GROUND MOBILE TARGET TRACKING BY HIDDEN MARKOV MODEL

Huynh Quan Hieu, Vu Dinh Thanh

Abstract

In recent few years, there are many researches of finding the efficient method for tracking moving objects using some fundamental and traditional instruments such as GPS, video camera,… The most important task is to find out the accompanying algorithm to realise this tracking with a better precision. In this paper, an algorithm based on Hidden Markov Model (HMM) for ground mobile target tracking is introduced. The HMM is a good choice for modelling moving process because of its character in modelling sequential processes. Suppose the moving object is on a known ground plane and theHMM parameters are the motion capture data (position, speed, steering angle,…). Once the HMM for the target motion has been constructed, the Viterbi algorithm is applied to find the trajectory of the target. Some illustrating results in modelling a moving target (for instance, a mobile cell phone) will be presented.

***Keywords**: Hidden Markov model, target tracking, Viterbi algorithm, observation,sensor.*

I. Introduction

The mobile target tracking is one of new researches which appear in the next generation of mobile communication, where all the mobile system require higher performances in operation. In the real mobile system , the positions of object are not known exactly and directly but we can detect them through a time sequence of measurements. This shows that there are two processes in parallel: the first process involves the real movement of the target which has to recognize and the other is the accumulated observation sequences which are provided by the first one. Such problems are the same for speech recognition or video tracking. This paper uses the results of works based on image sequences obtained by fixed surveillance cameras, or by oriented signals from BTS or by sensor array to find the series of positions of the target versus time.

While the target moves, the eventual motions and the object position are updated and so the data-base is changed. For the motion on a plane surface, e.g. mobile target moving on roads, the 2D_tracking is sufficiently used. Since the positioning data are directly updated, one smoothing path process is activated and then applied the HMM. These data are now used as the input parameters of HMM system.

In this paper, we only focus on a Hidden Markov Model with discrete hidden states and discrete observations from the states, the signal processing is not involved here. The simulations of the model, implementing on Matlab, shows the results of tracking paths and the respective accuracies according to the learning or non-learning modes.

II. System Overview

The input system is presented in Fig. 1[1]. Its inputs are the signals transmitted by target. A sensor receives these signals and then conveys them into data_base as the input of a smooth function. The output of a smooth function will be the input parameters of HMM system.

Besides, the medium noise plays an important effect on the input signals which can influence much on the accuracy of the model. The HMM is also able to decrease this effect due to its training capacity.

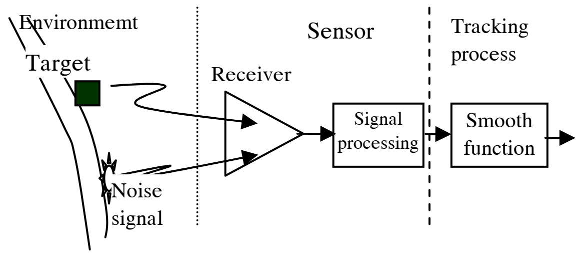

Figure 1: *Overall input system*

The block scheme of the HMM system is shown in Fig.2 [3] , where the smoothened input signals are used to characterize the object motion and to generate the observations. The HMM provides the estimated locations of the target and then produces its tracking path.

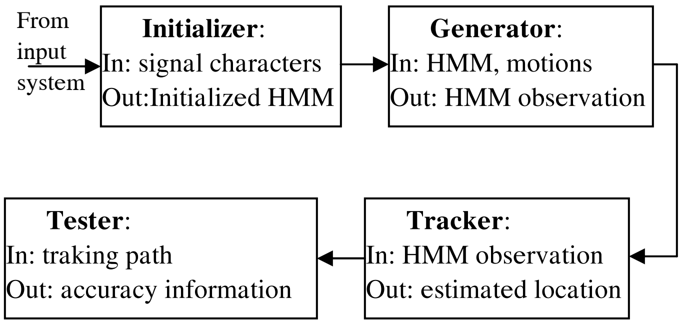

Figure 2: *HMM system*

For instance, we consider a real tracking system based on the BTSs in a mobile communication as in Fig. 3.

