Open Geospatial Consortium

Publication Date: 2016-01-18

Approval Date: 2015-09-17

Posted Date: 2015-08-24

Reference number of this document: OGC 15-096

Reference URL for this document: http://www.opengis.net/doc/DP/mf-app

Category: Discussion Paper

Editor(s): Akinori Asahara, Hideki Hayashi, Carl Reed

Use Cases and Applications of the OGC Moving Features Standard: The Requirements for a Moving Feature API

Warning

This document is not an OGC Standard. This document is an OGC Discussion Paper and is therefore not an official position of the OGC membership. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard. Further, an OGC Discussion Paper should not be referenced as required or mandatory technology in procurements.

Document type:	OGC® Discussion Paper
Document subtype:	NA
Document stage:	Approved for public release
Document language:	English

License Agreement

Contents

Figures

Page

Tables

Page

Abstract

This OGC Discussion Paper provides examples of some actual and potential geospatial applications using the OGC Moving Features encoding. These applications can be used to define the next steps in the development of the OGC Moving Features Standard: The definition of a "Moving Features API". As a conclusion, the Moving Features SWG recommends that a new Moving Features API standard should target the following three kinds of operations: retrieval of feature information, operations between a trajectory and a geometric object, and operations between two trajectories. Additionally, the Moving Features SWG recommends establishing an abstract specification for these three kinds of operations because only a part of operations for trajectories is defined by ISO 19141:2008 - Schema for moving features.

Keywords

ogc, ogcdoc, Moving Features, mf, use cases, api

Use Cases and Applications of the OGC Moving Features Standard: The Requirements for a Moving Feature API

1 Introduction

1.1 Scope

This OGC Discussion Paper provides examples of some actual and potential geospatial applications using the OGC Moving Features encoding. These applications can be used to define the next steps in the development of the OGC Moving Features Standard: The definition of a "Moving Features API". In the first half of this paper, past discussions on OGC Moving Features encoding are summarized to clarify the scope of the current OGC Moving Features encoding. In the second half, actual and potential applications of OGC Moving Features are described to specify requirements for the next revision and/or enhancement to the OGC Moving Features Standard.

1.2 Document contributor contact points

All questions regarding this document should be directed to the editor or the contributors:

Name	Organization
Akinori Asahara	Hitachi Ltd.
Hideki Hayashi	Hitachi Ltd.
Carl Reed	Carl Reed and Associates

1.3 Future work

Improvements in this document are desirable to adding more applications.

1.4 Forward

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium shall not be held responsible for identifying any or all such patent rights.

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

2 References

The following documents are referenced in this document. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. For undated references, the latest edition of the normative document referred to applies.

ISO19141:2008. *Geographic information -- Schema for moving features*. (http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=41445)

OGC 06-121r3, *OGC® Web Services Common Standard 2.0* (http://portal.opengeospatial.org/files/?artifact_id=38867)

NOTE This OWS Common Standard contains a list of normative references that are also applicable to this Implementation Standard.

OGC 14-083r2. *OGC Moving Features Encoding Part 1: XML Core* (http://docs.opengeospatial.org/is/14-083r2/14-083r2.html)

OGC 14-084r2. *OGC Moving Features Encoding Extension: Simple Comma Separated values* (http://docs.opengeospatial.org/is/14-084r2/14-084r2.html)

In addition to this document, this report includes several XML Schema Document files as specified in Annex A.

3 Terms and definitions

For the purposes of this report, the definitions specified in Clause 4 of the OWS Common Implementation Standard [OGC 06-121r3] and in ISO19141:2008 *Geographic information -- Schema for moving features* [ISO19141:2008] shall apply. In addition, the following terms and definitions apply.

4 Conventions

4.1 Abbreviated terms

API	Application Program Interface
GIS	Geographic Information System
MF	Moving Features
COTS	Commercial Off The Shelf

4.2 Used parts of other documents

This document uses significant parts of document [OGC 14-083r2]. To reduce the need to refer to that document, this document copies some of those parts with small modifications. To indicate those parts to readers of this document, the largely copied parts are shown with a light grey background (15%).

5 OGC Moving Features encoding

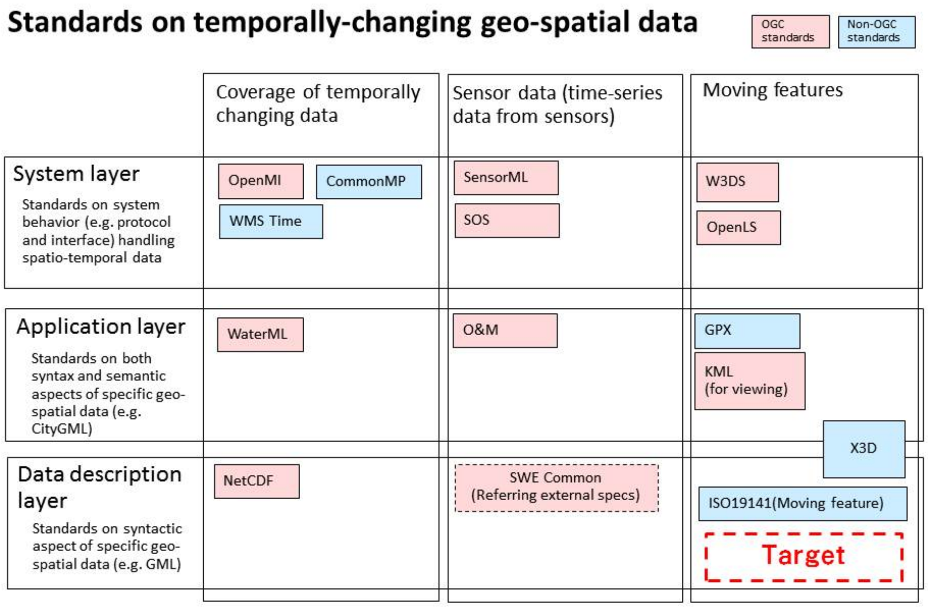
5.1 Demands for standards

Demand is rapidly increasing in the GIS community for better handling of massive volumes of moving feature data. Example applications for moving feature data include:

- Traffic congestion information services using probe cars or taxis equipped with GPS to measure the travel time of each road link,
- Tracking systems on auto-trucks for logistics management, and agent-based road traffic simulation systems for forecasting traffic situations.

Systems relying on single-source moving feature data are now evolving into more integrated systems. Integration of moving feature data from different sources is a key to developing more innovative and advanced applications. Section 6 provides examples of such efforts.

Moreover, the growth of location enabled smart phones makes it much easier to acquire large amounts of data on user trajectories reflecting the movement of people and vehicles on a global scale and in real-time. This will create a market for geospatial applications that requires the integration of moving feature data from many heterogeneous sources with a GIS platform.


Standards on temporally-changing geo-spatial data
OGC standards
Non-OGC standards
Coverage of temporally changing data
Sensor data (time-series data from sensors)
Moving features
System layer
Standards on system behavior (e.g. protocol and interface) handling spatio-temporal data
OpenMI
CommonMP
WMS Time
SensorML
SOS
W3DS
OpenLS
Application layer
Standards on both syntax and semantic aspects of specific geo-spatial data (e.g. CityGML)
WaterML
O&M
GPX
KML (for viewing)
X3D
Data description layer
Standards on syntactic aspect of specific geo-spatial data (e.g. GML)
NetCDF
SWE Common (Referring external specs)
ISO19141(Moving feature)
Target

■CommonMP:
Common platforms for numerical calculation developed in Japan
http://framework.nilim.go.jp/eng/index.html
■WMS Time:
WMS interface implementation used by MapServer
http://mapserver.org/ogc/wms_time.html

■GPX:
GPS eXchange format
http://www.topografix.com/gpx.asp
■X3D:
http://www.web3d.org/x3d/

Figure 5.1: Existing Standards and OGC Moving Features

The OGC Moving Features encoding standard was developed to support the growing market for moving feature data. Figure 5.1 summarizes standards that can be used in temporally changing geospatial data applications. Many standards for coverage data and sensor observation data have been established and widely implemented. However, the only international standard for moving features is ISO19141:2008, which is defines an abstract model. OGC Moving Features was developed as an implementation standard based on ISO19141:2008.

Figure 5.2 diagrams the modularity of the OGC Moving Features standard. Moving Features core XML is a fundamental standard designed for easy extensibility. Moving Features Simple CSV is encoding style designed to reduce data size even if large volumes of data are encoded. Moving Features 1D/2D and Moving Features 3D in the figure are standards for adding shapes of features to Moving Features core XML. They will be defined in a future version of the MF standard because most of existing use cases require only trajectory data without shapes.

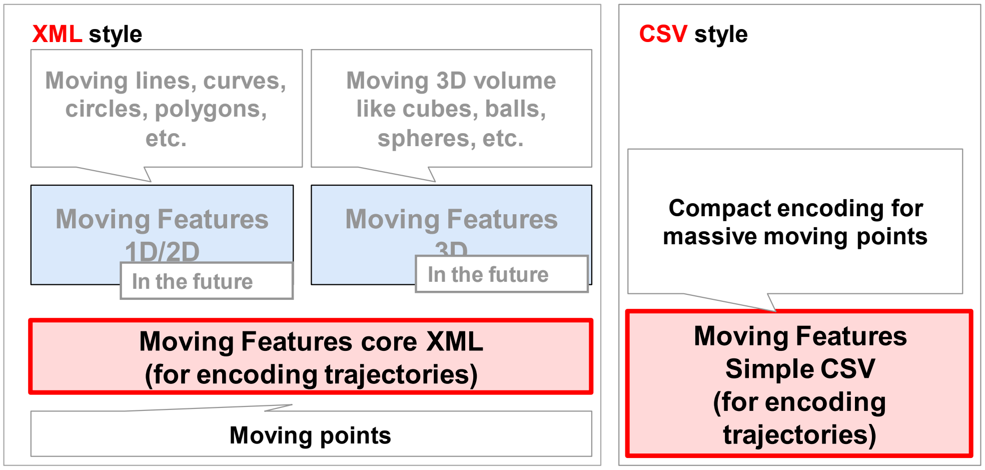

XML style
Moving lines, curves, circles, polygons, etc.
Moving 3D volume like cubes, balls, spheres, etc.
Moving Features 1D/2D
In the future
Moving Features 3D
In the future
Moving Features core XML (for encoding trajectories)
Moving points
CSV style
Compact encoding for massive moving points
Moving Features Simple CSV (for encoding trajectories)

Figure 5.2: Modularity of OGC Moving Features

5.1.1 Extensible Encoding Style: OGC Moving Features XML Core

The scope of the OGC Moving Features encoding is illustrated in Figure 5.3. The OGC Moving Features core XML standard defines an XML element to encode line-string like tracks. The other types of curves such as spline curve, constant acceleration, and circular movement will be defined as extensions in the future. Types of features in scope of the current Moving Features Simple CSV standard are constrained by the scope of Moving Features core XML. That is, Moving Features core XML can always encode features which are encoded by Moving Features Simple CSV.

