Moving Objects and KML Files

Karine Reis Ferreira[#1], Lúbia Vinhas[#2], Antônio Miguel Vieira Monteiro[#3], Gilberto Camara[#4]

[#]*Image Processing Division (DPI), National Institute for Space Research (INPE)*
Av. dos Astronautas, 1758, Jardim da Granja, São José dos Campos – SP, Brazil
[1]karine@dpi.inpe.br
[2]lubia@dpi.inpe.br
[3]miguel@dpi.inpe.br
[4]gilberto@dpi.inpe.br

Abstract— **Although KML files can be used to describe journeys, there is not a standard way to represent them as moving object trajectories for further analysis. In the KML schema, there is not a predefined element to describe a moving object trajectory. Each software or mobile device that generates KML files with trajectories uses its own structure for representing them. Therefore, this work proposes an interoperable way to extract moving object trajectories from any KML file, based on the processing of an additional metadata file. This metadata file is an XML that must be compliant with an XML schema proposed in this paper. This proposal has been implemented in a geographical software library as a proof of concept.**

I. Introduction

The recent technological advances in geospatial data collection, such as Earth observation and GPS satellites, mobile computing, and sensor networks, have motivated new applications that deal with spatiotemporal information. Examples of these applications are animal tracking and environmental change monitoring based on satellite images. Therefore, there is a growing demand for geographical information systems (GIS) that can represent and analyse such information.

Moving objects is a well-known class of spatiotemporal data, which represents objects whose spatial positions or extents change continuously over time [1]. Examples of moving objects are cars, aircraft, ships, mobile phone users, polar bears, hurricanes, forest fires, and oil spills on the sea. This work focuses on this class of spatiotemporal data.

Regarding moving object treatment in database management systems (DBMS), there are two main initiatives of moving object database systems: SECONDO [2] and Hermes [3]. Both extend a DBMS SQL type system with new data types and functions to represent and handle moving objects, based on the algebra proposed by Erwig et al. [1].

Moving objects can also be represented in flat data files, such as KML files. KML is an Open Geospatial Consortium (OGC) standard for encoding representations of geographic data, mainly for display in an Earth browser [4]. It can be used to different purposes, such as: (1) to specify icons and labels to identify locations on the Earth surface, (2) to define image overlays, (3) to define styles of appearance, and (4) to describe journeys or trajectories of moving objects.

KML uses a tag-based structure with nested elements and attributes and is based on the XML standard. An XML Schema describes the structure of an XML document. The XML schema for KML files, called KML Schema, defines elements to describe geometry objects and time stamps. It does not define a particular element to describe trajectories. Thus, a trajectory is often described by a set of geometry objects associated to time stamps, without using a predefined element for it. Each software or mobile device that generates KML files uses its own way to organize geometry objects associated to time stamps as trajectories.

Virtual Globes, such as Google Earth, can display geometry objects associated to time stamps in KML files, automatically configuring timelines and generating animations over time. However, most of these software tools are not able to process or analyze them as moving object trajectories. For example, they cannot answer questions like "*When did object o_1 enter a specific region r_{10} and how long did it stay in this region?*" or "*Where and when was there a spatiotemporal cluster of objects?*". This requires a more specialized tool that is able to: (1) translate geometry objects associated to time stamps in KML files into data structures that represent moving object trajectories, and (2) analyze trajectories, by providing functions over its data structures that can answer questions like the ones presented above. The authors are developing a software module in a geographical library called TerraLib [5] to meet these two requirements, as described in Section V.

Since there is not a standard way to represent trajectories in KML files, this work proposes the use of an additional metadata file to describe how to translate geometry objects associated to time stamps in KML files into moving object trajectories. This metadata file is an XML that must be compliant with the XML schema proposed in Section IV.

This proposal has been implemented in the TerraLib library as an interoperable strategy to extract moving object trajectories from any KML file.

II. Background

This section provides a brief overview of some subjects that are essential for our proposal understanding.