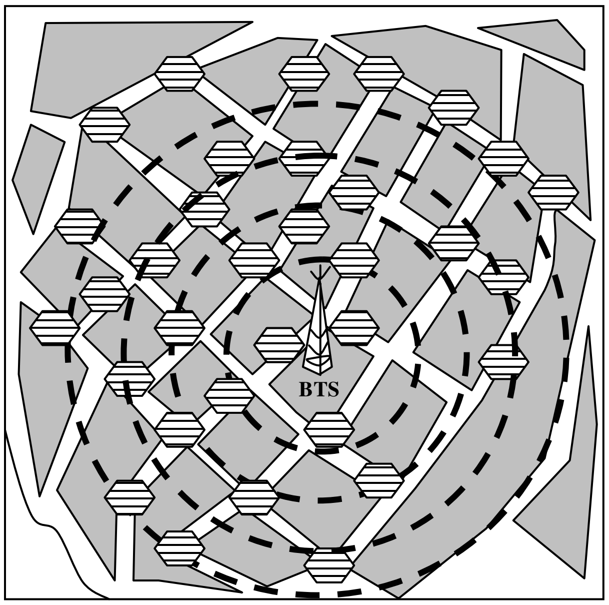

Figure 3 : *A real tracking BTS*

In Fig. 3, the road is discreted into a grid of cells . The surveillance area of BTS is depicted by circles in which the target moves. In tracking time, while the target is moving from state to another state, it will send its changed location signals (azimuth angle, elevation angle, TOA, range error, ..) to BST which, in turn, processes signals and then transmits its data to the "**Server** ". The server does calculate the coarse positions of the target before giving the HMM system. [1][5].

III. Overview Hidden Markov Model

A Hidden Markov Model (HMM) consists of a set of N states, each of which is associated with a set of M possible observations. The parameters of the HMM include:

- An initial matrix of state probabilities:

$$\pi = [p_1, p_2, ..., p_N]^T \qquad (1)$$

whose elements $p_i, i \in [1, N]$ describe the position distribution probabilities of the target over the initial state set at the beginning t = 1.

- A transition matrix

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NN} \end{pmatrix} \qquad (2)$$

whose elements $a_{ij}; i, j \in [1, N]$ are the transition probabilities from state i to state j .

- An observation matrix

$$\mathbf{B} = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1M} \\ b_{21} & b_{22} & \cdots & b_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ b_{N1} & b_{N2} & \cdots & b_{NM} \end{pmatrix} \qquad (3)$$

whose elements b_{im} are the probabilities of observing symbol m $\in$ [1, M] given that the system is at the state i $\in$ [1, N].

The HMM parameter set is denoted by $\lambda = (\mathbf{A}, \mathbf{B}, \pi)$.

The model order pair (N,M) and restrictions on allowed transitions and observed symbols define the structure of the model. The more (N,M) is, the more exactly the model is and the larger the surveillance area is. The transition probabilities express which type the model is (the model can be ergodic or left-right or coupled). When M is equal to N, it means that the observations are, in fact, the primary estimated positions of the target, otherwise the whole of a set of observed symbols will determine the primary position. As usual, the HMM has three problems [4]:

- *Evaluating problem:* what is the probability of the observation O, given the model λ, i.e. $P(O|\lambda)$? ⇒ Solution: Forward or Backward algorithm.
- Decoding problem: What is the most likely state sequence given the observation O, i.e. $\arg_S[\max P(S,O|\lambda)]$? ⇒ Solution: Viterbi algorithm.
- Estimating problem: How can we estimate parameters given the training observation sequences, $\lambda^* = \arg_\lambda[\max P(O|\lambda)]$? ⇒ Solution: Baum-Welch algorithm.

The HMM which we use is a discrete Markov model, i.e the output of HMM is a sequence of discrete states. Furthermore, the state transition probabilities depend only on the previous state, so that the model is first order model.

