

OGC Moving Features Encoding Extension - JSON

Contents

1. Scope	5
2. Conformance	7
3. References	8
4. Terms and Definitions	9
4.1. base representation	9
4.2. coordinate	9
4.3. coordinate reference system	9
4.4. curve	9
4.5. domain	10
4.6. dynamic attribute	10
4.7. encoding	10
4.8. engineering coordinate reference system	10
4.9. feature	10
4.10. feature attribute	10
4.11. feature collection; collection	11
4.12. foliation	11
4.13. geometric object	11
4.14. geometric primitive	11
4.15. instant	11
4.16. leaf	11
4.17. life span	11
4.18. motion	12
4.19. moving feature	12
4.20. one parameter set of geometries	12
4.21. one parameter set of values	12
4.22. parametric coordinate reference system	12
4.23. parametric datum	12
4.24. period	12
4.25. position	13
4.26. prism	13
4.27. temporal coordinate reference system	13
4.28. temporal datum	13
4.29. temporal geometry	13
4.30. trajectory	13
4.31. value	13
5. Conventions	14
5.1. Identifiers	14
5.2. JSON notation	14

5.3. UML notation	14
5.4. Abbreviated terms	14
6. Overview of Moving Features JSON Encodings (Informative)	16
7. Moving Features JSON Encodings (MF-JSON) Specification	19
7.1. MF-JSON Trajectory Encoding	20
7.2. MF-JSON Prism Encoding	24
7.2.1. TemporalGeometry Object	27
7.2.2. TemporalProperties Object	37
7.2.3. CoordinateReferenceSystem Object	41
7.2.4. MovingFeature Object	43
7.2.5. MovingFeatureCollection Object	45
7.2.6. LifeSpan Object	47
7.2.7. BoundingBox Object	48
7.2.8. Geometry Object	48
7.2.9. Properties Object	48
7.2.10. MotionCurve Objects	48
8. Media Types	56
Annex A: Conformance Class Abstract Test Suite (Normative)	57
A.1. Conformance Test Class: MF-JSON Trajectory Encoding	57
A.1.1. MF-JSON Trajectory file	57
A.1.2. LinearTrajectory object	57
A.1.3. Datetimes	58
A.1.4. Constraints	58
A.2. Conformance Test Class: MF-JSON Prism Encoding	59
A.2.1. MF-JSON Prism file	59
A.2.2. Conflict	59
A.2.3. TemporalGeometry object	59
A.2.4. TemporalPrimitiveGeometry object	60
A.2.5. TemporalPrimitiveGeometry object type	60
A.2.6. 3D model	61
A.2.7. TemporalComplexGeometry object	62
A.2.8. TemporalProperties object	62
A.2.9. @propertyN object	63
A.2.10. CoordinateReferenceSystem object	63
A.2.11. MovingFeature object	64
A.2.12. MovingFeatureCollection object	64
A.2.13. LifeSpan object	65
A.2.14. BoundingBox object	65
A.2.15. MotionCurve object	66
Annex B: Sample Data of Moving Feature (Informative)	67
B.1. Sample Data of OGC Moving Features XML and CSV encoding	67

B.2. Sample Data of OGC Moving Features JSON Trajectory encoding	69
B.3. Sample Data of OGC Moving Features JSON Prism encoding	70
Annex C: Deformation of Rigid Body (Informative).....	72
Annex D: Revision History	75
Annex E: Bibliography	76

Open Geospatial Consortium

Submission Date: 2019-07-30

Approval Date: <yyyy-mm-dd>

Publication Date: <yyyy-mm-dd>

External identifier of this OGC® document: <http://www.opengis.net/doc/{doc-type}/{standard}/{m.n}>

Internal reference number of this OGC® document: 19-045r3

Version: 1.0

Category: OGC® Implementation Standard

Editor: Kyoung-Sook KIM, Nobuhiro ISHIMARU

OGC Moving Features Encoding Extension - JSON

Copyright notice

Copyright © 2018 Open Geospatial Consortium

To obtain additional rights of use, visit <http://www.opengeospatial.org/legal/>

Warning

This document is not an OGC Standard. This document is distributed for review and comment. This document is subject to change without notice and may not be referred to as an OGC Standard.

Recipients of this document are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

Document type: OGC® Standard

Document subtype: Encoding

Document stage: Draft

Document language: English

License Agreement

Permission is hereby granted by the Open Geospatial Consortium, ("Licensor"), free of charge and subject to the terms set forth below, to any person obtaining a copy of this Intellectual Property and any associated documentation, to deal in the Intellectual Property without restriction (except as set forth below), including without limitation the rights to implement, use, copy, modify, merge, publish, distribute, and/or sublicense copies of the Intellectual Property, and to permit persons to whom the Intellectual Property is furnished to do so, provided that all copyright notices on the intellectual property are retained intact and that each person to whom the Intellectual Property is furnished agrees to the terms of this Agreement.

If you modify the Intellectual Property, all copies of the modified Intellectual Property must include, in addition to the above copyright notice, a notice that the Intellectual Property includes modifications that have not been approved or adopted by LICENSOR.

THIS LICENSE IS A COPYRIGHT LICENSE ONLY, AND DOES NOT CONVEY ANY RIGHTS UNDER ANY PATENTS THAT MAY BE IN FORCE ANYWHERE IN THE WORLD.

THE INTELLECTUAL PROPERTY IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE DO NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE INTELLECTUAL PROPERTY WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION OF THE INTELLECTUAL PROPERTY WILL BE UNINTERRUPTED OR ERROR FREE. ANY USE OF THE INTELLECTUAL PROPERTY SHALL BE MADE ENTIRELY AT THE USER'S OWN RISK. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR ANY CONTRIBUTOR OF INTELLECTUAL PROPERTY RIGHTS TO THE INTELLECTUAL PROPERTY BE LIABLE FOR ANY CLAIM, OR ANY DIRECT, SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM ANY ALLEGED INFRINGEMENT OR ANY LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR UNDER ANY OTHER LEGAL THEORY, ARISING OUT OF OR IN CONNECTION WITH THE IMPLEMENTATION, USE, COMMERCIALIZATION OR PERFORMANCE OF THIS INTELLECTUAL PROPERTY.

This license is effective until terminated. You may terminate it at any time by destroying the Intellectual Property together with all copies in any form. The license will also terminate if you fail to comply with any term or condition of this Agreement. Except as provided in the following sentence, no such termination of this license shall require the termination of any third party end-user sublicense to the Intellectual Property which is in force as of the date of notice of such termination. In addition, should the Intellectual Property, or the operation of the Intellectual Property, infringe, or in LICENSOR's sole opinion be likely to infringe, any patent, copyright, trademark or other right of a third party, you agree that LICENSOR, in its sole discretion, may terminate this license without any compensation or liability to you, your licensees or any other party. You agree upon termination of any kind to destroy or cause to be destroyed the Intellectual Property together with all copies in any form, whether held by you or by any third party.

Except as contained in this notice, the name of LICENSOR or of any other holder of a copyright in all or part of the Intellectual Property shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Intellectual Property without prior written authorization of LICENSOR or such copyright holder. LICENSOR is and shall at all times be the sole entity that may authorize you or any third party to use certification marks, trademarks or other special designations to indicate compliance with any LICENSOR standards or specifications. This Agreement is governed by the laws of the Commonwealth of Massachusetts. The application to this Agreement of the United Nations Convention on Contracts for the International Sale of Goods is hereby expressly excluded. In the event any provision of this Agreement shall be deemed unenforceable, void or invalid, such provision shall be modified so as to make it valid and enforceable, and as so modified the entire Agreement shall remain in full force and effect. No decision, action or inaction by LICENSOR shall be construed to be a waiver of any rights or remedies available to it.

i. Abstract

This standard defines how to encode and share the various movements of geographic features by using JavaScript Object Notation (JSON). It provides an alternative encoding for OGC Moving Features instead of XML Core [OGC 14-083r2] and Simple CSV [OGC 14-084r2]. A moving feature, typically like a vehicle and a pedestrian, contains a temporal geometry whose location continuously changes over time and dynamic non-spatial attributes whose values vary with time. This Moving Features JSON encoding defines a set of keywords to implement the conceptual schema of moving features defined in ISO 19141:2008 [ISO 19141:2008], accompanied with IETF GeoJSON Format [IETF RFC 7946].

ii. Keywords

The following are keywords to be used by search engines and document catalogues.

ogcdoc, OGC document, moving features, encoding, JSON

iii. Preface

This document specifies JSON encodings for the implementation of the conceptual schema of moving features defined in ISO 19141:2008, i.e., features whose locations change over time. OGC Moving Features Encoding standards XML Core [OGC 14-083r2] and Simple CSV [14-084r2] have focused on the movement of 0-dimensional geometric primitives (Points), called trajectories. In addition, the ISO schema excludes deformation and changes of non-spatial attributes of the feature. However, there are requirements to represent and share the movements of 1-dimensional curve LineStrings, 2-dimensional surface Polygons, and 3-dimensional solid polyhedrons with time-varying non-spatial attributes in various application areas, including: Location Based Services, Intelligent Transportation Systems, Disaster Risk Management Systems, and Smart City Applications. For example, the traffic congestion on roads and the hotspot of air pollution are typical moving features in the real world.

Therefore, OGC Moving Features JSON encodings (MF-JSON) define data members for one-parameter geometries and properties of the moving feature in time not only trajectory. MF-JSON are inspired by the IETF GeoJSON Format [IETF RFC 7946].

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.

iv. Submitting organizations

The following organization has submitted this Document to the Open GeoSpatial Consortium, Inc.:

- Artificial Intelligence Research Center, National Institute of Advanced Industrial Science and Technology
- Defense Systems Company, Hitachi, Ltd.
- Pusan National University

v. Submitters

All questions regarding this submission should be directed to the editor or the submitters:

Name	Organization
Kyoung-Sook KIM	Artificial Intelligence Research Center, National Institute of Advanced Industrial Science and Technology
Taehoon KIM	Artificial Intelligence Research Center, National Institute of Advanced Industrial Science and Technology
Nobuhiro ISHIMARU	Defense Systems Company, Hitachi, Ltd.
Ki-Joune LI	Pusan National University

Chapter 1. Scope

This document defines the structure and content of JSON (JavaScript Object Notation) [IETF RFC7159] encoding implementation of the conceptual model for moving features described in ISO 19141:2008. In common, a moving feature is a geographic feature whose location changes over time containing zero or more thematic attributes. Such a feature may be a car, a pedestrian, an airplane, a ship, or something alike. The geometry of a moving feature to express its motion is defined as a one parameter set of geometries that may be viewed as a set of leaves or a set of trajectories as shown in the Figure 1. A leaf represents the geometry of the moving feature at a particular value of the parameter (e.g., a point in time) and a trajectory is a curve that represents the path of a point in the geometry of the moving feature as it moves with respect to the parameter (e.g., time).

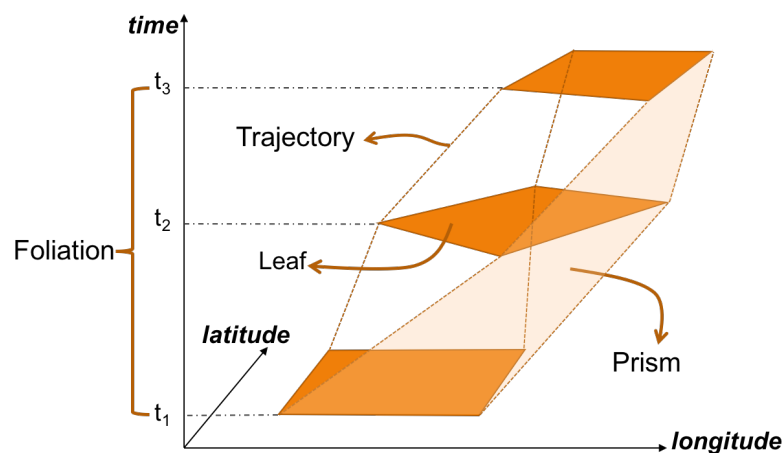


Figure 1. Feature Movement: trajectory, leaf, foliation, and prism [ISO 19141:2008]

In the illustration above, a 2D rectangle moves, rotates, and shrink through time. Each representation of the rectangle at a given time (e.g., t_1 , t_2 , and t_3) is a leaf. The path traced by each corner point of the rectangle (and by each of its other points) is a trajectory. The set of points contained in all of the leaves, and in all of the trajectories, forms a prism. The set of leaves also forms a foliation, meaning that there is a complete and separate representation of the geometry of the feature for each specific time. If viewed in a 4 dimensional spatiotemporal coordinate system, the points on the feature's geometry at different times are different points.

