2009 Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing

Comparison of User Trajectories Based on Coordinate Data and State Transitions

Junichi Oda and Ruck Thawonmas
Intelligent Computer Entertainment Lab,
ISE, Ritsumeikan University
Kusatsu, Shiga, 525-8577, Japan
ruck@ci.ritsumei.ac.jp

Kuan-Ta Chen
Multimedia Networking and Systems Lab
IIS, Academia Sinica
Taipei 115, Taiwan
swc@iis.sinica.edu.tw

Abstract—**At present, one can acquire a great deal of trajectory data as positioning technology has prevailed and become sophisticated. In accordance with this fact, data mining technology for extracting important information and knowledge from trajectory data is gaining recognition. Clustering of trajectories and giving meanings to movements are active areas of research. In this paper, we focus on the Markov chain model for representing trajectory data and propose a method for defining states that can cope with the distinguished characteristic of each trajectory and a method for comparing trajectories transitioning over different sets of states in the Markov chain model. From an experiment using real trajectory data, we confirm the advantage of the proposed approach over others.**

I. INTRODUCTION

In recent years, because positioning technology such as GPS and wireless network has become widely available, one can acquire moving objects' trajectories, such as those of people or vehicles, easily. In accordance with this fact, data mining technology for extracting important information and knowledge from trajectory data is gaining recognition. Clustering of trajectories, extracting features or relationship of trajectories, and approximating time-series data are active areas of research. This type of spatial-temporal data mining is applicable to not only the area of movement locus analysis but also to many other areas such as clustering of object shapes and images.

Important issues in the study of trajectories of mobile objects include (i) determining the distance or similarity among trajectories, (ii) clustering the trajectories based on distance or similarity indices, (iii) finding the meaning in each mobile object movement. A main objective of this study is to enable understanding the current status of a mobile object of interest. Another objective is to predict the next action or position of a given mobile object. Our research findings have wide applicability. If this study is applied in order to find a set of basic motions of human, an architect can design a building based on this set. In home security, one can find a suspicious individual whose trajectory is too different from a pre-derived set of basic motions.

In the area of movement trajectory analysis, a number of methods were previously proposed that express and compare trajectories by state transitions. They divide target space into multiple areas and define these areas as states. Each trajectory is then expressed by a set of transitions between these common states. However, this approach has two problems: (1) some trajectories are overly approximated and (2) small movements in a given state cannot be expressed exactly. Although these problems can be solved by increasing the number of states, this leads to an increase in computational cost and thus degrades performance. We argue that the target space should be divided into suitable states for each trajectory and a method is needed to compare trajectories modeled by transitions in different state sets.

In this paper we propose a method for defining states that can cope with the distinguished characteristic of each trajectory. The proposed method adopts dynamic space division based on the quadtree representation. In addition, we propose a method for comparing user trajectories that uses two clustering steps: one based on coordinate data and the other based on state transitions. The former clustering step uses the Hamming distance between a pair of quadtrees, each being represented as a bit sequence, as distance measure. Resulting clusters are used in the latter clustering step. In each cluster, a Markov chain model for each of its trajectories is built, and the latter clustering is performed for these trajectories based on a proposed distance measure. Dynamic space division enables elucidation of trajectory details and prevention of trajectory over-approximation. In addition, two-step clustering leads to less computational costs and easier interpretation of clustering results.

II. RELATED WORK

In order to compare player trajectories in Massive Multiplayer Online Game (MMOG), the following method was proposed in [1], as an improved version of the method in [2]. First, a quadtree is generated based on states derived through dynamic space division using all players' trajectories. Landmarks in the given map are then defined based on the density of coordinate data of all trajectories. Each trajectory is approximated by the state transition probabilities between landmarks. Finally, such transition probabilities of each pair of trajectories are compared and used for clustering players. The speed of this method is fast, but the method might overly

978-0-7695-3762-7/09 $26.00 © 2009 IEEE
DOI 10.1109/IIH-MSP.2009.288

approximate trajectories. In addition, when players exist whose trajectories are errant, this method loses accuracy.