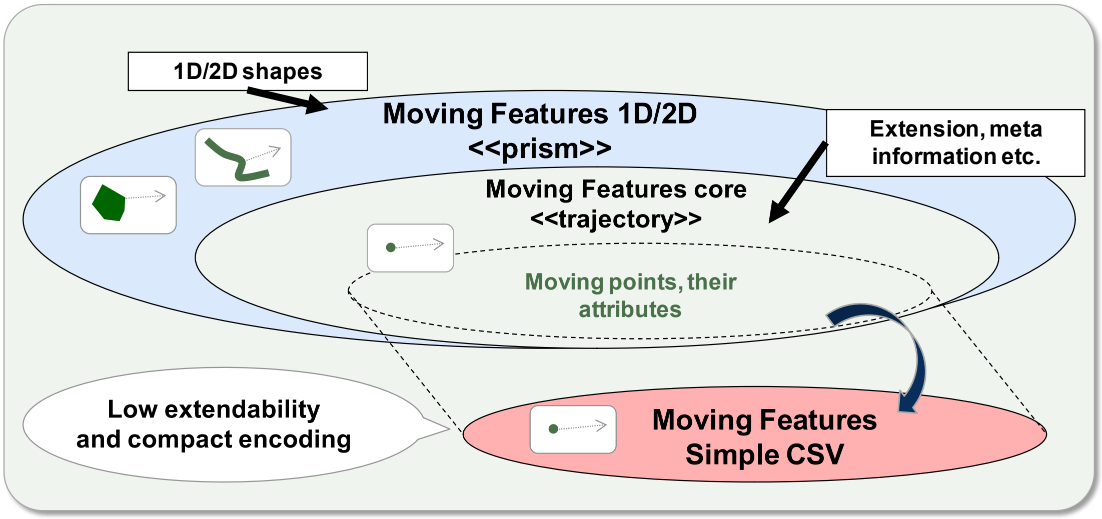

1D/2D shapes
Moving Features 1D/2D
<<prism>>
Extension, meta information etc.
Moving Features core
<<trajectory>>
Moving points, their attributes
Low extendability and compact encoding
Moving Features Simple CSV

Figure 5.3: Scopes of OGC Moving Features Encoding styles

XML elements to describe trajectories are shown in Figure 5.4. mf:AbstractTrajectory is a fundamental superclass for all types of trajectories (as if gml:AbstractCurve with an attribute to specify the temporal interpolation). mf:LinearTrack is used for line-string like tracks.

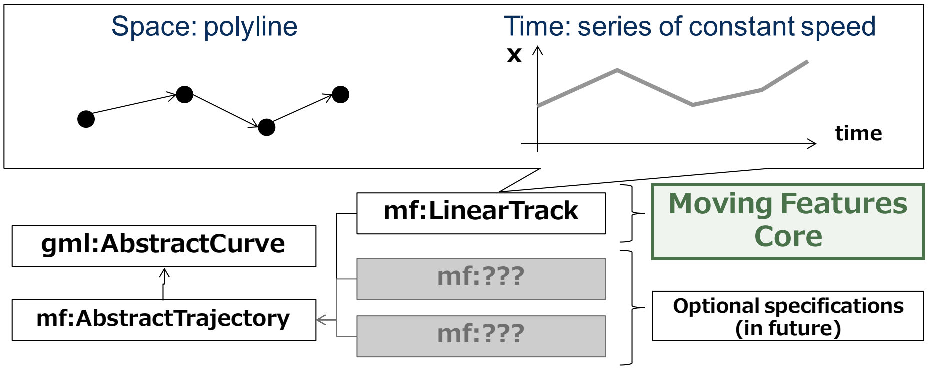


Figure 5.4: Trajectory classes in Moving Features XML Core

5.1.2 Simple Encoding Style: OGC Moving Features Simple CSV

The OGC Moving Features Simple CSV standard was developed to reduce the size of moving features data encoded by OGC Moving Features core XML. Table 5.1 compares typical encoding styles. 'Data size' column lists roughly estimated data sizes in the table (600,000 features, 1440 points in a day for every feature). According to the comparison, data sizes by text and binary are half of that by JSON and XML. In addition, software to handle data encoded by the text style is easily developed. Text encoding style thus was developed as the first step.

Table 5.1: Rough estimation of data sizes

Encoding style	Data size	Pros	Cons
Text	69.12GB	Simple format (readable)	
Binary	51.84GB	Small data size High performance of parsing	Complicated parser
GeoJSON	184.9GB	Java Script parser is applicable	Large data size
XML core	164GB	XML parser is available for parsing	Large data size

5.1.3 Existing issues

The OGC Moving Features core XML standard defines a model to encode moving features. The OGC Moving Features Simple CSV standard was developed for practical and compact encoding. The functions provided by the standards satisfy requirements for simple use cases. Moreover, the following standards will enable more types of applications:

- Binary encoding for larger datasets
- Data exchange API using the encoding styles

6 Applications

6.1 Transportation survey with smart phones

Traffic congestion is a serious problem in many large cities. To solve such problems, two types of solutions are generally applied: road-construction and public transportation enhancements. Either solution requires travel demand data, which is the number of movements between places. This is because roads and public transportation should make connection between pairs of places with the highest trip demand. Therefore, travel demand data are collected using transportation surveys.

Questionnaires are traditionally used for the trip generation data collection. However, with the broad availability of location enabled mobile devices, GPS based data collection of trip information is applicable as the alternative. In the latter case, the GPS tracks are encoded by using the Moving Features standard to enable sharing by many stakeholders such as local governments, bus companies, and so on.

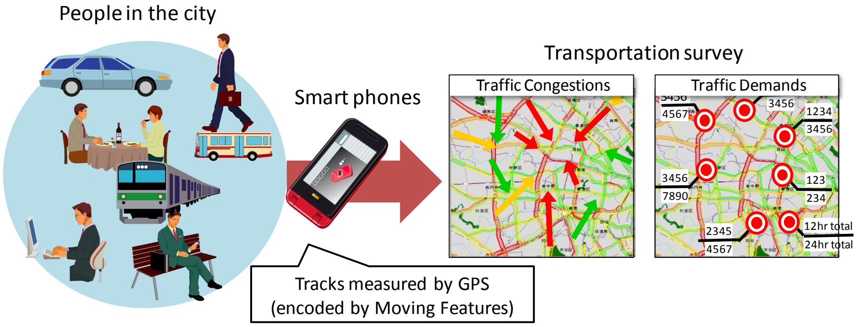
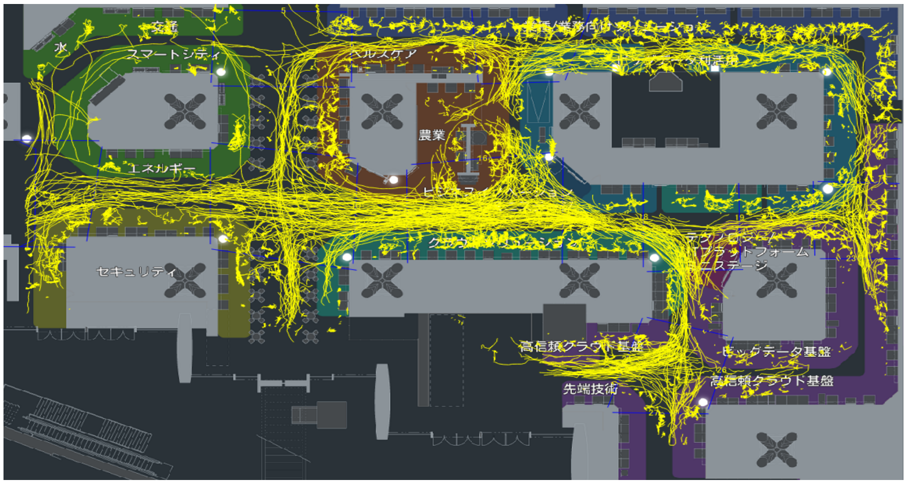


Figure 6.1: Transportation Survey

Background map: ©Open Street Map

6.2 Layout design with LIDAR

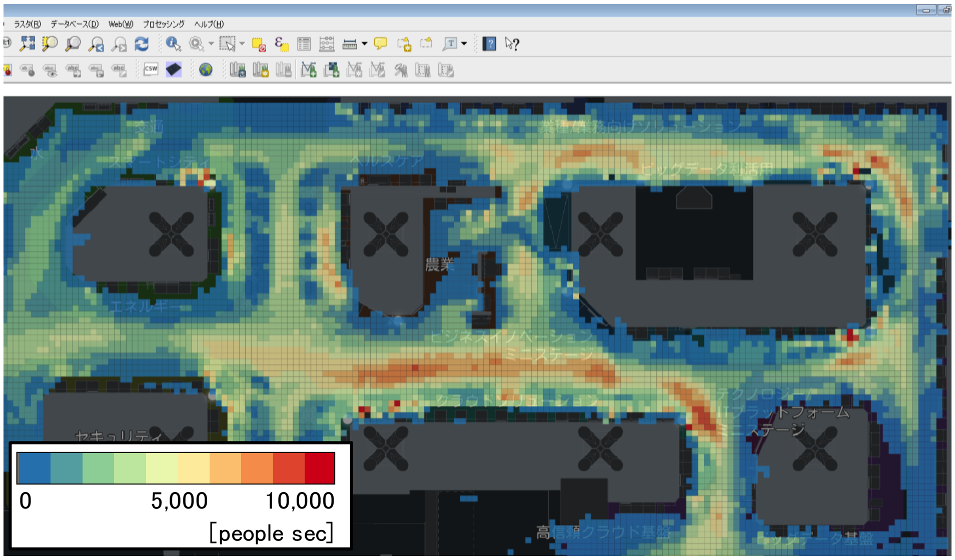
LIDAR (Laser Imaging Detection and Ranging) is a technique useful for detecting and tracking pedestrians [2][3][4]. In this use case, a pedestrian-tracking system using LIDAR is used to understand pedestrian movement behavior in a large facility such as shopping malls and train terminals.



©Hitachi, Ltd.

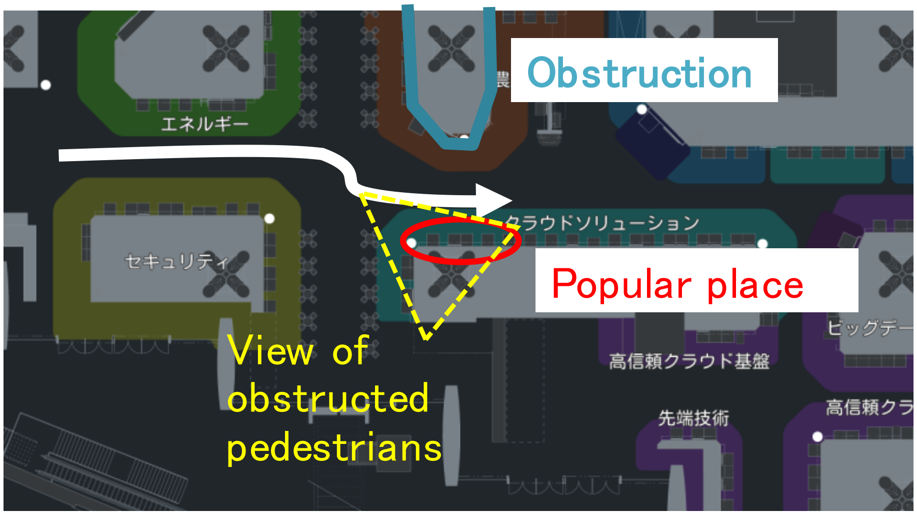
Figure 6.2: Tracks in Exhibition Event (2013)

In Figure 6.1, pedestrian tracks obtained by LIDAR at a 2013 exhibition event are shown by the yellow lines. Such trajectory data are recorded and stored to a spatio-temporal database. The data can then be output as a data file encoded using the OGC Moving Features standard to send to another trajectory analysis system. Figure 6.2 shows population distribution of the pedestrian tracks, which was calculated by the trajectory analysis system. This population map can be calculated by counting the number of people in each grid.



©Hitachi, Ltd.