This specification of Moving Features JSON encodings are inspired by IETF GeoJSON Format [IETF RFC 7946] to describe two types of dynamic attribute of a feature or feature collection in a JSON document as follows:

- *Temporal geometry*: The spatial change over time, representing the movement of the rigid or nonrigid body of a feature; and
- *Temporal property*: The thematic change over time, representing the variation of the value of any descriptive characteristic of a feature.

This JSON encodings provide an alternative to the OGC® Moving Features Encodings XML Core [OGC 14-083r2] and Simple Comma Separated Values (CSV) [OGC 14-084r2] standards by using IETF GeoJSON [IETF RFC 7946]. Moreover, it can encompass the movement of 1-dimensional, 2-dimensional, 3-dimensional geometric primitives, and their aggregation, as well as moving points and their trajectories. For example, the representation of the following phenomena in a

spatiotemporal domain is scope of this specification:

- *Discrete phenomena*, which exist only on a set of instants, such as road accidents;
- *Step phenomena*, where the changes of locations are abrupt at an instant, such as administrative boundaries;
- *Continuous phenomena*, whose locations move continuously for a period in time, such as vehicles, typhoons, or floods.

This specification does not address partial motions of parts of bodies in a primitive movement and the dimensional deformation of the feature (e.g., from 1-d geometry to 0-d geometry). Also the succession of either features or their association is out of scope in this specification.

Chapter 2. Conformance

This standard defines 2 requirements classes with URI:

- <http://www.opengis.net/spec/movingfeatures/json/1.0/req/trajectory>
- <http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism>

with two accompanying conformance classes with URI:

- <http://www.opengis.net/spec/movingfeatures/json/1.0/conf/trajectory>
- <http://www.opengis.net/spec/movingfeatures/json/1.0/conf/prism>

Conformance with this standard shall be checked using all the relevant tests specified in Annex A (normative) of this document. The framework, concepts, and methodology for testing, and the criteria to be achieved to claim conformance are specified in the OGC Compliance Testing Policies and Procedures and the OGC Compliance Testing web site.

Chapter 3. References

The following normative documents contain provisions that, through reference in this text, constitute provisions of 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.

- OGC: [OGC 18-005r4](#), *OGC Abstract Specifications - Topic 2 - Spatial referencing by coordinates*, version 5.0, 2019.
- OGC: [OGC 15-042r3](#), *TimeseriesML 1.0 – XML Encoding of the Timeseries Profile of Observations and Measurements*, version 1.0, 2016.
- OGC: [OGC 14-083r2](#), *OGC Moving Features Encoding Part I: XML Core*, 2015.
- OGC: [OGC 14-084r2](#), *OGC Moving Features Encoding Extension: Simple Comma Separated Values (CSV)*, 2015.
- ISO: [ISO/IEC Directives](#), *Part 2. Rules for the structure and drafting of International Standards*
- ISO: [ISO 8601:2004](#), *Data elements and interchange formats - Information interchange - Representation of dates and time*, 2004.
- ISO: [ISO 19101:2014](#), *Geographic information — Reference model — Part 1: Fundamentals*, 2014.
- ISO: [ISO 19107:2003](#), *Geographic Information - Spatial schema* , 2003.
- ISO: [ISO 19108:2002](#), *Geographic Information - Temporal schema* , 2002.
- ISO: [ISO 19141:2008](#), *Geographic information - Schema for moving features*, 2008.
- ISO/IEC: [ISO 11179-1:2004](#), *Information technology — Metadata registries (MDR) — Part 1: Framework*, 2004.
- IETF: [IETF RFC 3339](#), *Date and Time on the Internet: Timestamps*, 2002.
- IETF: [IETF RFC 3986](#), *Uniform Resource Identifier (URI): Generic Syntax*, 2005.
- IETF: [IETF RFC 7159](#), *The JavaScript Object Notation (JSON) Data Interchange Format*, 2014.
- IETF: [IETF RFC 7464](#), *JavaScript Object Notation (JSON) Text Sequences*, 2015.
- IETF: [IETF RFC 7946](#), *The GeoJSON Format*, 2016.

Chapter 4. Terms and Definitions

This document uses the terms defined in Sub-clause 5.3 of [OGC 06-121r8], which is based on the ISO/IEC Directives, Part 2, Rules for the structure and drafting of International Standards. In particular, the word “SHALL” (not “must”) is the verb form used to indicate a requirement to be strictly followed to conform to this standard.

For the purposes of this document, the following additional terms and definitions apply.

4.1. base representation

<moving features> representation, using a local origin and local ordinate vectors, of a geometric object at a given reference time

[ISO 19141:2008]

NOTE 1: A rigid geometric object may undergo translation or rotation, but remains congruent with its base representation.

NOTE 2: The local origin and ordinate vectors establish an engineering coordinate reference system (ISO 19111), also called a local frame or a local Euclidean coordinate system.

4.2. coordinate

one of a sequence of n numbers designating the position of a point in n -dimensional space

[OGC 18-005r4]

NOTE: In a coordinate reference system, the coordinate numbers are qualified by units.

4.3. coordinate reference system

coordinate system that is related to an object by a datum

[OGC 18-005r4]

NOTE: For geodetic and vertical datums, the object will be the Earth.

4.4. curve

1-dimensional geometric primitive, representing the continuous image of a line

[ISO 19107:2003]

NOTE: The boundary of a curve is the set of points at either end of the curve. If the curve is a cycle, the two ends are identical, and the curve (if topologically closed) is considered to not have a boundary. The first point is called the start point, and the last is the end point. Connectivity of the curve is guaranteed by the "continuous image of a line" clause. A topological theorem states that a continuous image of a connected set is connected.

4.5. domain

well-defined set

[ISO 19109:2015]

4.6. dynamic attribute

characteristic of a feature in which its value taken from the domain of the feature attribute type varies with time

4.7. encoding

conversion of data into a series of codes

[ISO 11179-1:2004]

4.8. engineering coordinate reference system

coordinate reference system based on an engineering datum

[OGC 18-005r4]

EXAMPLES: Local engineering and architectural grids; coordinate reference system local to a ship or an orbiting spacecraft.

4.9. feature

abstraction of real world phenomena

[ISO 19101:2014]

4.10. feature attribute

characteristic of a feature

[ISO 19109:2015]

Note: A feature attribute type has a name, a data type and a domain associated with it. A feature attribute instance has an attribute value taken from the domain of the feature attribute type.

4.11. feature collection; collection

a set of features from a dataset

NOTE: In this specification, 'collection' is used as a synonym for 'feature collection'. This is done to make, for example, URI path expressions shorter and easier to understand for those that are not geo-experts.

4.12. foliation

one parameter set of geometries such that each point in the prism of the set is in one and only one trajectory and in one and only one leaf

[ISO 19141:2008]

4.13. geometric object

spatial object representing a geometric set

[ISO 19107:2003]

NOTE: A geometric object consists of a geometric primitive, a collection of geometric primitives, or a geometric complex treated as a single entity. A geometric object may be the spatial representation of an object such as a feature or a significant part of a feature.

4.14. geometric primitive

geometric object representing a single, connected, homogeneous element of space

[ISO 19107:2003]

NOTE: Geometric primitives are non-decomposed objects that present information about geometric configuration. They include points, curves, surfaces, and solids.

4.15. instant

zero-dimensional geometric primitive representing position in time

[ISO 19108:2002]

4.16. leaf

<one parameter set of geometries> geometry at a particular value of the parameter

[ISO 19141:2008]

4.17. life span

period during which something exists

4.18. motion

a change in position measured by distance and time

4.19. moving feature

feature whose position changes over time

NOTE: Its base representation uses a local origin and local coordinate vectors of a geometric object at a given reference time.

4.20. one parameter set of geometries

function f from an interval $t \in [a, b]$ such that $f(t)$ is a geometry and for each point $P \in f(a)$ there is a one parameter set of points (called the trajectory of P) $P(t): [a, b] \rightarrow P(t)$ such that $P(t) \in f(t)$

[ISO 19141:2008]

EXAMPLE: A curve C with constructive parameter (variable) t is a one parameter set of points $c(t)$.

4.21. one parameter set of values

function f from an interval $t \in [a, b]$ such that $f(t)$ is a measure value m in a plane with coordinate (t, m)

4.22. parametric coordinate reference system

coordinate reference system based on a parametric datum

[OGC 18-005r4]

4.23. parametric datum

datum describing the relationship of a parametric coordinate system to an object

[OGC 18-005r4]

NOTE The object is normally the Earth.

4.24. period

one-dimensional geometric primitive representing extent in time

[ISO 19108:2002]

NOTE: A period is bounded by two different temporal positions.

4.25. position

data type that describes a point or geometry potentially occupied by an object

4.26. prism

<one parameter set of geometries> set of points in the union of the geometries (or the union of the trajectories) of a one parameter set of geometries

[ISO 19141:2008]

4.27. temporal coordinate reference system

coordinate reference system based on a temporal datum

4.28. temporal datum

datum describing the relationship of a temporal coordinate system to an object

4.29. temporal geometry

one parameter set of geometries in which the parameter is time

4.30. trajectory

path of a moving point described by a one parameter set of points

[ISO 19141:2008]

4.31. value

element of a type domain

[ISO/IEC 19501:2005]

Chapter 5. Conventions

This sections provides details and examples for any conventions used in the document. Examples of conventions are symbols, abbreviations, or special notes regarding how to read the document.

5.1. Identifiers

The normative provisions in this specification are denoted by the URI

<http://www.opengis.net/spec/movingfeatures/json/1.0>

All requirements and conformance tests that appear in this document are denoted by partial URIs which are relative to this base.

All examples in this document are informative only.

5.2. JSON notation

The notation of JSON in this document is based on the specification of [RFC 7159].

The ordering of the members of any JSON object must be considered irrelevant. Some examples use a JavaScript single line comment (//) and an ellipsis (...) as placeholder notation for a specific JSON instance. Whitespace is used in the examples inside this document to help illustrate the data structures, but is not required. Unquoted whitespace is not significant in JSON.

5.3. UML notation

Unified Modeling Language (UML) static structure diagrams appearing in this document are used as described in Subclause 5.2 of OGC Web Services Common [OGC 06-121r9].

5.4. Abbreviated terms

The following symbols and abbreviated terms are used in this standard:

CRS	Coordinate Reference Systems
CSV	Comma Separated Values
IETF	Internet Engineering Task Force
ISO	International Organization for Standardization
JSON	JavaScript Object Notation
OGC	Open Geospatial Consortium
UML	Unified Modeling Language
URI	Uniform Resource Identifiers
URL	Uniform Resource Locators

XML	Extensible Markup Language
0D	Zero Dimensional
1D	One Dimensional
2D	Two Dimensional
3D	Three Dimensional

Chapter 6. Overview of Moving Features

JSON Encodings (Informative)

This document provides an alternative to the OGC® Moving Features Encodings XML Core [OGC 14-083r2] and Simple Comma Separated Values (CSV) [OGC 14-084r2] standards by using JSON objects that is easily readable by machines and by humans. The JSON encoding formats for sharing moving feature data specified in this document leverages **IETF GeoJSON** [IETF RFC 7946] that has already been commonly used in many applications and well supported by tools and software libraries. This standard defines two JSON encoding forms: **MF-JSON Trajectory** and **MF-JSON Prism**. If an application focuses on only the linear movement (i.e., the spatiotemporal line string) of moving points based on World Geodetic System 1984, with longitude and latitude units of decimal degrees, and the ISO 8601 standard for representation of dates and times using the Gregorian calendar, the application can share the trajectory data by using **only** IETF GeoJSON, called **MF-JSON Trajectory**. For other cases, **MF-JSON Prism** can be used for expressing more complex movements of moving features. **MF-JSON Prism** is a GeoJSON-like format reserving new members of JSON objects ("**temporalGeometry**", "**temporalProperties**", "**crs**", "**trs**", "**time**", and etc.) as "foreign members" to represent spatiotemporal geometries, variations of measure, coordinate reference systems, and the particular period of moving features in a JSON document. Figure 2 presents the overview of two forms of Moving Features JSON encodings.

