Dynamic Clusters Graph for Detecting Moving Targets using WSNs

[1]Farzaneh R. Armaghani, [1]Iqbal Gondal, [1]Joarder Kamruzzaman, [2]David G. Green

[1]Gippsland School of Information Technology, [2]Clayton School of Information Technology
Monash University, Australia
{Farzaneh.Armaghani, Iqbal.Gondal, Joarder.Kamruzzaman, David.Green}@monash.edu

Abstract— **Efficient target tracking applications require active sensor nodes to track a cluster of moving targets. Clustering could lead to significant cost improvement as compared to tracking individual targets. This paper presents accurate clustering of targets for both coherent and incoherent movement patterns. We propose a novel clustering algorithm that utilises an implicit dynamic time frame to assess the relational history of targets in creating a weighted graph of connected components. The proposed algorithm employs key features of localisation algorithms in target tracking, namely, estimated current and predicted locations to determine the relational directions and distances of moving targets. Our simulation results show a significant improvement on the clustering accuracy and computation time by dynamically adjusting the history-window size and predicting the relationships among targets.**

Keywords- Clustering Algorithms; Target Tracking; Wireless Sensor Networks, Weighted Graph

I. INTRODUCTION

Tracking multiple targets is a crucial task of Wireless Sensor Networks (WSNs), especially when they are deployed for military purposes. In many such applications, targets rarely act individually, but in contrast they coordinate a collaborative movement toward one or more destinations. Likewise, in order to obtain their goals malicious targets sometimes attempt to deceive the surveillance system by displaying incoherent movement patterns. Therefore, they could join or leave a team at any time, or move in a manoeuvring order, for example, a group of aircrafts conducting a mission towards a destination. In these scenarios, the high cost of tracking targets individually could be significantly reduced by estimating the groups and then tracking the groups of moving targets. In this paper, we aim to investigate a clustering strategy for the collaborative targets.

Typical clustering techniques treat the moving objects as static datasets at each time instant and then relate the created clusters within certain time intervals [1-5]. For example, the work in [3] considers the spatial-temporal pattern among the moving objects in the form of *Moving Clusters* (MC). MC employs the traditional static DBSCAN algorithm [6] to find a set of compact clusters at a time. The clusters are then correlated with the clusters in adjacent time intervals to create a moving set of clusters. The techniques in this category suffer from lower accuracy when different clusters overlap. Moreover, measuring compactness is not sufficient to capture the changes in spatial-temporal datasets where objects coordinate collaboratively.

On the other hand, in trajectory mining techniques [7-11], clusters are created based on the historical knowledge of objects' locations within a specified window of time. Such window indicates a time frame of the past locations of the objects. A general assumption is to consider a fixed time interval as the size of history-window to find the strongly related objects. Object density [8-10] is commonly used as the clustering metric. Introduction of history can improve accuracy in forming clusters; however, constructing such data structure requires storing the past locations of the objects and as a result, such clustering algorithms could have high complexity. To reduce the complexity and increase the accuracy, in *Dynamic Density Based Clustering* (DDBC) [12], instead of storing all the past locations of objects, an implicit history-window by means of a weighted relationship graph is proposed. The relationship graph contains one vertex per object; and relationships between objects are represented as weighted edges. DDBC uses the distance between two objects to estimate the likelihood that objects are related over the time window interval. Therefore, the relationships are maintained without storing the past locations of the objects and by just updating the likelihood at each time. DDBC uses a fixed window size in assessing the relational history of targets; however, a dynamic window size can reduce the complexity by ignoring those timeslots that do not contribute in improving clustering accuracy. Moreover, measuring just the distance could lead to an algorithm's failure in detecting overlapped clusters and a large time window is needed to assess the relationships.