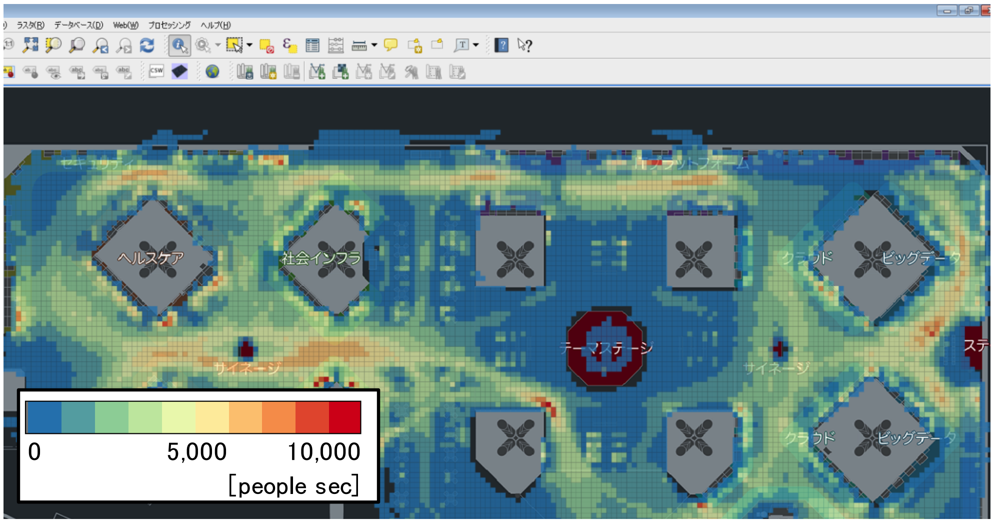
Figure 6.3: Population map (2013)



©Hitachi, Ltd.

Figure 6.4: Causal analysis

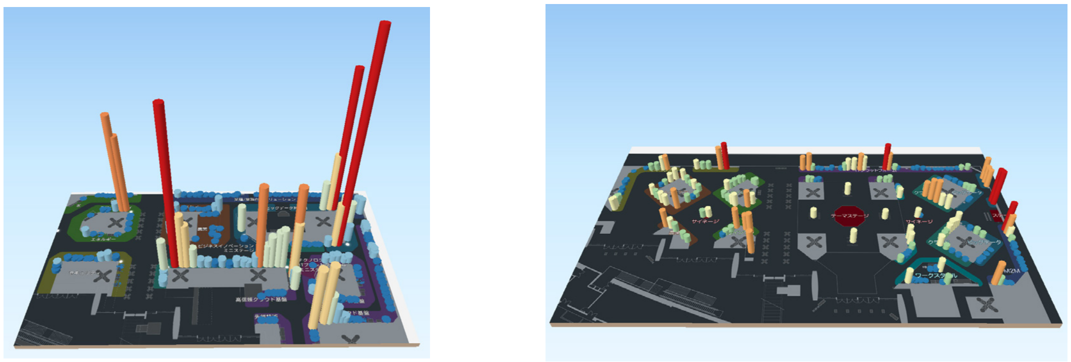
Popular booths could be determined by determining how many pedestrians at the event visited a booth and how long they stayed around the booth. Figure 6.3 illustrates the causal analysis to find such popular spots. As shown in the figure, flows of pedestrians should be bent by obstructions in order to make pedestrians stay at exhibition booths. According to the analysis, the exhibition event at the next year (2014) was designed to make more obstructions. Figure 6.4 shows a population map at 2014 as the result: pedestrian-dense areas are widely distributed to field of the event.



©Hitachi, Ltd.

Figure 6.5: Population map (2014)

The durations of stay by visitors at booths are compared in Figure 6.5 in terms of height of cylinders. At a glance, it is confirmed that the durations of stay of visitors at 2014 were highly equalized in comparison to 2013.



2013

2014

©Hitachi, Ltd.

Figure 6.6: Contrast between "before" (2013) and "after" (2014)

The duration can be calculated by summing duration of trajectories around each booth, where a trajectory is "around a booth" if distance between the booth and the track is less than 3 meters.

6.3 Disaster Management

As an example of using MF for disaster management, an estimation system of tsunami damage partly implemented by the authors of this OGC Discussion Paper is introduced. A tsunami is a very large wave typically generated by a sudden up-thrust or sinking of the sea bed in association with an undersea earthquake, and causes serious damage along the coastal regions. Huge tsunamis have recently caused serious damage in Sumatra, Indonesia in 2004 and Tohoku, Japan in 2011. In order to reduce the amount of damage from tsunamis, measures and policies are required, such as the construction of breakwaters, the designation of evacuation areas, and the provision of evacuation guidance. A simulation for tsunami evacuation scenarios is helpful for creating measures and formulating policies.

The simulation system for tsunami evacuation facilitates well-informed decision making for the appropriate allocation of tsunami evacuation buildings by integrating tsunami simulations and people flow or evacuation simulations. Tsunami simulations precisely estimate the flooded areas when a tsunami hits a coastal area. The estimation is based on the topography of the sea bed and land. The people flow or evacuation simulation estimates the location of individuals when an earthquake and possible tsunami occurs and computes the evacuation movements to the nearest tsunami evacuation building. The system computes how many people could successfully evacuate under different scenarios to evaluate the effectiveness of the evacuation guidance, tsunami warning services, location/capacity of tsunami evacuation buildings among others. Figure 6.6 is a screen shot from the simulation system. The blue to purple color gradation indicates the height of the simulated tsunami. Each point denotes residents (a point for 50 people). The yellow points are evacuating people, while the red ones are people who could not escape. The green squares are tsunami evacuation buildings. The system requires the functionality of exchanging temporally-changing inundation area/depth and moving features in a three-dimensional space.

For integrated simulation for disaster risk management, location data is collected and integrated from different simulation systems such as people evacuation simulations, road vehicle simulations including emergency vehicles, and tsunami simulations. Many of such location-based simulations, except the tsunami simulations, are agent-based simulation systems that explicitly output the trajectories of individual agents, i.e., pedestrians and vehicles. It is necessary to collect location data from cellular phones in order to increase accuracies of these simulations.

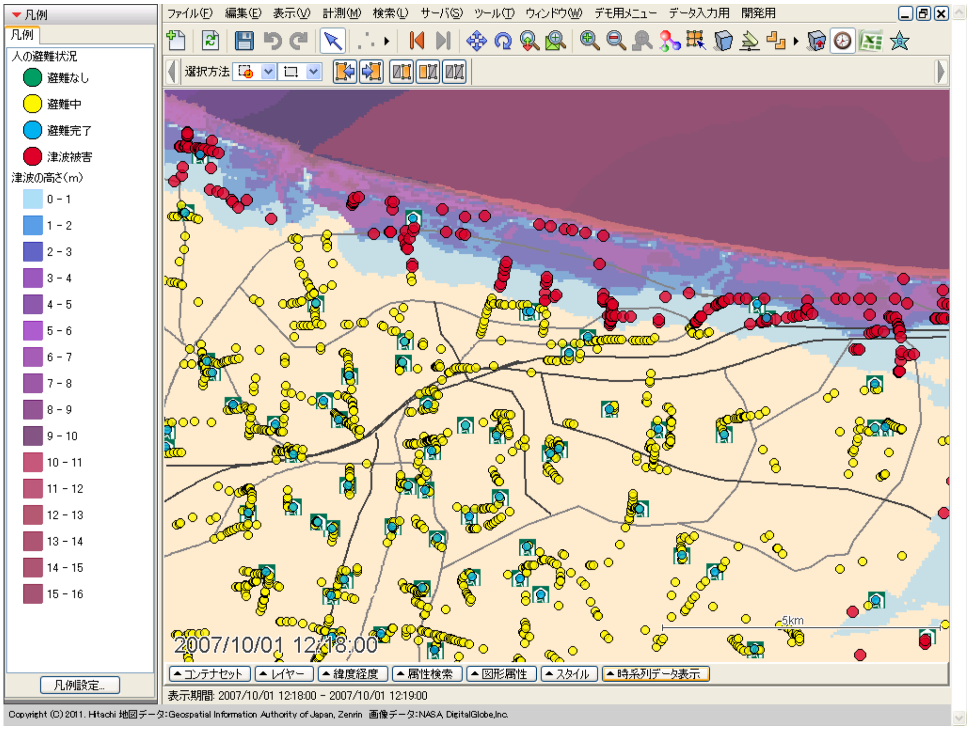

凡例
凡例
人の避難状況
● 避難なし
● 避難中
● 避難完了
● 津波被害
津波の高さ(m)
0 - 1
1 - 2
2 - 3
3 - 4
4 - 5
5 - 6
6 - 7
7 - 8
8 - 9
9 - 10
10 - 11
11 - 12
12 - 13
13 - 14
14 - 15
15 - 16
凡例設定...
ファイル(F) 編集(E) 表示(V) 計測(M) 検索(L) サーバ(S) ツール(T) ウィンドウ(W) デモ用メニュー データ入力用 開発用
選択方法
2007/10/01 12:18:00
5km
コンテナセット
レイヤー
緯度経度
属性検索
図形属性
スタイル
時系列データ表示
表示期間: 2007/10/01 12:18:00 - 2007/10/01 12:19:00
Copyright (C) 2011. Hitachi 地図データ:Geospatial Information Authority of Japan, Zenrin 画像データ:NASA, DigitalGlobe,Inc.

Figure 6.7: Example of Simulation of Evacuation from Tsunami for Coastal City

©Hitachi, Ltd.

6.4 Maritime sector use case

Several devices have been used by vessels to track their positions. Some of these devices are mandatory as requested by regional/international standardization initiatives (i.e. EU Directives, IMO conventions, etc …).

The objectives for tracking the position of the vessels are twofold: increase the safety and security within the maritime sector.

In order to address maritime use cases, the Moving Feature standard has to include at least the following information: (i) ship position, which can be provided by different data source (AIS, LRIT, etc …); and (ii) ship voyage, which describe the track of a ship. Additional information about the vessel incidents and the ship particulars has to be considered optional and therefore as possible extensions to the Moving Feature standard. For example, the following figure summarize the information collected by the Canonical Data Format produced by the European Maritime Safety Agency