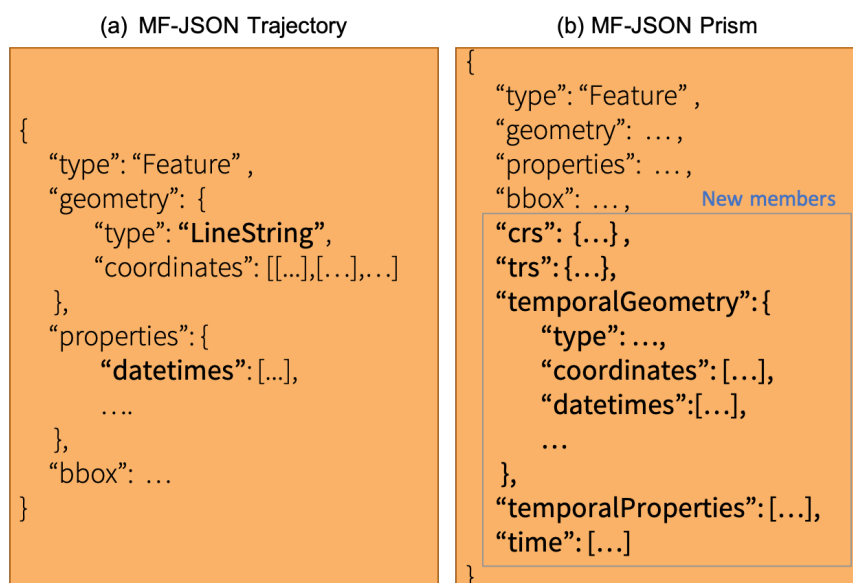


Figure 2. JSON encodings for moving feature data



Coordinate Reference System (CRS)

The IETF GeoJSON format recommends a single coordinate reference system based on World Geodetic System 1984, with longitude and latitude units of decimal degrees. However, a moving feature needs a temporal coordinate system and may require an engineering coordinate reference system or parametric coordinate reference system to represent the movement of the geometry of the feature in the case of an application request. Therefore, MF-JSON Prism includes the "**crs**" field as described in GeoJSON "prior arrangement" (also known as GeoJSON:2008[4]).



The existing GeoJSON implementation can ignore the new members of "**temporalGeometry**", "**temporalProperties**", "**trs**", and "**time**" in an MF-JSON document if there is no processing module for foreign members.

Figure 3 illustrates an example of the movement of a hurricane with a time-varying 2-dimensional geometry as a moving feature. Suppose the pressure centroid of the hurricane reported four locations at time t_1 , t_2 , t_3 , and t_4 . Its trajectory may be drawn depending on the interpolation method between two consecutive locations, such as A (linear) or B (Quadratic). ISO 19141:2008 defines a conceptual schema of a moving feature based on geometry types as shown in Figure 4. The ISO conceptual schema represents the motion consisting of translation and/or rotation of a moving feature, but not including deformation of the geometry, over time.

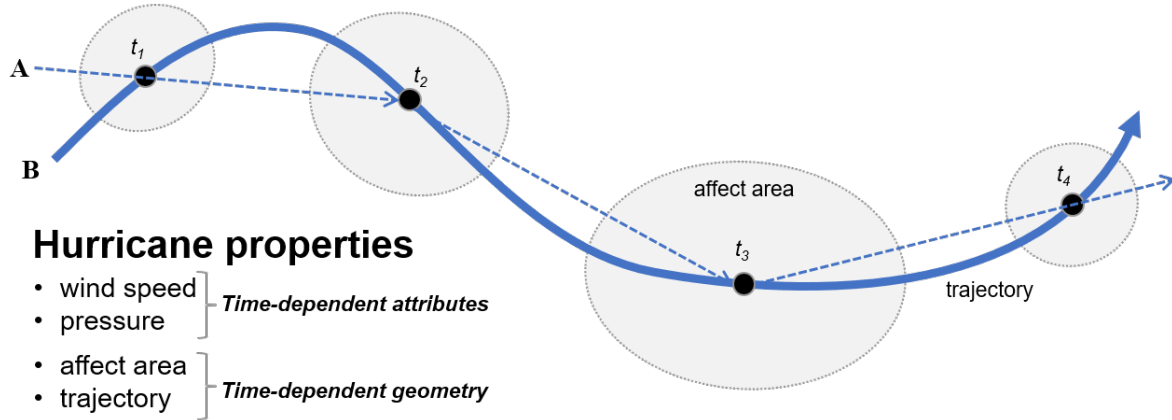


Figure 3. Example of a moving feature: a hurricane and its properties

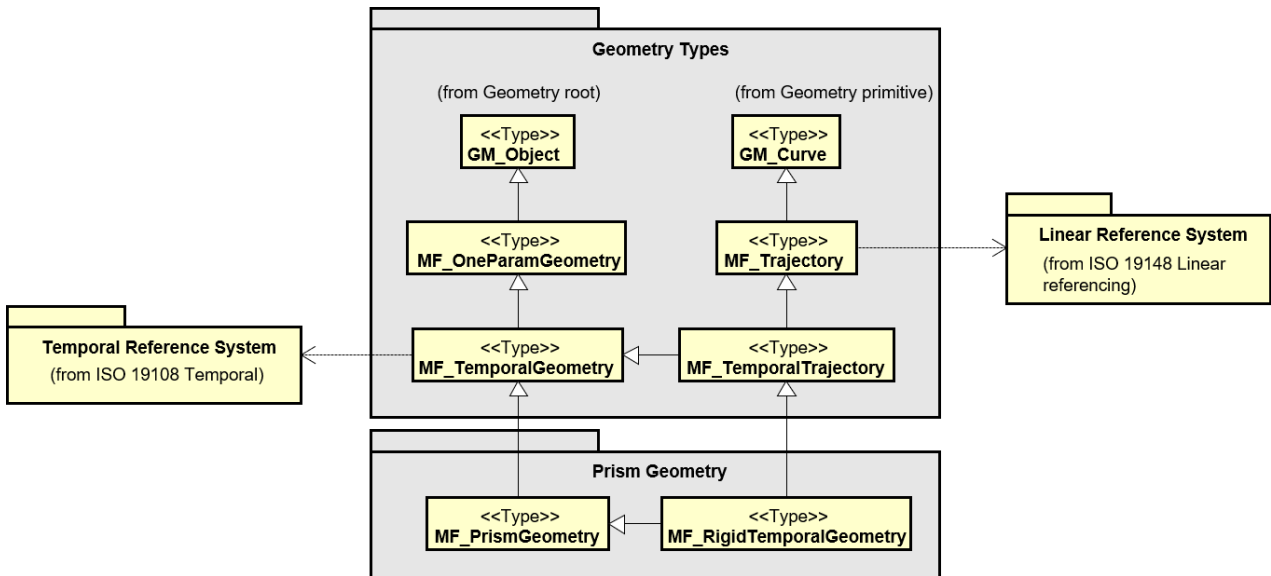


Figure 4. Components of the moving feature packages in ISO 19141:2008

The data model defined in ISO 19141:2008 is based on two geometric types: **MF_OneParamGeometry** and **MF_Trajectory**. **MF_OneParamGeometry** is the type to describe a function f from an interval $t \in [a, b]$ such that $f(t)$ is a geometry. A leaf of a one parameter set of geometries is the geometry $f(t)$ at a particular value t of the parameter. **MF_Trajectory** describes a one-parameter geometry whose cross section is a point as a leaf. **MF_OneParamGeometry** and **MF_Trajectory** is specialized as **MF_TemporalGeometry** and **MF_TemporalTrajectory** respectively, when the parameter is time representing a multiple of a single unit of measure such as year, day, or second for those types. **MF_TemporalTrajectory** is also a sub-type of

MF_TemporalGeometry. The OGC Moving Features XML and CSV standard only provide the encoding formats to represent linear trajectories of moving points as instances of **MF_TemporalTrajectory**, typically representing vehicles or pedestrians. A sample data of OGC Moving Features XML and CSV encoding is provided in [Annex B.1](#). However, Moving Features JSON Encodings can cover **MF_TemporalGeometry**, **MF_PrismGeometry**, and **MF_RigidTemporalGeometry**.

Chapter 7. Moving Features JSON Encodings (MF-JSON) Specification

This clause specifies the structure and content of JSON (JavaScript Object Notation) [IETF RFC7159] encoding implementation for the conceptual model of moving features described in ISO 19141:2008. Moving Features JSON Encodings (MF-JSON) has two encoding formats: **MF-JSON Trajectory** for instances of the MF_TemporalGeometry type with linear interpolation and **MF-JSON Prism** that contains new members to describe instances of the MF_PrismGeometry (or MF_RigidTemporalGeometry) type in the ISO conceptual model.



An object in this specification is a JSON object. The JSON object is an unordered collection of zero or more name/value pairs, where a name is a string and a value can be a JSON **null**, **true**, **false**, string, number, array, or object. An array consists of elements where each element is a value as a JSON **null**, **true**, **false**, string, number, array, or object.

In an MF-JSON object, there are three types of members: MANDATORY, OPTIONAL, and DEFAULT.

MANDATORY

If an object member is **mandatory**, its value SHALL NOT be a JSON **null** value. The mandatory member of an object and its value SHALL appear in a MF-JSON document.

DEFAULT

If an object member is **default**, the member is mandatory and has a default value defined in this specification when its pair (member/value) does not appear in a MF-JSON document.

OPTIONAL

If an object member is **optional**, its value allows a JSON **null** value. If the number has a JSON **null** value, its pair can be omitted from a MF-JSON document.

7.1. MF-JSON Trajectory Encoding

MF-JSON Trajectory specifies how to map/interpret linear trajectories of moving points into/from the **GeoJSON** [IETF RFC 7946] object. It is based on the data model of *mf:MovingFeatures* defined in **OGC Moving Features XML Core** [OGC 14-083r2].



The *mf:MovingFeatures* Class is implemented by *mf:Foliation* including movements of moving points, which are instances of *mf:LinearTrajectory*. In XML Core, the *mf:LinearTrajectory* Class is expressed by a single spatiotemporal segment with linear interpolation, having two elements of *gml:PosList* and *mf:Attr*, as well as three attributes of *mfIdRef*, *start*, and *end*. The *mfIdRef* attribute is the text specifying the moving feature (i.e. person ID, vehicle ID, etc.). The *start* and *end* attribute of *mf:LinearTrajectory* describes the temporal offset from the *gml:beginPosition* element in the *mf:sTBoundedBy* Class. Two or more coordinate tuples are specified in *gml:posList* and linearly interpolated from time *start* to *end*. The *mf:Attr* element contains the attribute information as a text. The values of the attributes are constant while the feature moves along the *mf:LinearTrajectory*. The detailed description refers to **OGC Moving Features XML Core**[OGC 14-083r2].

The UML diagram for MF-JSON Trajectory is depicted in Figure 5.

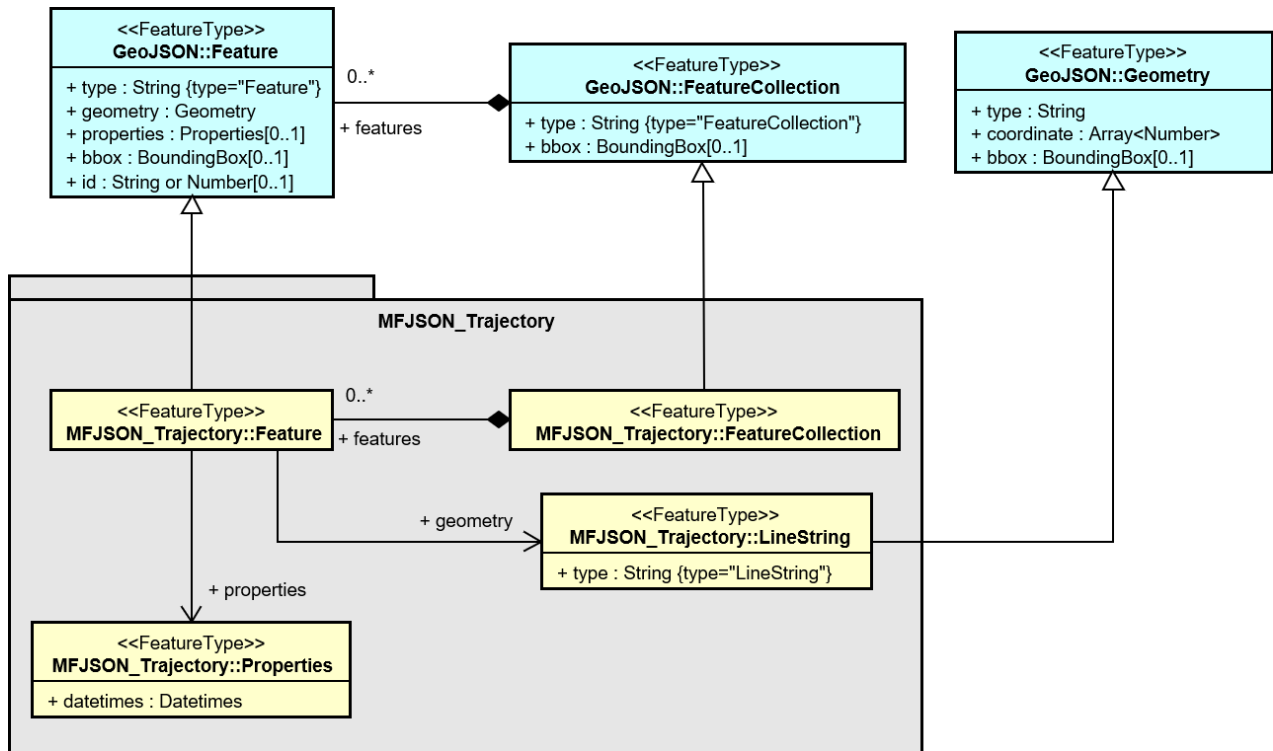


Figure 5. Class diagram for MF-JSON Trajectory

Comparing to the segment-based encodings of **OGC XML Core** and **CSV**, MF-JSON Trajectory encodes a spatiotemporal linestring or a collection of linestrings to reduce the size of a GeoJSON document. The spatiotemporal linestring is a one-dimensional object representing a sequence of spatiotemporal points and the trajectory segments connecting them. The GeoJSON encoding rules of MF-JSON Trajectory are set as follows:

- An MF-JSON Trajectory document presents an instance of the *mf:LinearTrajectory* Class or as a

GeoJSON Feature or a set of those instances as a GeoJSON FeatureCollection object. An *mf:LinearTrajectory* instance is mapped into a GeoJSON Feature object with a linestring geometry; i.e., the value of member "type" in the Geometry object is "LineString".



