

A Hierarchical Position-Prediction Algorithm for Efficient Management of Resources in Cellular Networks

Tong Liu[†] Paramvir Bahl[‡] Imrich Chlamtac^{*}

[†] Tellabs Wireless Systems Division

[‡] Microsoft Research

^{*} University of Texas at Dallas

Abstract— We propose a novel hierarchical position-prediction algorithm which improves connection reliability and overall system performance by accurately predicting the future movement pattern of the mobile user. Our algorithm adopts a two-level approach – at the top (global) level, approximate pattern matching is applied to determine the mobile’s overall inter-cell movement direction, and at the bottom (local) level, an optimum self-learning Kalman estimator is applied that uses real-time signal strength measurements to estimate the mobile’s intra-cell movement direction and velocity. Simulation results show that this two-tier prediction algorithm promises to provide a high degree of prediction accuracy as it is robust in the presence of random movement patterns and noise corrupted measurement data.

I. INTRODUCTION

The management of channel resources in cellular communications presents a challenging set of problems for wireless network designers. One of the key problems within this set is mobility management. In existing systems, mobility management generally entails passive network involvement in keeping track of subscriber locations and in maintaining connections. No per user resources are pre-allocated and the probability of blocking is high as the subscriber moves between cells. In order to reduce the blocking probability, recently, the concept of predictive mobility management has been proposed which aggressively anticipates the movement behavior of subscribers. In [1], the author proposes to determine user location based on their quasi-deterministic mobility behavior, which can be represented as a set of movement patterns stored in a user profile. This method was further pursued by [2], in which a user’s moving behavior is modeled as repetitions of some elementary movement patterns, based on which a pattern-matching based mobile motion prediction algorithm is proposed to estimate the future location for a mobile user. The major drawback of these algorithms are their sensitivity to the randomness of user movement. As reported in [2], the prediction performance decreases linearly with the random factor, the proportion in choosing next cell the is due to pure uniformly random.

To provide prediction for all users with different mobility characteristic, an accurate model is needed to explore the regularity and rationality in the seemingly random movement. To achieve this, we propose to model the user’s quasi-deterministic inter-cell movement by editing her regular movement pattern with *inserting*, *deleting* and *changing* operations. Additionally, we propose to model the mobile’s micro-movement as a non-stationary process with its dynamic states being non-linearly related to a time correlated Gaussian process whose mean value behaves as a

semi-Markov process. Based on this model, we develop a set of low-complexity, recursive mobility prediction algorithms that can be implemented easily in real-time systems. Our algorithms are composed of a *approximate pattern matching* algorithm that extracts any existing regular movement pattern that might exist to predict the global inter-cell direction, and a *Extended Self-learning Kalman filter* that deals with “unclassifiable” random movement by tracking intra-cell trajectory and predicting the next-cell crossing. We study the performance of our methods in the presence of path loss, shadow fading and random user movement. Simulation results show that our algorithms are robust in the presence of noisy input, being able to predict the speed and direction-of-travel of the mobile user with a high degree of accuracy.

II. USER MOBILITY MODEL

The mobility model we proposed in this paper is aimed to minimize the uncertainty of user inter-cell mobility by exploring inter-cell moving intention and tracking intra-cell movement. To achieve this, we model a user’s movement in terms of a two level hierarchy. At the higher level is the *Global Mobility Model (GMM)* whose resolution is in terms of inter-cell movement, the cell sequence she may cross during the connection’s lifetime. At the lower level is the *Local Mobility Model (LMM)*, whose resolution is in terms of intra-cell movement modeled as a stochastic process with state variables (speed, direction and position) that vary dynamically with time.