A. XML and XSD

XML stands for e**X**tensible **M**arkup **L**anguage and is a markup language designed to transport and store structured data. It is a W3C recommendation and has been widely used to carry and share data mainly in the Web environment [6].

An XML file is structured through user-defined tags, that is, its tags are not predefined. An XML document can be described by a XML Schema. The purpose of an XML Schema is to define the legal building blocks of an XML document in terms of elements and attributes that can appear in an XML file. The XML Schema language is called **X**ML **S**chema **D**efinition (XSD) [7].

B. *KML*

KML stands for **K**eyhole **M**arkup **L**anguage and is an OGC standard for encoding and transporting representations of geographic data, mainly for data display in an Earth browser [4]. It is an XML file that follows a predefined XML schema. Such schema describes the grammar which KML file instances must be compliant with. All components of the KML schema are defined in the namespace with the identifier "http://www.opengis.net/kml/2.2".

A KML file can be used to different purposes, such as: (1) to specify icons and labels to identify locations on the Earth surface, (2) to define image overlays, (3) to define styles of appearance, and (4) to describe journeys. This work focuses on the fourth purpose.

C. *Moving Objects and Trajectory*

Erwig et al. defines *moving objects* as entities whose spatial positions or extents change continuously over time [1]. They propose an algebra to deal with moving objects which defines three main data types, *moving points*, *moving lines* and *moving regions*, and a set of operations over them, such as *trajectory* and *distance* [2].

Two examples of moving objects are animals and deforested regions, illustrated in Fig. 1 (a) and (b). The former is an example of a *moving point* because the object position is changing over time, not its extent. The latter is an example of a *moving region,* since the object extent is changing over time.

Although the moving object spatial positions or extents change continuously over time, they are often represented by discrete observations. For instance, Fig. 1 (a) shows an animal tracking composed of a set of observations where each one contains a spatial position, represented by a point, and a time instant when the animal was in this position. Fig. 1 (b) presents the evolution of a deforested region in three observations. Each observation contains the spatial extent of the deforested region, represented by a polygon, and the year when it was detected.

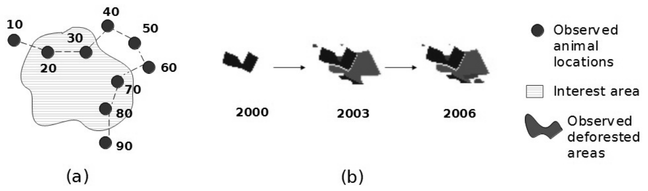


Fig. 1 Examples of moving objects: (a) an animal tracking and (b) the evolution of a deforested region.

According to Spaccapietra et. al [8], *trajectories* are countable journeys associated to objects which are moving in space over time. Different kinds of trajectories can be extracted from a moving object. For example, if an application is interested in studying the daily behaviour of an animal, it can extract its trajectories by grouping its daily observations.

III. The Problem

A journey is often represented in KML files by a set of observations, where each one describes the spatial location of a moving object in a specific time. Each observation is represented by a `kml::PlacemarkType` data type.

Nevertheless, there is not a standard way to organize these observations as a moving object trajectory. In the KML schema, there is not a predefined element to represent trajectories. So, each software or mobile device that generates KML files uses its own structure to group these observations as trajectories. For example, all observations of the same trajectory can be grouped together either in a `kml::FolderType` component or in a `kml::DocumentType` type. Besides that, a common problem arises when a KML file stores trajectories of more than one moving object. In this case, there is not an ordinary way to separate the observations of each object.

We illustrate the problems described above with two real KML files, generated by distinct devices and for different motivations. Both are valid KML files that contain trajectories of moving objects:

1) The former was generated by a project that monitors sea elephants in the Antarctica. This KML file contains trajectories of eight monitored animals and groups the observations of each one in a distinct `kml::FolderType` element. Each animal observation is represented by a `kml::PlacemarkType` element that is composed by its spatial location, represented by a `kml::PointType` element, and the time instant when it was at this location, represented by a `kml::TimeStampType` element. This file also contains visual style elements to describe how the data should be visualized. Fig. 2 shows the display of this KML file in the Google Earth, where the red lines represent the sea elephant trajectories.