(GeoJSON) A Feature object represents a spatially bounded thing. A Feature object has, at least 3 members: a "type" member with the value "Feature", a "geometry" member, and a "properties" member. Geometry only has 2 members: "type" and "coordinates". The "type" value of geometry can be one of "Point", "MultiPoint", "LineString", "MultiLineString", "Polygon", "MultiPolygon", and "GeometryCollection". The GeoJSON extensibility is limited by the interpretation of the sentence in the IETF RFC 7946 "Implementations MUST NOT extend the fixed set of GeoJSON types: FeatureCollection, Feature, Point, LineString, MultiPoint, Polygon, MultiLineString, MultiPolygon, and GeometryCollection."

- The identifier of a moving feature (i.e., *mfIdRef*) is mapped into the member named "id" of the GeoJSON Feature object.



(GeoJSON) If a Feature has a commonly used identifier, that identifier is included as a member of the Feature object with the name "id", and the value of this member is either a JSON string or number.

- The value of the "coordinates" member of a linestring geometry object has two or more positions that draws a sequence of trajectory segments. Each position is each projection in the spatial domain. All trajectory segments that have the same identifier are merged into a linestring geometry object as long as any consecutive segments meet each other at a time instant.



(GeoJSON) A position is an array of numbers. There MUST be two or more elements. The first two elements are longitude and latitude, or easting and northing, precisely in that order and using decimal numbers. Altitude or elevation MAY be included as an optional third element. The coordinate reference system for all GeoJSON coordinates is a geographic coordinate reference system, using the World Geodetic System 1984 (WGS 84)[1] datum, with longitude and latitude units of decimal degrees.

- The "properties" member of a GeoJSON Feature object has a member with the name "datetimes". The value of the "datetimes" member is an array of a sequence of monotonic increasing time instants, having the same number of positions of the "coordinates" member. The value of a time instant is a JSON string using Z (e.g., 1985-04-12T23:20:50.52Z) encoded by IETF RFC 3339, an Internet profile of the ISO 8601 standard for representation of dates and times using the Gregorian calendar, or a numeric value of milliseconds since midnight (00:00 a.m.) on January 1, 1970 in UTC (e.g., 1465621816590).



(GeoJSON) A Feature object has a member with the name "properties". The value of the properties member is an object (any JSON object or a JSON null value).

- If a Feature has any variable attribute by time and its value (i.e., *mf:AttrDef* and *mf:Attr*), the attribute can be a member inside the "properties" member with the value of a JSON array, the size of its array can be distinguished into three types as below:
 - The array of a time-varying attribute with step interpolation has elements with one less than the number of positions of the "datetimes" member.
 - The array of a time-varying attribute with linear interpolation has elements with the same number of positions of the "datetimes" member.
 - The array of a fixed attribute during the whole of "datetimes" has only one element.



MF-JSON Trajectory considers that a member in the "properties" which has a JSON-array value is a temporal attribute.

- A set of *mf:LinearTrajectory* instances is packed in the elements of the "features" member of a FeatureCollection object.



(GeoJSON) A FeatureCollection object has a member with the name "features". The value of "features" is a JSON array of Feature objects. It is possible for this array to be empty.

A sample data of MF-JSON Trajectory is provided in [Annex B.2](#).

Requirements Class: MF-JSON Trajectory	
http://www.opengis.net/spec/movingfeatures/json/1.0/req/trajectory	
Target type	JSON object
Dependency	<ul style="list-style-type: none"> • ISO 19141 • IETF RFC 3339 • IETF RFC 7946
Requirement 1.1	http://www.opengis.net/spec/movingfeatures/json/1.0/req/trajectory/GeoJSON A MF-JSON Trajectory file SHALL contain a LinearTrajectory object or a set of LinearTrajectory objects having a compliance with GeoJSON[IETF RFC 7946].
Requirement 1.2	http://www.opengis.net/spec/movingfeatures/json/1.0/req/trajectory/lineartrajectory A LinearTrajectory object SHALL be a GeoJSON Feature object that has two MANDATORY members of "geometry" and "properties".

Requirement 1.3	http://www.opengis.net/spec/movingfeatures/json/1.0/req/trajectory/geometry The value of the "geometry" member SHALL be a LineString Geometry object, having "type" = "LineString". The number of elements in the array of the "coordinates" value in the Geometry object SHALL more than two positions.
Requirement 1.4	http://www.opengis.net/spec/movingfeatures/json/1.0/req/trajectory/properties The value of the "properties" member SHALL be a GeoJSON object that has at least a member with the named "datetimes". The value of the "datetimes" member is a JSON array. If a Feature has any variable attribute by time and its value, the attribute can be a member inside the "properties" member with the value of a JSON array, the size of its array SHALL be same as 1 (const), N-1 (step), or N (linear). Note that N is the number of elements in the array of the "datetimes" value.
Requirement 1.5	http://www.opengis.net/spec/movingfeatures/json/1.0/req/trajectory/datetimes Each element in the array of the "datetimes" value SHALL be an instant object. An instant object SHALL be only a JSON string to represent a timestamp encoded by the IETF RFC 3339 format using Z or the numeric value of milliseconds since midnight (00:00 a.m.) on January 1, 1970, in UTC.
Requirement 1.6	http://www.opengis.net/spec/movingfeatures/json/1.0/req/trajectory/datetimes/monotonic The array of the "datetimes" value SHALL be a monotonic increasing sequence. There SHALL be no instant object that has the same value as any other element.
Requirement 1.7	http://www.opengis.net/spec/movingfeatures/json/1.0/req/trajectory/constraints The number of elements in both arrays of the "coordinates" value and the "datetimes" value SHALL be equal.

7.2. MF-JSON Prism Encoding

This clause specifies the format of MF-JSON Prism to implement the package of Prism Geometry defined in ISO 19141:2008 (Figure 6). The package contains five types used to describe the prism of a moving geometric object: **MF_PrismGeometry**, **MF_RigidTemporalGeometry**, **MF_LocalGeometry**, **MF_TemporalOrientation**, and **MF_RotationMatrix**. The type **MF_PrismGeometry** represents the movement of an object through geographic space. **MF_RigidTemporalGeometry** specializes **MF_PrismGeometry**, which in turn specializes **MF_TemporalGeometry** for the case of an object that moves without deformation, where the object's basic shape is immutable and may be translated or rotated over time. The type **MF_LocalGeometry** is a geometric object to define the local geometry of the moving object in an engineering coordinate reference system (usually 3D). The engineering coordinate reference system is accessible through the Coordinate Reference System association inherited from **GM_Object**. The remaining types support description of the possible rotation of such an object.

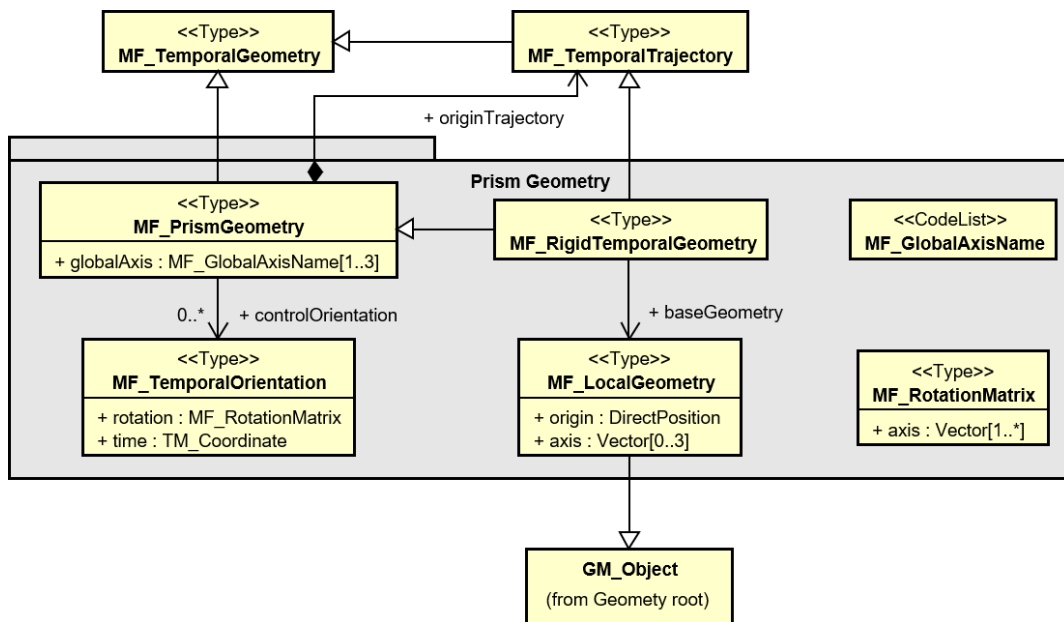


Figure 6. Types of the package of Prism Geometry in ISO 19141:2008

The MF-JSON Prism encoding can represent the movement of **MF_PrismGeometry** and **MF_RigidTemporalGeometry** of a feature which may be 0D, 1D, 2D, 3D geometric primitives, or their aggregations. MF-JSON Prism reserves the following members as foreign members in a GeoJSON Feature object that represents a moving feature:

- The "**temporalGeometry**" member describes a movement of a moving feature.
- The "**temporalProperties**" member describes a set of dynamic non-spatial attributes and their time-dependent values of a moving feature.
- The "**time**" member describes the life time of a moving feature.
- The "**trs**" member describes a Temporal Coordinate Reference System object referred by time-expression to indicate the time of positions or attributes' values.
- The "**crs**" member describes a Spatial Coordinate Reference System object which relates the positions of features in the real world.



Coordinate Reference System (CRS)

The IETF GeoJSON format recommends a single coordinate reference system based on World Geodetic System 1984, with longitude and latitude units of decimal degrees. However, a moving feature needs a temporal coordinate system and may require an engineering/parametric coordinate reference system to represent the movement of the feature's geometry (positions) in the case of an application request.

A MF-JSON Prism document may contain JSON objects that represent instances with the following types, as well as primitives types of JSON null, true, false, string, number, and array.

- TemporalGeometry object (see [7.2.1](#))
 - TemporalPrimitiveGeometry object (see [7.2.1.1](#))
 - TemporalComplexGeometry object (see [7.2.1.2](#))
- TemporalProperties object (see [7.2.2](#))
 - ParametricValues object (see [7.2.2.1](#))
- CoordinateReferenceSystem object (see [7.2.3](#))
- MovingFeature object (see [7.2.4](#))
- MovingFeatureCollection object (see [7.2.5](#))
- LifeSpan object (see [7.2.6](#))
- BoundingBox object (see [7.2.7](#))
- Geometry object (see [7.2.8](#))
- Properties object (see [7.2.9](#))
- MotionCurve object (see [7.2.10](#))

The schema of MF-JSON Prism is depicted by a UML diagram as shown in [Figure 7](#).

Prism has priority to interpret the movement of a feature. A sample data of MF-JSON Prism is provided in [Annex B.3](#).

Requirements Class: MF-JSON Prism	
http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism	
Target type	JSON object
Dependency	<ul style="list-style-type: none"> • ISO 19141 • IETF RFC 3339 • IETF RFC 7946
Requirement 2.1	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/GeoJSON A MF-JSON Prism SHALL use the same meaning of members defined in GeoJSON[IETF RFC 7946].
Requirement 2.2	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/object A JSON object in an MF-JSON Prism file SHALL be encoded by one of the Object types defined in this standard, i.e., TemporalGeometry, TemporalProperties, CoordinateReferenceSystem, MovingFeature, MovingFeatureCollection, LifeSpan, BoundingBox, Geometry, Properties, and MotionCurve.
Requirement 2.3	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/conflict The MF-JSON Prism format SHALL be taken priority over MF-JSON Trajectory when interpreting the movement of a feature.