A. Global Mobility Model

The Global Mobility Model, as shown in Figure 1-(a), is motivated by the fact that most mobile users exhibit some *regularity* in their daily movement, and such regularity can best be characterized by a number of *User Mobility Patterns* (or UMPs), recorded in a *profile* for each user and indexed by the most possible occurrence time. The UMPs proposed here are similar to the movement patterns in [1][2], but are more feasible, since we decrease UMPs’ sensitivity to small deviations of *User Actual Path (UAP)* while keeping the capacity in exploring inter-cell moving intention by *approximate pattern matching* (explained in section III). As a result, the number of UMPs necessary to cover a network can be greatly reduced, which in turn, dramatically reduce the computation time needed for pattern matching. To be specific, we model the regular movement of a mobile user as an *edited UMP* by allowing the following

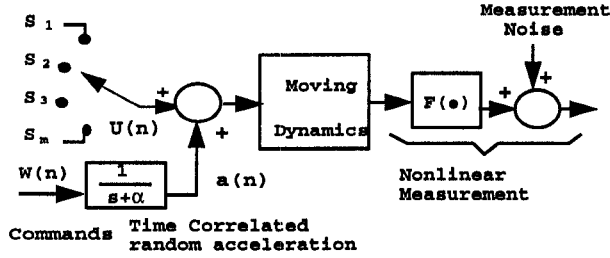
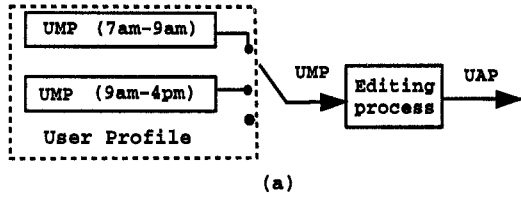


Fig. 1. (a) Global Mobility Model. (b) Local Mobility Model

legal operations, with $UMP = a_1 a_2 \dots a_{i-1} a_i a_{i+1} \dots a_n$.

- inserting a cell c at position i of the UMP gives UAP: $a_1 a_2 \dots a_{i-1} c a_i \dots a_n$.
- deleting the cell a_i at position i of the UMP gives UAP: $a_1 a_2 \dots a_{i-1} a_{i+1} \dots a_n$.
- changing a cell a_i to another cell c gives UAP: $a_1 a_2 \dots a_{i-1} c a_{i+1} \dots a_n$.

The degree of resemblance of UAP and UMP can be measured by the *edit distance*, a well-known metric of finite string comparison[3]. The simplest way to find this distance is the smallest number of *insertions*, *deletions* and *changes* by which the two finite cell sequences can be made alike. In order to reflect the geometric relationship of UAP and UMP in physical domain, we assign a non-negative number to each edit operation as the spatial weight. As a result, the *edit distance* between UMP and UAP becomes the total of the weights of the *editing* operations which are chosen not only so as to make the two cell sequences alike, but also to have the smallest possible total weight. If the *edit distance* is less than a matching threshold t , an approximately matched UMP is found, indicating the general moving intention of the user.

For large systems or systems with complex network topology, calculation of the spatial weight can be quite involved. For simple computation, we limit the temporary deviation of a mobile user within the neighboring cells of the edited UMP. To be specific, we define the weight as follows:

- The cost of *inserting* a cell c at position i :
$$W_{Ici} = \begin{cases} 1, & c \text{ is the adjacent cell of } a_i \text{ for } i = 1, \dots, n \\ \infty & \text{otherwise} \end{cases}$$
- The cost of *deleting* the cell a_i :
$$W_{Da_i} = \begin{cases} 0, & a_1, \dots, a_{i-1} \text{ have already been deleted} \\ 1 & \text{otherwise} \end{cases}$$
- The cost of *changing* a cell to another cell c at position i :
$$W_{Cci} = \begin{cases} 1, & c \text{ is the adjacent cell of } a_i \text{ for } i = 1, \dots, n \\ \infty & \text{otherwise} \end{cases}$$

B. Local Mobility Model

Our motivation for Local Mobility modeling is to further reduce the uncertainty of inter-cell mobility by tracking the user's intra-cell movement. The seemingly random choice of the next crossing cell is actually a logic function of the user's position, velocity, moving direction, and cell geometry. In order to track these time-varying values, we model the moving mechanism of a moving user as a linear, dynamical system, on which linear minimum mean-square estimator can be built in real time to estimate and predicate the dynamic states in two-dimensional Cartesian coordinates.

