Extended General Motion Model: an Assembled Object Oriented Model for Moving Objects

HUANZHUO YE [1 a], TENGHONG LIU [1 b], JIANPING PAN [2], JING CHEN [3]
[1] Information School
Zhongnan University of Economics and Law
114, Wuluo Road, Wuhan, Hubei Province
P. R. CHINA

[2] School of Civil Engineering & Architecture
Chongqing Jiaotong University
No.66 Xuefu Road Nan'an District，Chongqing
P. R. CHINA

[3] State Key Laboratory of Information Engineering in Surveying and Mapping
Wuhan University
129, Luoyu Road, Wuhan, Hubei Province
P. R. CHINA

Abstract: - Moving object information management, which is based on the model of moving object, is necessary in many applications. Lots of models of moving object are proposed. Each of them works well in certain scenario. In order to fulfill the needs of most application, General Motion Model (GMM) is studied. And based on this model, Extended GMM (EGMM) is proposed to make motion functions be able to be registered in real time so as to not ask the motion functions to be predefined and introduce access control to each motion function. Besides, it also changes the model structure to optimize the performance of GMM.

Key-Words: - Moving Objects, Information Management, Modeling, GMM, EGMM, Motion

1 Introduction

GPS, PDA and other electronic equipments are becoming must haves in today's life, esp. in traveling. These equipments help to locate individuals, cars or other moving objects, and provide environment information, such as locations of points of interest, directions to certain places, etc. with help of GIS. With this information, people can easily find interested places and the way to get these places.

This technique is widely applied in ITS (Intelligent Transport System), digital battle field, logistics management, mobile e-commerce, traveler service and other LBS (Location Based Service) systems. At the beginning of the application, the services are mostly provided as static and 'one-way'. The information provided is pre-stored and not changed in real time. The moving objects retrieve the location and direction based on their current states, which are not collected by the server and used for further service. Now this static and 'one way' technique is no longer satisfied by users. In many cases, people like to know the traffic situation as well as the location and direction, because sometimes the shortest way is not the fastest way. In other scenarios, history states of users are needed for better service, such as rebuilding history track, extracting users' behavior, finding out users' habit, etc. These require the server to record the current and history states of each user. And the information provided for one user is not only based on the current state of this user, but also based on the current states of other users and/or the history states.

The problem is how to organize the motion information of a moving object and thus to store it. At a certain time, a moving object, regardless of its shape and extend, can be represented by its position and orientation. In 2D space, which is widely applied in digital map and many other applications, given a coordinate system, the position of the moving object is represented by $P(x, y)$, and the orientation can be represented by the angle $O(\alpha)$ between predefined starting vector, e.g. X axis, and the predefined front direction of the moving object. In 3D space, a moving object can be represented by its position $P(x, y, z)$ and orientation $O(\varphi, \omega, \kappa)$.

However, (P, O) is the spatial state of a moving object in a certain time. It can not show the moving process. Therefore, how to represent the motion of moving object, in another word, how to establish a

moving object model, becomes the basic issue of moving object information management.

2 Related Works

There are lots of models proposed for moving objects.

The simplest way is to record (*P, O*) periodically. This, which can be called as point model, is somewhat like motion picture, using discrete static picture to simulate a continuously changing process. In the period of moving, (*t, P, O*) is recorded for each object to show that at time *t* the object is at position *P* with orientation *O*. The (*t, P, O*) series is stored in database, organized by DBMS and queried using SQL.

As an improvement of point model, interval model introduces time interval to state how long the recorded position and orientation is valid. This improvement expands discrete points to "lines". But these "lines" are only limited in time dimension. Ref. [1] is a good example of interval model.

Polyline model simplifies trajectory of moving object as connected line segments, and processes the motion data of moving objects by recording and processing the segments in polylines. This kind of model can be used to manage the moving objects traveling in urban area or roads. Many studies of M. Vazirgiannis [2], D. Pfoser [3] and H. D. Chon [4] [5] are based on polyline model.

Function model depicts a trajectory as a time function, and gets the position at certain time by recording and calculating the corresponding function value. Most current moving object studies are based on this kind of model. R. Ding abstracts a moving object as a point in space, whose position *f(t)* is a function of time. When any parameter of *f(t)* changes, the database is updated, so that the moving trend can be recognized to predict the position of moving object at future time [6].