7.2.1. TemporalGeometry Object

A TemporalGeometry object represents the movement of a moving feature, i.e., changes of the position. Comparing to MF-JSON Trajectory, MF-JSON Prism provides an encoding rule for conceptual schemas specifying prisms of the set of leaves (**MF_PrismGeometry** in ISO 19141) or trajectories (**MF_TemporalTrajectory** in ISO 19141). The TemporalGeometry object is modeled as a mapping function from $t \in \text{time}$ to a Geometry object g of Point, LineString, Polygon, or GeometryCollection.

- A TemporalGeometry object is a JSON object that has at least three members with the name "type", "crs", and "trs" as shown in [Example 1](#).
- The value of the "type" member is one of string "MovingPoint", "MovingLineString", "MovingPolygon", "MovingPointCloud", and "MovingGeometryCollection". The "type" member is a mandatory member of a TemporalGeometry object. The TemporalGeometry object is specified with additional members depending upon the value of the "type" member.
- The value of the "crs" and "trs" member is a JSON object to represent a spatial and temporal CoordinateReferenceSystem (CRS) object respectively (see [7.2.3](#)). If an object has no "crs" and "trs" member, then it may acquire the CRS instances of the "crs" and "trs" member from

upper-level braces of a JSON object. If no CRS instance can be so acquired in a file, the TemporalGeometry object assumes the default CRSs (see 7.2.3.3).

Example 1: Common members of a TemporalGeometry object

```
{
  "type": "...", //(MANDATORY)
  "crs" : {...}, //(DEFAULT)
  "trs" : {...}, //(DEFAULT)
  ....., // different expression with respect to the value of "type"
}
```

Requirement 2.4	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry A TemporalGeometry object SHALL have at least a mandatory member with the named "type" and its value SHALL be one of string "MovingPoint", "MovingLineString", "MovingPolygon", "MovingPointCloud", and "MovingGeometryCollection".
Requirement 2.5	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/crs A TemporalGeometry object has two default members: "crs" and "trs". The value of the "crs" and "trs" member SHALL be a JSON object to represent a CoordinateReferenceSystem (CRS) object. If no CRS instance can be so acquired from upper-level braces of a JSON object, the default spatial and temporal CRS SHALL be applied for the "crs" and "trs" member, respectively.

7.2.1.1. TemporalPrimitiveGeometry Object

A TemporalPrimitiveGeometry object describes the movement of a geographic feature whose leaf geometry at a time instant is drawn by a primitive geometry such as a point, linestring, and polygon in the two- or three-dimensional spatial coordinate system, or a point cloud in the three-dimensional spatial coordinate system. The TemporalPrimitiveGeometry object is defined with additional members named "datetimes", "coordinates", "interpolation", "base", and "orientations" as shown in Example 2, as well as the common members in Example 1.

Example 2: Members of a TemporalPrimitiveGeometry Object

```
{
  // vbar | as a means to select ONE type.
  "type": "MovingPoint | MovingLineString | MovingPolygon | MovingPointCloud", //(MANDATORY)
  "datetimes" : [...],      //(MANDATORY)
  "coordinates": [...],     //(MANDATORY)
  "interpolation": "...",   //(DEFAULT)
  "base": {...},            //(OPTIONAL)
  "orientations": [...],   //(OPTIONAL)
  "crs": {...},             //(DEFAULT)
  "trs": {...}              //(DEFAULT)
}
```

- **"type"**: The **"type"** member is **mandatory** and its value is one of string **"MovingPoint"**, **"MovingLineString"**, **"MovingPolygon"**, and **"MovingPointCloud"**. The value specifies the type of a TemporalPrimitiveGeometry object for the interpretation of the array value of the **"coordinates"** member. Each type represents as follows:

Types	Description
MovingPoint	The type represents a trajectory of a time-parametered 0-dimensional (0D) geometric primitive (Point), representing a single position at a time position (instant) within its temporal domain. Intuitively a temporal geometry of a continuous movement of point depicts a set of curves in a spatiotemporal domain. It supports more complex movements of moving features, as well as linear movement like MFJSON Trajectory. For example, the non-linear movement information of people, vehicles, or hurricanes can be shared as a TemporalPrimitiveGeometry object with the "MovingPoint" type.
MovingLineString	The type represents the prism of a time-parametered 1-dimensional (1D) geometric primitive (LineString), whose leaf geometry at a time position is a 1D linear object in a particular period. Intuitively a temporal geometry of a continuous movement of curve depicts a set of surfaces in a spatiotemporal domain. For example, the movement information of weather fronts or traffic congestion on roads can be shared as a TemporalPrimitiveGeometry object with the "MovingLineString" type.
MovingPolygon	The type represents the prism of a time-parametered 2-dimensional (2D) geometric primitive (Polygon), whose leaf geometry at a time position is a 2D polygonal object in a particular period. The list of points are in counterclockwise order. Intuitively a temporal geometry of a continuous movement of polygon depicts a set of volumes in a spatiotemporal domain. For example, the changes of flooding areas or the movement information of air pollution can be shared as a TemporalPrimitiveGeometry object with the "MovingPolygon" type.

Types	Description
MovingPointCloud	The type represents the prism of a time-parametered point cloud whose leaf geometry at a time position is a set of points in a particular period. Intuitively a temporal geometry of a continuous movement of point set depicts a set of curves in a spatiotemporal domain. For example, the tacking information by using Light Detection and Ranging (LiDAR) can be shared as a TemporalPrimitiveGeometry object with the "MovingPointCloud" type.

- **"datetimes"**: The **"datetimes"** member is **mandatory**. Its value is a JSON array of a sequence of monotonic increasing instants, having at least one element that is not **null**. It does NOT allow a JSON **null** value. The expression of element instant is based on its temporal coordinate reference system that is given by the **"trs"** member.

Types	Formats of the "datetimes" value	Comments
TimeInstant	[t1, t2, t3, ...]	<i>a list of monotonic increasing instants, i.e., $t1 < t2 < t3 < \dots$</i>

When the **"trs"** member has the default temporal CRS (i.e., ISO8601) object, an instant is generally as a JSON string encoded by [IETF RFC 3339](#) or a numeric value of milliseconds since midnight (00:00 a.m.) on January 1, 1970 in UTC (e.g., **1465621816590**). Here are some examples of instant.



```

YYYY (e.g., 1997)
Year and month:
  YYYY-MM (e.g., 1997-07)
Complete date:
  YYYY-MM-DD (e.g., 1997-07-16)
Complete date plus hours and minutes:
  YYYY-MM-DDThh:mmTZD (e.g., 1997-07-16T19:20+01:00)
Complete date plus hours, minutes and seconds:
  YYYY-MM-DDThh:mm:ssTZD (e.g., 1997-07-16T19:20:30+01:00)
Complete date plus hours, minutes, seconds and a decimal fraction of a second
  YYYY-MM-DDThh:mm:ss.sTZD (e.g., 1997-07-16T19:20:30.450Z)
where:
  YYYY = four-digit year
  MM   = two-digit month (01=January, etc.)
  DD   = two-digit day of month (01 through 31)
  hh   = two digits of hour (00 through 23) (am/pm NOT allowed)
  mm   = two digits of minute (00 through 59)
  ss   = two digits of second (00 through 59)
  s    = one or more digits representing a decimal fraction of a second
  TZD  = time zone designator (Z or +hh:mm or -hh:mm)

```

- **"coordinates"**: The **"coordinates"** member is **mandatory**. Its value is a JSON array of a sequence of leaf geometries of a temporal geometry, having the same number of elements as **"datetimes"**. It does NOT allow a JSON **null** value nor an empty array. Each leaf geometry corresponds to

each instant element in the array value of **"datetimes"** in order. Also, it has the same structure for all elements in the array and the same number of single positions in the elements, except **"MovingPointCloud"**. The value expression of this member is different depending on the **"type"** value as follows:

Types	Formats of the "coordinates" value	Comments
Moving Point	[[x1,y1(z1)], [x2,y2(z2)], ...]	<i>a list of Point coordinates to construct a 0D leaf geometry (point) corresponding to each instant in order</i>
Moving LineString	[[[x11,y11(z11)], [x12,y12(z12)], ...], [[x21,y21(z21)], [x22,y22(z22)], ...], ...]	<i>a list of LineString coordinate arrays to construct a 1D leaf geometry (linestring) corresponding to each instant in order</i>
Moving Polygon	[[[[ox11,oy11(oz11)], [ox12,oy12(oz12)], ...], [[ix11,iy11(iz11)], [ix12,iy12(iz12)], ...], ...], [[[ox21,oy21(oz21)], [ox22,oy22(oz22)], ...], [[ix21,iy21(iz21)], [ix22,iy22(iz22)], ...], ...], ...]	<i>a list of Polygon coordinate arrays to construct a 2D leaf geometry (polygon) corresponding to each instant in order</i>
Moving PointCloud	[[[x11,y11(z11)], [x12,y12(z12)], ...], [[x21,y21(z21)], [x22,y22(z22)], ...], ...]	<i>a list of MultiPoint coordinate arrays to construct a set of points as a leaf geometry corresponding to each instant in order</i>



- A single position is represented by a JSON array of numbers, which must contain two or more elements corresponding to the order of axis of a spatial coordinate reference system. When it refers to the default spatial CRS (i.e., WGS84), the first two elements are longitude/easting (x) and latitude/northing (y), precisely in that order and using decimal numbers. Altitude/elevation (z) MAY be included as an optional third element.
- A Point coordinate is a single position.
- A LineString coordinate array is expressed as a JSON array of two or more single positions, allowing the same positions.
- A Polygon coordinate array is expressed as a JSON array of LinearRing coordinate arrays. A LinearRing is closed LineString with 4 or more single positions to represent the boudary of a surface or the boundary of a hole in a surface, allowing the same positions. The first and last position is equivalent (they represent equivalent points). For Polygons with multiple rings, the first must be the exterior ring and any others must be interior rings (holes). A LinearRing MUST follow the right-hand rule with respect to the area it bounds, i.e., exterior rings are counterclockwise, and interior rings are clockwise.
- A MultiPoint coordinate array is a JSON array of single positions.

- **"interpolation"**: MF-JSON Prism separates out translational motion and rotational motion. The **"interpolation"** member is default and represents the translational motion of the geometry described by the **"coordinates"** value. Its value is a MotionCurve object described by one of predefined five motion curves (i.e., **"Discrete"**, **"Step"**, **"Linear"**, **"Quadratic"**, and **"Cubic"**) or a URL (e.g., <http://www.opengis.net/spec/movingfeatures/json/1.0/prism/example/motioncurve>). The name of predefined five motion curves is a case-sensitive string. In order to support various types of dynamic movements, MF-JSON Prism allows the external reference encoding by the URL to represent a MotionCurve object. The URL indicates a JSON document containing a user-defined curve of parametric motion for the movement of a feature in two- or three-dimensional space. The detail interpretation and representation of method is described in 7.2.10. If the TemporalPrimitiveGeometry object has no member with the name **"interpolation"**, it is interpreted by **"Linear"** as the default value. If a moving feature has both translational and rotational motion, MF-JSON Prism recommends using the **"base"** and **"orientations"** member with the **"coordinates"**, which represents a MovingPoint for avoiding ambiguities of the object boundary. The only usage of **"interpolation"** may bring unintended consequences for object motion encoding. Annex C shows some examples of unintended consequences of the linear interpolation without the values of **"base"** and **"orientations"**. If a moving feature has two motions of translation and rotation, using **"base"** and **"orientations"** is recommended as well as **"interpolation"**.
- **"base"**: The **"base"** member is optional, and its value is a JSON object or a JSON **null** value. When the moving feature has the base representation, the value of **"type"** is **"MovingPoint"** for its translational motion. The JSON object has two members: **"type"** and **"href"**. The **"type"** member has a JSON string to represent a 3D File format such as STL[6], OBJ[7], PLY[8], and glTF[9]. The **"href"** member has a URL to address a 3D model data. The 3D model represents a base geometry of a 3D shape, and the combination of the **"base"** and **"orientations"** members represents a 3D temporal geometry of the MF_RigidTemporalGeometry type in ISO 19141. In MF-JSON Prism, the 3D model is transformed into the fixed local coordinate reference system whose bound is **-0.5 to 0.5** for each axis concerning the right-handed local engineering coordinate reference system (i.e., right-handed orthogonal axis system, $Z = X \times Y$) and unit is **meter**. The *X* axis points forward from the base model. The *Y* axis points to the left-hand side of the model, as viewed when facing forward. The *Z* axis points up from the ground as shown in Figure 8. The origin of the local coordinate reference system is 0 for each axis and is translated into the position of each leaf point in **"coordinates"**.
- **"orientations"**: The **"orientations"** member is optional and represents rotational motion of the base representation of a member named **"base"**. It allows a JSON **null** value or an empty array if and only if the **"base"** member has a JSON **null** value. When the temporal geometry has the base representation, its value is a JSON array as a sequence of JSON objects which have two members of **"scales"** and **"angles"**, having the same number of elements of the **"datetimes"** value. The **"scales"** member has a JSON array value of numbers along the *X*, *Y*, and *Z* axis in order as three scale factors, and the **"angles"** member has a JSON array value of numbers along the *X*, *Y*, and *Z* axis in order as Euler angles in degree. Angles are defined according to the right-hand rule; a positive value represents a rotation that appears clockwise when looking in the positive direction of the axis, and a negative value represents a counter-clockwise rotation. A pair of **"scales"** and **"angles"** generates a transform matrix of the base representation at each time of the elements in **"datetimes"**. The transformation matrix at each time poses the base representation from the right-handed local engineering coordinate reference system to a geographical Cartesian coordinate system. This standard attaches to the Universal Transverse