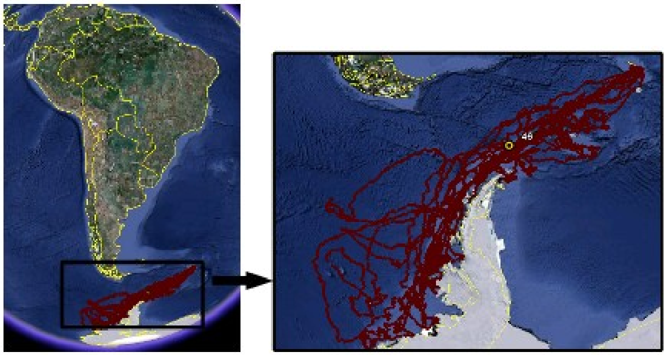


Fig. 2 Trajectories of sea elephants: display of the first KML in the Google Earth software.

2) The latter was generated by a mobile device collecting the positions of a runner during a training section. This KML file contains the trajectory of a single object whose

observations are spread in different `kml::FolderType` elements inside a `kml::DocumentType` element. Each observation is also represented by a `kml::PlacemarkType` element, where the object spatial location is represented by a `kml::PointType` element and the time instant when it was at this location is represented by a `kml::TimeStampType` element.

Since there is not a predefined element to describe trajectories in KML files, we propose a strategy that allows different software packages to interpret and extract moving object trajectories from any KML file. Considering the first example, this strategy is able to distinguish the trajectories of each animal, during different periods of time, from the collection of observations in the file. In the second example, it is able to group all observations in the file into one single trajectory. In both cases, our goal is to analyse the moving object trajectories by using the TerraLib library, in a similar way.

IV. Metadata about KML Moving Objects

Our approach consists in using an additional XML metadata file. We propose an XML schema to describe how to interpret data stored in KML files as moving objects. The original KML data file is than annotated with a XML metadata file that follows the proposed schema.

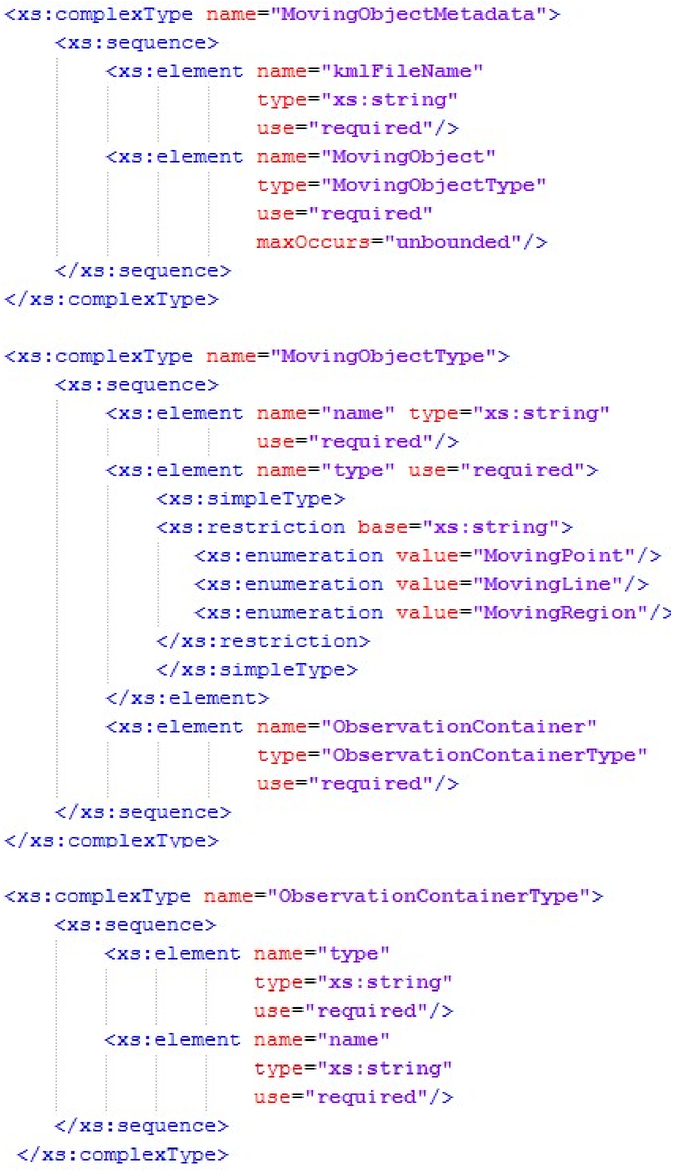
The proposed schema defines three complex elements: `MovingObjectMetadata`, `MovingObjectType` and `ObservationContainerType`, as can be seen in Fig. 3.