In real situations, a moving user has a wide acceleration range. Traffic light and turns of the road may cause abrupt changes of speed in x and y directions respectively. In order to follow such sudden changes quickly and in the mean time be able to keep tracking slow variations. As shown in Figure 1-(b), we model the driving input to the dynamic systems as a combination of a semi-Markov process U_n and a time-correlated random process $a(n)$. The states S_1, S_2, \dots, S_m are discrete levels selected to cover the whole acceleration range. Transition from one state to another corresponds to the dramatic change of moving behavior. Random acceleration $a(n)$ has Gaussian distribution, with zero mean and variance chosen to cover the "gap" between adjacent states. Such process can be generated by passing a white Gaussian process W_n to a one-pole shaping filter, $\frac{1}{s+\alpha}$ with α as the reciprocal of random acceleration time constant. This modeling method was once successfully applied in tactical weapon systems for maneuvering target tracking[4][5].

Based on this model, we can get the dynamic equation for a moving user:

$$X_{n+1} = AX_n + BU_n + W_n \quad (1)$$

where, $X_n = [x(n) \ v_x(n) \ a_x(n) \ y(n) \ v_y(n) \ a_y(n)]^T$ and, $U_n = \begin{bmatrix} u_x(n) \\ u_y(n) \end{bmatrix}$ $W_n = \begin{bmatrix} w_x(n) \\ w_y(n) \end{bmatrix}$

In the above, X_n represents the system state vector, with $x(n)$, $y(n)$, $v_x(n)$, $v_y(n)$, and $a_x(n)$, $a_y(n)$ as the position, speed and random acceleration in the x and y directions respectively. A and B are constant state and disturbance transition matrices. In the driving input vector U_n , $u_x(n)$ and $u_y(n)$ are independent semi-Markovian processes which may take m possible discrete values respectively s_1, s_2, \dots, s_m . W_n is discrete time Gaussian white noise with $w_x(n)$ and $w_y(n)$ uncorrelated.

In practical systems, observation of the dynamic states can be obtained from RSSI measurement. The measured value (in dB) is the sum of two terms, one due to path loss and the other due to shadow fading, which can be regarded as a zero-mean Gaussian random variable with standard deviation 4-8 dB[6]. Such measurement is non-linearly related to the dynamic state X_n and can be expressed by:

$$Z_n = h(X_n) + \xi_n \quad (2)$$

where Z_n is the measurement vector, and ξ_n is the shadowing component.

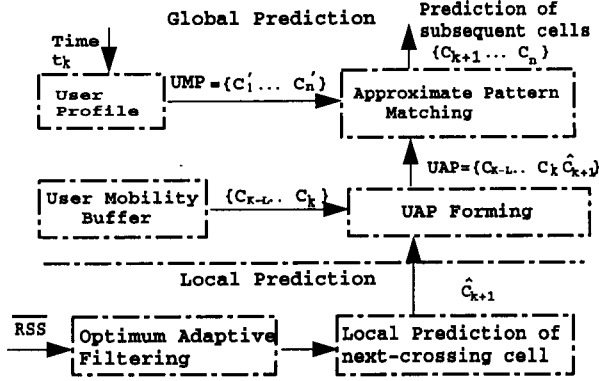


Fig. 2. Mobility Prediction

In order to track the mobile station in 2 dimensional domain, 3 independent measurement data are needed. Such measurement can be easily obtained in real systems, like GSM, in which a mobile station in active mode is able to measure and report to the down link signal strength of current cell and the best six neighboring cells every 480 ms. For this problem, we select the three largest measurements as valid observation. With Eq.1 and Eq.2 as the mathematical description of user's intra-cell movement, an *Extended Self-Learning Kalman filter* can be designed that adaptively tracks the user's trajectory and predicts her dynamic states in terms of speed and position.