The most popular function model used is Moving Objects Spatio-Temporal (MOST) by A. Prasad Sistla. MOST employs object-oriented method to present dynamic attribute of moving objects. A dynamic attribute *A* consists of three sub-attributes, *A.value*, *A.updatetime*, *A.Function*, in which *A.Function* is a function of time *t* and its value is 0 when t=0, *A.updatetime* is the update time of the record, *A.value* is the value of dynamic attribute *A* at time *A.updatetime*. At any time between *A.updatetime* and the time of next update *A.updatetime*+t_0, the value of dynamic attribute *A* is *A.value*+*A.Function*(t_0) [7].

Ref. [8] proposes general motion model (GMM), which separates the static attributes and dynamic attributes of moving objects and manage them respectively. Not like MOST, GMM expresses attributes in set(s) and does not ask the function of attributes to be explicit and incremental. The dynamic attribute set S (S=$\{C_i\}$, in which C_i is dynamic attribute, $1 \leqslant i \leqslant n$, n is the number of dynamic attributes in set S) of moving object *Obj* is composed of two sub-attributes, *S.Data* and *S.Functions* (as shown in Fig. 1). *S.Data* contains the values of the variables related to the dynamic attributes including their changes and/or the parameters of the motion functions. *S.Functions* describes the ways to calculate the value of attribute(s) in set *S*, i.e. the definition of the motion functions and/or the way to get or form the motion functions. When $S(t)$ or $C_i(t)$ is to be calculated, the predefined function(s) or the calculation procedure(s) is found in *S.Functions*, and then the related parameters or sample data are found in *S.Data*. According to the value of t, $S(t)$ or $C_i(t)$ can be calculated.

3 Analysis of GMM

GMM tries to solve the problem of moving objects representation by forming a kind of superset of other existing models. It does solve some problem. However, it still has some disadvantages.

3.1 Advantages of GMM

It offers both interpolation/extrapolation and discrete processing, unifies sampling method and function method. Its multiple definitions of functions or function construction methods meet the requirement of level of details (LOD) for motion data, and combines linear and nonlinear functions. It also offers flexibility in dynamic attributs choosing and grouping, so that it can be used in 1D, 2D and 3D space or even higher. GMM is encapsulated with object oriented method and thus easy to use by application programmers.

3.2 Disadvantages of GMM

However, applications are various and always challenging. GMM is not satisfactory in some cases. It has the following deficiencies.

3.2.1 Function must be predefined

As showin in Fig. 1, the dynamic attribute set S of moving object *Obj* is composed of two sub-attributes,

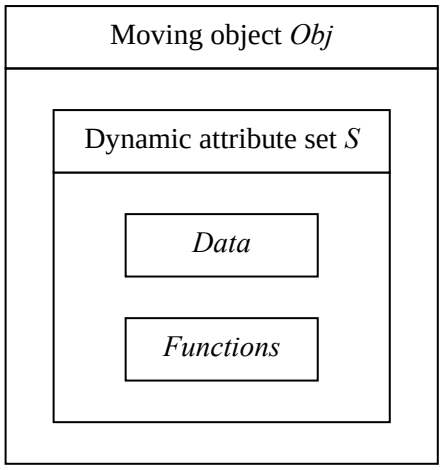

Fig. 1: Structure of GMM

S.Data and *S.Functions*. *S.Data* is recorded as the motion data of the object is collecting. But *S.Functions* must be predefined. Otherwise the system does not know the structure of the functions and the functions can not be referenced by *S.Data*.

This requirement limits the use of GMM, because sometimes the functions can not be predefined and need to be added to *S.Functions* dynamically.

In many scenarios, the function set is expected to be expandable when new objects or new moving signatures are added to the system. But the predefining requirement makes this impossible.

3.2.2 Function sets are not shared

From the structure in Fig. 1, it is easy to find out that *S.Data* and *S.Functions* are dependent on dynamic set *S*. As sub attributes, they are defined and stored within *S*. For different dynamic sets $S_1, S_2, S_3, \ldots, S_m$, there are *m* different *Functions* subsets. Are they all necessary?

Though function set is expected to be expandable, in most cases, different moving objects or different dynamic sets have many functions in common. The models discussed in section 2 can fulfill the needs of most applications. So it is obvious that it will be more efficient if the function set can be shared within different dynamic sets or different objects.

3.2.3 New functions must be explicitly added

The function set shows all the functions in a list. In a certain period of time, the motion of an object falls into one of the functions in the list. When the motion is recorded, a reference of the function is given and filed into data set with time indicator and corresponding parameters. This procedure requires that all the function that will be referenced must be in the list, i.e. in the function set. If any new function will be used, it must be known and added to the function set first. Even if dynamic defining of functions is possible, in some cases, it is still not practice to add all the functions into the function set.