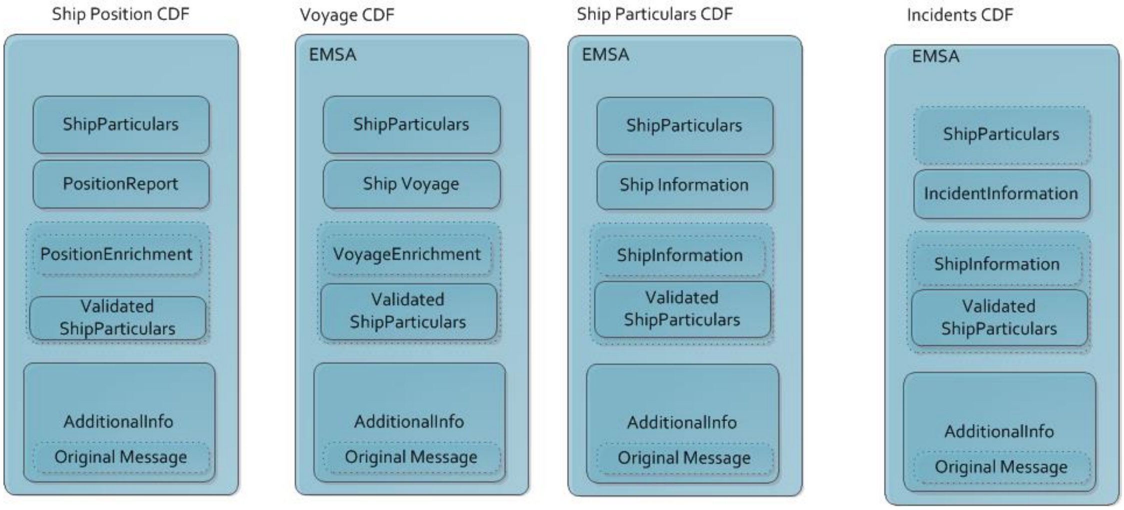
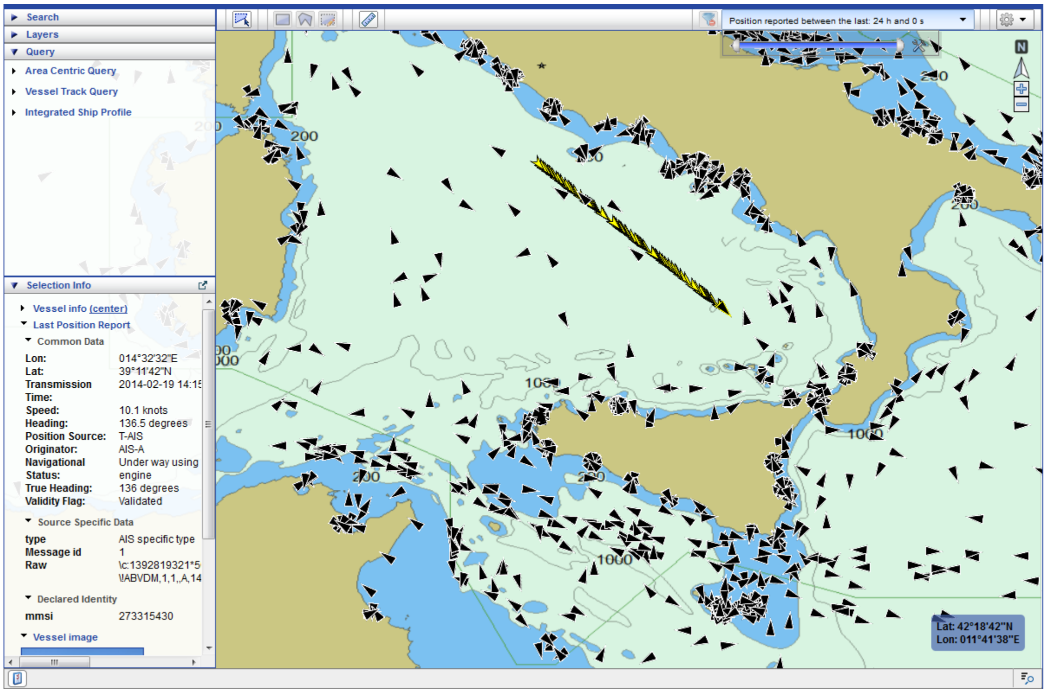


Figure 6.8: Canonical Data Format

The following map reports the last position of vessels within a specific area (black triangle), and the track of a specific vessel (yellow triangles). In this case the yellow triangles report the position of a specific vessel every 6 minutes in the last 24 hours. Furthermore additional information as: heading, true heading and speed is recorded to enrich the description of the vessel track.



Lon:	014°32'32"E
Lat:	39°11'42"N
Transmission Time:	2014-02-19 14:15
Speed:	10.1 knots
Heading:	136.5 degrees
Position Source:	T-AIS
Originator:	AIS-A
Navigational Status:	Under way using engine
True Heading:	136 degrees
Validity Flag:	Validated
type	AIS specific type
Message id	1
Raw	\c:1392819321*5 \!ABVDM,1,1,,A,14
mmsi	273315430

Figure 6.9: vessels positions map

A vessel has a unique identification at worldwide scale (IMO number, or MMSI). The position of a vessel (ship position) is a point feature over the time. The voyage feature of a vessel (ship voyage) is characterized by several attributes: speed over the water, course over the water, heading and true heading.

In conclusion, to build the trajectory of a vessel two main abstract feature types are relevant for Moving Feature standard: ship position and ship voyage.

6.5 Aviation sector use case

Aircraft and other airborne vehicles are moving features and the aviation sector has a (large) number of methods to track aircraft and to represent their position.

The main sources of tracking data include the following

- Primary surveillance radar: Radar equipment measuring position and heading of aircraft.
- Secondary surveillance radar: position detection is augmented with an active response from the aircraft's transponder. It supplies additional information such as its identity.
- ADS-B: aircraft with on board GPS or other GNSS equipment may also provide position information based on the GPS measurement. This mode of position and

information broadcasting is called ADS-B and is seen as an important future tool for increasing air traffic capacity and safety.

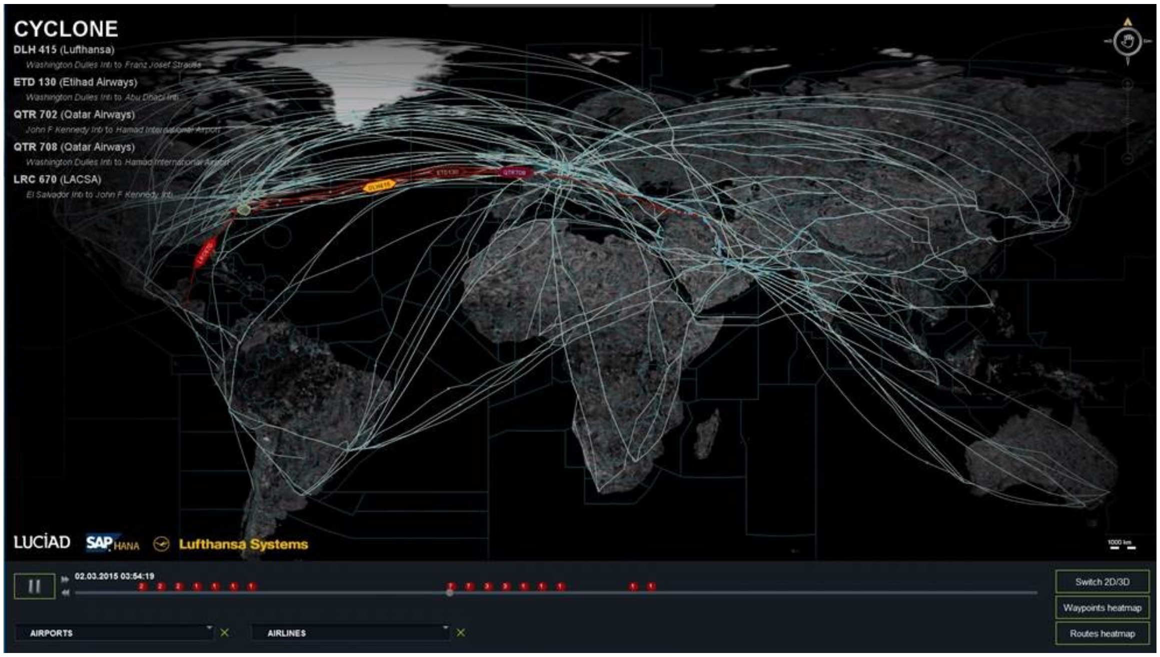

CYCLONE
DLH 415 (Lufthansa)
Washington Dulles Intl to Franz Josef Strauss
ETD 130 (Etihad Airways)
Washington Dulles Intl to Abu Dhabi Intl
QTR 702 (Qatar Airways)
John F Kennedy Intl to Hamad International Airport
QTR 708 (Qatar Airways)
Washington Dulles Intl to Hamad International Airport
LRC 670 (LACSA)
El Salvador Intl to John F Kennedy Intl
LUCIAD
SAP HANA
Lufthansa Systems
1000 km
02.03.2015 03:54:19
AIRPORTS
AIRLINES
Switch 2D/3D
Waypoints heatmap
Routes heatmap

Figure 6.10: Flight Management and Visualization (http://vimeo.com/133130574)

The position and related data of aircraft is encoded according to different standards. One standard is Eurocontrol ASTERIX (http://www.eurocontrol.int/asterix). ASTERIX defines different categories. Most categories contain position data according to a different detection method.

For instance, category 48 defines the following fields (copied from the Cat 48 specification on http://www.eurocontrol.int/services/specifications-documents)

Data Item	Description	Description System Units
I048/010	Data Source Identifier	N.A.
I048/020	Target Report Descriptor	N.A.
I048/030	Warning/Error Conditions	N.A.
I048/040	Measured Position in Slant Polar Co-ordinates	RHO: 1/256 NM, THETA: 360°/(2 16)
I048/042	Calculated Position in Cartesian Co-ordinates	X, Y: 1/128 NM
I048/050	Mode-2 Code in Octal Representation	N.A.
I048/055	Mode-1 Code in Octal Representation	N.A.
I048/060	Mode-2 Code Confidence Indicator	N.A.
I048/065	Mode 1 Code Confidence Indicator	N.A.
I048/070	Mode-3/A Code in Octal Representation	N.A.

I048/080	Mode-3/A Code Confidence Indicator	N.A.
I048/090	Flight Level in Binary Representation	1/4 FL
I048/100	Mode-C Code and Confidence Indicator	N.A.
I048/110	Height Measured by a 3D Radar	25 ft
I048/120	Radial Doppler Speed	(2-14) NM/s
I048/130	Radar Plot Characteristics	N.A.
I048/140	Time of Day	1/128 s
I048/161	Track/Plot Number	N.A.
I048/170	Track Status	N.A.
I048/200	Calculated Track Velocity in Polar Representation	Speed: (2-14) NM/s, Heading:360°/(2 16)
I048/210	Track Quality	N.A.
I048/220	Aircraft Address	N.A.
I048/230	Communications / ACAS Capability and Flight Status	N.A.
I048/240	Aircraft Identification	N.A.
I048/250	Mode S MB Data	N.A.
I048/260	ACAS Resolution Advisory Report	N.A.

The items in bold relate to the position, heading, time, and identity of the aircraft. Other information fields provide metadata on quality, data source, etc.

Other ASTERIX categories provide different sets of information where position and other information may also be encoded in different ways. Some interesting aspects of ASTERIX:

- The positions may be expressed in relation to the radar equipment position. A (separately available) position of the radar is thus needed to correctly geo-reference the aircraft.

- The data has a binary encoding optimized to obtain a small message size. Verbose GML/XML/JSON encoding may be a barrier for adoption of the Moving Feature standard for large data sets in the aviation world.

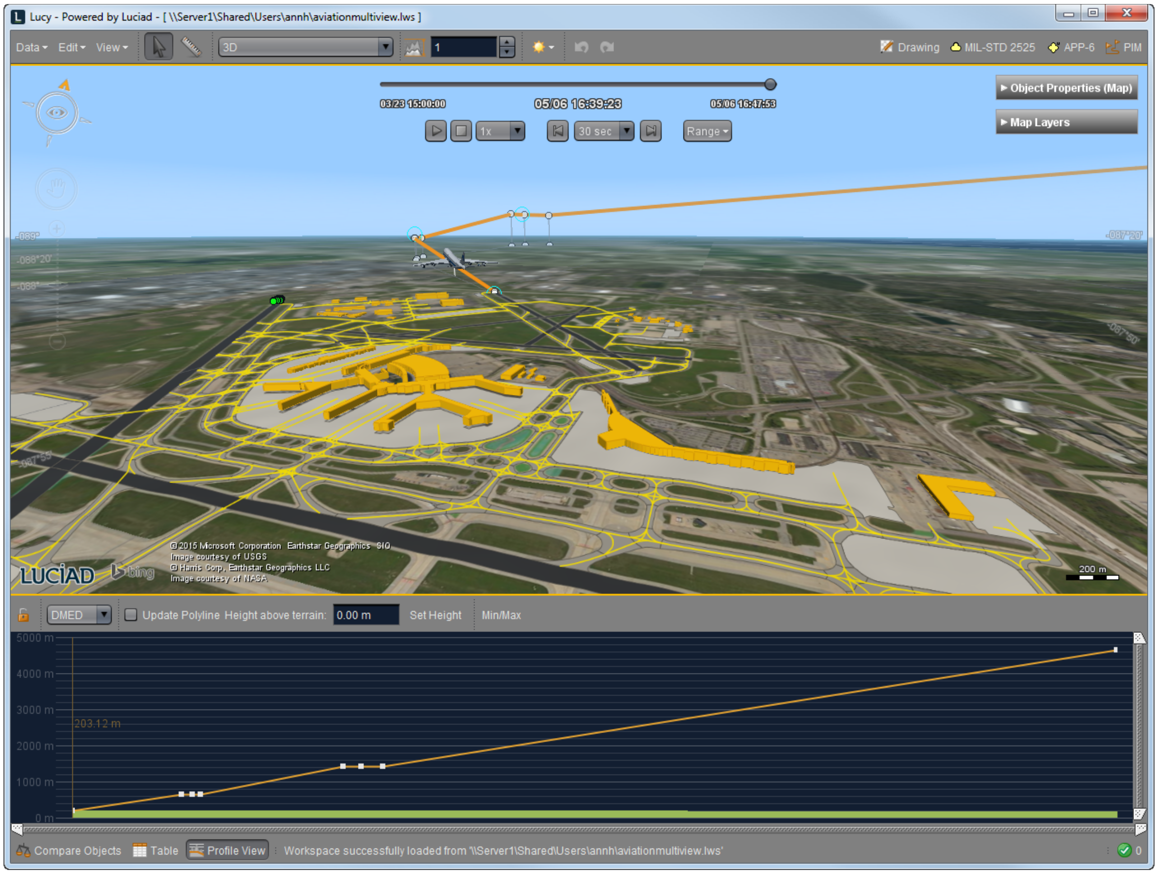
- Many data sets do not include all fields. Optional elements in the Moving Features may accommodate representing such data sets.