In the area of mobile-object tracking, the work in [3] adopted the Markov chain model for modeling the mobility statistic. In order to conduct multidimensional analysis, data-cube like logical representation of mobility histogram was introduced. Under this representation, each state is represented by a unique number based on Z-ordering. Binary representation for these unique numbers is used to represent a state transition of interest, and adjustment when comparing transitions among states of different depths, drill-down or roll-up, is performed. To build the histogram, a tree structure was introduced. Adaptive extension of the tree is implemented, which enables memory decrement. However, time for building the histogram can be a bottleneck when a great number of mobility objects are monitored.

The work in [4] also proposed a comparing method based on the Markov chain model. This work decides states manually according to the layout of a real store. It aims at detection of outliers. In order to detect a person who has an errant trajectory, a detection method was proposed which has two steps. First, each trajectory is modeled by transitions between Markov chain states, and probabilistic distances between trajectories are evaluated. Users who have high distances to the others are considered outliers and are removed. Remaining trajectories are projected into lower dimensional space by multidimensional scaling. In this new space, trajectories are clustered, and outliers are additionally detected using the likelihood of trajectories. K-means is used for the clustering task where the initial centroids are those minimizing the distortion within clusters. For determining the number of optimal clusters, the rating index proposed in [5] is used. However, multidimensional scaling makes it difficult to locate reasons behind results.

III. Proposed Method

In [1-4], in order to use the Markov chain model, a common set of states is defined based on all trajectories. In general, important areas and the number of coordinate data at each divided area are different for each trajectory. To cope with these issues, we argue not to use a common set of states for all trajectories but to generate and use an individual state set for each trajectory. However, comparison of trajectories modeled by different state sets is difficult. Thereby, we propose to adopt the roll-up mechanism in [3] for this task.

A. Dynamic map division based on quadtree

For each trajectory, the map is divided from trajectory data distribution. Let D indicate the level of division. First, the initial state ($D = 0$) is divided into four areas. For each area, its data density is evaluated, and it will be further divided into four areas if the density is higher than a given threshold (Fig. 1).

In the above operation, dynamic map division is implemented by the quadtree, where a node represents an area. The initial state is the root of the tree with node number = 0, then extension of the tree corresponds to map division. If there are some nodes that have no trajectory, all such nodes will be deleted (Fig. 2).

B. Distance measure for quadtrees

Each node in a quadtree is assigned a unique number based on Z-ordering. Because Z-ordering is used by all trajectories, a same node number indicates the same area.

Comparing a pair of quadtrees is done through comparing bit sequences that represent the quadtrees. The n-th bit indicates existence or nonexistence of the n-th node. If the n-th bit is 1, the n-th node exists; otherwise, the n-th node does not exist (Fig. 3).

In Fig. 4, $Dist_{AB}$ shows the distance between two quadtrees A and B and is defined as the Hamming distance between the corresponding two bit sequences. This distance is used in the 1st-step clustering as an element in the $(N-1)\times(N-1)$ distance matrix, where N is the number of trajectories.

C. 1st-Step clustering based on coordinate data

Here, all trajectories are clustered with the Ward method [6] based on the aforementioned distance matrix. The number of clusters is decided with the rating index introduced in [5]. Any cluster with the number of members less than ten percent of N is excluded because we consider them outliers.

D. 2nd-Step clustering based on state transitions

In each cluster, time-series trajectory data are changed into state-transition data based on the Markov chain model, where a leaf node represents a state. Because quadtrees are different from one trajectory to another, the positions of non-empty elements in the state-transition matrices are also different. To compare a given pair of different quadtrees, we need to derive a common structure from them. This can be achieved by logical multiplication of the corresponding two bit sequences (Fig. 5).