The `MovingObjectMetadata` element contains the name of the KML file that stores moving objects (element name: `kmlFileName`) and one or more moving object description (element name: `MovingObject`) which is of the `MovingObjectType` type.

Each moving object is described by the `MovingObjectType` type that contains the object name or identity (element name: `name`), its type (element name: `type`) and information about the container that carries its observations (element name: `ObservationContainer`). The possible types of a moving object are `MovingPoint`, `MovingLine` and `MovingRegion`, as explained in Section II.

The container that carries the moving object observations is described through the `ObservationContainerType` type. Such type is composed of two elements, the container type (element name: `type`) and its name (element name: `name`).

In our proposal, we assume that each moving object observation is represented in a KML file by the `kml::PlacemarkType` type. This assumption is based on the fact that `kml::PlacemarkType` is the only KML schema type that is associated to a geometry type. The geometry types defined in the KML schema are: `kml:PointType`, `kml:LineStringType`, `kml:LinearRingType`, `kml:PolygonType`, and `kml:MultiGeometryType`.

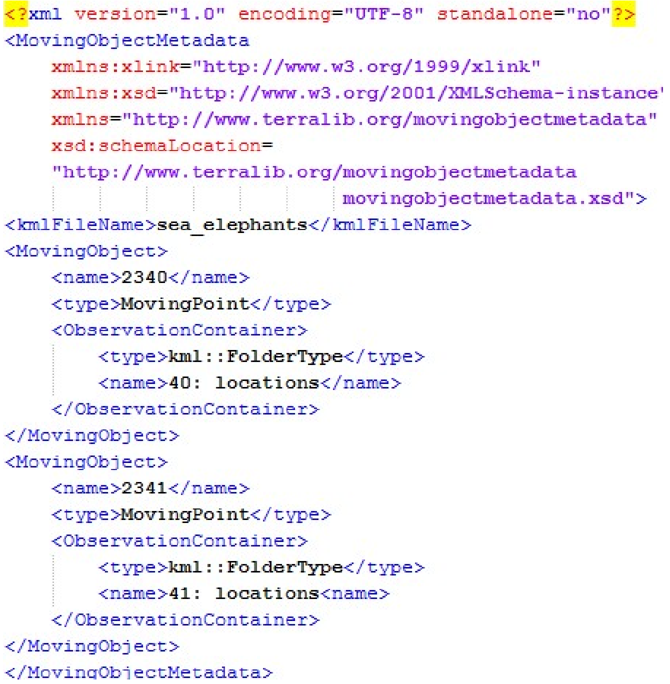


```
<xs:complexType name="MovingObjectMetadata">
    <xs:sequence>
        <xs:element name="kmlFileName"
                    type="xs:string"
                    use="required"/>
        <xs:element name="MovingObject"
                    type="MovingObjectType"
                    use="required"
                    maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>

<xs:complexType name="MovingObjectType">
    <xs:sequence>
        <xs:element name="name" type="xs:string"
                    use="required"/>
        <xs:element name="type" use="required">
            <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:enumeration value="MovingPoint"/>
               <xs:enumeration value="MovingLine"/>
               <xs:enumeration value="MovingRegion"/>
            </xs:restriction>
            </xs:simpleType>
        </xs:element>
        <xs:element name="ObservationContainer"
                    type="ObservationContainerType"
                    use="required"/>
    </xs:sequence>
</xs:complexType>

<xs:complexType name="ObservationContainerType">
    <xs:sequence>
        <xs:element name="type"
                    type="xs:string"
                    use="required"/>
        <xs:element name="name"
                    type="xs:string"
                    use="required"/>
    </xs:sequence>
 </xs:complexType>
```