Mercator (UTM) system. A coordinate of the base 3D model, (x, y, z) , transforms into a target 3D coordinate, (x', y', z') , by using the vector rotations with quaternions and the scaling factors, $(x, y, z) \rightarrow (x', y', z')$, as follows:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} S_x(1 - 2(Y^2 + Z^2)) & 2S_y(XY + ZW) & 2S_z(XZ - YW) & 0 \\ 2S_x(XY - ZW) & S_y(1 - 2(X^2 + Z^2)) & 2S_z(YZ + XW) & 0 \\ 2S_x(XZ + YW) & 2S_y(YZ - XW) & S_z(1 - 2(X^2 + Y^2)) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

The scale factor takes the form of a vector $S = \langle S_x, S_y, S_z \rangle$, where S_x , S_y and S_z respectively is a coefficient against each axis to extend or shrink the base representation from the local engineer coordinate reference system to the UTM system. The quaternion takes the form of a vector $Q = \langle X, Y, Z, W \rangle$, where X , Y and Z are coefficients defining an axis of rotation in an imaginary spherical space surrounding the moving feature, and W is a scalar representation of the amount of rotation applied around the axis. At this point, it should be noted that the X , Y and Z components of a quaternion are not the same as the components of a three-dimensional unit vector, nor is W a direct representation of a rotation in degree. To convert Euler angles to quaternion form, defining the angles around the X-axis (roll) as ψ , the Y-axis (pitch) as θ , and the Z-axis (yaw) as ϕ . The quaternion Q can then be calculated as follow:

$$[X = \sin(\frac{\psi}{2})\cos(\frac{\theta}{2})\cos(\frac{\phi}{2}) - \cos(\frac{\psi}{2})\sin(\frac{\theta}{2})\sin(\frac{\phi}{2})]$$

$$[Y = \cos(\frac{\psi}{2})\sin(\frac{\theta}{2})\cos(\frac{\phi}{2}) + \sin(\frac{\psi}{2})\cos(\frac{\theta}{2})\sin(\frac{\phi}{2})]$$

$$[Z = \cos(\frac{\psi}{2})\cos(\frac{\theta}{2})\sin(\frac{\phi}{2}) - \sin(\frac{\psi}{2})\sin(\frac{\theta}{2})\cos(\frac{\phi}{2})]$$

$$[W = \cos(\frac{\psi}{2})\cos(\frac{\theta}{2})\cos(\frac{\phi}{2}) + \sin(\frac{\psi}{2})\sin(\frac{\theta}{2})\sin(\frac{\phi}{2})]$$

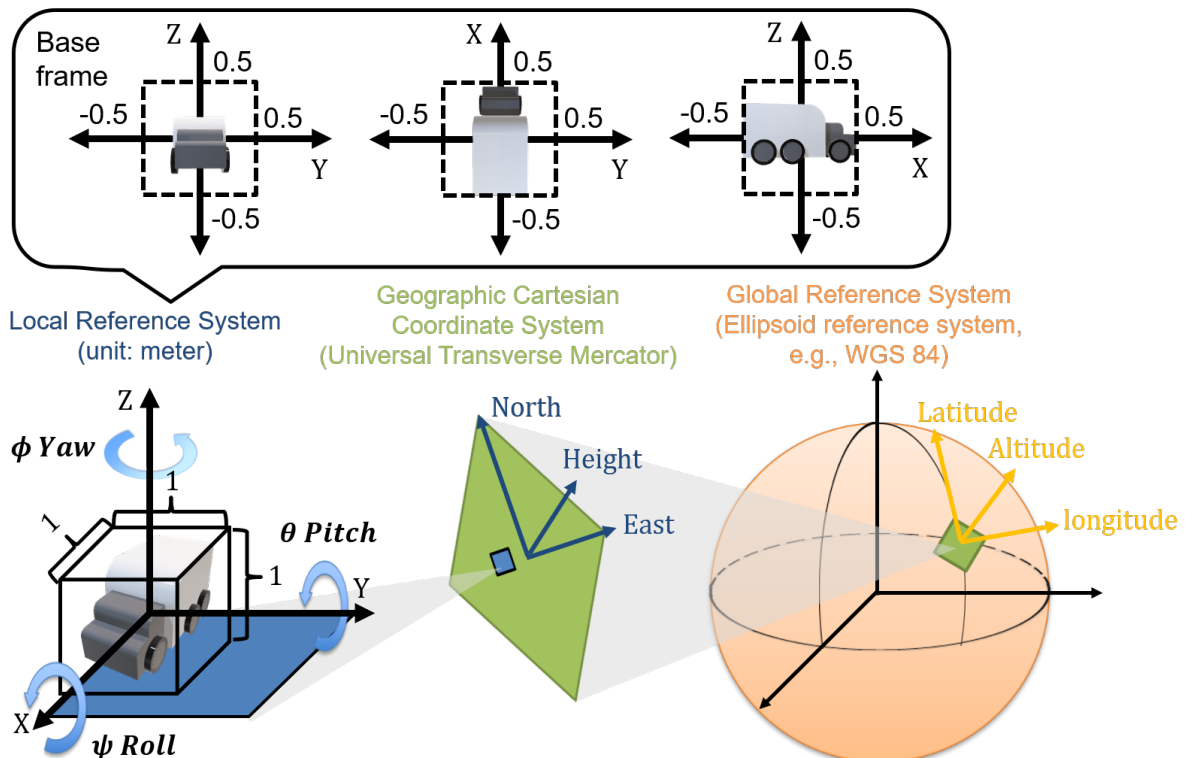


Figure 8. The transformation of a 3D model by using the orientation matrices

Requirement 2.6	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/primitive A TemporalPrimitiveGeometry object SHALL have at least three mandatory members of "type", "datetimes", and "coordinates". The value of the "type" member SHALL be one of string "MovingPoint", "MovingLineString", "MovingPolygon", and "MovingPointCloud". The value of the "datetimes" member SHALL be a JSON array of a sequence of monotonic increasing instants, having at least one element that is not null. The value of the "coordinates" member SHALL be a JSON array of a sequence of leaf geometries of a temporal geometry, having at least one element that is not null and its expression is depending on the "type".
Requirement 2.7	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/primitive/constraint The number of elements in both arrays of the "coordinates" value and the "datetimes" value SHALL be equal. The number of elements in both arrays of the "orientations" value and the "datetimes" value SHALL be equal.
Requirement 2.8	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/primitive/movingpoint A MovingPoint object SHALL have the value of the "type" = "MovingPoint". And the value of the "coordinates" member SHALL be a list of Point coordinates to construct a 0D leaf geometry (point) corresponding to each instant in order.
Requirement 2.9	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/primitive/movinglinestring A MovingLineString object SHALL have the value of the "type" = "MovingLineString". And the value of the "coordinates" member SHALL be a list of LineString coordinate arrays to construct a 1D leaf geometry (linestring) corresponding to each instant in order.
Requirement 2.10	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/primitive/movingpolygon A MovingPolygon object SHALL have the value of the "type" = "MovingPolygon". And the value of the "coordinates" member SHALL be a list of Polygon coordinate arrays to construct a 2D leaf geometry (polygon) corresponding to each instant in order.
Requirement 2.11	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/primitive/movingpointcloud A MovingPointCloud object SHALL have the value of the "type" = "MovingPointCloud". And the value of the "coordinates" member SHALL be a list of MultiPoint coordinate arrays to construct a set of points as a leaf geometry corresponding to each instant in order.

Requirement 2.12	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/primitive/interpolation The value of the "interpolation" member SHALL be a MotionCurve object. If the TemporalPrimitiveGeometry object has no member with the name "interpolation", the default MotionCurve object ("Linear") SHALL be applied.
Requirement 2.13	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/primitive/3dmodel The 3D model SHALL be transformed into the fixed local coordinate reference system whose bound is -0.5 to 0.5 for each axis and unit is meter. The coordinate reference system for the 3D model SHALL be a right-handed coordinate system.
Requirement 2.14	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/primitive/base The value of the "base" member SHALL be a JSON object having two members of "type" and "href". The "type" member has a JSON string to represent a file format, and the "href" member has a web accessible URL to address a 3D model data. If the TemporalPrimitiveGeometry object has a member with the name "base", the value of "type" in the TemporalPrimitiveGeometry object SHALL be a "MovingPoint".
Requirement 2.15	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/primitive/orientations The value of the "orientations" member SHALL be a JSON array of a JSON object that has two members of "scales" and "angles". The number of the element in array of the "orientations" value SHALL be the same as "datetimes". The "orientations" member SHALL be accompanied with the "base" member.
Requirement 2.16	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/primitive/orientations/scales The value of the "scales" member SHALL be a JSON array value of numbers along the X, Y and Z axis in order.
Requirement 2.17	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/primitive/orientations/angles The value of the "angles" member SHALL be a JSON array value of numbers of Euler angles along the X, Y and Z axis in order. The value of the "angles" member SHALL be defined according to the right-hand rule and unit is degree.

Requirement 2.18

<http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/primitive/crs>

The value of the "crs" and "trs" member SHALL be a JSON object to represent a CoordinateReferenceSystem (CRS) object. If no CRS instance can be so acquired from upper-level braces of a JSON object, the default spatial CRS (and temporal CRS) SHALL be applied.

7.2.1.2. TemporalComplexGeometry Object

A TemporalComplexGeometry object represents a set of TemporalPrimitiveGeometry objects. When a TemporalGeometry object has a "type" member is "MovingGeometryCollection", the object is specialized as a TemporalComplexGeometry object with one additional **mandatory** member named "prisms". The value of the "prisms" member is represented by a JSON array of a set of TemporalPrimitiveGeometry instances, having at least one element in the array.

```
{
  "type": "MovingGeometryCollection", // (MANDATORY)
  "prisms": [ //(MANDATORY)
    {
      "type": "MovingPoint | MovingLineString | MovingPolygon | MovingPointCloud", // (MANDATORY) vbar | as a
means to select ONE type.
      "datetimes": [...], //(MANDATORY)
      "coordinates": [...], //(MANDATORY)
      "interpolation": "...", //(DEFAULT)
      ...
    },
    ...
  ],
  "crs": {...}, //(DEFAULT)
  "trs": {...} //(DEFAULT)
}
```

- **MovingGeometryCollection:** Each element of "prisms" can be an TemporalPrimitiveGeometry with one of types of "MovingPoint", "MovingLineString", "MovingPolygon", and "MovingPointCloud". The leaf geometry at a time position is the union of each leaf of any temporal geometries at the same time.

Requirement 2.19

<http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/complex>

A TemporalComplexGeometry object SHALL have at least two mandatory members of "type" and "prisms". The value of the "type" member SHALL be a "MovingGeometryCollection" string. The value of the "prisms" member SHALL be a JSON array of a set of TemporalPrimitiveGeometry instances, having at least one element in the array.

Requirement 2.20	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/complex/movinggeometrycollection A MovingGeometryCollection object SHALL the value of the "type" = "MovingGeometryCollection". Each element of "prisms" SHALL be a TemporalPrimitiveGeometry instance with one of types of "MovingPoint", "MovingLineString", "MovingPolygon", and "MovingPointCloud". The leaf geometry at a time position must be an instance of type "GeometryCollection" of GeoJSON, which is the union of each leaf of any temporal geometries at the same time.
Requirement 2.21	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/complex/crs The value of the "crs" and "trs" member SHALL be a JSON object to represent a CoordinateReferenceSystem (CRS) object. If no CRS instance can be so acquired from upper-level braces of a JSON object, the default spatial CRS (and temporal CRS) SHALL be applied.

7.2.2. TemporalProperties Object

A TemporalProperties object is a JSON array of ParametricValues objects that groups a collection of dynamic non-spatial attributes and its parametric values with time.

7.2.2.1. ParametricValues Object

A ParametricValues object is a JSON object to represent a collection of parametric values of dynamic non-spatial attributes that are ascertained at the same times. A parametric value may be a time-varying measure, a sequence of texts, or a sequence of images. Even though the parametric value may depends on the spatiotemporal location, MF-JSON Prism only considers the temporal dependencies of their changes of value.

