convex set S, $\langle \cdot, \cdot \rangle$ denotes the scalar product, function $\psi(\cdot)$ maps θ to the natural parametrization and $J(\cdot)$ is the log-partition function. For SemiTMC-GMM, the definition of statistics is

$$s^{(1)}_{n',lk} = \mathbb{1}\{(v_{n'}, d_{n'}) = l, (v_{n'+1}, d_{n'+1}) = k\}, \tag{21}$$
$$s^{(2)}_{n',k} = \mathbb{1}\{(v_{n'}, d_{n'}) = k\}, \tag{22}$$
$$s^{(3)}_{n',ij} = \mathbb{1}\{v_{n'} = i, h_{n'} = j\}, \tag{23}$$
$$s^{(4)}_{n',ij} = \mathbb{1}\{v_{n'} = i, h_{n'} = j\}\boldsymbol{y}_{n'}, \tag{24}$$
$$s^{(5)}_{n',ij} = \mathbb{1}\{v_{n'} = i, h_{n'} = j\}\boldsymbol{y}_{n'}^{\mathsf{T}}\boldsymbol{y}_{n'}, \tag{25}$$

where $\mathbb{1}\{\cdot\}$ is the indicator function, $n' = 1, \ldots, N$. Then, the statistics vector at time n' is of the form $s_{n'} = \left\{s^{(1)}_{n',lk}, s^{(2)}_{n',k}, s^{(3)}_{n',ij}, s^{(4)}_{n',ij}, s^{(5)}_{n',ij}\right\}$. Consequently, the sufficient statistics S_n is the expectation of $s_{n'}$ conditioned on $\boldsymbol{y}_1^n$

$$S_n = \frac{1}{n}\mathbf{E}_\theta\left(\sum_{n'=1}^{n} s_{n'}\right)\Bigg|\boldsymbol{y}_1^n. \tag{26}$$

Denote $S_n = \left\{S^{(1)}_{n,lk}, S^{(2)}_{n,k}, S^{(3)}_{n,ij}, S^{(4)}_{n,ij}, S^{(5)}_{n,ij}\right\}$, in which the elements are the expectation of the ones with respect to $s_{n'}$. Now, comparing the equation groups (11)-(19) and (21)-(26), we can reform the parameter update Equations (15)-(19) with sufficient statistics

$$\tilde{S}^{(2)}_{n,i} = \sum_{d_n} S^{(2)}_{n,(i,d_n)}, \tag{27}$$

$$\zeta_k = S^{(2)}_{1,k}, \tag{28}$$

$$a_{n,lk} = S^{(1)}_{n,lk}\Big/S^{(2)}_{n,k}, \tag{29}$$

$$c_{n,ij} = S^{(3)}_{n,ij}\Big/\tilde{S}^{(2)}_{n,i}, \tag{30}$$

$$\boldsymbol{\mu}_{n,ij} = S^{(4)}_{n,ij}\Big/S^{(3)}_{n,ij}, \tag{31}$$

$$\Sigma_{n,ij} = S^{(5)}_{n,ij}\Big/S^{(3)}_{n,ij} - \boldsymbol{\mu}_{n,ij}^{\mathsf{T}}\boldsymbol{\mu}_{n,ij}. \tag{32}$$

Remark 1. *Replacing n with N in Equation (26), which means all the observed data $\boldsymbol{y}_1^N$ are used, S_N is then called as complete sufficient statics. Therefore, using S_N to compute the parameters in Equations (28)-(32) will be exactly the batch mode parameter learning that given in Equations (15)-(19).*

4.3. On-line estimation

In the previous section we have discussed about how to use sufficient statistics to learn θ in batch mode. In order to apply the on-line estimation, a common way [29] is to update the sufficient statistics when a new observed data come in

$$S_{n+1} = (1 - \rho_{n+1}) \cdot S_n + \rho_{n+1} \cdot \mathbf{E}_{\theta_n} \left(s_{n+1} | \boldsymbol{y}_{n+1} \right), \tag{33}$$

where ρ_n is the stepsize sequence that satisfies $\sum_{n=1}^{\infty} \rho_n = \infty$, $\sum_{n=1}^{\infty} \rho_n^2 < \infty$. Normally it is set to $\rho_n = 1/n$. Then, the new parameter θ_{n+1} is available by Equations (27)-(32). The estimation of x_{n+1}, u_{n+1} can be obtained by Equation (10).

While in this paper, we do not update θ at every sampling time. Instead, we set a window length W_l and accumulate the latest W_l observed data first. Then use Equations (11)-(14) to get the smoothed result, compute the sequenced statistics $s_n|_1^{W_l}$ for all the W_l data by Equations (21)-(25). Afterwards, update the sequenced sufficient statistics $S_n|_1^{W_l}$ and $\theta_n|_1^{W_l}$ by Equation (33) and Equations (27)-(32), respectively. It should be noticed that in on-line mode, the initial probability ζ_k is not necessary.