IV. Application of HMM in tracking target.

The proposed method consists of following steps:

1. *Initiating the model*

Firstly, we assume that the target is moving in a known surveillance area. The discrete model makes the area be divided into N cells corresponding to the states of the object. For instance, when the target is in cell i at time t , its model is in the state $q_i(t)$.

The movement of the target from the state i to the state j is described by transition probabilities a_{ij} in matrix **A**. Their values often depend on the speed distribution of the target, on the geographical feature of the area and on the allowed transitions.

The state q_i of the target will emit an observation symbol o(t). The observation symbol probability distributions **B**=$[b_{ij}]$ are estimated from the observed signals from sensors. Generally, there is many ways to obtain the observed signals such as TOA, AOA, data_comparision, cell_id. In this paper, we suppose these signals are available. The number of symbols of each state M is set equal to N.

Finally, we establish the intial state distribution π, which is the probability that the model is in states at beginning. For easy tracking, we assume that the initial state is known.

2. *Training the model*

Once the model has been established, we wish that the observations received from model are of the highest prtobability. That is why we change the HMM's parameters for their matching to reality as much as possible.

Imagine that a target moves within some defined and typical trajectories such that its positions along the time are detected by sensors. The outputs of the sensors accumulate in a set of observation sequences. Each sequence $\mathbf{O}^{(m)} = \{O_1^{(m)}, O_2^{(m)}, \ldots, O_K^{(m)}\}$ is a string of states which the target has occupied (m: mth sequence, O_k : state at the time instant number k). Now, we use forward-backward procedure to calculate $P(O|\lambda)$ [5].

Forward procedure:

- $\alpha_1(i) = \pi_i . b_i(O_1)$, $1 \le i \le N$ (4)

- $\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] . b_j(O_{t+1})$, with $1 \le t \le T-1$, $1 \le j \le N$ (5)

- And $P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$ (6)

Backward procedure:

- $\beta_T(j) = 1$ $1 \le j \le N$ (7)

- $\beta_{t-1}(i) = \left[\sum_{j=1}^{N} \beta_t(j) a_{ij}\right] . b_j(O_{t-1})$ with $2 \le t \le T$, $1 \le i \le N$ (8)

- And $P(O|\lambda) = \sum_{i=1}^{N} \beta_1(i)$ (9)

The Baum – Welch algorithm:

- $$\xi_k(i,j) = \frac{\alpha_k(i).a_{ij}.b_j(O_{k+1}).\beta_{k+1}(j)}{P(O|\lambda)} \quad (10)$$

with $\beta_{k+1}(i) = \sum \beta_{k+2}(j).a_{ij}.b_j(O_{k+1})$; $\beta_K(i) = 1$ (11)

- $$\gamma_k(i) = \sum_{j=1}^{N} \xi_k(i,j) = \frac{\alpha_k(i).\beta_{k+1}(i)}{P(O|\lambda)} \quad (12)$$

- And
$$\begin{cases} a'_{ij} = \sum_{k=1}^{K-1} \xi_k(i,j) \Big/ \sum_{k=1}^{K-1} \gamma_k(j) \\ b'_j(O_k = i) = \sum_{\substack{k=1\\O_k=i}}^{K} \gamma_k(j) \Big/ \sum_{k=1}^{K} \gamma_k(j) \\ \pi'_i(k=1) = \gamma_1(i) \end{cases} \quad (13)$$

Equation (13) produces a new set of training parameters of HMM system.

The trained model $\lambda' = \{A', B', \pi'\}$ has a property $P(O|\lambda') \geq P(O|\lambda)$.

Furthermore, we can learn model parameters from K observation sequences in [4]

3. Extracting the target's track

Commonly, HMM is only a model of tracking which is based on observations with an optimal algorithm. Hence, the more accurate observations, the better target's tracking is. In real system, HMM, therefore, is used after the primary estimator (for extracting the observations).