- Some categories or data sets do not include an identification of the moving features. Such data is often referred to as ‘plots’. Since multiple plots cannot be related to a single aircraft, the data set basically is a bag of time-stamped points.

FAA has an Aircraft Situation Display to Industry (ASDI) feed which provides aircraft position reports as a convenient data feed to industry. The feed also includes information on flight plans (the planned route of the aircraft). See http://www.fly.faa.gov/ASDI/asdi.html for more information.

In order to address aviation use cases at least this information must be managed in the Moving Feature standards. Other information is important as well and may be encoded as use-case specific metadata.

Use cases can also involve analysis afterwards (incident analysis, statistics, etc.) or simulated air traffic. The Moving Features standard may be useful to convert raw feeds into a standard moving features representation for use in generic analysis and processing tools.


Lucy - Powered by Luciad - [\\Server1\Shared\Users\annh\aviationmultiview.lws]

Workspace successfully loaded from '\\Server1\Shared\Users\annh\aviationmultiview.lws'

Figure 6.11: Flight visualization and analysis

6.6 Moving Feature Use case for Hurricanes

Tropical storms, hurricanes and typhoons are extremely energetic and dangerous weather phenomena with a pronounced centre of rotation, the 'eye'. Their position can be forecast, with increasing accuracy, several days in advance, and even more accurately only a day in advance. These forecasts are normally presented to the general public and emergency services as a simple graphical map, often using PNG or JPEG formats.

Hurricanes and typhoons are the same phenomenon. Traditionally the names for these phenomena in the western Pacific and western Atlantic are different. A tropical storm is a weaker version, causing less damage, but can become more energetic and become a hurricane/typhoon. Currently a wind speed threshold of 65 knots (120Km/hour, 75 miles/hour) is the only distinguishing attribute.

Typically a deterministic chart will show a single forecast storm trajectory as a single track with the location of the storm centre or 'eye' indicated at regular intervals, say every 6 or 12 hours. The location at a specific time is usually marked by specific symbols, standardized across all countries for both meteorology and aviation.

Sometimes a probabilistic chart is preferred, and a range of possible trajectories or an envelope of these trajectories is used. The set of trajectories may come from a single forecast centre using an ensemble of possible forecasts, or the set may be from several forecast centers which all produce slightly different forecasts. Generally, if the spatial, and temporal, spread of the set of trajectories is small, then more confidence is attached to them than if the spread is great.

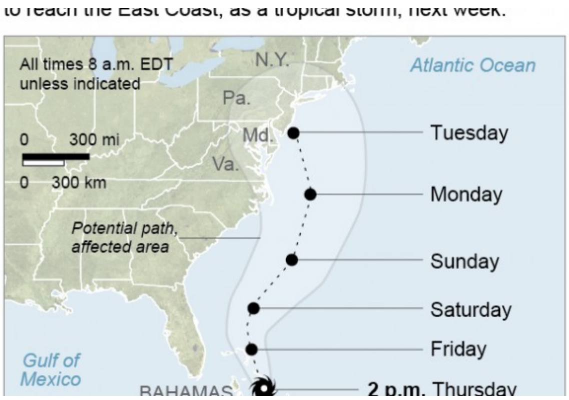
Expert users, such as forecasters, may also use a chart that displays ranges of trajectories from successive forecast times, and the actual trajectory so far. Again, similarity in these successive forecasts gives increased confidence in the accuracy of the forecasts.

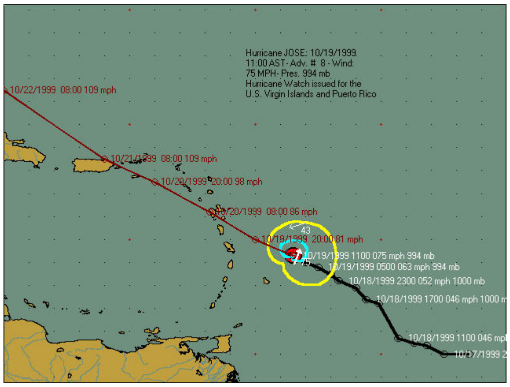
If graphical output is inappropriate, there is a standardized simple text message used to transmit the current and forecast locations across emergency communications channels. These messages also include the actual and forecast wind speeds at various distances and directions from the 'eye'. The atmospheric pressure near the eye may also be included.

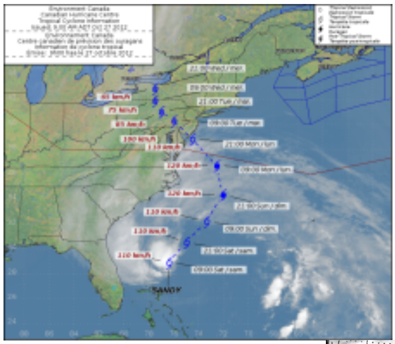
The central pressure is an important indicator of damage, like the wind speed, as the reduced pressure causes the local sea surface to rise by a few meters, causing flooding and destructive waves.

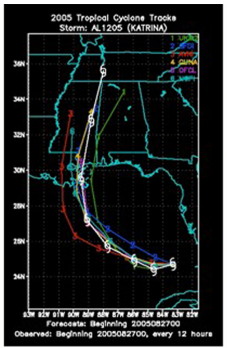
The sea surface temperature in the path of the storm is also important, as this determines whether the storm will increase or decrease in strength.

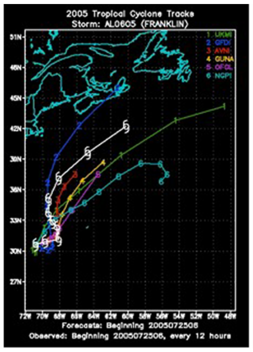
So there are a number of changing attributes need to be associated with the storm and its trajectory.

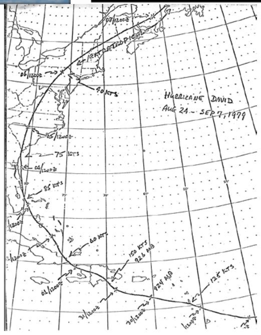

to reach the East Coast, as a tropical storm, next week.


Hurricane JOSE: 10/19/1999
11:00 AST - Adv. # 8 - Wind:
75 MPH - Pres. 994 mb
Hurricane Watch issued for the
U.S. Virgin Islands and Puerto Rico
10/22/1999 08:00 109 mph
10/21/1999 08:00 109 mph
10/20/1999 20:00 98 mph
10/20/1999 08:00 86 mph
43
10/19/1999 20:00 81 mph
10/19/1999 1100 075 mph 994 mb
10/19/1999 0500 063 mph 994 mb
10/18/1999 2300 052 mph 1000 mb
10/18/1999 1700 046 mph 1000 m
10/18/1999 1100 046 mph
10/17/1999 2

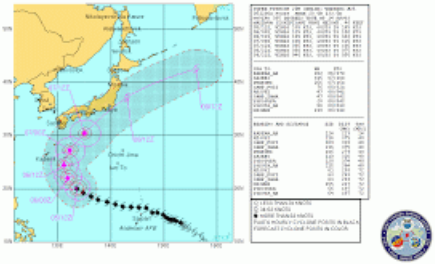



2005 Tropical Cyclone Tracks
Storm: AL1205 (KATRINA)
1 UKMI
2 GFDI
3 AVNI
4 GUNA
5 OFCL
6 NGPI
36N
34N
32N
30N
28N
26N
24N
93W 92W 91W 90W 89W 88W 87W 86W 85W 84W 83W 82W
Forecasts: Beginning 2005082700
Observed: Beginning 2005082700, every 12 hours


2005 Tropical Cyclone Tracks
Storm: AL0605 (FRANKLIN)
1 UKMI
2 GFDI
3 AVNI
4 GUNA
5 OFCL
6 NGPI
51N
48N
45N
42N
39N
36N
33N
30N
27N
72W 70W 68W 66W 64W 62W 60W 58W 56W 54W 52W 50W 48W
Forecasts: Beginning 2005072506
Observed: Beginning 2005072506, every 12 hours



Le 01/01/2014 à 04h
Heures locales La Réunion
Les Comores
Mayotte
Agaléga
BEJISA
Madagascar
01/04h
02/04h
03/04h
04/04h
05/04h
Maurice
La Réunion
Rodrigues
10°00'S
15°00'S
20°00'S
25°00'S



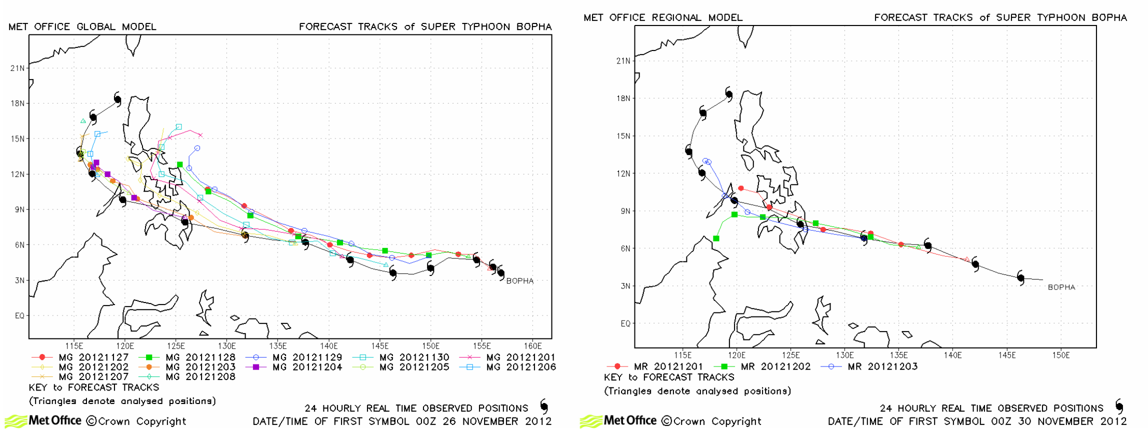

MET OFFICE GLOBAL MODEL
FORECAST TRACKS of SUPER TYPHOON BOPHA
21N
18N
15N
12N
9N
6N
3N
EQ
115E
120E
125E
130E
135E
140E
145E
150E
155E
160E
BOPHA
MG 20121127
MG 20121128
MG 20121129
MG 20121130
MG 20121201
MG 20121202
MG 20121203
MG 20121204
MG 20121205
MG 20121206
MG 20121207
MG 20121208
KEY to FORECAST TRACKS
(Triangles denote analysed positions)
24 HOURLY REAL TIME OBSERVED POSITIONS
Met Office ©Crown Copyright
DATE/TIME OF FIRST SYMBOL 00Z 26 NOVEMBER 2012
MET OFFICE REGIONAL MODEL
FORECAST TRACKS of SUPER TYPHOON BOPHA
21N
18N
15N
12N
9N
6N
3N
EQ
115E
120E
125E
130E
135E
140E
145E
150E
BOPHA
MR 20121201
MR 20121202
MR 20121203
KEY to FORECAST TRACKS
(Triangles denote analysed positions)
24 HOURLY REAL TIME OBSERVED POSITIONS
Met Office ©Crown Copyright
DATE/TIME OF FIRST SYMBOL 00Z 30 NOVEMBER 2012

Figure 6.12: Hurricane tracks

6.7 Predicting Crime patterns by simulating movement of criminals, victims and police on map

Goal: To replicate the conditions under which crime is committed and predict crime patterns, for example predicting regions more vulnerable to crime based on Routine Activity Theory.