5. Experimental results

Two datasets are used to validate the proposed algorithm. The first dataset is the Sport and Daily Activities (SDA) dataset [10], in which eight subjects were enrolled to perform 19 daily and sport activities while wearing five IMUs on their torso, left arm, right arm, left thigh and right thigh. The sensor sampling rate was set to 25 Hz, and each activity lasted about 5 minutes. Because the objective of the proposed algorithm is to detect lower limb locomotion activities that have gait cycle or leg cycle, only 11 activities out of the total are selected in this work: walk in parking lot, walk on treadmill with incline, walk on treadmill on flat, stair descent, stair ascent, run on treadmill, jump, exercise on stepper, exercise of cycling in vertical position, exercise of cycling in horizontal position, exercise on cross trainer. These 11 locomotion activities of SDA dataset are referred as D1A1 to D1A11 in the remaining of this paper.

There are only 1200 samplings for each experiment of SDA, the data length is not long enough to use on-line EM recognition. Therefore we utilize the second dataset for the validation of the proposed on-line EM algorithm. This second dataset is described in [7], is called Locomotion of Foot-mounted IMU (LMFIMU) dataset[1]. 10 subjects were enrolled to perform a specific experiment that lasts nearly 30 minutes with an IMU mounted on the shoe. Each experiment contained two identical sections of a sequence of 4 locomotion activities: walking, running, stair ascent and stair descent. Therefore, the performance of the second section will be improved compared to the first section, if the on-line algorithm can gradually learn the activity pattern of the subject. The 4 locomotion activities are referred as D2A1 to D2A4 in the rest of this paper. The sensor sampling rate was set to 100 Hz, so the data length is long enough for the on-line EM algorithm.

The proposed SemiTMC-GMM model is compared with TMC-GMM to see the advancement of semi-Markov structure in recognising lower limb locomotion activities. While GMM is implemented by different κ to see the impact of the GMM components number that has on recognition accuracy.

5.1. SDA dataset

The batch mode recognition is tested by a leave-one-out cross-validation (LOOCV) strategy, *i.e.* taking one subject for testing and the others for training, then make the test for all the subjects. The sliding window length of feature extraction is set to 5. Both SemiTMC-GMM and TMC-GMM model

[1] The dataset and its details are available on the website: https://github.com/unilee/TMC_LowerLimbActs.

are involved in the validation, the GMM mixture number κ is set to 1, 3, 6, 9 respectively. Particularly when $\kappa = 1$, the conditioned observation density yields to the conventional Gaussian distribution.

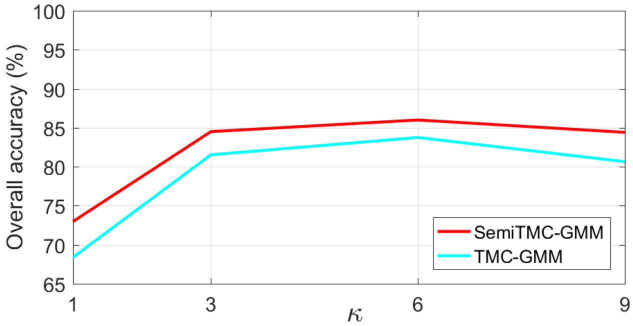


Figure 3. The overall batch mode recognition accuracy on SDA dataset, according to different GMM mixture number κ.

Table 1. The sensitivity, specificity, F1 score, MCC value of the batch mode recognition, for each activity of SDA dataset. Up: TMC-HIST; middle: TMC-GMM when $\kappa = 6$; down: SemiTMC-GMM when $\kappa = 6$.

	Activity (TMC-HIST)					
	D1A1	**D1A2**	**D1A3**	**D1A4**	**D1A5**	**D1A6**
Sensitivity	0.4900	0.5463	0.6997	0.9017	0.7885	1.0000
Specificity	0.9392	0.9883	0.9649	0.9839	0.9222	0.9939
F1 Score	0.4687	0.6574	0.6837	0.8708	0.6057	0.9709
MCC	0.4128	0.6461	0.6511	0.8587	0.5781	0.9684
	D1A7	**D1A8**	**D1A9**	**D1A10**	**D1A11**	**Total**
Sensitivity	0.8308	0.7116	0.9489	0.9972	0.6618	0.7797
Specificity	0.9911	0.9924	1.0000	1.0000	0.9813	0.9779
F1 Score	0.8654	0.7966	0.9737	0.9986	0.7168	0.7826
MCC	0.8535	0.7854	0.9715	0.9985	0.6936	0.7652