III. USER MOBILITY PREDICTION

The algorithm proposed for User Mobility Prediction is summarized in Figure 2. As shown, and corresponding to our mobility modeling, user mobility prediction is carried out at two levels – Local Prediction (LP) and Global Prediction (GP). LP provides the best estimate of the next crossing cell based on trajectory tracking and cell geometry. GP is responsible for movement pattern identification. With LP, a high degree of accuracy for next-crossing cell prediction can be achieved without any assumption of user's mobility history. Moreover, UMP identification error can be greatly reduced, since GP is able to looking-ahead before make the decision about the matched pattern.

A. Global Prediction

In Global Prediction, we will solve the following: we are given a number of UMPs related to current time interval and a UAP which is composed of recent crossed cells in User Mobility Buff (UBF) and the LP of the next-crossing cell. Both UAP and UMP are finite length sequences, and we assume the memory length of UAP is less or equal to the length of UMP. The task is to find out that whether a UMP contains an particular interval that UAP most resembles in the sense that the edit distance is the minimum and less than matching threshold t . If such an approximately matched sequence is found, the remaining sequence in UMP

simply becomes the output of Global Prediction.

The basic solution of the problem is the following dynamic programming method[3]: Suppose that $UAP = a_1, \dots, a_m$, $UMP = b_1, \dots, b_n$, we can compute the edit distance $d(a_1 \dots a_m, b_1, \dots, b_n)$ by constructing an $(m+1) \times (n+1)$ matrix (d_{ij}) , defined by an recursive procedure:

$$d(0, j) = 0, \quad 0 \leq j \leq n$$

$$d(i, 0) = \begin{cases} i, & i = 0, 1 \\ d(i, 0) = \infty, & 0 \leq i \leq m \end{cases}$$

$$d_{ij} = \begin{cases} d_{i-1, j-1}, & \text{if } a_i = b_j \\ \min\{d_{i-1, j-1} + W_{Cij}, d_{i-1, j} + W_{Aij}, \\ d_{i, j-1} + W_{Dij}\} & \text{otherwise} \end{cases} \quad (3)$$

where $i = 0$ or $j = 0$ means $a_1 \dots a_i = 1$ or $b_1 \dots b_j = 1$, with 1 representing the empty sequence. As defined in section II, W_{Cij} is the weight of changing b_j to a_i , W_{Aij} is the weight of inserting cell a_i at position j , W_{Dij} is the weight of deleting b_j .

Matrix (d_{ij}) can be evaluated column-by-column in time $O(mn)$, whenever $d(m, j)$ is found to be at most t for some j , there is an approximate occurrence of UAP ending at j th cell, b_j , of the UMP with edit distance $d(m, j) \leq t$. If the edit distance is the minimum among all the approximately matched UMPs found, the remaining cell sequence b_{j+1}, \dots, b_n becomes the global prediction. An example is given in the simulation part to illustrate the operation of this procedure.

Based on the editing relationship between UAP and UMP, other matching criteria, like elapsed time matching [2] can further be applied if multiple UMP subsequences have similar *edit distance* from UAP.

B. Local Prediction

Observing the fact that the trajectory of a moving user is non-stationary, Kalman filter is the best candidate, which can be easily implemented as a simple software process. However, conventional Kalman filter cannot resolve our problem, because the deterministic input U_n is a semi-Markovian process with m possible states. Such hidden randomness requires a bank of m filters with each filter operating on a possible state. Fortunately, when certain practical assumptions are made, as discussed in [5], the filter bank can be reduced to a single Kalman filter augmented by a recursive technique of estimating U_n . The adaptive state estimator then becomes:

$$\hat{X}_{n+1} = A\hat{X}_n + B\hat{U}_n + K_{n+1}(Z_{n+1} - HA\hat{X}_n - HB\hat{U}_n) \quad (4)$$

$$\text{where } \hat{U}_n = \sum_{i=1}^n U_n(S_i)P(S_{n+1}^i/Z_{n+1}) \quad (5)$$

$$P(S_{n+1}^i/Z_{n+1}) = (\text{const})f(z_{n+1}/S_{n+1}^i, Z_n) \sum_{\alpha=1}^m \theta_{\alpha i}P(S_n^i/Z_n) \quad (6)$$

where the following are true:

1. Probability density function $f(z_{n+1}/S_{n+1}^i, Z_n)$ has a Gaussian distribution with mean $H_{n+1}AX_n(s_i) + H_{n+1}BU_n(s_i)$ and variance $H_{n+1}[AM_{n/n}A^T + Q_n]H_{n+1}^T + R_n$; $M_{n/n}$ is the state estimation matrix; R_n is the measurement error covariance matrix and Q_n is the Gaussian disturbance covariance matrix.
2. Probability $\theta_{\alpha i} = P(U_n = S_i | U_{n-1} = S_\alpha)$ is obtained from semi-Markov considerations. This parameter can be approximated by a value p near unity for $i = \alpha$ and $(1 - p)/(m - 1)$ for $i \neq \alpha$ for many tracking situations.
3. The constant (const) is evaluated from $\sum_{i=1}^m P(S_n^i/Z_n) = 1$.

Based on this result, we are able to complete the adaptive optimum filter to predict the dynamic states from the received signal strength measured at the mobile station. The resulting algorithm turns out to be very simple:

$$\text{Prediction : } X_{n+1/n} = AX_n/n + B\hat{U}_n \quad (7)$$

Minimum Prediction MSE Matrix:

$$M_{n+1/n} = AK_n/nA^T + Q \quad (8)$$

Kalman Gain Matrix:

$$K_{n+1} = M_{n+1/n}H_{n+1}^T[R_{n+1} + H_{n+1}M_{n+1/n}H_{n+1}] \quad (9)$$

Correction:

$$\hat{X}_{n+1/n+1} = \hat{X}_{n+1/n} + K_{n+1}[Z_{n+1} - h(\hat{X}_{n+1/n})] \quad (10)$$

Deterministic input update:

$$\hat{U}_n = \sum_{i=1}^n U_n(S_i)P(S_{n+1}^i/Z_{n+1}) \quad (11)$$

Minimum MSE matrix update:

$$M_{n+1/n+1} = [I - K_{n+1}H_{n+1}]M_{n+1/n} \quad (12)$$

$$\text{where } H_{n+1} = \left. \frac{\partial h}{\partial X_{n+1}} \right|_{X_{n+1}=\hat{X}_{n+1/n}} \quad (13)$$

$P(S_{n+1}^i/Z_{n+1})$ for $i = 1, \dots, m$ can be obtained using the recursive equation Eq.6.

Based on the estimation of dynamic states, cell-crossing probability $P(\text{Cell}_i), i = 1, \dots, 6$ can be calculated when the user moves close to the cell boundary, where we assume that no chance left for the user to make dramatic change. Variables involved in this calculation are the position $(x(n), y(n))$, moving direction θ and bearings of the cell vertex $\theta_i, i = 1, \dots, 6$. Moving direction, θ can be simply obtained from the velocity in x and y direction. Defining $f(X_n)$ as the probability density function (pdf) of the dynamic state X_n and $f(\theta/X_n)$ as the pdf of the moving direction given X_n , $P(\text{cell}_i)$ can be calculated by:

$$P(\text{cell}_i) = \int f(X_n)P(\text{Cell}_i/X_n)dX_n \quad (14)$$

$$\text{where } P(\text{Cell}_i/X_n) = \int_{\theta_i}^{\theta_{i+1}} f(\theta/X_n)d\theta \quad (15)$$

$$\text{with } i = 1, 2, \dots, 6 \quad \theta_7 = \theta_1$$

For unconstraint cell, we assume uniform distribution of X_n , then $P(\text{Cell}_i)$ is uniquely determined by $P(\text{Cell}_i/X_n)$. In this case the local prediction of the next crossing cell becomes:

$$\text{Next Cell} = \text{argmax}_i \{P(\text{Cell}_i/X_n)\} \quad i = 1, \dots, 6 \quad (16)$$