First, some functions are in explicit format like $y = f(x)$, which is easy and straightforward to add into function set. However, some functions are not. For implicit functions and complicated functions, it is difficult to add the functions themselves directly on the fly. Sometimes it is even impossible.

Second, in some applications, functions are not able to be added into functions set, because of their system structure or some other reasons. For example, in distributed computing, recording and calculating are assigned to different hosts. It is not necessary and not efficient to add all these functions into one set on the central server. And in some scenarios, for security reason, certain functions are not allowed to be added to the list and accessed freely by others.

To get rid of the above deficiencies, new method must be studied.

4 Extended GMM

As discussed in section 3, GMM works well with most applications. The question is how to solve the problem when the motion function of the moving object is not able to be predefined, or can not be added into function set and be accessed by other programs or hosts, and how to rearrange the model structure to make most used basic function be shared among different objects. Comparatively, motion function sharing is easier, but function defining on the fly and not explicitly listing in the function set needs more work.

4.1 Improving Solution

Taken all the disadvantages into consideration, in order to keep the existing advantages of GMM as well as to solve the problems, the following steps are employed to improve the existing model.

4.1.1 Sharing Function Set

Take function set out of dynamic attribute set *S*, even out of the moving object model, so as to make it shared by different moving objects.

The data sub-attribute is still remain in dynamic attribute set *S*, but the reference to motion function in recording data should be changed to the object outside of moving object.

Build a new object, moving objects depository, to hold the motion function set and all the moving objects depend on this set.

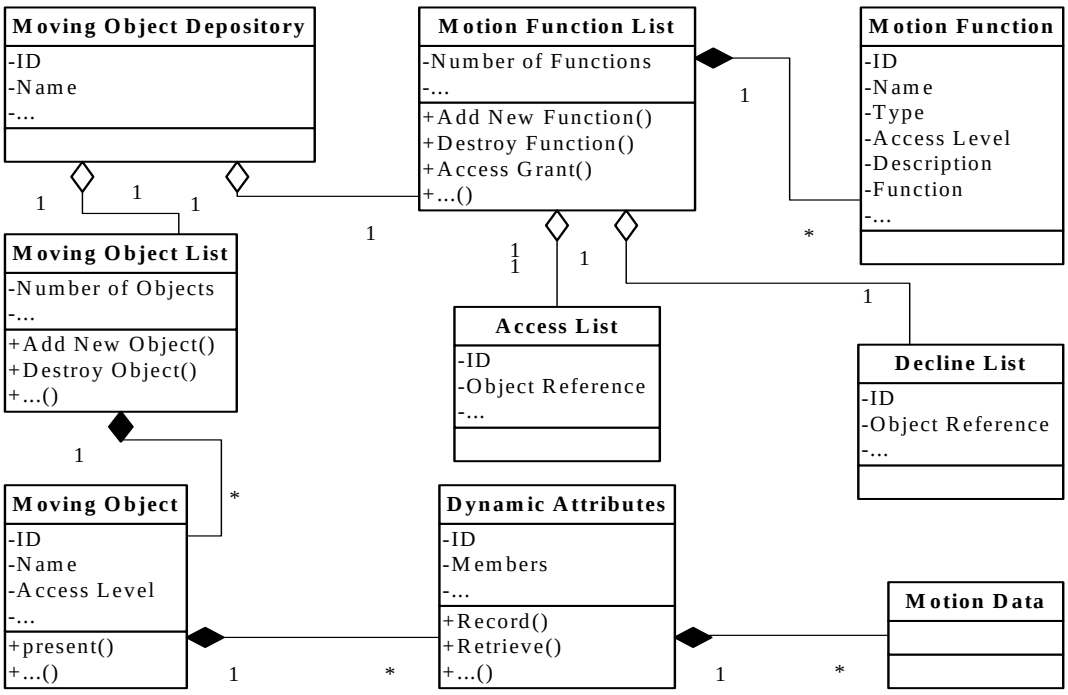

Fig. 2: Simplified Structure of Extended General Motion Model

In this way, all the related moving objects or the objects which have similar motion signature and want to share the function list are fallen into one group. Unrelated moving objects can either be put into the same group or put into different group as the application preference.