	Activity (TMC-GMM)					
	D1A1	**D1A2**	**D1A3**	**D1A4**	**D1A5**	**D1A6**
Sensitivity	0.6784	0.6797	0.5483	0.9146	0.8980	1.0000
Specificity	0.9322	0.9993	0.9866	0.9689	0.9465	0.9995
F1 Score	0.5777	0.8059	0.6525	0.8164	0.7305	0.9978
MCC	0.5353	0.8067	0.6382	0.8025	0.7151	0.9975
	D1A7	**D1A8**	**D1A9**	**D1A10**	**D1A11**	**Total**
Sensitivity	0.8843	0.8917	0.8602	0.9876	0.8784	0.8383
Specificity	0.9961	0.9940	0.9987	0.9998	0.9999	0.9838
F1 Score	0.9197	0.9140	0.9184	0.9930	0.9348	0.8419
MCC	0.9129	0.9059	0.9132	0.9923	0.9309	0.8319

	Activity (SemiTMC-GMM)					
	D1A1	**D1A2**	**D1A3**	**D1A4**	**D1A5**	**D1A6**
Sensitivity	0.6672	0.7247	0.6182	0.9638	0.8767	0.9990
Specificity	0.9457	0.9972	0.9860	0.9773	0.9563	0.9990
F1 Score	0.6054	0.8273	0.7039	0.8752	0.7509	0.9944
MCC	0.5644	0.8223	0.6862	0.8666	0.7327	0.9939
	D1A7	**D1A8**	**D1A9**	**D1A10**	**D1A11**	**Total**
Sensitivity	0.9025	0.9410	0.8561	0.9956	0.9215	0.8606
Specificity	0.9936	0.9922	0.9996	0.9994	1.0000	0.9860
F1 Score	0.9175	0.9324	0.9208	0.9948	0.9590	0.8620
MCC	0.9096	0.9255	0.9165	0.9943	0.9560	0.8516

The overall accuracy of batch mode recognition on SDA dataset is shown in Figure 3. As it can be seen, SemiTMC-GMM achieves an accuracy improvement of about 2%-3% compared to TMC-GMM. The proposed model reaches the highest accuracy of 86.00% when $\kappa = 6$, while the one of TMC-GMM is 83.76%. Meanwhile, TMC-HIST obtains the lowest accuracy of 77.91%. Table 1 shows the sensitivity, specificity, F1 score, and Matthews correlation coefficient (MCC) of each individual activity. Particularly for the sensitivity of each individual activity, it equals to the accuracy of corresponding activity. Activities D1A1 to D1A3 are recognised with relatively poor performance, it is because that these three activities are all walking and are very easily misclassified. As reported in [20], the classifiers of kNN, SVM and decision tree are tested on SDA dataset using all the five sensors. The accuracies are 78.97%, 84.03% and 84.63% respectively. In [21], the authors used SDA dataset and showed single sensor recognition accuracy of four classifiers: kNN, decision tree, discriminant analysis and Naive Bayes. Specifically for the right leg sensor that is used in our paper, the four classifiers obtained accuracy of 81.72%, 78.78%, 87.03%, 76.93%. Therefore, we can state that SemiTMC-GMM outperforms the generic classifiers like kNN, SVM, decision tree and Naive Bayes, and obtains a similar performance of discriminant analysis. On the other hand, the authors in [30] used CNN to recognise human daily activities in OPPORTUNITY dataset [31], which contains activities such as open (close) door, open (close) drawer, clean table, drink cup... They obtained an accuracy of 85.8% by using 23 body-worn sensors, 12 object sensors and 21 ambient sensors. Also for the OPPORTUNITY dataset, [18] used CNN obtains an accuracy of 77.99% by using the body-worn sensors only. While in [32], CNN obtained an accuracy of 93.75% on six activities: walking, stair ascent, stair descent, sitting, standing and laying. Because of the prevalent CNNs can generate high dimensional features that suit for the recognition task, then CNNs may probably be suited for sophisticated activities. But it requires huge quantity of data to train the network, and it is difficult to make CNN work for adaptive on-line scenario. So, maybe CNN could obtain higher accuracy than SemiTMC-GMM, we still believe that our proposed model is competent in some scenarios.

5.2. LMFIMU dataset

For this dataset, the size of sliding window for computing features is set to 15. Firstly, the batch mode recognition is performed using LOOCV strategy. Figure 4 shows the recognition accuracy *w.r.t.* different κ. The accuracy of SemiTMC-GMM when $\kappa = 9$ is 95.16%, while the one of TMC-GMM is 92.57%. Meantime, the choice of κ has less impact on accuracy for SemiTMC-GMM. The recognition accuracy obtained by TMC-HIST is 80.42%, which is lower than the ones of TMC-GMM and SemiTMC-GMM when $\kappa > 1$. Table 2 shows the sensitivity, specificity, F1 score, and MCC of each individual activity. By comparing the batch mode recognition shown in Table 1 and 2, both TMC-GMM and SemiTMC-GMM outperform TMC-HIST. It means that considering the observation correlation improves the recognition performance.