Deriving the common tree structure of quadtrees leads to degradation of trajectory representation. Thereby, all related transition probabilities have to be adjusted. This adjustment corresponds to the roll-up algorithm in [3]. The transition probabilities related to a common node of interest are adjusted as follows:

i) In the case where both the source and the destination of the transition are a common node resulting from roll-up,

$$a_{l'l'} = \frac{1}{A(l')} \sum_{i=1}^{A(l')} \sum_{j=1}^{A(l')} a_{l'(i)l'(j)} \tag{1}$$

ii) In the case where only the source of the transition is a common node resulting from roll-up,

$$a_{l'm} = \frac{1}{A(l')} \sum_{i=1}^{A(l')} a_{l'(i)m} \tag{2}$$

iii) In the case where only the destination of the transition is a common node resulting from roll-up,

$$a_{ml'} = \sum_{i=1}^{A(l')} a_{ml'(i)} \,, \tag{3}$$

where a_{xy} is the transition probability from state x to y, l' indicates the common node resulting from roll-up, $A(l')$ contains the indices to all unified nodes involving in calculation of the transition probability to or from l'. In Fig. 5 if l' is 3, $A(3)$ contains 15, 16, 56, 57 and 58. As a result of roll-up, an intermediate node might become a state. For example, in Fig. 5, intermediate nodes 4 and 8 become states after roll-up.

Next, the distance between two trajectories after roll-up is defined as the average in the differences in the elements of the corresponding transition matrices. This distance is used as an element of the distance matrix in the 2nd-step clustering. Because the division level is different from area to area and some areas might have lost the detail, we need to give weights to the transition probabilities in accordance with D as follows:

$$w_l = \frac{D_{\max} - D_l + 1}{D_{\max}}, \tag{4}$$

where D_l is the division level of area l, and D_{max} is the maximum level of D. The aforementioned distance between trajectories i and j is given as

$$Dist_{ij} = \frac{1}{h_{num}} \sum_{l=1}^{h_{num}} \sum_{m=1}^{h_{num}} w_l w_m \left| a_{lm}(i) - a_{lm}(j) \right|, \tag{5}$$

where $a_{lm}(x)$ is the transition probability from states l to m in trajectory x, and h_{num} indicates the number of common states between i and j. If D_{max} after roll-up is 0, the distance between these particular two trajectories will be given the maximum possible value of (5), i.e., 2.

After obtaining the distance matrix for a cluster of interest, the trajectories in this cluster will be further clustered using the same method as described in Ⅲ.C.

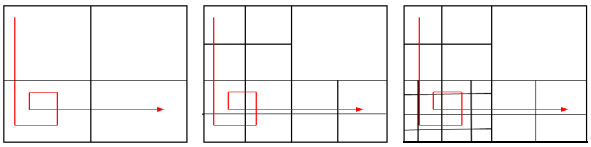

Figure 1 Dynamic map division from trajectory data distribution. From left to right D=1, D=2, D=3.

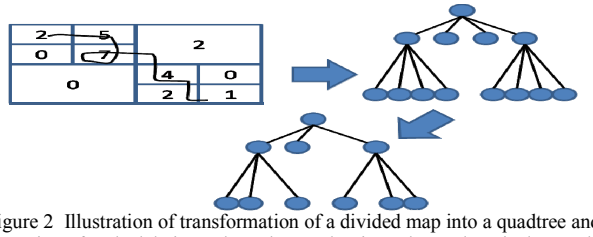

Figure 2 Illustration of transformation of a divided map into a quadtree and illustration of node deletion, where the number in each area here is the number of coordinate data in the trajectory.

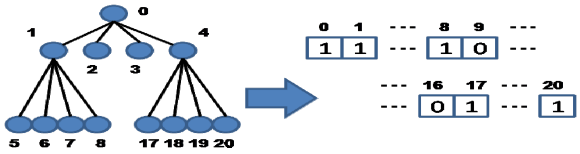

Figure 3 From a quadtree to a bit sequence.