Parameters	Comments
$T = 0.5s$	Sampling interval
$\sigma_m^2 = 0.5m/s^2$	Variance of random acceleration
$ A_{\max} = 10m^2/s$	Maximum acceleration
$V \in [30, 60]$ miles/hr	Speed range
$1/\alpha = 10s$	Random acceleration constant
$\sigma_\xi = 5dB$	Standard deviation of lognormal shadowing
$p_0 = 20w$	Base station transmission power
$g_b = 6dB$	Power gain of base station
$g_m = 1dB$	Power gain of mobile station
λ	Wavelength of RF signal
$R = 4km$	Radiance of cell
$UMP_1 = c_2c_5c_8c_{12}c_{14}$	Movement pattern
$UMP_2 = c_2c_5c_4c_9c_{19}c_{18}c_{17}c_{16}$	Movement pattern
$UAP = c_1c_2c_5c_8c_9c_{10}c_{17}$	Actual path

IV. SIMULATION

To demonstrate prediction performance, a moving user is simulated in the conventional hexagon cell environment, who is able to move to any cell in the network along unknown trajectories with non-constant speed. According to her mobility history, Two movement patterns UMP_1 and UMP_2 are possible in the time-interval observed. On-line mobility related information are signal-strength measurement to different base stations and the recently crossed cell lsequence stored in UBF. Parameters involved in the simulation is summarized in the table above.

A. Local Prediction Result

The result of trajectory tracking is shown in figure 3, with the dashed curve as the actual trajectory, and solid curve indicating the predicted trajectory. Figure 4 demonstrates the result of time-varying velocity prediction. As we can see the proposed algorithm demonstrates a good estimation performance. The result of LP of the next crossing cell is summarized in the following table:

Current Cell	c_1	c_2	c_5	c_8	c_9	c_{10}	c_{17}
Predicted Cell	c_6	c_5	c_8	c_9	c_{10}	c_{17}	

In this case, the prediction accuracy rate is 83%. On analyzing prediction result of the next-crossing cell, together with users' trajectory tracking, we find that a high degree of prediction accuracy is achieved once the Kalman filter becomes stable, and the error caused by the initial instability is limited to the prediction of the first-crossing cell at the beginning of the journey. In the stable state, the prediction accuracy is related to the geometric relation of user trajectory and cell boundary, which can be compensated by UMP matching.

B. Global Prediction Result

Recursive *approximate pattern matching* algorithm is implemented with matching threshold t as 3 and UBF size equal to 4, which means GP output is available when the user reaches to the forth cell, c_8 . GP result along the user moving path is given in the following table

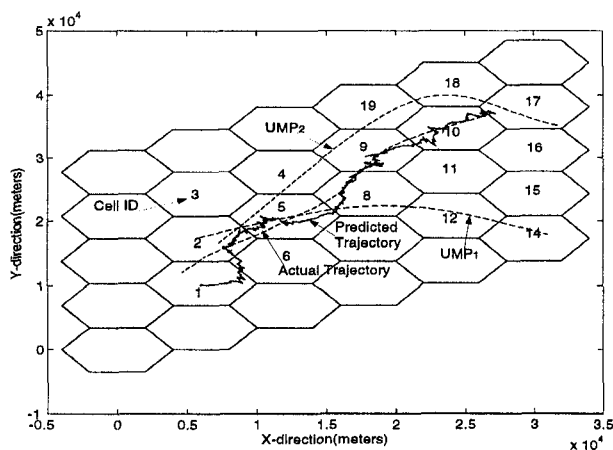


Fig. 3. Actual and predicted user trajectory

Current Cell	d_{UMP_1}	d_{UMP_2}	GP output
c_8	2	2	c_9
c_9	∞	2	$c_{10}c_{18}c_{17}c_{16}$
c_{10}	∞	3	$c_{17}c_{16}$