Fig. 3 XML schema to describe metadata about KML moving objects.

Each moving object observation is represented by a `kml::PlacemarkType` type that has, at least, two elements, one to represent the geometry of the object and the other to represent the time instant associated to this geometry. The geometry of a moving point observation is represented by the `kml::PointType` type, of a moving line observation by the `kml:LineStringType` type and of a moving region by the `kml:PolygonType` type. Finally, the time instant of all observations is represented by the `kml::TimeStampType` type.

The proposed XML schema defines the structure of an XML document that describes how to interpret data stored in KML files as moving objects. Fig. 4 shows an example of an XML document that contains the metadata about the sea elephant trajectories stored in the KML described in Section III.



```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<MovingObjectMetadata
    xmlns:xlink="http://www.w3.org/1999/xlink"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema-instance'
    xmlns="http://www.terralib.org/movingobjectmetadata"
    xsd:schemaLocation=
    "http://www.terralib.org/movingobjectmetadata
                                  movingobjectmetadata.xsd">
<kmlFileName>sea_elephants</kmlFileName>
<MovingObject>
    <name>2340</name>
    <type>MovingPoint</type>
    <ObservationContainer>
        <type>kml::FolderType</type>
        <name>40: locations</name>
    </ObservationContainer>
</MovingObject>
<MovingObject>
    <name>2341</name>
    <type>MovingPoint</type>
    <ObservationContainer>
        <type>kml::FolderType</type>
        <name>41: locations<name>
    </ObservationContainer>
</MovingObject>
</MovingObjectMetadata>
```

Fig. 4 XML document with moving object metadata.

The XML document presented in Fig. 4 contains the metadata about a KML file named “sea_elephants”, as described in the tag `<kmlFileName>`. This document contains the metadata about two moving objects and each one is described by the tag `<MovingObject>`.

The first moving object is a moving point, as described in the tag `<type>`, and its name or identity is “2340”, as described in the tag `<name>`. Its observations are grouped together in a folder (`kml::FolderType`) named “40: locations”, as described in the tag `<ObservationContainer>`.

The second moving object is also a moving point, as described in the tag `<type>`, and its name or identity is “2341”, as described in the tag `<name>`. Its observations are grouped together in a folder (`kml::FolderType`) named “41: locations”, as described in the tag `<ObservationContainer>`.

An advantage of using a metadata file, as proposed in this paper, is that a user can extract a subset of moving objects from a KML file. For example, the “sea_elephants” file contains trajectories of more than two animals, but its metadata file, presented in Fig. 4, describes only two of them. So, by using this metadata file, only these two animals will be extracted from the “sea_elephants” KML file.

V. Software Module

This section describes a software module that is being developed in an open source geographical library called TerraLib [5] to provide functionalities for moving object analysis. This module has two components, **MovingObjectLoader** and **MovingObjectAnalyser**, as presented in Fig. 5.

The **MovingObjectLoader** component is responsible for extracting moving object trajectories from KML files, following the proposed strategy. It receives two files: (1) a KML file that contains observations of trajectories, and (2) an XML file that contains the metadata about how to interpret the data stored in the KML file as moving objects. The second file must be compliant with the XML schema proposed in the previous section. Based on these two files, this component can extract the moving object trajectories from any KML file and complete the data structures and types needed by the analyser module.