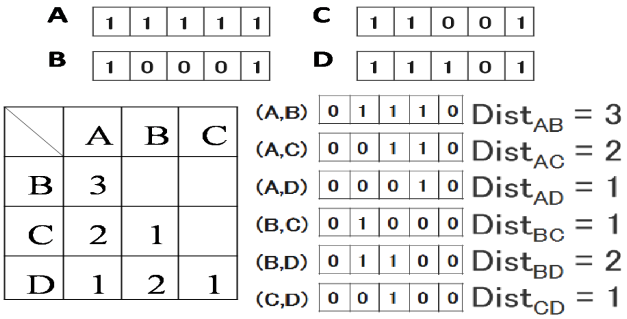

Figure 4 Example of the distance matrix: N=4.

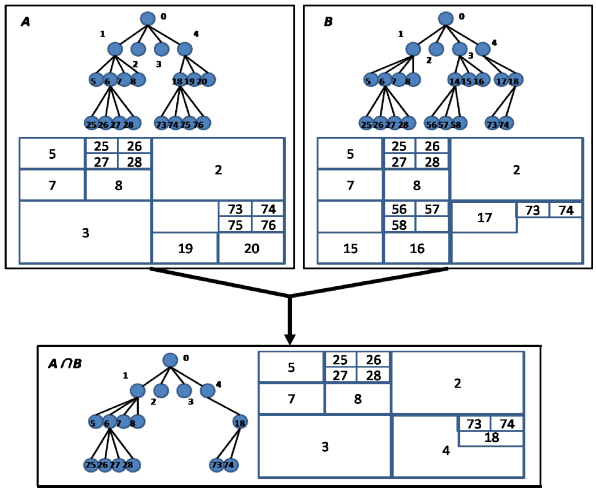

Figure 5 Illustration of derivation of a common tree structure for two different quadtrees A and B, where the node number is shown in each area. The area with no number does not contain any coordinate data in the trajectory.

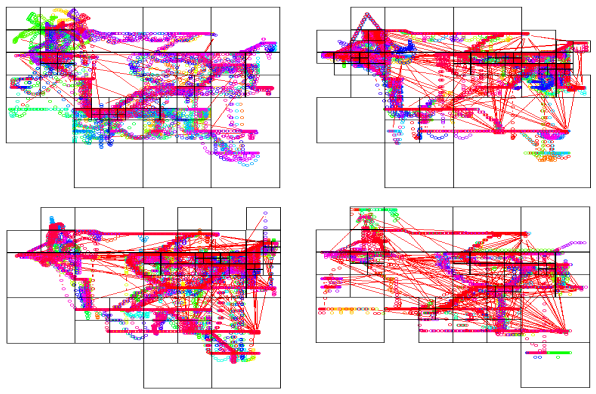

Figure 6 Examples of trajectories for players and bots (Top-left: player, Top-right: Crbot, Bottom-left: Ice , Bottom-right: Eraser).

IV. EVALUATION

We conducted an experiment on a trajectory data set of a First-Person Shooting game called Quake Ⅱ. This set of data [7] consists of movement data from human players and three kinds of bots (Fig. 6). The number of trajectories is 171 consisting of 105 players and 66 bots of three types: 25 Crbots, 21 Erasers and 20 Ices. To evaluate the effectiveness of our approach, three other methods are used. One is dynamic map division based on all trajectories data that divides a map of interest according to all trajectories data distribution, considers each divided area as a state, and computes transition probabilities between states for each trajectory [1]. The other two methods are the one that clusters the trajectory data with only the first-step clustering (III.C) and the one with only the second-step clustering (III.D), respectively. In our experiment, we adjusted Dmax to 5 and the threshold for dividing map to 5%.

For showing the quality of derived clusters, we show the entropy of player types summed for all clusters in Table 1. The above results show that the entropy of our method is 0, indicating no existence of mixed player types in any derived cluster. In addition, use of each of the other methods cannot correctly divide players. Although not shown here, dynamic map division based on all trajectories correctly divided players and bots, but could not correctly cluster bots into proper types.