We propose a new real-time clustering algorithm: *Predictive-Clustering of Moving Targets* (PCMT) by adopting the *weighted relationship graph* from DDBC to find clusters of targets and their constituent members. PCMT employs the predicted locations of targets derived from *localisation algorithms*. This is to determine the *distance* as well as the *movement directions* of targets to estimate the relationship likelihoods on real-time basis as targets traverse the network. Movement directions of targets contribute in finding the overlapped groups as well as finding the groups that are not dense enough but still move together. Unlike DDBC, our approach attempts to dynamically set the history-window size leading to significant reduction in complexity. Moreover, we assign a credit value to each timeslot within the history-window indicating a fading memory where the recent records

978-1-4673-1881-5/12/$31.00 ©2012 IEEE

are more important. Credits are used in estimating the graph weights. Our results show a significant reduction in time required to form the clusters and improvement in detecting accurate clusters as compared to existing algorithms.

This paper is organised as follows. Section II describes our proposed algorithm and simulation results are presented in Section III. Finally, section IV concludes the paper. In summary, main contributions are as following:

1. Development of weighted graphs, representing the relationships between targets in proposed clustering criterion which incorporates predicted relative-distances and movement-directions of targets.
2. Development of a history-window of dynamic size.
3. Incorporation of credit values in weights estimation.

II. Predictive Clustering of Moving Targets

We consider a large number of sensor nodes are randomly deployed in a rectangular region to track moving targets and a central server maintains the whole network. Targets move in various groups either in a coherent or incoherent manner. In an incoherent movement, they may move in a deceitful order to be perceived as individuals. Sensor nodes collect the measurements from the detected targets and transmit them to the central server; the central server then employs a localisation algorithm to estimate the locations. Targets can be represented by a 4-tuple $\left(i, \hat{x}_{i,t}, \hat{x}^p_{i,t+1}, t\right)$, where $\hat{x}_{i,t}$ and $\hat{x}^p_{i,t+1}$ are the estimated current and predicted locations of target i at time t, respectively. Finally, the localisation results are utilised by the clustering algorithm PCMT to find various clusters of targets. Fig. 1 shows the hierarchy of underlying algorithms to accomplish the clustering.

We model our clustering algorithm based on the weighted graph concept whose vertex consist of estimated targets, in which a sliding history-window for each pair of connected targets is required to determine the edge weight. A positive non-zero weighted edge between two targets represents an existence of relationship at one or more times within the history-window. The following sections explain different phases of the clustering algorithm.

A. Instant Relationship Assignment

The first phase of our algorithm is to estimate the likelihood that each pair of targets is related. The likelihood estimation is done by measuring the relative distance as well as the moving directions.

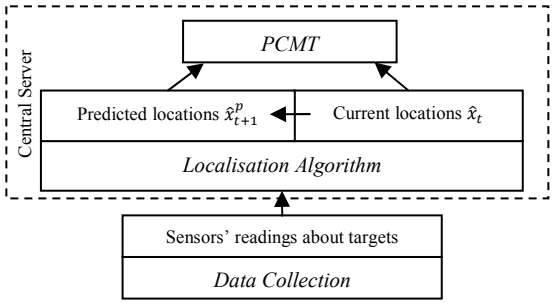

Fig. 1: Hierarchy of algorithms' flow to accomplish clustering

Assuming there are m estimated targets moving in the region at time t, the current and predicted locations of the targets calculated using a localisation algorithm are $\hat{x}_{1,t}, \hat{x}_{2,t}, \dots, \hat{x}_{m,t}$ and $\hat{x}^p_{1,t+1}, \hat{x}^p_{2,t+1}, \dots, \hat{x}^p_{m,t+1}$, respectively. Suppose $\vec{x}_{i,t} = \hat{x}^p_{i,t+1} - \hat{x}_{i,t}$, and $\vec{x}_{j,t} = \hat{x}^p_{j,t+1} - \hat{x}_{j,t}$ represent the location vectors of target i and target j with the same initial points, the relative moving direction can be expressed as the angle between their location vectors:

$$\theta_t(i,j) = \arccos \frac{\vec{x}_{i,t} \cdot \vec{x}_{j,t}}{|\vec{x}_{i,t}||\vec{x}_{j,t}|} \tag{1}$$

Subsequently, the instant relationship assignments between two targets i and j at current time t can be determined as:

$$f_t(i,j) = \begin{cases} 1, & \theta_t(i,j) \le \theta_{\text{th}} \;\&\; d_t(i,j) \le d_{\text{th}} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

where $d_t(i,j)$ is the Euclidean distance between the predicted locations of two targets, and θ_{th} and d_{th} are the threshold values that are dictated by the application. An edge with positive weight exists between targets i and j if they are instantly related to each other.