The **MovingObjectLoader** component uses two well-known software libraries internally, OGR and Xerces. OGR is a C++ open source library that provides read (and sometimes write) access to a variety of geographical vector file formats, including KML files (http://www.gdal.org/ogr/). This component uses the OGR LIBKML Driver to read KML files (http://www.gdal.org/ogr/drv_libkml.html).

Xerces-C++ (http://xerces.apache.org/xerces-c/) is a C++ open source library that is able to read and write XML data. This library is used by this component to read the XML metadata file.

The **MovingObjectAnalyser** component provides a set of functions to process and analyse moving objects. For example, there are functions to calculate the distance between two moving objects over time and the intersection between a moving object and a region of interest.

By using the functions provided by this component, a user can answer questions like: (1) *Where was object o_1 at time t_5?* (2) *When did object o_1 enter a specific region r_{10} and how long did it stay in this region?* (3) *When and where did objects o_1 and o_2 meet each other (considering a meeting when the distance between two objects is less than 2 meters)?* (4) *Where and when was there a spatiotemporal cluster of objects?*

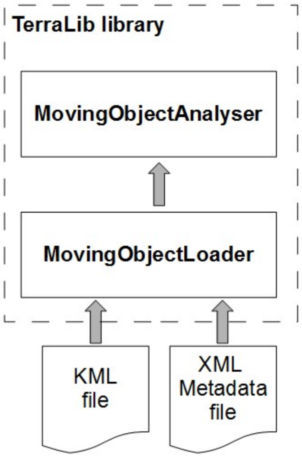


Fig. 5 Software module.

VI. Conclusion

This paper proposes an interoperable way to extract moving object trajectories from KML files, based on the processing of an additional metadata file. This metadata file describes how the moving object trajectories are represented in a KML file and must be compliant with an XML schema proposed in this work. This proposal has been implemented in a geographical software library called TerraLib.

The main advantages of our proposal are: (1) the ability to extract moving object trajectories from any KML file that stores journeys; (2) the capacity of extracting a subset of trajectories from a KML; and (3) the ability to process and analyze the extracted moving object trajectories.

References

[1] M. Erwig, R. H. Guting, M. Schneider, M. Vazirgiannis, "Spatio-Temporal Data Types: An Approach to Modeling and Querying Moving Objects in Databases". *GeoInformatica*. vol. 3, pp. 265-291, 1999.

[2] R. H. Guting and M. Schneider, *Moving Objects Databases*, San Francisco, Morgan Kaufmann, 2005.

[3] N. Pelekis, E. Frentzos, N. Giatrakos, Y. Theodoridis, "HERMES: Aggregative LBS via a Trajectory DB Engine" in *Proceedings of the ACM SIGMOD' 08 Conference*. Vancouver, BC, Canada, 2008.

[4] Open Geospatial Consortium (OGC), "OGC KML", reference number: OGC 07-147r2. [Online]. Available at: http://www.opengeospatial.org. 2008.

[5] G. Câmara, L. Vinhas, K. Ferreira, G. Queiroz, R. C. Souza, A. M. V. Monteiro, M. T. Carvalho, M. A. Casanova, U. M. Freitas, "TerraLib: An Open Source GIS Library for Large-scale Environmental and Socio-economic Applications". *Open Source Approaches to Spatial Data Handling*, Berlin, Springer-Verlag., 2008.

[6] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, F. Yergeau, "Extensible Markup Language (XML) 1.0 (Fifth Edition)", W3C recommendation. Technical report, W3C, 2008.

[7] D. C. Fallside and P. Walmsley, "XML Schema Part 0: Primer (Second Edition)", W3C recommendation. Technical report, W3C, 2004.

[8] S. Spaccapietra, C. Parent, M. Damiani, J. A. F. Macedo, F. Porto, C. Vangenot, "A conceptual view on trajectories", *Data & Knowledge Engineering*, vol. 65, pp. 126-146, 2008.