Table 2 shows the detail of the result from our method for both steps. The 1st column is the cluster number of the 1st-step clustering, and the 2nd column shows the cluster number of the 2nd-step clustering. C, E and I stand for Crbot, Eraser and Ice bots, respectively. H(k) shows the entropy of cluster k. The 4th cluster of the 1st-step clustering includes very different trajectories from others. Because the number of these trajectories is lower than ten percent of the number of all trajectories, these trajectories were not considered in the 2nd-step clustering. Note that the 2nd-step clustering contributes to correct clustering of data according to player types.

V. CONCLUSIONS AND FUTURE WORK

The above results show that our method correctly clusters and classifies trajectories into human players as well as Crbot, Eraser and Ice Bots. Although dynamic map division based on all trajectories could roughly classify the trajectories, it could not categorize bots into each type. In our method, the 1st-step clustering roughly classified the trajectories, and the 2nd-step clustering finely classified them. Note that use of the 1st-step clustering only cannot correctly divide players and that use of the 2nd-step clustering only will take much longer time.

Movement speed can also be derived from trajectory data in order to find the detail meaning in each mobile object movement. In future, we plan to incorporate speed information into our method and to extend the method to three-dimensional space. We also intend to experiment with many other kinds of data, such as traffic data, real human trajectories, MMOG player movements.

ACKOWLEDGEMENTS

This work was supported in part by Japan Society for Promotion of Science (JSPS) under the Grant-in-Aid for Scientific Research (C) 20500146 as well as by the National Science Council under the grant NSC97-2221-E-001-009 (Taiwan).

REFERENCES

[1] Masayoshi Kurashig, "Determination of Landmarks and Player Clustering in MMOG," Master Thesis, Graduate School of Science and Engineering, Ritsumeikan University, March, 2008.

[2] Ruck Thawonmas, Masayoshi Kurashige and Kuan-Ta Chen, "Detection of Landmarks for Clustering of Online-Game Players," International Journal of Virtual Reality, vol. 6, no. 3, pp. 11-16, 2007.

[3] Yoshiharu Ishikawa, Yoji Machida, and Hiroyuki Kitagawa, "A Dynamic Mobility Histogram Construction Method Based on Markov Chains," Proc. 18th International Conference on Scientific and Statistical Database Management (SSDBM 2006), pp.359-368, 2006.

[4] Naohiko Suzuki, Kosuke Hirasawa, Kenichi Tanaka, Yoshinori Kobayashi, Yoichi Sato, and Yozo Fujino, "Learning Motion Patterns and Anomaly Detection by Human Trajectory Analysis," Proc. IEEE Int. Conf. Systems, Man and Cybernetics (SMC2007) , pp. 498-503, 2007.

[5] T. Calinski and J. Harabasz, "A dendrite method for cluster analysis," Communication in Statistics, Vol.3, pp.1-27, 1974.

[6] J.H. Ward, "Hierarchical Grouping to optimize an objective function," Journal of American Statistical Association, 58(301), pp. 236-244, 1963.

[7] Kuan-Ta Chen, Hsing-Kuo Kenneth Pao, and Hong-Chung Chang, "Game Bot Identification based on Manifold Learning," Proceedings of ACM NetGames 2008, 2008.

TABLE 1 Result of four clustering methods

Method	Entropy
Dynamic map division for all trajectory data	1.58
1st-step clustering	1.56
2nd-step clustering	1.59
Our Method	0

TABLE 2 Detail of clustering results.

		Number of members				$H(k)$	
1st	2nd	*player*	*C*	*E*	*I*	1st	2nd
1	1	27	0	0	0	0	0
	2	9	0	0	0		0
2	1	9	0	0	0	0	0
	2	31	0	0	0		0
3	1	9	0	0	0	0.83	0
	2	0	25	0	0		0
4	1	15	0	0	0	0	0
5	1	5	0	0	0	0.72	0
	2	0	0	0	20		0
6	1	0	0	13	0	0	0
	2	0	0	8	0		0