B. Relationship Strength

Consider a situation where the group members are conducting an incoherent movement, or multiple groups are overlapping. Then clusters formation would not be accurate with consideration of relationship likelihoods at the current time only. Estimation of the weight for an edge between each pair of targets is preceded by applying a sliding history-window structure. This history-window denotes a time period h to monitor the instant relationship assignment records. Each record is a timeslot within h that is associated with a *credit* value $C_{k\in[1,h]}$. Credits express a fading memory where the recent, rather than the older, records have a greater effect on calculation of the weights. A calculated weight with a value in unit interval represents the level of relationship strength between two targets. The weight value of 1 represents the strongest relationship. Let $(t-h, t]$ be the time interval for a history-window of size h, the relationship strength between targets i and j is given by:

$$W_t(i,j) = \frac{\sum_{k=1:h} f_{t-k+1}(i,j) \times C_k}{\sum_{k=1:h} k} \tag{3}$$

where $C_k = h - k + 1$, and $\sum_{k=1:h} k$ is a normalisation factor.

C. Clusters Formation

The last phase of the PCMT concentrates on finding the clusters using modified Tarjan's algorithm [13]. The variation in Tarjan's algorithm is in determining adjacent vertex. A vertex i is said to be adjacent to another vertex j if the relationship graph contains an edge whose weight is more than a threshold, i.e., $w_t(i,j) \ge \alpha$. The threshold value α is the requirement of application in assessing clusters with different

relationship strengths. Output of this phase includes sub-graphs of connected targets, whereby each sub-graph is representing a cluster.

To improve the time complexity of the algorithm, *dynamic* history-window size is considered. Initially, the window size is set to an upper-bound h_{max} when a non-zero weight is attained for a pair of targets right after a zero weight at previous time instant. This considers all the timeslots in the interval $(t - h_{max}, t]$ in the weight estimation at a time t. Once the two targets are decided as a cluster, the window size minimises to only considering the current timeslot. The window size stays on its minimum size as long these two targets remain in a common cluster; otherwise it incrementally grows to the upper-bound h_{max}. No history-window is considered for the targets whose edge's weight is zero. Fig. 2 explains an example of the dynamic setting of the history-window for two targets at six subsequent time instances with $h_{max} = 5$. Algorithm 1 explains the pseudo-code of the PCMT algorithm.

III. Performance Evaluation

Matlab simulations have been conducted to evaluate the performance of our proposed clustering algorithm (PCMT) for different values of: h_{max} value, number of generated clusters and constituent targets. We compared our algorithm with the existing DDBC [12] and MC [3] algorithms. DDBC employs a fixed history-window to generate a graph of connected components, whereas MC finds a moving set of clusters by investigating the static behaviour of objects at each time step. Next, we introduce the simulation settings followed by the results.

A. Simulation Settings

Simulation scenarios consisted of 200 randomly deployed stationary sensors within a 1 × 1 km^2 square region. Due to the lack of appropriate real moving-targets dataset, we developed a synthetic targets' trajectory generator which accepts several input parameters including number of real clusters (n), location variance (δ), noise variance (Q), and the transition matrix for movement patterns (A). Depending on the simulation scenario, different values were considered for input parameters in our generator. Initial positions of clusters' centres were randomly distributed in the region. Each cluster contains a random number of member targets varying from 2 to 20.