Setup: Criminals tend to move only along roads on the map. There are certain fixed marked spots on the map. These include Criminal's Home, Office, Mall, Bar, Restaurants. There is a weekly schedule and the criminal agent moves between these spots based on the schedule.

Forms of Motion:

- Beeline Movement: The criminal agent chooses one of the shortest 3 paths between two spots randomly and moves along it. This is time bound - if there are less than 3 routes that can be covered in the amount of time the agent has - only that number of routes are considered for random selection.

- Random Movement: Agent starts from source and at each road intersection. Agent chooses a road randomly and moves along it. This is also time bound - when time runs out, the agent moves to the destination along the shortest path from its current location.

- Hybrid Movement: Agent moves between source and destination along the shortest path with deviations. Agent randomly chooses two points on the shortest route and takes a longer different route between these two points. If there is time remaining, agent repeats this, otherwise moves to the destination along the shortest path.

Prediction of Crime Patterns: Criminal Agent, Victim Agents and Police all keep moving along the map as per their schedules. There are a number of designated crime spots on the map. When the criminal agent passes through the vicinity of a crime spot, they decide whether or not to commit crime based on factors like profit value associated with crime spot, risk involved, presence of police nearby, absence of guardian of spot and the number of times the spot has been visited before. Crime Patterns are predicted by running this simulation a very large number of times.

Discussion:

- The simulation needs a city description with roads and buildings of interest. CityGML could be used here.

- The simulation needs to represent the agent's trajectory in the above-cited city. Moving Features standard would be used here.

- The trajectory is repeated weekly with random variations. That may be represented as a "base line" moving feature. This base line would not describe a real moving feature, but would be used as a template.

- Interactions between the environment (CityGML) and the moving features may trigger events. For example when <mf:Trajectory> value become close to a CityGML feature or interest having a low "presence of police" attribute, a crime may occur depending on the value of that attribute and other attributes like "absence of guardian", "profit", and "risk".

Example: The criminal agent moves between Home, Office, Bar, Lunch-A, Lunch-B, Mall marked on the map as per a weekly schedule. The result corresponds to the weekly schedule run for 500 times on the city: Indianapolis. The small squares on the map correspond to crime spots. The color of each crime spot represents the number of times crime has been committed at that spot, normalized on a scale of 0-100.

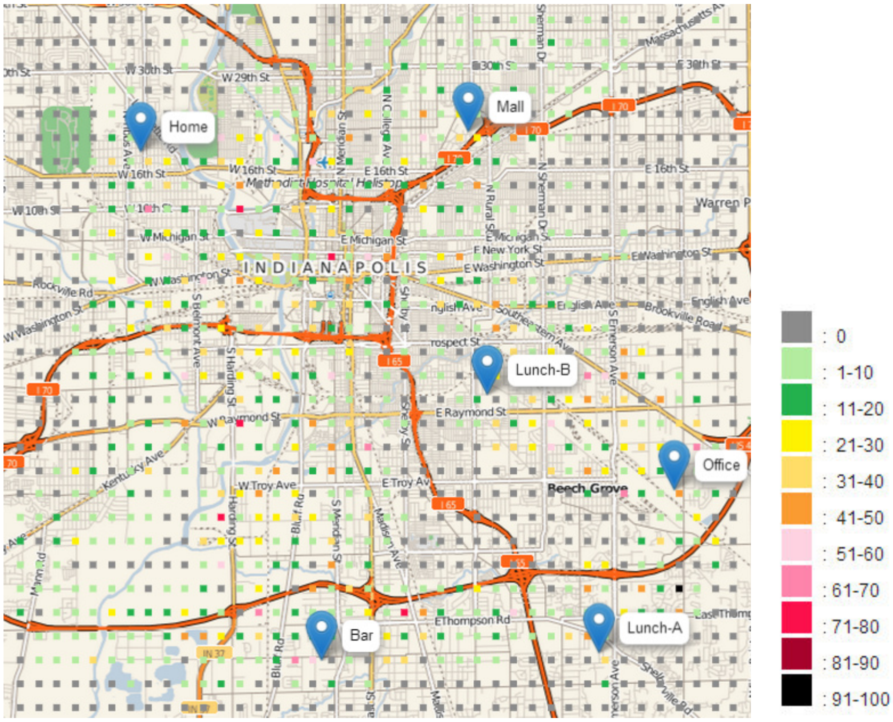


Figure 6.13: Crime spots

6.8 Use-Cases for Soccer Matches

Background: It is an important task of soccer coaches to analyze completed matches for developing proper tactics and preparing for future matches. With recent progress in several video analysis and sensor technologies, it is now possible to gather match data including trajectories of both players and the ball.

Feature Modelling [5]: A soccer match is composed of 23 moving features including 22 players and ball, where each moving feature has *p*(*x*,*y*) or *p*(*x*,*y*,*z*) coordinates for each time instance during a match. A sequence of moving point for a player or ball forms a trajectory. A basic feature model for soccer match in terms of moving feature is given as the following figure.

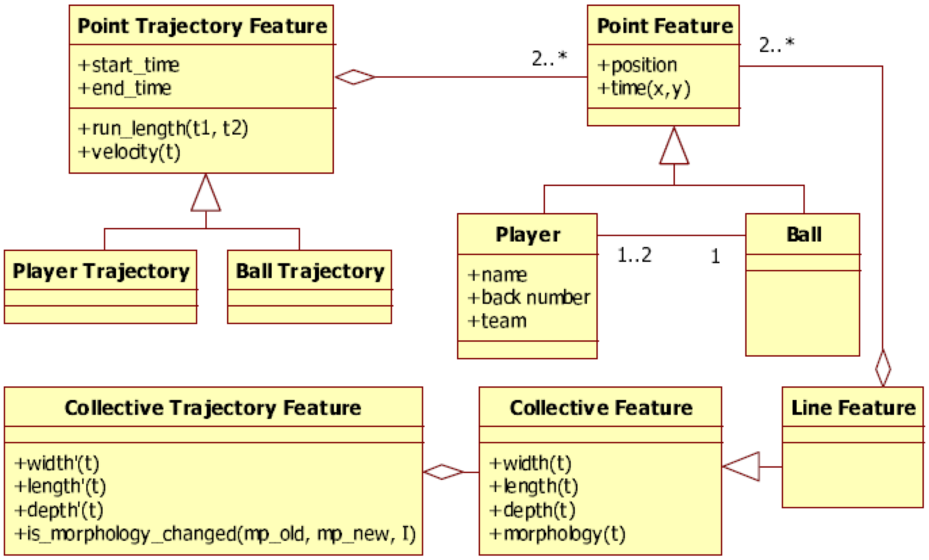


Figure 6.14: Basic Feature Model for Soccer Match

We define two abstract feature types Point Feature and Point Trajectory Feature as the root classes of the model of figure 1, which correspond to spatial and spatiotemporal objects respectively. And Ball and Player, which inherit from Point Feature, have the coordinates of the ball and players at each sampling time instance. Ball Trajectory and Player Trajectory represent the feature types of ball and player trajectories respectively. Each object of Point Feature has (*x*, *y*) coordinates at each time instance *t*. Note that we employ (*x*, *y*)-coordinate reference system to specify the position of spatial objects from the left-bottom corner of the field. And we assume the continuity of temporal domain even though there may be several breaks of match such as ball-out. Due to the breaks, we cannot assure the continuity of the ball trajectory. Since soccer is a collective sport, there is a requirement to analyze the positions of players as a collection as well as position of each individual player. For example, the defensive tactics would be better understood by analyzing the shape of the defensive players in a collective way, rather than the position of each individual defender separately. For this reason, we define Collective Feature and

Collective Trajectory. We may define additional properties for player such as direction.

Based on the information of this basic feature model, we can derive more useful information from primitive behaviors such as kicks, ball possession, dribbles (see figure 2), intercepts, passes, to complicated strategies such as formation, defense or attack systems.

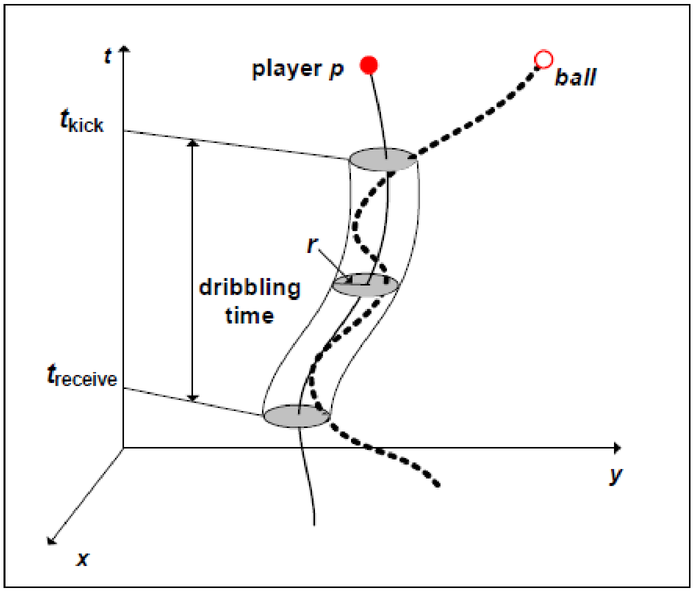


Figure 6.15: Dribbles [6]

6.9 Real-time location data collection

Collecting real time location data is important for some use cases. This use-case presents system architecture, operations, operation sequence for collecting location data in real-time.

6.9.1 System architecture

A location collection protocol can be designed with MQTT (Message Queuing Telemetry Transport), which takes a publish-subscribe message pattern to provide one-to-many message distribution.