4.1.2 Allowing Reference in Function Set

In GMM, only explicit function expression or function procedure is listed in function set. In order to add motion function on the fly and allowing implicit function procedure exist, motion function reference must allowed to be listed in function set. It means that all kind of function format, expression, procedure and reference can all be presented in the new function set.

Therefore, when a function is needed to be added into function set, it does not need to be explicitly presented. It can be any form of procedure or expression as long as it can return the necessary motion state when certain parameters are given. The reference of this function can be added to the function set at any time before it is visited, so that it does not need to be predefined.

4.1.3 Adding Access Control

For security reason, every function or procedure itself can define its own access control policy. Nevertheless, to make the model self-dependent, it is necessary to add access control to each function into function set.

This can be easily implemented by defining access level with each function. Only the moving object with proper access authority can visit that function.

4.1.4 Defining Interface for Function Registering

In order to add or register new functions on the fly, an official register interface must be defined. The register interface should allow all forms, including reference, of function with access level and parameter list declaration, etc. to be added into function set and make it to be accessible to all the proper moving objects.

4.2 EGMM Structure

Based on the steps discussed above, the structure of extended general motion model can be drawn. Fig. 2 illustrates a simplified model, in which functions set is relatively independent to the objects set and some attributes are added to combine two sets together under a unified holder.

Moving Object Depository (MOD) is a container to hold all the related moving objects and corresponding motion functions. It contains two lists, Motion Function List and Moving Object List, and encapsulates all the moving objects and motion

functions to make an easy interface for application user.

Motion Function List (MFL) is the function set discussed above. It adds new function at request and deletes functions that will not be used anymore (of course, this operation is very rare used). The main part of MFL is a list of motion functions, which with each entry stated the type of the function and the way to access it as well as access level. MFL also maintains two other lists, Access List and Decline List to accelerate the accession with access control.

Moving Object List (MOL) organizes the moving objects in this depository. It adds new object or delete object on demand, and maintains each moving object properly. Besides static attributes information, each moving object contains one or more dynamic attribute set, which groups related dynamic attributes together according to the motion function requirement. Naturally, each dynamic attribute set contains a motion data set, which records all the motion data related to the attribute(s) in the dynamic attribute set.

4.3 Working with EGMM

An application system can have one or more instances of MOD. When a MOD is instanced, an empty MOL and a MFL with common used function set are generated.

New moving objects are added through MOL. When a new moving object is created in MOL, it can record and retrieve motion data immediately with the existing functions in MFL. When there is a new function needed, it can be added to MFL at runtime, and then be referenced by moving objects.

The other operations are similar to GMM. In this instance, EGMM preserves all the advantages of GMM, thus works well with moving objects information management.

4.4 Characteristics of EGMM

Through the structure of EGMM and the way it works, it is not difficult to find out the characteristics of EGMM.

Besides the problems EGMM solves, it keeps the advantages of GMM and have the following feature in moving objects information management.

1. Not like some other moving object models, which only present position in their data management, EGMM presents orientation in spatial state as well. And the elements in spatial state can be freely chosen at the request of different application.
2. The velocity, acceleration and other motion states in movement and rotation, which are also collectable, can all be included in EGMM.
3. The collectable data are various in different applications. EGMM is able to deal with different cases, including the information management of 1D and 2D moving objects.
4. When applied sampling method with EGMM, interpolation and extrapolation can be added to discrete samples, so as to make the state at any time when the movement is accessible, and the state in any future time within valid period is predictable.
5. In the organization of motion data within EGMM, DBMS as well as file system can both be applicable.
6. EGMM can be applied not only in the management of objects in uniform motion, but also in the management of objects in non uniform motion.
7. EGMM is able to be applied in the cases that the movement path is unknown as well as the cases that the movement path is known.
8. EGMM is not limited to any specific motion function. It should be able to process nonlinear functions as well as linear functions, even the functions which are not able to be expressed explicitly.
9. With EGMM, not only the information of system-controlled objects and the objects whose motion functions are obtainable is manageable, but also the information of the objects, whose motion functions are not obtainable, can be managed.
10. Encapsulation mechanism is employed to EGMM, so that the users can operate the model through interfaces in an easy manner.
11. Different solutions with tradeoff of precision and computation complexity should be applied to different requirement according to the application, i.e. level of details (LOD) for motion data is taken into consideration in EGMM and can be applied by the use of different motion functions for the same period of motion..