Algorithm 1: PCMT

Input: localisation algorithm's output, θ_{th}, d_{th}, α

for every target pair $\{(i,j)|i,j \in \{1,2,\dots,m\} \,\&\, i \neq j\}$
compute $\vec{x}_{i,t}$, $\vec{x}_{j,t}$, $\theta_t(i,j)$, $d_t(i,j)$, and $f_t(i,j)$
if instant relationship assignment is one $f_t(i,j) = 1$
  if history-window size is zero $h(i,j) = 0$
    $h(i,j) = h_{max}$
compute the weight $W_t(i,j)$
if targets form a cluster, $W_t(i,j) \geq \alpha$
  minimise the history-window size, $h(i,j) = 1$
else if a positive weight is attained $0 < W_t(i,j) < \alpha$, and $h < h_{max}$
  $h(i,j) = h(i,j) + 1$
run the *Tarjan's* algorithm to find sub graphs with $W_t \geq \alpha$

Output: set of clusters

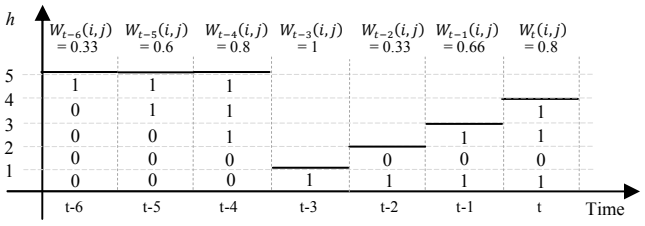

Fig. 2: An example of dynamic setting of the history-window size for targets *i* and *j* at different time instances for $h_{max} = 5$ and $\alpha = 0.7$.

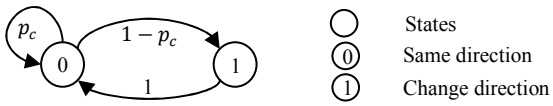

Fig. 3: Two states Markov Chain trajectory model for incoherent targets

Initial location of a target satisfied the Gaussian distribution as $x_{init} \sim N(x_{centre}^{cluster}, \delta)$. Velocities of every cluster were chosen randomly and the members of a cluster move according to that velocity.

We evaluated both coherent and incoherent movement patterns. In case of coherent movement, members of a cluster stayed together as they moved in the same direction toward a destination according to a linear function as $x_t = A_{t-1}x_{t-1} + q_{t-1}$, where A is a transition matrix, and q is a zero-mean Gaussian process noise with variance Q. In case of incoherent movement, a two states Markov Chain location model was considered as depicted in Fig. 3. In this model members of a group stayed together and continued to move along the same direction with probability p_c and changed direction with probability 1-p_c. Targets will get back again to the same direction at next time step inducing a manoeuvring movement. We used the two states Markov Chan model in the trajectory linear function. The generated trajectories were used as an input to localisation algorithm for locations estimation.

We used the Rao-Blackwellized particle filtering [14] algorithm as the localisation algorithm. Note that PCMT does not rely on any particular underlying localisation algorithm; rather any algorithm could be used. Finally, each data point in the result represents an average of 1000 simulation runs. Table 1 offers an overview of the parameters used in the simulations.

B. Performance Metrics

We evaluated our algorithm in terms of *CPU time* and *clustering error*. Clustering error is defined as the total error in similarity between the set of clusters estimated by the clustering algorithm $\hat{G}_t$ and the generated set of actual clusters by the generator G_t over simulation time T [12, 15] and is defined as:

$$clustering\_error = \sum_{t=1:T} 1 - \frac{\sum_{a=1:|\hat{G}_t|} \max_{1 \leq b \leq |G_t|} sim\left(\hat{G}_t^a, G_t^b\right)}{max(|G_t|, |\hat{G}_t|)}$$

$$sim\left(\hat{G}_t^i, G_t^j\right) = \frac{\left|\hat{G}_t^a \cap G_t^b\right|}{\left|\hat{G}_t^a \cup G_t^b\right|}$$

where $|G_t|$ and $\left|\hat{G}_t\right|$ denote the number of true and estimated clusters at time *t*, respectively.