As a matter of fact, Figures 3, 4 and 5 show that introducing semi-Markov structure into the TMC model can improve the accuracy. Meanwhile, using GMM with $\kappa > 1$ also improves the recognition significantly. But it does not mean that using a larger κ allows higher accuracy to be achieved. In Figure 3, the accuracy when $\kappa = 9$ is slightly lower than that obtained when $\kappa = 6$, it is because the observation of SDA dataset is more closer to a GMM mixture of 6 densities. A too much larger κ may probably lead to an over fitting problem. It is sure that κ can be automatically acquired through the methods such as BIC [33] and AIC [34], to make κ consistent with different activities. While for simplification in this paper, we manually set κ to 6 for all the activities based on the experimental results.

Then, the on-line EM algorithm is performed to validate the adaptive on-line recognition performances. The proposed algorithm is implemented in Matlab code, running on a 64-bit system computer with 3.2GHz CPU and 32G RAM. In the dataset, the average experiment time is 32.33 minutes, while the computing time of SemiTMC-GMM when $\kappa = 1, 3, 6, 9$ are 9.72, 14.72, 21.53 and 27.65 minutes respectively. Thus, using on-line EM is applicable in on-line scenarios. The window

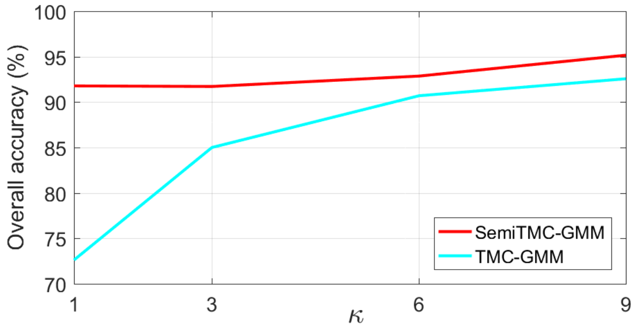


Figure 4. The overall batch mode recognition accuracy on LMFIMU dataset, according to different GMM mixture number κ.

Table 2. The sensitivity, specificity, F1 score, MCC value of the batch mode recognition, for each activity of LMFIMU dataset. Up: TMC-HIST; middle: TMC-GMM when $\kappa = 9$; down: SemiTMC-GMM when $\kappa = 9$.

	Activity (TMC-HIST)				
	D2A1	**D2A2**	**D2A3**	**D2A4**	**Total**
Sensitivity	0.7007	0.9721	0.7705	0.9385	0.8454
Specificity	0.9858	0.8931	0.9174	0.9595	0.9389
F1 Score	0.8169	0.8258	0.6885	0.8596	0.7977
MCC	0.7194	0.7833	0.6317	0.8382	0.7431

	Activity (TMC-GMM)				
	D2A1	**D2A2**	**D2A3**	**D2A4**	**Total**
Sensitivity	0.9399	0.9475	0.9105	0.8590	0.9142
Specificity	0.9720	0.9996	0.9512	0.9787	0.9754
F1 Score	0.9547	0.9723	0.8327	0.8641	0.9060
MCC	0.9130	0.9654	0.8044	0.8419	0.8812

	Activity (SemiTMC-GMM)				
	D2A1	**D2A2**	**D2A3**	**D2A4**	**Total**
Sensitivity	0.9608	0.9829	0.9483	0.8749	0.9417
Specificity	0.9831	0.9987	0.9634	0.9910	0.9841
F1 Score	0.9713	0.9891	0.8799	0.9071	0.9368
MCC	0.9445	0.9861	0.8600	0.8932	0.9210

length W_l for updating the parameters is set to 1000, which means that parameters are updated every 10 seconds.

Figure 5 shows the recognition accuracy obtained by LOOCV strategy. The solid lines are higher than the dashed lines which means that the on-line EM algorithm can improve the recognition performance. Also the GMM with $\kappa > 1$ can significantly improve the accuracy. When $\kappa = 9$, SemiTMC-GMM has an accuracy improved from 95.48% in the first section to 96.93% in the second section, while TMC-GMM achieves an improvement from 93.83% to 95.04%. By contrast, the adaptive on-line algorithm using TMC-HIST in our previous work, the accuracy was improved from 95.32% to 96.93%. However, this high accuracy is mainly because of the gait cycle complete detection in the adaptive on-line algorithm, which manually set the activity of all the samplings in one gait cycle to be identical. If without using the gait cycle complete detection, TMC-HIST will fail in the on-line recognition, with the accuracies of 78.32% in the first section and 65.20% in the second section. Comparing SemiTMC (when $\kappa = 1$) and TMC-HIST, we can conclude that semi-Markov structure is more robust for recognising the hidden states which have sojourn time. Therefore, the results

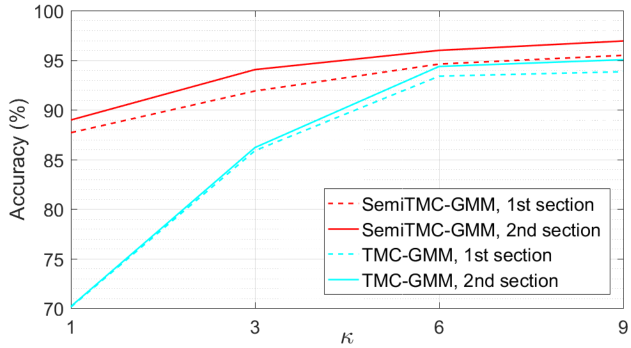


Figure 5. The on-line mode recognition accuracy of the two experiment sections in LMFIMU dataset, according to different GMM mixture number κ.