We can obtain the target's trajectory through two algorithms:

- The former is deriving the most likely state q_t at each time step individually:

$$q_t = \arg\max_{1\le i\le N} [\gamma_t(i)], \quad 1 \le i \le T \quad (14)$$

If the matrix **A** has the transition state probability $a_{ij} = 0$, the best state sequence may not be valid. Therefore, it is required to find an algorithm estimating the single best state sequence over the entire obvervation time: the Viterbi algorithm.

- Viterbi algorithm:

We define the quantity:
$\delta_t(i) = \max_{q_1,q_2,\ldots,q_{t-1}} P(q_1 q_2 \ldots q_{t-1}, q_t = S_i, o_1 o_2 \ldots o_t | \lambda)$ $\delta_t(i)$ is the highest probability along a single path at time t, which accounts for the first t observations and ends in state S_i. By induction, we have

$$\delta_{t+1}(j) = \left[\max_i \delta_t(i) a_{ij}\right] b_j(o_{t+1}) \quad (15)$$

The whole Viterbi algorithm is described as follows [5]:

- o Initialization:

$$\delta_1(i) = \pi_i . b_i(o_1), \quad 1 \le i \le N \qquad \varphi_1(i) = 0 \quad (16)$$

- o Recursion:

$$\delta_t(j) = \max_{1\le i\le N} \left[\delta_{t-1}(i) a_{ij}\right] \cdot b_j(o_t) \qquad \varphi_t(j) = \arg\max_{1\le j\le N} \left[\delta_{t-1}(i) a_{ij}\right] \quad (17)$$

with $2 \le t \le T$; $1 \le j \le N$

- o Termination:

$$P^* = \max_{1\le i\le N} [\delta_T(i)] \qquad q_T^* = \arg\max_{1\le i\le N} [\delta_T(i)] \quad (18)$$

- o State sequence backtracking:

$$q_t^* = \varphi_{t+1}(q_{t+1}^*), \; t = T-1, T-2, \ldots, 1 \quad (19)$$

There are some modified versions of training and tracking. For example, the backward variable is defined on partial observation sequence, other applications produce observations with AWGN.

To perform Viterbi algorithm, we must obtain observation sequences. In order to get these data, we use an array of sensors, each sensor gives an obsevation at each time instant. A data fusion approach is then used for linking all observations. The final observation has the highest accurate state where the target occupies.
The detection of a target on each sensor depends on the correlation between the target and the sensor; such as the range and Doppler, the TOA (Time of Arrival), the AOA (Angle of Arrival), … [6][7].

Data comparision [2] is commonly used as a fusion approach which is not mentioned here. The results of data comparision are considered as observation sequences being the inputs of HMM.

V. The model simulation and result

A real model as in Fig. 4 is considered:

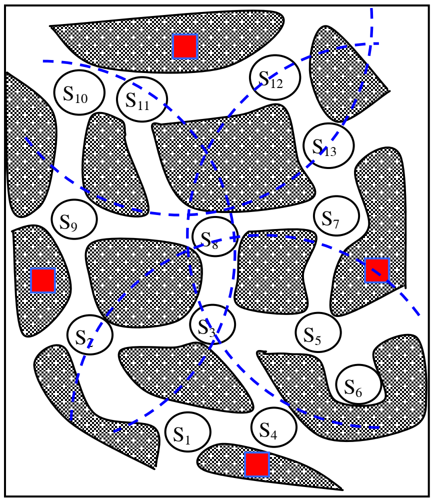

Figure 4: *A real model.*

In this model:

S_i : the state i of the target

▨ : the obstacles such as building blocks

🟥 : the sensor such as BTS, camera or acoustic sensor.

╌ : the surveillance area of a sensor.

In Matlab simulation, all the states of target are defined as the accessible points, while the obstacles are just the unaccessible points which the target can not reach.

For simulating the model, we initial its parameter as follows:

- The transition probability is determined by the target's kinematic constraints and the surveillance area constraints [8]. For example, a_{ij} approximates to 1 if the state j is near the state i and becomes less in case of farther states. We can build up a sparse matrix **A** by choosing some values of a_{ij} equal to 0, on condition that the state i can not reach the state j at each time step.