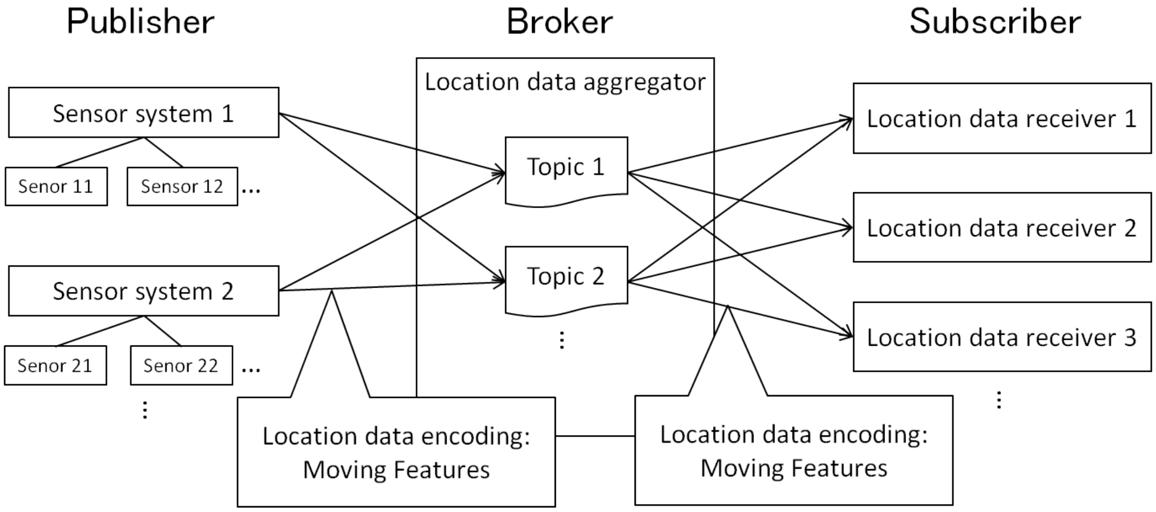


Figure 6.16: System architecture for collecting location data

- ***Sensor*** - Entity that provides data of an observed property as output.

- ***Sensor System*** - System which multiple sensors compose.

- ***Location data aggregator* -** Entity that receives location data from sensor systems and sends location data to location data receivers.

- ***Location data receiver* -** Entity that receives location data from location data aggregators.

6.9.2 Operations

- ***RegisterSensor*** **–** an operation for registering information of a sensor system to a location data aggregator. The data aggregator deletes the information of a sensor system when it receives an empty *RegisterSensor* message from the sensor system.

- ***InsertTrajectory*** **–** an operation for sending location data to an location data aggregator. The location data aggregator sends the location data to location receivers.

6.9.3 Operation sequence

6.9.3.1 Registration for sensor system

1. The sensor system sets a topic “lcp/RegisterSensor/[SensorSystemID]” and a schema definition encoded by OGC Moving Features Simple CSV header lines, where “lcp” comes from acronyms of “location collection protocol”.

6.9.3.2 Registration for location data receiver

1. The Location data receiver sends a request for registration as a subscriber of topic “lcp/RegisterSensor/#” to location data aggregator. “#” is a wild card in topic definition of MQTT.

2. The Location data aggregator registers the location data receiver.

3. The Location data aggregator sends the latest message of topic “lcp/RegisterSensor/#”. This message includes sensor system IDs and schema definitions generated by the sensor systems.

4. The Location data receiver sends a request for registration as a subscriber of topic “lcp/InsertTrajectory/[SensorSystemID]”

5. The Location data aggregator registers the location data receiver.

6.9.3.3 Sending and receiving location data

1. The sensor system sends location data encoded by OGC Moving Features Simple CSV data body to a location data aggregator on topic “lcp/InsertTrajectory/[SensorSystemID]”.

2. The Location data aggregator sends the location data to subscribers of topic “lcp/InsertTrajectory/[SensorSystemID]”.

6.9.3.4 Sending and receiving location data

1. The sensor system sends a message with an empty content to the location aggregator on topic “lcp/RegisterSensor/[SensorSystemID]”.

   The Location data aggregator sends the message to subscribers of “lcp/RegisterSensor/[SensorSystemID]”.

Acknowledgement

The work in Clauses 6.9 is supported by consignment research and development of techniques about use and application of real-time information in G-space platform from the Ministry of Internal Affairs and Communications, Japan.

7 Toward OGC Moving Features API

7.1 Needs for "OGC Moving Features API"

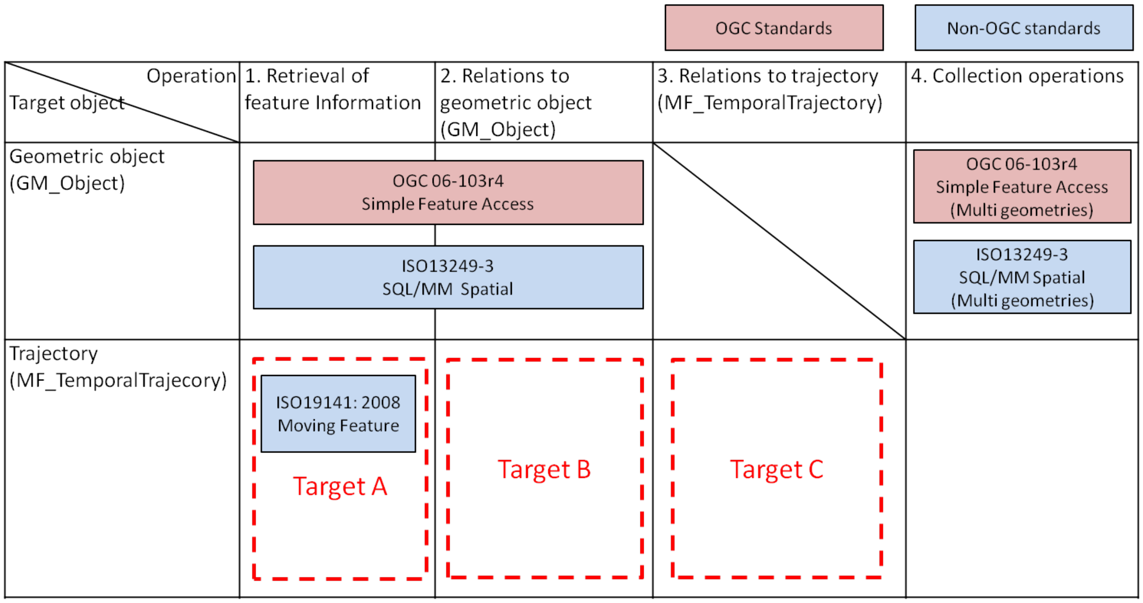
Based on the use cases and applications for OGC Moving Features as described in Clause 6 above, the following API operations are needed:

1. Retrieval of feature information
   For examples, these operations retrieve positions, trajectories, and velocities of a moving feature such as a car (see Clauses 6.1, 6.3, 6.7, and 6.8), a person (see Clauses 6.2 and 6.3), a vessel (see Clause 6.4), an aircraft (see Clause 6.5), and a hurricane (see Clauses 6.6).

2. Operations between MF_TemporalTrajectory and GM_Object
   An example of these operations is "intersection" between a geometric object and a trajectory of a moving feature like a car (see Clauses 6.1, 6.3, 6.7, and 6.8), a person (see Clauses 6.2 and 6.3), a vessel (see Clause 6.4), an aircraft (see Clause 6.5), and a hurricane (see Clauses 6.6).

3. Operations between two MF_TemporalTrajectory
   An example of these operations is to calculate a distance of the nearest approach of a trajectory to another trajectory. The case studies are distance between a criminal agent and a police agent for predicting crime patterns (see Clause 6.7) and distance between soccer players for making proper tactics (see Clause 6.8).

Therefore, the Moving Features SWG recommends that a new Moving Features API standard supporting these operations should be defined. Figure 7.1 summarizes existing APIs and Moving Features API. Operations for geometric objects have been supported by OGC 06-103r4 (OpenGIS Implementation Standard for Geographic information - Simple feature access - Part 1: Common architecture) and ISO 13249-3 (Information technology — Database languages — SQL multimedia and application packages Part3: Spatial). On the other hand, a part of operations for trajectories was defined by ISO 19141:2008, which is an abstract specification for moving features. Thus, the new Moving Features API should target the following three kinds of operations: retrieval of feature information ("Target A" shown in Figure 7.1), operations between a trajectory and a geometric object ("Target B" shown in Figure 7.1), and operations between two trajectories ("Target C" shown in Figure 7.1). Collection operations, of which targets are collection data, such as clustering of multiple trajectories, obtaining centroid of multiple trajectories are positioned as future works until the demands are clarified.



Operation / Target object	1. Retrieval of feature Information	2. Relations to geometric object (GM_Object)	3. Relations to trajectory (MF_TemporalTrajectory)	4. Collection operations
Geometric object (GM_Object)	OGC 06-103r4 Simple Feature Access ISO13249-3 SQL/MM Spatial			OGC 06-103r4 Simple Feature Access (Multi geometries) ISO13249-3 SQL/MM Spatial (Multi geometries)
Trajectory (MF_TemporalTrajecory)	ISO19141: 2008 Moving Feature Target A	Target B	Target C	

Figure 7.1: Existing APIs and Moving Features API

Examples of Moving Features APIs are shown in Clauses 7.2, 7.3, 7.4, and 7.5. In these examples, MF_TemporalTrajectory and GM_object deal with 2 or 3 dimensional objects.

7.2 Target A – 1. retrieval of feature information

The operations on retrieval of feature information are based on operations of “MF_TemporalTrajectory” (see ISO19141:2008).

□ ***pointAtTime*** – accepts a time in the domain of the trajectory as input, and returns the direct position of the trajectory at that time.

```
MF_TemporalTrajectory::pointAtTime(t: TM_GeometricPrimitive):
DirectPosition
```

□ ***timeAtPoint*** – accepts a direct position as input and returns the set of times at which the trajectory passes through that direct posit*ion.*

```
MF_TemporalTrajectory::timeAtPoint(p: DirectPosition):
Set<TM_GeometricPrimitive>
```

□ ***velocity*** – accepts a time as input and returns the velocity as a vector at that time.

```
MF_TemporalTrajectory::velocity(t: TM_Coordinate): Vector
```

□ ***acceleration*** – accepts a time as input and returns the acceleration as a vector at that time.

```
MF_TemporalTrajectory::acceleration(t: TM_Coordinate): Vector
```

- ***timeToDistance* –** returns a graph of the time to distance function as a set of curves in the Euclidean space consisting of coordinate pairs of time, distance.

```
MF_TemporalTrajectory::timeToDistance( ): GM_Curve[1..*]
```

- ***timeAtDistance* –** returns an array of TM_GeometricPrimitive that lists in ascending order the time or times a particular point (determined by the Set<Distance> in the trajectory's GM_GenericCurve::paramForPoint(p:DirectPosition) : Set<Distance>, DirectPosition) is reached.

```
MF_TemporalTrajectory::timeAtDistance(d: Distance):
TM_GeometricPrimitive[0..*]
```

- ***cummulativeDistanceAtTime* –** accepts a time as input and returns the cumulative distance travelled (including all movements forward and retrograde as positive travel distance) from the beginning of the trajectory at that time "t".

```
MF_TemporalTrajectory::cumulativeDistanceAtTime(t: TM_Coordinate):
Distance
```

- ***timeAtCummulativeDistance* –** accepts a distance as input and returns the time at which the trajectory's total length (including all movements forward and retrograde as positive travel distance) reaches that cumulative travel distance.

```
MF_TemporalTrajectory::timeAtCumulativeDistance(d: Distance):
TM_GeometricPrimitive
```

- ***subTrajectory* –** accepts two times in the domain of the trajectory and return a trajectory that is a subset of the given trajectory for the specified time interval.

```
MF_TemporalTrajectory::subTrajectory(newStartTime: TM_Coordinate,
newEndTime: TM_Coordinate): MF_TemporalTrajectory
```

- ***positionAtTime* –** accepts a time in the domain of the trajectory and return the position of the moving feature on the trajectory at that time, expressed as a liner reference system position.

```
MF_TemporalTrajectory::positionAtTime(t: TM_Coordinate):
LR_PositionExpression
```

7.3 Target B – 2. relations between MF_TemporalTrajectory and GM_Object

These operations are similar to OGC - Simple Feature Access - Part 1: Common architecture (see OGC 06-103r4).