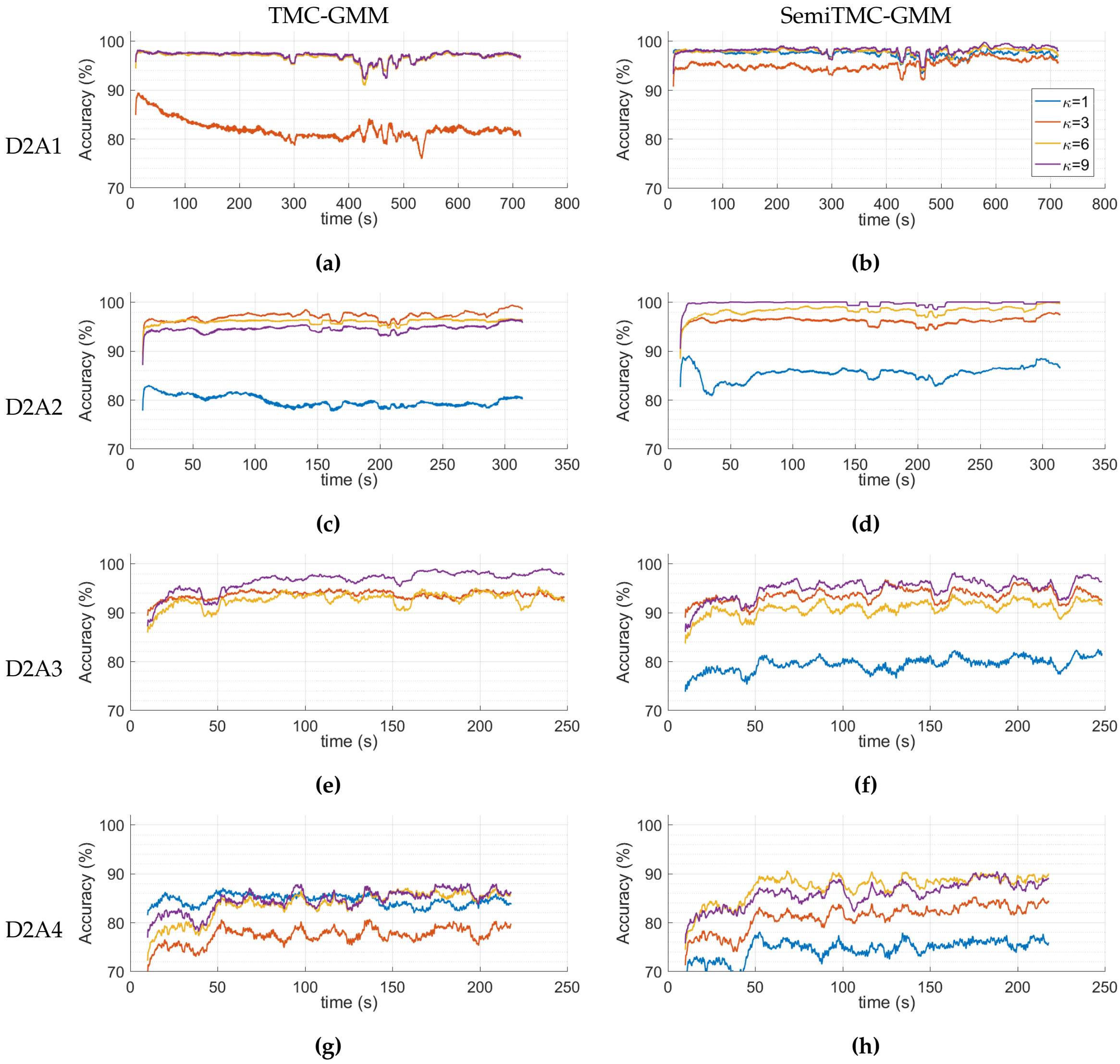
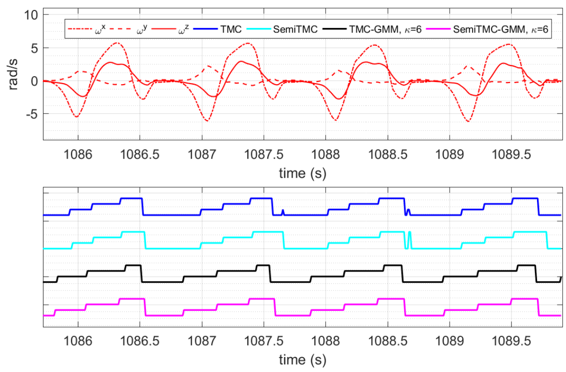


Figure 6. Recognition accuracy computed in the latest 10 seconds *w.r.t.* each activity of LMFIMU dataset. Left column: TME-GMM, right column: SemTMC-GMM.