- The observation probability distribution gets its value by the best quantity at the diagonal of matrix **B** which means that the target in the state i will produce the most observation i and b_{ij} $(i \neq j)$ has lower distribution (other authors uses Gaussian distribution).

- The initial marix can be simplified, because the initial state is supposed to be known. For example, $\pi = [0,0,1,0,\ldots,0]'$ means the first state is i = 3.

The final diagram is depicted as the following Fig.5.

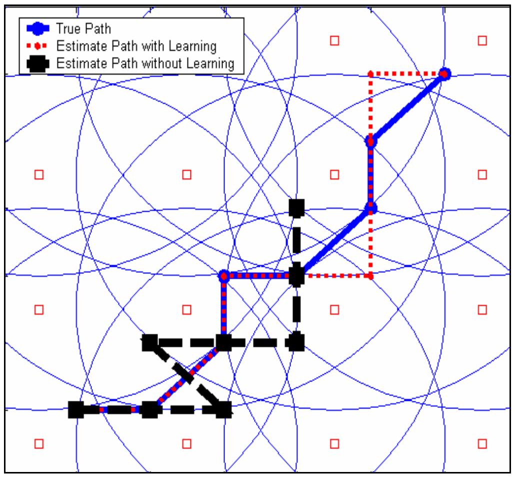

Figure 5: *The mobile target's trajectory in the surveillance area with learning and non-learning methods.*

Clearly, the estimated path without learning can not track the true path, while the estimated path with learning can.The learning method also depends on the number of recursive trainings.

Figure 6: *The distant errors of the mobile target versus the number of recursions*

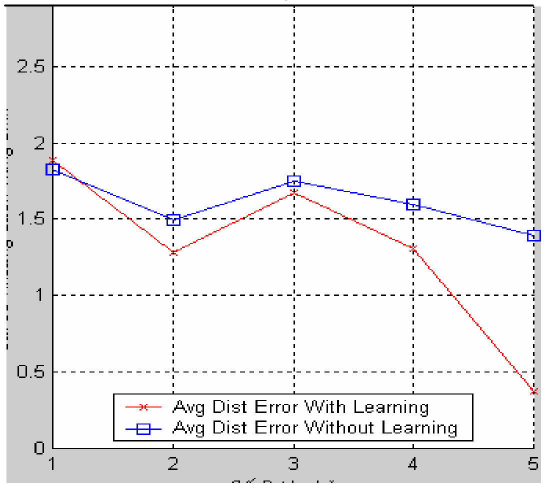

Fig. 6 represents the relation between the average distant error and the number of recursion. Generally, the more we train the model, the less the error is, and the training method attains, of course, more accuracy than the non-training method. Consider another result as Fig. 7

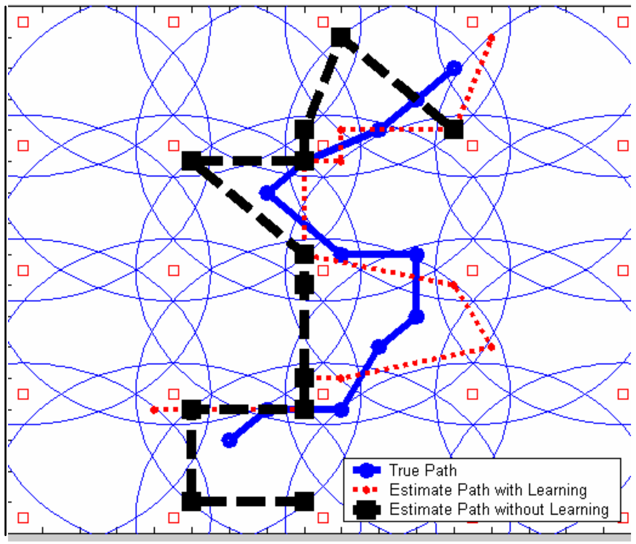

Figure 7: *The estimated path in case of increasing the number of states.*

The algorithm is still valid when the number of states increases. However, if the number of states is overwhelming, the time spent for running the algorithm is not ensure for real-time applications. The tester calculates the distant errors versus time steps as in Fig.8.