Example 3: Examples of a ParametricValues object

```
{
  "datetimes": [...], //(MANDATORY) a JSON Array of time instants
  "@property0": {      // @property0 whose name is any string defined by an application.
    "type": "Measure",  //(MANDATORY) a predefined string among `Measure`, `Text`, and `Image`
    "values": [...],    //(MANDATORY) a JSON Array of values
    "interpolation": "...", //(DEFAULT) a predefined string or a URL
    "form": "...",      //(OPTIONAL) a unit of measurement
    "description": "any string" //(OPTIONAL) any string content for an application
  },
  "@property1": {      // @property1 whose name is any string defined by an application, but the name is not the same as
    // @property0.
    "type": "Text", // @property1 has values at the same time instants of @property0
    ...
  },
  "@property2" : {     // @property1 whose name is any string defined by an application, but the name is same neither
    // @property0 nor @property1.
    "type": "Image",
    ....
  },
  ...
}
```

A ParametricValues object has one **"datetimes"** member and more than one member with the name *@propertyN*, where *@propertyN* is any string defined by an application as a dynamic attribute. The value of the *"@propertyN"* member is a JSON object that has the following fields:

- **"type"**: The **"type"** member is mandatory and has a string as one of the followings:

Type strings	Descriptions
Measure	The "values" member contains any numeric values.
Text	The "values" member contains any strings.
Image	The "values" member contains Base64 strings converted from images or URLs to address images.

- **"values"**: The **"values"** member is mandatory and has a JSON array whose element is a string (including null, true and false) or numeric value to represent sample values. The number of elements is the same as the **"datetimes"** ones. There is an one-to-one correspondence between the elements of **"values"** of a *@propertyN* object and **"datetimes"** as a temporal sequence of pairs (v, t) , where v is a value of measurement and t is its sampling time.
- **"interpolation"**: The **"interpolation"** member is default and has a JSON string or a URL to describe an interpolation method. A dynamic attribute also needs an interpolation method to estimate its value at any time between two successive instants like the TemporalPrimitiveGeometry object. Given the same number of $V = (v_0, v_1, v_2, ..., v_n)$ in **"values"** and $T = (t_0, t_1, t_2, ..., t_n)$ in **"datetimes"**, this standard defines four predefined methods for a parametric value with time: **"Discrete"**, **"Step"**, **"Linear"**, and **"Regression"**. Each method interpolates (or extrapolates) from sample values as follows.

Names	Descriptions
Discrete	The sampling of the attribute occurs such that it is not possible to regard the series as continuous; Thus, there is no interpolated value if t is not an element in " datetimes ".
Step	The values are not connected at the end of a subinterval with two successive instants. The value just jumps from one value to the other at the end of a subinterval.
Linear	The values are essentially connected and a linear interpolation estimates the value of the property at the indicated instant during a subinterval.
Regression	The value of the attribute at the indicated instant is extrapolated from a simple linear regression model with the whole values corresponding to the all elements in " datetimes ".

The available interpolation method restricts according to "**type**" value as follows.

Type strings	Available interpolations
Measure	Discrete, Step, Linear, Regression
Text	Discrete, Step
Image	Discrete, Step

If the *@propertyN* object has no member with the name "**interpolation**", it is interpreted by "**Discrete**" as the default value. For a URL, this standard refers to the [InterpolationCode Codelist](#) defined in [OGC TimeseriesML 1.0](#)[OGC 15-042r3] between neighboring points in a timeseries, e.g., "<http://www.opengis.net/def/timeseries/InterpolationCode/Continuous>", "<http://www.opengis.net/def/timeseries/InterpolationCode/Discontinuous>", and etc.

- "**form**": The "**form**" member is optional and its value is a JSON string as a common code (3 characters) described in the Code List Rec 20 by the UN Centre for Trade Facilitation and Electronic Business (UN/CEFACT)[2] or a URL specifying the unit of measurement. This member is applied only for a temporal property whose value type is **Measure**.
- "**description**": The "**description**" member is optional and its value is any string to describe a short description.

Requirement 2.22	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tproperties A TemporalProperties object SHALL be a JSON array of ParametricValues objects.
-------------------------	---

Requirement 2.23	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tproperties/pvalues A ParametricValues object SHALL have at least a mandatory member with the named "datetimes" and more than one member with the name @propertyN, where @propertyN is any string defined by an application as a temporal property.
Requirement 2.24	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tproperties/pvalues/property A @propertyN object SHALL have at least two mandatory members of "type" and "values". The value of the "type" member SHALL be one of string "Measure", "Text", and "Image". The value of the "values" member SHALL be a JSON array whose element is a string (including null, true and false) or numeric value.
Requirement 2.25	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tproperties/pvalues/property/constraint The number of elements in both arrays of the "datetimes" value and the "values" value SHALL be equal.
Requirement 2.26	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tproperties/pvalues/property/interpolation The value of the "interpolation" member SHALL be one of string "Discrete", "Step", "Linear", and "Regression" or a URL to address a code list defined in OGC TimeseriesML 1.0 between neighboring points in a timeseries. If the @propertyN object has no member with the name "interpolation", the default interpolation method ("Discrete") SHALL be applied.
Requirement 2.27	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tproperties/pvalues/property/interpolation/constraint The available interpolation method SHALL be restricted according to "type" value as follows. <ul style="list-style-type: none"> • Measure: Discrete, Step, Linear, Regression • Text: Discrete, Step • Image: Discrete, Step
Requirement 2.28	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tproperties/pvalues/property/form The value of the "form" member SHALL be a common code (3 characters) described in the list of Code List Rec 20 [UN/CEFACT] or a URI denoting a unit-of-measure defined in a web resource.

7.2.3. CoordinateReferenceSystem Object

A coordinate reference system (CRS) is usually comprised of two components: one coordinate system and one datum (reference frame). The IETF GeoJSON format recommends that the CRS for all GeoJSON coordinates is a geographic coordinate reference system, using the World Geodetic System 1984 (WGS 84)[1] datum, with longitude and latitude units of decimal degrees. This is equivalent to the CRS identified by the [EPSG:4326](#) with the axis Lat-Long. For a 3D coordinate with an altitude/elevation value, the value refers to distance above mean sea level (MSL or AMSL). However, a moving feature needs a temporal coordinate system and may be described by an engineering or parametric coordinates in a certain application, such as robot or vessel navigation. Therefore, MF-JSON Prism allows the `"crs"` member to describe an alternative CRS whose representation follows the way as described in GeoJSON-with-CRS[4]. Also it defines the `"trs"` member to describe a temporal CRS against which time is measured. A temporal CRS is always 1D with a temporal datum (origin) and its time unit.

- A non-null `CoordinateReferenceSystem` (CRS) object is a JSON object with two mandatory members of the name `"type"` and `"properties"`.
- The value of the `"type"` member is a JSON string as one of the string `"Name"` and `"Link"`, indicating the type of CRS object.
- The value of the `"properties"` member is a JSON object with three optional members named `"name"`, `"href"`, and `"type"` whose value is a JSON string or JSON `null` value.
- CRS shall not change coordinate ordering.

Depending on the value of `"type"`, there are three expressions: Named CRS, Linked CRS, and Default CRS.

7.2.3.1. Named CRS

A Named CRS object indicates a coordinate reference system by name. In this case, the value of its `"type"` member is the string `"Name"`. The value of its `"properties"` member is a JSON object containing a `"name"` member whose value is a string identifying a coordinate reference system (not JSON `null` value). The value of `"href"` and `"type"` is a JSON `null` value. This standard recommends an [EPSG\[3\]](#) code as the value of `"name"`, such as `"EPSG::4326"`.

Example 4: Example of a named CRS object

```
{
  "type": "Name",
  "properties": {
    "name": "urn:ogc:def:crs:OGC:1.3:CRS84"
  }
}
```

7.2.3.2. Linked CRS

A linked CRS object has one required member: `"href"` and one optional member: `"type"`. The value of the required `"href"` member is a dereferenceable URI. The value of the optional `"type"` member is a string that hints at the format used to represent CRS parameters at the provided URI. Suggested

values are: "Proj4", "OGCWKT", "ESRIWKT", but others can be used.

Example 5: Example of a linked CRS object

```
{
  "type": "Link",
  "properties": {
    "href": "http://example.com/crs/42",
    "type": "Proj4"
  }
}
```

Example 6: Example of a relative linked CRS object with an auxiliary file

```
{
  "type": "Link",
  "properties": {
    "href": "data.crs",
    "type": "OGCWKT"
  }
}
```

7.2.3.3. Default CRS

MF-JSON Prism defines two default CRS objects for a spatial CRS and a temporal CRS as follows:

- **WGS84** with longitude and latitude units of decimal degrees: This CRS object is the default value of the "crs" member.

```
"crs" : {
  "type": "Name",
  "properties": {
    "name": "urn:ogc:def:crs:OGC:1.3:CRS84"
  }
}
```

- **ISO8601** of dates and times using the Gregorian calendar and 24 hour local or Coordinated Universal Time (UTC): This CRS object is the default value of the "trs" member.

```
"trs" : {
  "type": "Name",
  "properties": {
    "name": "urn:ogc:data:time:iso8601"
  }
}
```

A JSON `null` value of the "crs" and "trs" is considered as **WGS84** and **ISO8601**, respectively.

Requirement 2.29	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/crs A CoordinateReferenceSystem object SHALL have two mandatory members of "type" and "properties". The value of the "type" member SHALL be one of string "Name" and "Link". The value of the "properties" member SHALL be a JSON object with three optional members named "name", "href", and "type" whose value is a JSON string or JSON null value.
Requirement 2.30	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/crs/named A named CRS object SHALL have the value of the "type" = "Name". And the value of the "properties" member SHALL be a JSON object containing a "name" member whose value is a string identifying a coordinate reference system.
Requirement 2.31	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/crs/linked A linked CRS object SHALL have the value of the "type" = "Link". And the value of the "properties" member SHALL be a JSON object containing a "href" member whose value is a dereferenceable URI.

7.2.4. MovingFeature Object

A MovingFeature object in a MF-JSON Prism document is a JSON object to represent a moving feature that is spatially and temporally bounded, and its position (continuously) changes over time with the members as shown in [Example 7](#).

Example 7: Members of a MovingFeature object

```
{
  "type": "Feature", //(MANDATORY)
  "temporalGeometry": {...}, //(MANDATORY)
  "temporalProperties": [...], //(OPTIONAL)
  "crs" : {...}, //(DEFAULT)
  "trs" : {...}, //(DEFAULT)
  "time": [...], //(OPTIONAL)
  "bbox": [...], //(OPTIONAL)
  "geometry": {...}, //(OPTIONAL)
  "properties": {...}, //(OPTIONAL)
  "id": ... //(OPTIONAL)
}
```

- A MovingFeature object has a mandatory member with the name "type" whose value is the "Feature" string.



Due to the compatibility issue with GeoJSON, MF-JSON Prism uses the same keyword to abstract a moving feature. If a GeoJSON document has no foreign member defined in this specification, applications can interpret the data by the semantics of GeoJSON members and types.

- A MovingFeature object has a mandatory member with the name `"temporalGeometry"` and its value is a TemporalGeometry object as defined above (see 7.2.1).
- A MovingFeature object has an optional member with the name `"temporalProperties"` as additional dynamic non-spatial attributes whose value varies over time. The `"temporalProperties"` member has a TemporalProperties object as defined in 7.2.2 or a JSON `null` value.
- A MovingFeature object has two default members with the name `"crs"` and `"trs"` for representing its spatial and temporal coordinate reference system respectively. The value of the `"crs"` and `"trs"` member is a CoordinateReferenceSystem (CRS) object (see 7.2.3). If a MovingFeature object has no `"crs"` and `"trs"`, the object refers to the default spatial and temporal CRS. The value expression of spatial and temporal coordinates of `"temporalGeometry"` and `"temporalProperties"` is obedient to these CRS objects.
- A MovingFeature object can optionally have a `"time"` member for the temporal coordinate range to cover all of temporal geometries and temporal properties with a LifeSpan object (see 7.2.6).
- A MovingFeature object can optionally have a `"bbox"` member for the spatial coordinate range with a BoundingBox object (see 7.2.7). It represents a spatial domain to cover a temporal geometry or a collection of temporal geometries.
- A MovingFeature object has an optional member with the name `"geometry"` allowing a JSON `null` value, as the same semantics of GeoJSON (see 7.2.8).
- A MovingFeature object has an optional member with the name `"properties"` allowing a JSON `null` value, as the same semantics of GeoJSON (see 7.2.9). The properties member is an object (i.e., any JSON object or a JSON `null` value).
- A MovingFeature object has an optional member with the name `"id"` to represent a commonly used identifier and the value of this member is either a JSON string or number.



The difference from `"temporalProperties"` is that a `"properties"` member represents static attributes whose value is fixed for the life span of a moving feature.