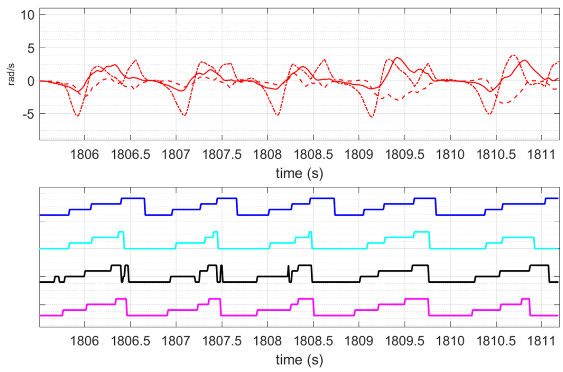
indicate that both GMM density and semi-Markov structure improve the on-line recognition, and the combination the two improves the performance the most.

In order to understand dynamic performance of the parameter updating, Figure 6 shows the recognition accuracy computed during the latest 10 seconds. Notice that the accuracies when $\kappa = 1$ are not displayed in Figures 6a, 6e because TMC obtains accuracies lower than 70% for D2A1 and D2A3. SemiTMC-GMM obtains a relatively fast convergence rate when κ equals to 6 and 9. The activities D2A1 and D2A2 reach high accuracy within 20 seconds in the first section of the experiment, 97.77% and 99.02% respectively. By contrast, D2A3 and D2A4 are slower than the former two activities, and obtain lower accuracies of 92.04% and 89.48% respectively. The main reason of this phenomenon is that the activity patterns of D2A3 and D2A4 vary much more differently among the subjects. But in a general view, we can still state that the on-line EM algorithm can dynamically improve the recognition accuracy to a reasonable level.

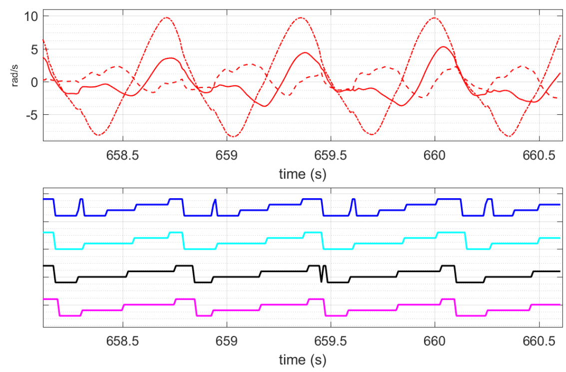
Figure 7 displays the estimated gait cycles of each activity, when the model converged, obtained by TMC-GMM and SemiTMC-GMM, κ is set to 1 and 6. ω^x, ω^y and ω^z are the sliding mean of angular rate along the three axes of sensor. The features are 12-dimensional, but here we only display the acceleration of along the three axes to show how the gaits proceed. Hence, the estimated gait cycles are displayed *w.r.t.* four models, *i.e.* TMC, SemiTMC, TMC-GMM and SemiTMC-GMM. In fact, the gait phases or leg phases are introduced in the model to improve the recognition accuracy of the lower limb locomotion activity. The Figure shows that SemiTMC-GMM obtains the most regular gait cycle, with no fluctuation in short period and no missing detection. As a consequence, the well estimated gait or leg cycle obtained from SemiTMC-GMM leads to a higher activity recognition performance.



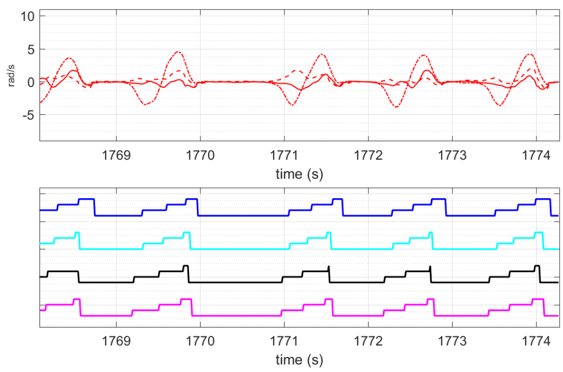
(a) D2A1.



(b) D2A3.



(c) D2A2.



(d) D2A4.

Figure 7. Estimated gait cycle of each activity. The blue, cyan, black and magenta represent the gait obtained by TMC, SemiTMC, TMC-GMM and SemiTMC-GMM respectively.