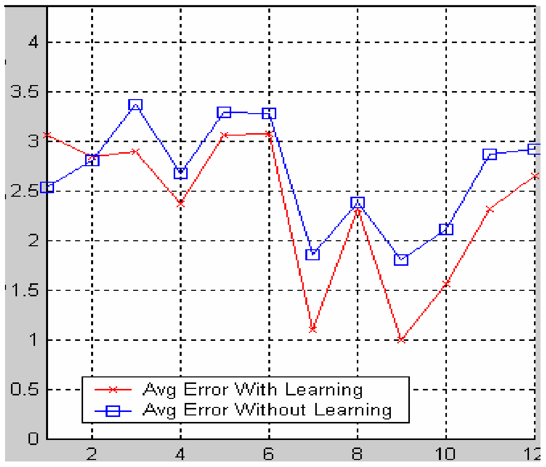

Figure 8: *The distant errors versus time steps*

Nearly, the estimated path without learning produces more distant errors than the learning one all time steps. The results indicate that the learning model, in other words a model with its parameters matching the surveillance area, is better than the non-learning algorithm.

VI. Conclusion

A method for tracking the target's trajectory is described in the paper. By dividing a path into discrete cells, it is suitable for applying discrete-HMM to solve the tracking problem.

The accurate observation sequences, the careful choice of model's parameters and the good algorithm are the principal criteria to contribute to successful results. The observation is extracted from available data of sensors. The data fusion is beyond this paper, but we assume that the observations repeat the true states with errors due to "non-line of sight" between target and sensors.

When an ergodic HMM (i.e. $a_{ij} \geq 0$, and every transition is possible) is used, it takes much time to run the algorithm. We can improve this fact by reducing the number of states (for example, take states at the corner of streets and omit states at the middle of them), preparing a sparse transition matrix (some parameters a_{ij} are set up to zeros). Once the processing time decreases , we can extract target track in real time.

The optimistic outputs of designed HMM are perspective. The ultimate aim is how we can put it into real tracking system with real data. The further researches would master how and where the sensors give optimal observations and set up a real geographical-dependent transition matrix.

Inferences:

[1] Ashley W.Stroupe, Martin C.Martin, and Tucker Balch, " *Distributed Sensor Fusion for Object Position Estimation by Multi- Robot Systems*", The Robotics Institute, Carnegie Mellon University.

[2] Trond Nypan, Oddvar Hallingstad, " *Cellular positioning by database comparision and hidden Markov model*", Unik – Uni. Graduate center.

[3] Andrew Adam, Saleema Amershi,"*Identifying humans by their walk and generating new motions using hidden Markov models*", CPSC 532A Topics in AI: Graphical models and CPSC 526 computer animation, December 15, 2004.

[4] Lawrence Rabiner and Biing- Hwang Juang," *Fundamentals of speech recognition*", Prentice – Hall International Editions.

[5] Chih-Chung Ke, “ *Literature survey on ground target tracking problem”*, Center for Multisource Information Fusion State University of New York, No. CMIF 3-99.

[6] Richard I.A. Davis, Chrisrian J. Walder and Brian C. Lowell, “ *Improved classification using hidden Markov averaging from multiple observation sequences”*, School of Information technology and Electrical engineering, the Uni. Of Queensland, Australia.

[7] H.Yang and B. Sikdar, “ *A Protocol for Tracking Mobile Targets using Sensor Networks”*, Department of Electrical, Computer and Systems Enginnring, Research Polytechnie Institute, Troy, NY 12180.

[8] R. Fraile and S.J. Maybank*,”Vehicle trajectory approximation and classification”*, Department of computer science, the University of reading Whiteknights, UK.