Table 1: Parameters and their settings

Parameter	Setting
Region size	1 km ×1 km
n (number of clusters)	5, 10, 15, 20, 25, 30
h_{max}	10, 20, 30, 40, 50
θ_{th} (angle threshold)	20°, 40°
d_{th} (distance threshold)	10 m
α (weight threshold)	0.5, 0.7
δ (location variance)	1.5, 2
Q (noise variance)	0.1, 0.5
p_c	0.6
T (simulation time)	400 s. 900 s

$\hat{G}_t^a$ and G_t^b represents the a^{th} and b^{th} clusters in $\hat{G}_t$ and G_t, respectively. Finally, $sim(\hat{G}_t^a, G_t^b)$ measures the similarity ratio by finding the common member targets to the total member targets for both real and estimated clusters.

C. Simulation Results

1) Dynamic setting of history-window

We evaluated the effect of dynamic history-window on coherent set of moving targets. We ran the simulations for T = 900s in which θ_{th} and α were set to 20° and 0.7, respectively. The rest of parameters were set as stated in Table 1. Note that clusters are more scattered and they are less likely to overlap in scenarios with less number of clusters, e.g. 5 clusters; whereas scenarios with high number of clusters, e.g., 30 clusters, are highly overlapped. In fact, the possibility of overlapping increases as the number of clusters grows.

Fig. 4(a) shows the error reduction attained in clustering under higher values of h_{max}. The error reduction is more prominent in scenarios with higher number of clusters due to the high accuracy of PCMT in detecting the overlapped clusters by reviewing the historical assignments. Higher h_{max} would let the algorithm to assess more historical information resulting in a better accuracy. Fig. 4(b) shows an average size of history-window for different cluster densities and different values of h_{max}.

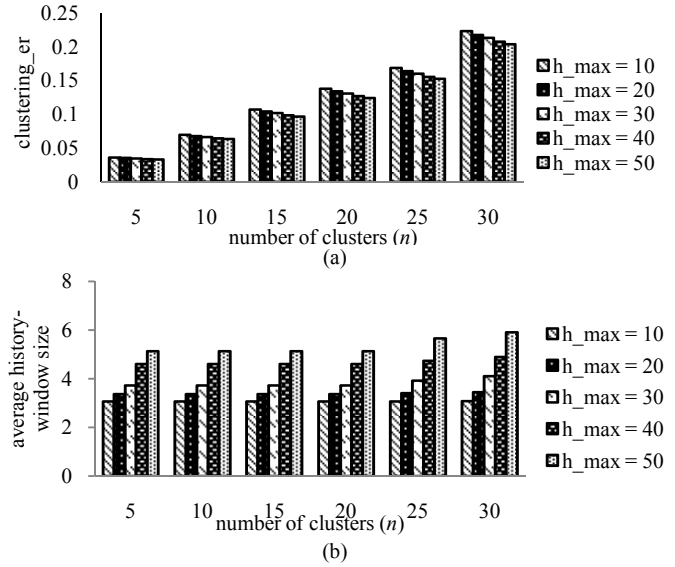

Fig. 4: Dynamic setting of history-window (a) impact of h_{max}value, (b) average size of history-window

2) Impact of weight values

Using the same simulation setting as in previous section, we compare the time complexity of clusters' construction at the initialisation phase for PCMT and DDBC algorithms. Initialisation phase represents the elapsed time since the declaration of the first non-zero weight until the detection of targets as one cluster. The reason for this comparison is that both algorithms use the concept of history-window to create a weighted graph.

Fig. 5 illustrates the CPU time at the initialisation phase for both algorithms. We observe that PCMT outperforms DDBC, and its performance is even better for higher h_{max} size Note that history-window size (h) for DDBC remains as h_{max} throughout simulation; whereas it is dynamically adjusted for PCMT as illustrated in Fig. 2. The reason for the better performance of PCMT is due to introduction of the weights associated with credits in (3) resulting in a more accurate and faster detection of clusters.

3) Clustering for the coherent movement of targets

This study was done by setting $h_{max} = 50$, as DDBC uses the constant history-window size which was also set to 50 in the simulations.