6. Conclusion

In this paper, we propose an adaptive on-line algorithm using wearable IMU sensor for recognising lower limb locomotion activities, with the help of introducing gait cycle or leg cycle into the model. The algorithm is based on the developed SemiTMC-GMM model, which naturally replicate the real motion. Our experiments show that semi-Markov structure and GMM density can better recover gait or leg cycles, which in return improve the activity recognition significantly. The adopted on-line EM algorithm can gradually improve the accuracy to a high level. The proposed algorithm is not only developed for the applications which require rum-time activity recognition, it is also helpful to those applications that requires gait cycles. For example, the detected gait phases can be beneficial for exoskeleton equipment to better assist impaired people in performing locomotion activities, by providing precise information to the equipment.

While, there are still some limitations. The proposed algorithms only takes periodic lower limb locomotion in consideration, neither the static activity nor non-periodic lower limb locomotion activity is involved in our current work, such as standing and making turn. To distinguish static and motion activities, it is possibly to include specific features into the observations. For example, standard deviation will be close to zero when a person is in static, otherwise it will vary according to the motion. Distinguishing periodic and non-periodic can be accomplished by periodic pattern mining method, such as fast Fourier transform-based [35] and principle component analysis-based [36] approaches. Our future work will focus on adopting more types of activities, including static activity and non-periodic locomotion activities.

Author Contributions: Conceptualization, H. L.; Methodology, H. L., S. D.; Software, H. L.; Validation, H. L., S. D. and W. P.; Formal analysis, H. L., S. D.; Investigation, H. L., S. D.; Resources, S. D., W. P.; Data curation, H. L.; Writing–original draft preparation, H. L.; Writing–review and editing, S. D., W. P.; Visualization, H. L.; Supervision, S. D., W. P.; Project administration, S. D.

Funding: This research received no external funding.

Acknowledgments: We would like to thank the Chinese Scholarship Council (CSC) which partially supports this research.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhu, Z.; Lu, Y.; You, C.; Chiang, C. Deep learning for sensor-based rehabilitation exercise recognition and evaluation. *Sensors* **2019**, *19*, 887.
2. Susnea, I.; Dumitriu, L.; Talmaciu, M.; Pecheanu, E.; Munteanu, D. Unobtrusive monitoring the daily activity routine of elderly people living alone, with low-cost binary sensors. *Sensors* **2019**, *19*, 2264.
3. Elhoushi, M.; Georgy, J.; Noureldin, A.; Korenberg, M.J. A survey on approaches of motion mode recognition using sensors. *IEEE Transactions on Intelligent Transportation Systems* **2017**, *18*, 1662–1686.
4. Waheed, S.A.; Khader, P.S.A. A novel approach for smart and cost effective IoT based elderly fall detection system using Pi camera. IEEE Int. Conf. on Computational Intelligence and Computing Research (ICCIC), 2017, pp. 1–4.
5. Hsu, Y.; Yang, S.; Chang, H.; Lai, H. Human daily and sport activity recognition using a wearable inertial sensor network. *IEEE Access* **2018**, *6*, 31715–31728.
6. Xie, L.; Tian, J.; Ding, G.; Zhao, Q. Human activity recognition method based on inertial sensor and barometer. IEEE Int. Symp. on Inertial Sensors and Systems (INERTIAL), 2018, pp. 1–4.
7. Li, H.; Derrode, S.; Pieczynski, W. An adaptive and on-line IMU-based locomotion activity classification method using a triplet Markov model. *Neurocomputing* **2018**. under review.
8. Gorynin, I.; Gangloff, H.; Monfrini, E.; Pieczynski, W. Assessing the segmentation performance of pairwise and triplet Markov models. *Signal Processing* **2018**, *145*, 183–192.
9. Pieczynski, W.; Hulard, C.; Veit, T. Triplet Markov chains in hidden signal restoration. Image and Signal Processing for Remote Sensing VIII. International Society for Optics and Photonics, 2003, Vol. 4885, pp. 58–68.
10. Barshan, B.; Yüksek, M.C. Recognizing daily and sports activities in two open source machine learning environments using body-worn sensor units. *The Computer Journal* **2014**, *57*, 1649–1667.
11. Parri, A.; Yuan, K.; Marconi, D.; Yan, T.; Crea, S.; Munih, M.; Lova, R.M.; Vitiello, N.; Wang, Q. Real-time hybrid locomotion mode recognition for lower limb wearable robots. *IEEE/ASME Transactions on Mechatronics* **2017**, *22*, 2480–2491.