5 Experiment

The experiment system in Ref. [8] is reprogrammed to test EGMM. This experiment is to present the satellites according to their real motion based on the method introduced by F. R. Hoots and R. L. Roehrich in *Spacetrack report No.3, models for propagation of NORAD element sets* [9]. The location and the

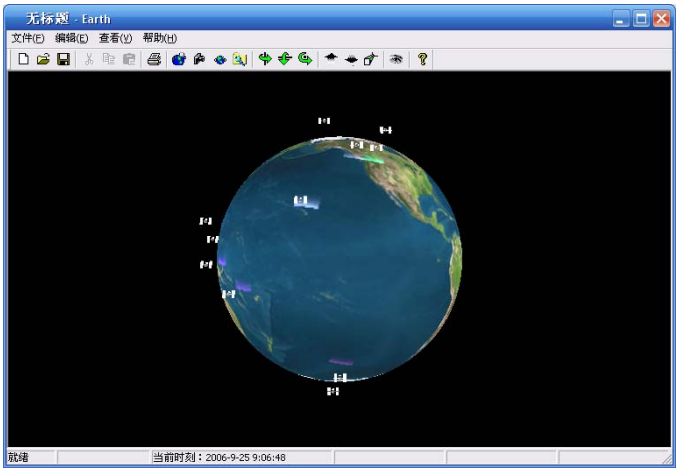

Fig. 3: Interface of Experiment System

velocity of any satellite can be retrieved and the motion can be replayed at the request.

In the new system, which employs EGMM, the upper level, i.e. application level is hardly changed, except for access control. But the lower level of the system, esp. the automation modular was completely rewritten.

First, based on EGMM structure in Fig. 2, a modular is constructed to replace the former Moving Object Management Modular, which complies with GMM.

Second, the moving object model is built according to the usage of EGMM.

Then, the satellite moving information system is built and tested.

In the new system, the motion function is not previously stated in EGMM it self, but organized independently first. While system is running, the motion function of satellite according to Ref. [9] is registered and be referenced by the management modular. In this way, the automation modular knows nothing about the detailed procedure of motion state calculation. It only registers the motion function at request and references it when motion state retrieval command arrives. Therefore, the motion function is not necessary to be predefined and the detailed calculation method is not necessary to be open. And of course with the access control feature, which is newly added to EGMM, the application can protect the motion function at its own control.

6 Acknowledgment

This research has been supported by

- China Scholarship Council
- Zhongnan University of Economics and Law (Startup Research Fund for Fetched in Talent of Zhongnan University of Economics and Law)
- Chongqing Science & Technology Commission of China (NO.2006BB2411)

References:

[1]. Y. Masunaga, N. Ukai. Toward a 3D moving object data model. *Proceedings of 1999 International Symposium on Database Applications in Non-Traditional Environments (DANTE'99)*, November 28 - 30, 1999 Kyoto, Japan. pp. 302-312.

[2]. M. Vazirgiannis and O. Wolfson. A spatiotemporal model and language for moving objects on road networks. *Proceedings of International Symposium on Spatial and Temporal Databases*, pp. 20-35, 2001.

[3]. D. Pfoser, C. S. Jensen, and Y. Theodoridis. Novel approaches in query processing for moving objects. *Proceedings of the 26th International Conference on Very Large Databases (VLDB)*, 2000.

[4]. H. D. Chon, D. Agrawal, A. El Abbadi. Query processing for moving objects with space-time grid storage model. *Proceedings of the International Conferenceon Mobile Data Management*, 2002.

[5]. H. D. Chon, D. Agrawal, A. El Abbadi. Storage and retrieval of moving objects. *Proceedings of Mobile Data Management 2001 conference*, pp. 173-184, 2001.

[6]. R. Ding, X. Meng. A quadtree based dynamic attribute index structure and query process, *Proceedings of International Conference on Computer Networks and Mobile Computing*, pp. 446-451, 2001.

[7]. P. Sistla, O. Wolfson, S. Chamberlain, and S. Dao, Modeling and querying moving objects, *Proceedings of the Thirteenth International Conference on Data Engineering (ICDE13)*, Apr. 1997.

[8]. H. Ye, T. Shang, J. Ye, J. Pan, General Motion Model: a general model for moving objects, *Multimedia on Mobile Device, Proc. SPIE Vol.5684*: 274-283 (2005. 3.).

[9]. F. R. Hoots, R. L. Roehrich. Spacetrack report No.3, models for propagation of NORAD element sets. December 1980. http://www.amsat.org/amsat/ftp/docs/spacetrk.pdf.