Fig. 6(a) shows the clustering error produced by all the algorithms. We observe that the total error improvement in clustering by PCMT is up to 65% as compared to DDBC; and about 81% as compared to MC since PCMT imposes the prediction of targets' direction in (2) as well as new definition of weights associated with credits in (3). Fig. 6(b) shows the average computational cost of all the algorithms for different cluster densities. We see that PCMT, with the dynamic setting of history-window, achieves a significant better CPU performance than DDBC which uses a fixed size. It is obvious that a dynamic history-window size is able to considerably reduce the amount of information that needs to be processed in comparison with DDBC. MC achieves best computation time as it only considers the location of targets at the current time instant but this consideration comes at the expense of high clustering error as shown in Fig. 6(a). Overall, PCMT performs significantly better than other algorithms.

4) Clustering for the incoherent movement of targets

To evaluate the performance of PCMT for a set of incoherent targets, we ran the simulations for T = 400 s in which θ_{th} and α were set to 40° and 0.5, respectively. Targets' trajectory is modelled as for Fig. 3 with $p_c = 0.6$. The rest of parameters were same as Table 1.

The results for clustering error in Fig. 7(a) show that PCMT is capable of accurately detecting clusters of incoherent targets in highly overlapped scenarios (high cluster numbers), whereas there is insignificant improvement when clusters are less overlapping. In fact, relationships between targets of a group appear as weak connections in the graph when they change directions due to the strict behaviour of PCMT in imposing the directions in relationship assignment estimation. Measuring only compactness provides accurate results when different clusters do not overlap.

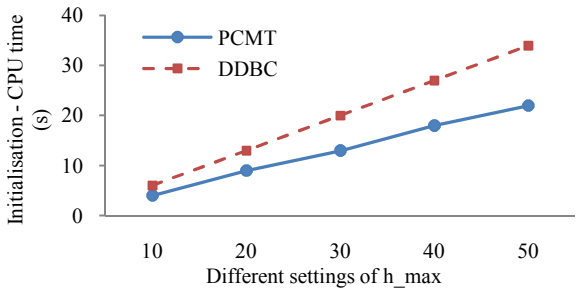

Fig. 5: Impact of the credit values on CPU time

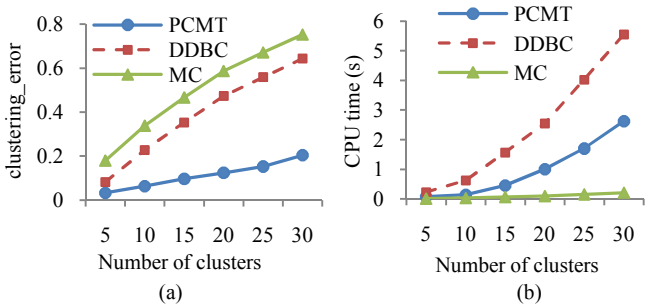

Fig. 6: Clustering for the coherent movement of targets, (a) clustering error, (b) CPU time at each algorithm run time

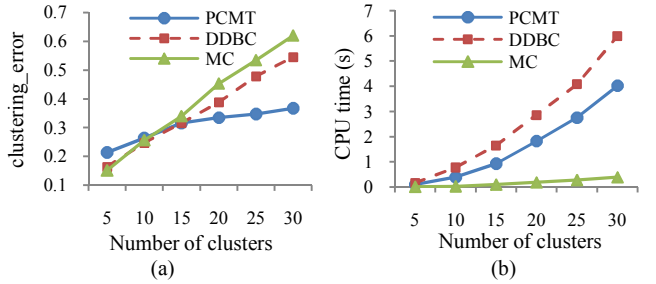

Fig. 7: Clustering for the incoherent movement of targets, (a) clustering error, (b) CPU time

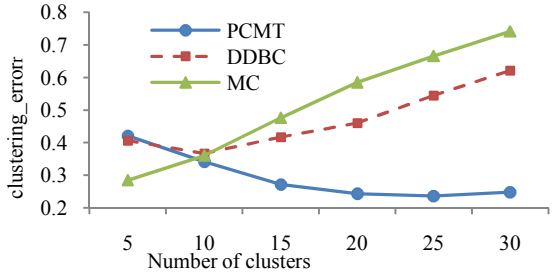

Fig. 8: Impact of split of and merge into clusters over times

We further find in Fig. 7(b) that PCMT still achieves a better CPU runtime than DDBC despite the oscillation of history-window to larger size in order to detect strong relationships between incoherent targets.