12. Chen, Z.; Zhu, Q.; Soh, Y.C.; Zhang, L. Robust human activity recognition using smartphone sensors via CT-PCA and online SVM. *IEEE Transactions on Industrial Informatics* **2017**, *13*, 3070–3080.
13. Wannenburg, J.; Malekian, R. Physical activity recognition from smartphone accelerometer data for user context awareness sensing. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **2017**, *47*, 3142–3149.
14. Zhao, H.; Wang, Z.; Qiu, S.; Wang, J.; Xu, F.; Wang, Z.; Shen, Y. Adaptive gait detection based on foot-mounted inertial sensors and multi-sensor fusion. *Information Fusion* **2019**, *52*, 157–166.
15. Van Kasteren, T.; Englebienne, G.; Kröse, B.J. Activity recognition using semi-Markov models on real world smart home datasets. *Journal of Ambient Intelligence and Smart Environments* **2010**, *2*, 311–325.
16. Sathyanarayana, A.; Joty, S.; Fernandez-Luque, L.; Ofli, F.; Srivastava, J.; Elmagarmid, A.; Taheri, S.; Arora, T. Impact of physical activity on sleep: a deep learning based exploration. *CoRR* **2016**, *abs/1607.07034*.
17. Mamoshina, P.; Vieira, A.; Putin, E.; Zhavoronkov, A. Applications of deep learning in biomedicine. *Molecular Pharmaceutics* **2016**, *13*, 1445–1454.
18. Ordóñez, F.; Roggen, D. Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition. *Sensors* **2016**, *16*, 115.
19. Ramasamy Ramamurthy, S.; Roy, N. Recent trends in machine learning for human activity recognition—A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* **2018**, *8*, 1254.
20. Bao, Y.; Chen, W. Automatic model construction for activity recognition using wearable devices. IEEE Int. Conf. on Pervasive Computing and Communications Workshops (PerCom Workshops), 2018, pp. 806–811.
21. Rokni, S.A.; Ghasemzadeh, H. Autonomous training of activity recognition algorithms in mobile sensors: a transfer learning approach in context-invariant views. *IEEE Transactions on Mobile Computing* **2018**, *17*, 1764–1777.
22. Schneider, T.; Helwig, N.; Schütze, A. Automatic feature extraction and selection for condition monitoring and related datasets. IEEE Int. Instrumentation and Measurement Technology Conf. (I2MTC), 2018, pp. 1–6.
23. Rezaie, H.; Ghassemian, M. An adaptive algorithm to improve energy efficiency in wearable activity recognition systems. *IEEE Sensors Journal* **2017**, *17*, 5315–5323.
24. Dao, M.S.; Nguyen-Gia, T.A.; Mai, V.C. Daily human activities recognition using heterogeneous sensors from smartphones. *Procedia computer science* **2017**, *111*, 323–328.
25. Martindale, C.F.; Sprager, S.; Eskofier, B.M. Hidden Markov model-based smart annotation for benchmark cyclic activity recognition database using wearables. *Sensors* **2019**, *19*.
26. Barbu, V.S.; Limnios, N. *Semi-Markov chains and hidden semi-Markov models toward applications: their use in reliability and DNA analysis*; Vol. 191, Springer Science & Business Media, 2009.
27. Lapuyade-Lahorgue, J.; Pieczynski, W. Unsupervised segmentation of hidden semi-Markov non-stationary chains. *Signal Processing* **2012**, *92*, 29–42.
28. Shetty, N.; Bendall, S. Understanding the gait cycle, as it relates to the foot. *Orthopaedics and Trauma* **2011**, *25*, 236–240.
29. Cappé, O. Online EM algorithm for hidden Markov models. *Journal of Computational and Graphical Statistics* **2011**, *20*, 728–749.
30. Yang, J.; Nguyen, M.N.; San, P.P.; Li, X.; Krishnaswamy, S. Deep convolutional neural networks on multichannel time series for human activity recognition. Twenty-Fourth International Joint Conference on Artificial Intelligence, 2015.
31. Roggen, D.; Calatroni, A.; Rossi, M.; Holleczek, T.; Förster, K.; Tröster, G.; Lukowicz, P.; Bannach, D.; Pirkl, G.; Ferscha, A.; others. Collecting complex activity datasets in highly rich networked sensor environments. 2010 Seventh international conference on networked sensing systems (INSS). IEEE, 2010, pp. 233–240.
32. Ronao, C.A.; Cho, S. Human activity recognition with smartphone sensors using deep learning neural networks. *Expert systems with applications* **2016**, *59*, 235–244.
33. Bhat, H.S.; Kumar, N. On the derivation of the Bayesian Information Criterion. *School of Natural Sciences, University of California* **2010**.
34. Arnold, T.W. Uninformative parameters and model selection using Akaike's Information Criterion. *The Journal of Wildlife Management* **2010**, *74*, 1175–1178.
35. Li, Z.; Ding, B.; Han, J.; Kays, R.; Nye, P. Mining periodic behaviors for moving objects. Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2010, pp. 1099–1108.

36. He, Z.; Wang, X.S.; Lee, B.S.; Ling, A.C. Mining partial periodic correlations in time series. *Knowledge and Information systems* **2008**, *15*, 31–54.