For cell c_8 , the *edit distance*, for both UMP_1 and UMP_2 , d_{UMP_1} and d_{UMP_2} equals to 2, since no pattern can be identified, GP only gives the prediction of next-crossing cell provided by LP. Figure 5 shows the matrices used for approximate pattern matching at cell c_9 to demonstrate the recursive procedure given in section III. Line segments in the matrices mark the direct arc in the *dependency graph*, going straight down or down at an angle or to the right. Notice that the path with minimum spatial weight may not be unique, which may result in different predictions. For example, in the matrix for UMP_2 , if \hat{c}_{10} is viewed as the result of *insertion*, c_{19} shall follow \hat{c}_{10} as part of the Global Prediction, otherwise, the cell after \hat{c}_{10} should be c_{18} , since \hat{c}_{10} is the result of *changing* c_{19} . In this case, we select the second possibility, which is reasonable, for c_{18} is closer to the destination.

It is also interesting to note that in this case, both UAP_1 and UAP_2 have two cells appeared in UAP , if the *state-matching* method [2] is used instead, UMP_2 can not be identified, since the UMP_1 and UMP_2 will have the same degree of similarity.

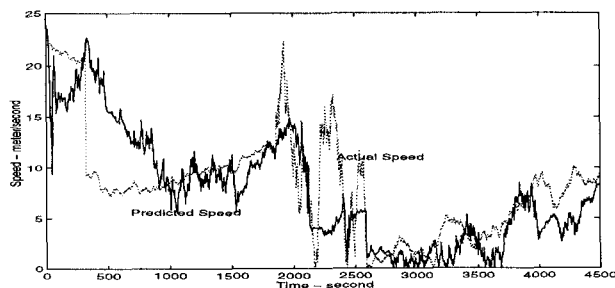


Fig. 4. Actual and predicted user speed

UMP_1	1	C_2	C_5	C_8	C_{12}	C_{14}
UAP	1	0	0	0	0	0
1	0	0	0	0	0	0
C_2	1	0	1	2	3	4
C_5	∞	1	0	1	2	3
C_8	∞	∞	1	0	1	2
C_9	∞	∞	∞	1	2	3
\hat{C}_{10}	∞	∞	∞	∞	∞	∞

UMP_2	1	C_2	C_5	C_4	C_9	C_{19}	C_{18}	C_{17}	C_{16}
UAP	1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0
C_2	1	0	1	2	3	4	5	6	7
C_5	∞	1	0	1	2	3	4	5	6
C_8	∞	∞	1	1	2	3	4	5	6
C_9	∞	∞	∞	2	1	2	3	4	5
\hat{C}_{10}	∞	∞	∞	∞	2	2	3	4	5

Fig. 5. Matrices used for *approximate pattern matching* for cell c_9 , with \hat{c}_{10} as the local prediction of the next-crossing cell.

V. CONCLUSION

In this paper, we recognize the fact that the performance of location tracking can be greatly improved by exploring the mobility information available from the mobile user and the network system structure superimposed on the user's current moving path. With this as motivation, we propose a hierarchical approach for user mobility modeling. Based on this model, a new location prediction strategy is developed which provides efficient prediction of next cell-crossing and subsequence cell sequence.

References

1. Sami Tabbane: "An Alternative Strategy for Location Tracking" IEEE Journal on Selected Areas in Communications Vol.13 No.5 June 1995
2. George Y. Liu and Gerald Q. Maguire Jr. "A predictive Mobility Management Algorithm for Wireless Mobile Computation and Communication" ICUPC'95
3. P.H.Sellers, "An algorithm for the distance between two finite sequences" J. of Algorithms 1, 1980, p. 359-373
4. R. A. Singer, "Estimating Optical Tracking Filter Performance for Manned Maneuvering Targets," IEEE Trans. on Aerospace and Electronic Systems, 1970
5. R. L. Moose, and H.F. Vanlandingham, "Modeling and Estimation for Tracking Maneuvering Targets," IEEE Trans. Aerospace and Electronic Systems, Vol. 15, No. 3, May 1979
6. Howard H. Xia, "An Analytical Model for Predicting Path Loss in Urban and Suburban Environments," PIRMC'96