5) Impact of split and merge of clusters

The final scenarios were created to evaluate PCMT's performance to detect changes where some member targets would leave the respective clusters or join other clusters. We generated the scenarios by adding/removing random number of targets to/from random clusters at random times. All the simulation settings are as for coherent targets with $h_{max} = 50$. Fig. 8 shows much superior performance of PCMT. The results confirm the fact that PCMT is able to find the stronger relationship assignments and detect the clusters changes. The achieved clustering error improvement by PCMT is up to 60% over DDBC and about 65% over MC in these scenarios.

I. CONCLUSION

Efficient clustering algorithms can significantly improve the accuracy of clustering of moving targets detected by wireless sensor networks. In this paper, we present an algorithm that uses predicted moving directions and distances of targets and dynamically adjusts the history-window to estimate the relationships among targets. Our results show an improvement in clustering accuracy between 65%-81% over existing algorithms that use a fixed history-window and simple criterion, namely, compactness to estimate the clustering relationships. Our algorithm also shows a superior computation time by a dynamic setting of history-window.

REFERENCES

1. Chen, J., et al., *Clustering moving objects in spatial networks*, in *Proceedings of the 12th international conference on Database systems for advanced applications*. 2007, Springer-Verlag: Bangkok, Thailand. p. 611-623.
2. Li, Y., J. Han, and J. Yang, *Clustering moving objects*, in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. 2004, ACM: Seattle, WA, USA. p. 617-622.
3. Kalnis, P., N. Mamoulis, and S. Bakiras, *On Discovering Moving Clusters in Spatio-temporal Data Advances in Spatial and Temporal Databases*, C. Bauzer Medeiros, M. Egenhofer, and E. Bertino, Editors. 2005, Springer Berlin / Heidelberg. p. 923-923.
4. Jensen, C.S., L. Dan, and O. Beng Chin, *Continuous Clustering of Moving Objects.* Knowledge and Data Engineering, IEEE Transactions on, 2007. **19**(9): p. 1161-1174.
5. Zhenjie, Z., et al., *Continuous k-Means Monitoring over Moving Objects.* IEEE Transactions on Knowledge and Data Engineering, 2008. **20**(9): p. 1205-1216.
6. Ester, M., et al. *A density-based algorithm for discovering clusters in large spatial databases with noise*. 1996: AAAI Press.
7. Ossama, O., H.M.O. Mokhtar, and M.E. El-Sharkawi, *An extended k-means technique for clustering moving objects.* Egyptian Informatics Journal, 2011. **12**(1): p. 45-51.
8. Nanni, M. and D. Pedreschi, *Time-focused clustering of trajectories of moving objects.* Journal of Intelligent Information Systems, 2006. **27**(3): p. 267-289.
9. Elnekave, S., et al. *Discovering regular groups of mobile objects using incremental clustering.* in *WPNC 2008.*
10. Rinzivillo, S., et al., *Visually driven analysis of movement data by progressive clustering.* Information Visualization, 2008. **7**(3-4): p. 225-239.
11. Brakatsoulas, S., D. Pfoser, and N. Tryfona. *Modeling, storing and mining moving object databases*. in *Proceedings of IDEAS,* 2004.
12. Rosswog, J. and K. Ghose, *Efficiently detecting clusters of mobile objects in the presence of dense noise*, in *Proceedings of the 2010 ACM Symposium on Applied Computing*. 2010, ACM: Sierre, Switzerland. p. 1095-1102.
13. Tarjan, R. *Depth-first search and linear graph algorithms*. IEEE ,1971: p. 114-121.
14. Särkkä, S., A. Vehtari, and J. Lampinen, *Rao-Blackwellized particle filter for multiple target tracking.* Information Fusion, 2007. **8**(1): p. 2-15.
15. Gavrilov, M., et al. *Mining the stock market (extended abstract): which measure is best?* ACM , 2000: p. 487-496.