- ***equals*** **–** accepts a point object and a time interval as input and returns 1 (TRUE) if this trajectory “spatially equal” to the point object. The parameter “timeInterval” shall restrict the search to a particular period of time.

  ```
  MF_TemporalTrajectory::equals(p: GM_Point, timeInterval: TM_Period):
  Interger
  ```

- ***disjoint*** **–** accepts a geometric object and a time interval as input and returns 1 (TRUE) if this trajectory “spatially disjoint” from the geometric object. The parameter “timeInterval” shall restrict the search to a particular period of time.

  ```
  MF_TemporalTrajectory::disjoint(geometry: GM_Object, timeInterval:
  TM_Period): Interger
  ```

- ***intersects*** **–** accepts a geometric object and a time interval as input and returns 1 (TRUE) if this trajectory “spatially intersects” the geometric object. The parameter “timeInterval” shall restrict the search to a particular period of time.

  ```
  MF_TemporalTrajectory::intersects(geometry: GM_Object, timeInterval:
  TM_Period): Interger
  ```

- ***distanceWithin***— accepts a geometric object, a time interval and a distance as input and returns 1 (TRUE) if this trajectory and the geometric object are within the specified distance of one another. The parameter “timeInterval” shall restrict the search to a particular period of time.

  ```
  MF_TemporalTrajectory::distanceWithin(geometry: GM_Object,
  timeInterval: TM_Period, d: Distance): Interger
  ```

- ***intersection*** — accepts a geometric object and a time interval as input and returns a sub-trajectory that represents the intersection of this trajectory to the geometric object. The parameter “timeInterval” shall restrict the search to a particular period of time.

  ```
  MF_TemporalTrajectory::intersection(geometry: GM_Object,
  timeInterval: TM_Period): MF_TemporalTrajectory
  ```

- ***difference*** — accepts a geometric object as input and returns a sub-trajectory that represents the difference of this trajectory from the geometric object. The parameter “timeInterval” shall restrict the search to a particular period of time.

  ```
  MF_TemporalTrajectory::difference(geometry: GM_Object, timeInterval:
  TM_Period): MF_TemporalTrajectory
  ```

- ***nearestApproach*** — accepts a geometric object and a time interval as input and returns a set of times and a distance of the nearest approach of this trajectory to the geometric object (see Figure 7.2). The parameter “timeInterval” shall restrict the search to a particular period of time.

```
MF_TemporalTrajectory::nearestApproach(geometry: GM_Object,
timeInterval: TM_Period): Distance, TM_GeometricPrimitive[1..*]
```

- ***nearestApprochPoint*** — accepts a geometric object as input and returns a set of times and a direct position of the nearest approach of this trajectory to the geometric object (see Figure 7.2). The parameter “timeInterval” shall restrict the search to a particular period of time.

```
MF_TemporalTrajectory::nearestApproachPoint(geometry: GM_Object,
timeInterval: TM_Period): DirectPosition,TM_GeometricPrimitive[1..*]
```

- ***snapToGrid*** — accepts a point object (origin) and an array whose elements are two cell sizes (x and y cell sizes) or three cell sizes (x, y, and z cell sizes) for a grid as input and returns a trajectory snapped to the grid.

```
MF_TemporalTrajectory::snapToGrid(p: GM_Point, cellsize[]: float):
MF_TemporalTrajectory
```

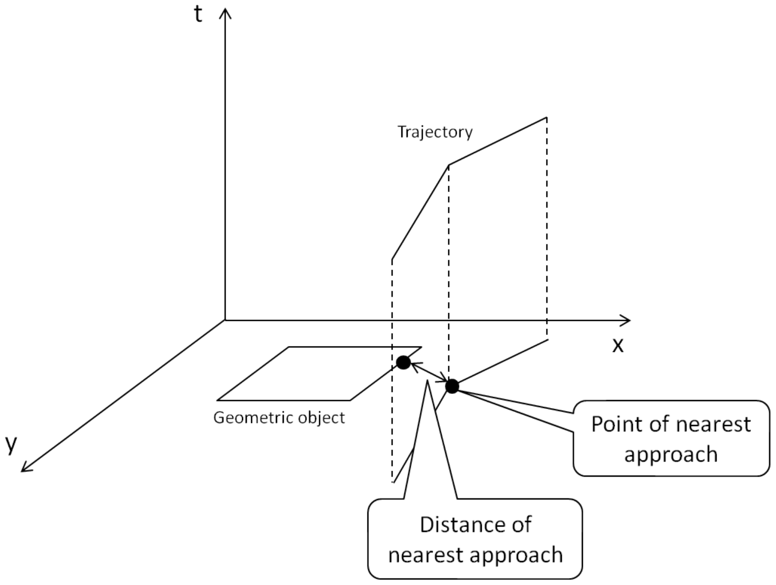


Figure 7.2: nearestApproach and nearestApproachPoint

between a trajectory and a geometric object

7.4 Target C – 3. relations between two MF_TemporalTrajectory

The following operations are similar to Target B operations. However, their input is replaced by MF_TemporalTrajectory.

- ***equals*** **–** accepts another trajectory and a time interval as input and returns 1 (TRUE) if this trajectory "spatially equal" to the other trajectory. The parameter "timeInterval" shall restrict the search to a particular period of time.

  ```
  MF_TemporalTrajectory::equals(anotherTemporalTrajectory:
  MF_TemporalTrajectory, timeInterval: TM_Period): Interger
  ```

- ***disjoint*** **–** accepts another trajectory and a time interval as input and returns 1 (TRUE) if this trajectory "spatially disjoint" from the other trajectory. The parameter "timeInterval" shall restrict the search to a particular period of time.

  ```
  MF_TemporalTrajectory::disjoint(anotherTemporalTrajectory:
  MF_TemporalTrajectory, timeInterval: TM_Period): Interger
  ```

- ***intersects*** **–** accepts another trajectory and a time interval as input and returns 1 (TRUE) if this trajectory "spatially intersects" the other temporal trajectory. The parameter "timeInterval" shall restrict the search to a particular period of time.

  ```
  MF_TemporalTrajectory::intersects(anotherTemporalTrajectory:
  MF_TemporalTrajectory, timeInterval: TM_Period): Interger
  ```

- ***distanceWithin***— accepts another trajectory a time interval, and a distance as input and returns 1 (TRUE) if this trajectory and the other trajectory are within the distance of one another. The parameter "timeInterval" shall restrict the search to a particular period of time.

  ```
  MF_TemporalTrajectory::distanceWithin(anotherTemporalTrajectory:
  MF_TemporalTrajectory, timeInterval: TM_Period, d: Distance):
  Interger
  ```

- ***intersection*** — accepts another trajectory and a time interval as input and returns direct points that represent the intersection of this trajectory and the other trajectory. The parameter "timeInterval" shall restrict the search to a particular period of time.

  ```
  MF_TemporalTrajectory::intersection(anotherTemporalTrajectory:
  MF_TemporalTrajectory, timeInterval: TM_Period): Set<DirectPosition>
  ```

- ***nearestApproach*** — accepts another trajectory and a time interval as input and returns a set of times and a distance of the nearest approach of this trajectory to the other trajectory (see Figure 7.3). The parameter "timeInterval" shall restrict the search to a particular period of time.

  ```
  MF_TemporalTrajectory::nearestApproach(anotherTemporalTrajectory:
  MF_TemporalTrajectory, timeInterval: TM_Period): Distance,
  TM_GeometricPrimitive[1..*]
  ```

- ***nearestApproachPoint*** — accepts another trajectory and returns a set of times and a direct position of the nearest approach of this trajectory to the other trajectory (see

Figure 7.3). The parameter “timeInterval” shall restrict the search to a particular period of time.

```
MF_TemporalTrajectory::nearestApproachPoint(anotherTemporalTrajector
y: MF_TemporalTrajectory):directPosition,TM_GeometricPrimitive[1..*]
```

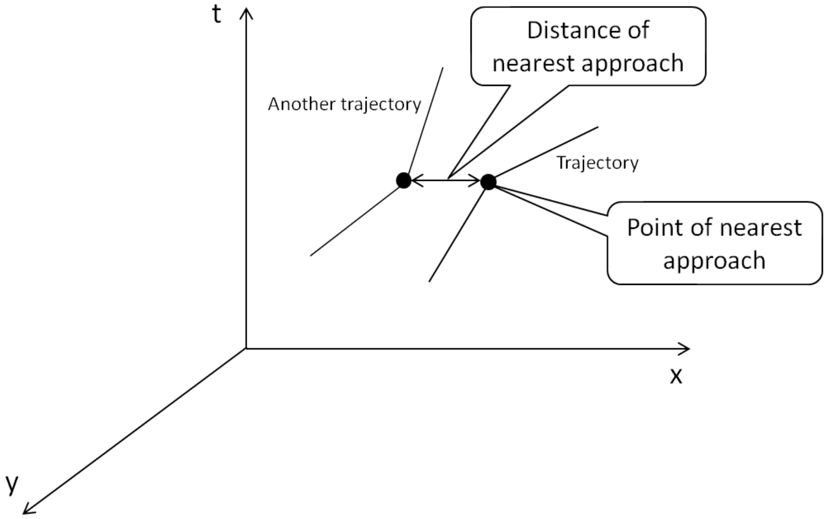


Figure 7.3: nearestApproach and nearestApproachPoint

between two temporal trajectories

7.5 Collection operations

The input of the operations classified into this category is multiple trajectories. For example, this category includes clustering of multiple trajectories, obtaining centroid of multiple trajectories, and so on. This category is positioned as future works until the demands are clarified.

7.6 Concluding remarks

The Moving Features SWG recommends that a new Moving Features API standard should target the following three kinds of operations: Target A (retrieval of feature information), Target B (operations between a trajectory and a geometric object), and Target C (operations between two trajectories). Additionally, the Moving Features SWG recommends establishing an abstract specification of these three kinds of operations because only a part of operations for trajectories was defined by ISO 19141:2008.

Bibliography

[1] Guidelines for Successful OGC Interface Standards, OGC document 00-014r1

[2] K. Nakamura. H. Zhao. R. Shibasaki. K. Sakamoto. T. Ooga. and N. Suzukawa, “Tracking Pedestrians by using Multiple Laser Range Scanners”, ISPRS commission IV WG IV/1 XXth ISPRS Congress. 2004, Jul-04

[3] M. Hashimoto, T. Konda, B. Zhitao and K. Takahashi, “Laser-based tracking of randomly moving people in crowded environments”, 2010 IEEE International Conference on Automation and Logistics (ICAL), pp 31-36

[4] K. Schenk, M. Eisenbach, A. Kolarow, H. M. Gross, “Comparison of Laser-Based Person Tracking at Feet and Upper-Body Height”, KI 2011: Advances in Artificial Intelligence, Vol. 7006, 2011, pp 277-288

[5] Ho-Chul Kim, Oje Kwon and Ki-Joune Li, "Spatial and Spatiotemporal Analysis of Soccer", Proceedings of ACM SIGSPATIAL GIS ’11, November 1-4, 2011. Chicago, IL, USA

[6] Chan-Hyun Kang, Jung-Rea Hwang, and Ki-Joune Li, "Trajectory analysis for soccer player", Spatial and Spatio-temporal Data Mining (SSTDM), the IEEE Computer Society Press

Revision history

Date	Release	Editor	Primary clauses modified	Description
5/21/2015		Carl Reed	Numerous	Editorial changes and suggestions
5/22/2015		Akinori Asahara	6.3, 6.8 and bibliography	The bibliography was listed at the end of this paper
7/30/2015		Hideki Hayashi	7	The API description was revised
9/11/2015		Frederic Houbie	6.5	Two additional images
10/6/2015		Akinori Asahara	Title	Title: "Search as used in Applications of OGC Moving Features"
10/21/2015		Carl Reed	Various	Prepare for publication