Requirement 2.32

<http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/feature>
A MovingFeature object SHALL be a GeoJSON Feature object that have two mandatory members of `"type"` and `"temporalGeometry"`. The value of the `"type"` member SHALL be a `"Feature"` string. The value of the `"temporalGeometry"` member SHALL be a JSON object to represent a TemporalGeometry object, not allowing the JSON `null`.

Requirement 2.33	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/feature/temporalProperties The value of the " temporalProperties " member SHALL be a JSON array of TemporalProperties objects. It allows a JSON null value.
Requirement 2.34	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/feature/crs The value of the " crs " and " trs " member SHALL be a JSON object to represent a CoordinateReferenceSystem (CRS) object. If no CRS instance can be so acquired in a file, the default spatial CRS (and temporal CRS) SHALL be applied to the object.
Requirement 2.35	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/feature/time The value of the " time " member SHALL be a JSON array to represent a LifeSpan object. It SHALL be represented as a temporal domain to cover a temporal geometry or a collection of temporal geometries. It allows a JSON null value.
Requirement 2.36	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/feature/bbox The value of the " bbox " member SHALL be a JSON array to represent a BoundingBox object. It SHALL be represented as a spatial domain to cover a temporal geometry or a collection of temporal geometries. It allows a JSON null value.
Requirement 2.37	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/feature/geometry The value of the " geometry " member SHALL be any JSON object or a JSON null value for the " geometry " member in a GeoJSON Feature object.
Requirement 2.38	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/feature/properties The value of the " properties " member SHALL be any JSON object or a JSON null value for the " properties " member in a GeoJSON Feature object.

7.2.5. MovingFeatureCollection Object

MovingFeatureCollection object is a JSON object with the following members as shown in [Example 8](#). It bundles a collection of moving features into a logical abstraction with a certain alias.

Example 8: Members of a MovingFeatureCollection object

```
{
  "type": "FeatureCollection", //(MANDATORY)
  "features": [...], //(MANDATORY)
  "crs" : {...}, //(DEFAULT)
  "trs" : {...}, //(DEFAULT)
  "bbox": [...], //(OPTIONAL)
  "time": [...], //(OPTIONAL)
  "label": "...", //(OPTIONAL)
}
```

- A MovingFeatureCollection object has a mandatory member with the name "type" whose value is the "FeatureCollection" string.
- A MovingFeatureCollection object has a mandatory member with the name "features". Its value is a JSON array and each element of the array is a Feature object as defined above (see 7.2.4).
- A MovingFeatureCollection object has two default members with the name "crs" and "trs" like a Feature object. If a MovingFeatureCollection object has the value of the "crs" or "trs" member, its elements of "features" refers to the coordinate reference systems of the collection object, not features' reference systems. The "crs" and "trs" member is on the top-level object in a hierarchy of type MovingFeatureCollection, MovingFeature, TemporalGeometry/TemporalProperties orders.
- A MovingFeatureCollection object has optional members with the name "bbox" and "time" for the spatial coordinate range and the temporal coordinate range of the life span to cover all element instances of the collection respectively. The value of "bbox" and "time" is a BoundingBox object and a LifeSpan object, respectively. They allow a JSON null value.
- A MovingFeatureCollection object may optionally have a "label" member with a JSON string value to indicate an alias of the collection. It allows a JSON null value.

Requirement 2.39	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/featurecollection A MovingFeatureCollection object SHALL be a GeoJSON FeatureCollection object that have two mandatory members of "type" and "features". The value of the "type" member SHALL be a "FeatureCollection" string. The value of the "features" member SHALL be a JSON array whose element is a MovingFeature object.
Requirement 2.40	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/featurecollection/constraints The number of elements in an array of the "features" value SHALL be more than 1.

Requirement 2.41	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/featurecollection/crs The value of the " crs " and " trs " member SHALL be a JSON object to represent a CoordinateReferenceSystem (CRS) object. If no CRS instance can be so acquired in a file, the default spatial CRS (and temporal CRS) SHALL be applied to the object.
Requirement 2.42	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/featurecollection/bbox The value of the " bbox " member SHALL be a JSON array to represent a BoundingBox object. It SHALL be represented as a spatial coordinate range to cover all element instances of the collection. It allows a JSON null value.
Requirement 2.43	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/featurecollection/time The value of the " time " member SHALL be a JSON array to represent a LifeSpan object. It SHALL be represented as a temporal coordinate range to cover all element instances of the collection. It allows a JSON null value.
Requirement 2.44	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/featurecollection/label The value of the " label " member SHALL be a JSON string to indicate an alias of the collection. It allows a JSON null value.

7.2.6. LifeSpan Object

A LifeSpan object is a JSON array of two strings encoded by [IETF RFC 3339](#) (e.g., "2016-09-11T06:10:32Z") to represent a particular period $[t_s, t_e]$ and $t_s \leq t_e$. Namely, its expression of an element in the array refers to the default temporal CRS object, i.e., ISO8601.

Example 9: An example of LifeSpan object with ISO 8601:2004

```
[ "2011-07-14T22:01:01Z", "2011-07-15T01:11:22Z" ]
```

Requirement 2.45	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/time A LifeSpan object SHALL be a JSON array having two elements of string encoded by IETF RFC 3339 .
Requirement 2.46	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/time/element The first element SHALL be less than or equal to the second element.

Requirement 2.47	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/time/crs The expression of an element in the array SHALL refer to the default temporal CRS object (ISO8601).
-------------------------	--

7.2.7. BoundingBox Object

A BoundingBox object is a JSON array of length $2 \cdot n$ where n is the number of dimensions represented in the spatial bounding box. The order of values follows the axes order of single position of longitude, latitude, and elevation. The expression of an element in the array refers to the default spatial CRS object, i.e., WGS84.

Example 10: An example of BoundingBox object with WGS 84

```
[139.757083, 35.627483, 0.0, 139.757716, 35.627701, 4.5]
```

Requirement 2.48	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/bbox A BoundingBox object SHALL be a JSON array of length $2 \cdot n$ where n is the number of dimensions represented in the spatial bounding box.
Requirement 2.49	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/bbox/element The elements in the array SHALL be two coordinates (lower-bound coordinate and upper-bound coordinate). The order of values SHALL follow the axes order of single position of longitude, latitude, and elevation.
Requirement 2.50	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/bbox/crs The expression of an element in the array SHALL refer to the default spatial CRS object (WGS 84).

7.2.8. Geometry Object

A Geometry object is the same as a GeoJSON Geometry object defined in GeoJSON [IETF RFC 7946](https://tools.ietf.org/html/rfc7946).

7.2.9. Properties Object

A Properties object is any JSON object or a JSON null value for the "properties" member in a GeoJSON Feature object defined in GeoJSON [IETF RFC 7946](https://tools.ietf.org/html/rfc7946).

7.2.10. MotionCurve Objects

In MF-JSON Prism, the concept of motion curves is important to represent various types of dynamic movements, especially non-linear movements of features.

7.2.10.1. Predefined curves

In order to reduce the encoding cost and the data size, the MF-JSON Prism encoding defines five strings of built-in motion curves for the value of the "interpolation" member in a TemporalPrimitiveGeometry object: "Discrete", "Step", "Linear", "Quadratic", and "Cubic". For these built-in motion curves, each leaf geometry of the TemporalPrimitiveGeometry object has the same structure for the elements in the value array of the "coordinates" member and the same number of single positions in the elements, except "MovingPointCloud". For example, if a leaf linestring consists of two single positions in a temporal geometry of type "MovingLineString", the other leaves also have two single positions. If a leaf polygon in a temporal geometry of type "MovingPolygon" has one interior ring, the others also have one interior ring and the number of positions of rings is same.

Example 11: Predefined motion curves of a TemporalPrimitiveGeometry object

```
{
  ....,
  "datetimes": [...], //T={t_0, t_1, ..., t_n}
  "coordinates": [...], //G={G_0, G_1, ..., G_n}
  "interpolation": "Discrete|Step|Linear|Quadratic|Cubic", // vbar | as a means to select ONE.
  ....
}
```

When a geographic feature is moving, each variable (i.e., x , y , z) of position can be expressed in terms of another variable called a parameter. The two most commonly used parameters are time and distance traveled along the path of motion. This standard interprets a value of the "interpolation" member in a TemporalPrimitiveGeometry object as parametric equations with the time parameter. In general, parametric equations of 3D movement are represented by any three such functions that depending on the same variable t as $x = x(t)$, $y = y(t)$, and $z = z(t)$. In other words, the position $P(t)$ is described as

$$P(t) = (x(t), y(t), z(t));$$

velocity v is the derivative of position (first derivative of position) with respect to time as follows:

$$V(t) = P'(x'(t), y'(t), z'(t));$$

and acceleration a is the derivative of velocity (second derivative of position) with respect to time as follows:

$$A(t) = V'(t) = P''(x''(t), y''(t), z''(t)).$$



For example, when a geographic feature continuously moves over time but there are only four explicit positions as shown in Figure 9. Depending on the parametric equation, the path of motion can be modeled with different curves in the plan. The "interpolation" member allows applications to design their time-varying positions at any time using few positions and a parametric curve. In particular, a parametric curve is associated with parametric continuity to describe the smoothness of the parameter's value with distance along the curve. The various order of parametric continuity can be described as follows:

- C^0 : Curves are continuous; zeroth-order parametric continuity.
- C^1 : First derivatives are continuous; first-order parametric continuity.
- C^2 : First and second derivatives are continuous; second-order parametric continuity.
- C^n : First through n -th derivatives are continuous.

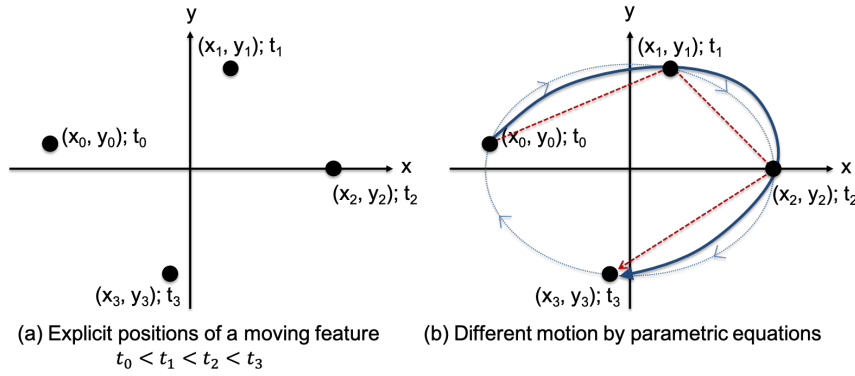


Figure 9. The dynamic motion of a geographic feature on parametrized curves

Given $N = (n + 1)$ leaf geometries $G = \{G_0, G_1, \dots, G_n\}$ at time instants in $T = \{t_0, t_1, \dots, t_n\}$ with the strictly increasing condition $t_0 < t_1 < \dots < t_n$ and one-to-one correspondence, a temporal geometry TG can be drawn by a bundle of k temporal trajectories (i.e., parametric curves with the time parameter) $\{C_0, C_1, \dots, C_k\}$, where k is the number of single positions in any leaf geometry. A temporal trajectory C is drawn for each position in $P = \{P_0, P_1, \dots, P_n\}$ at each time instant in $T = \{t_0, t_1, \dots, t_n\}$ such that $P_i \in G_i$ and $0 \leq i \leq n$, and its curve is realized by spline interpolation on subintervals. Namely, a temporal trajectory is implemented by a set of piecewise low-degree polynomials $\{SC_1, SC_2, \dots, SC_n\}$ whose derivatives satisfy some continuity constraints across sub-curve boundaries.

Each type of built-in parametric curves interpolates a time-varying position $P(t)$ for any time instant t in the particular period $I = [t_0, t_n]$ on the temporal trajectory as follows:

$$P(t) = C(t) = \begin{cases} SC_1(t), & t \in [t_0, t_1], \\ SC_2(t), & t \in [t_1, t_2], \\ \dots, & \\ SC_n(t), & t \in [t_{n-1}, t_n]. \end{cases}$$

Note that $C(t_0) = P_0, C(t_1) = P_1, \dots, C(t_n) = P_n$.

"Discrete"

The positions are NOT connected. The position is valid only at the time instant in T .

$$P(t) = \begin{cases} P_i & \text{if } t = t_i \in T \\ \emptyset & \text{otherwise.} \end{cases}$$

"Step"

It just jumps from one position to the next at the end of a subinterval. The curve is not continuous but would be useful for representing an accident or event. This interpolation requires at least two positions, i.e., $N \geq 2$.

$$\forall t \in [t_{i-1}, t_i]: P(t) = \begin{cases} P_{i-1} & \text{if } t < t_i \\ P_i & \text{otherwise.} \end{cases}$$

"Linear"

This method is the **default** value of the **"interpolation"** member. It connects straight lines between positions with zeroth-order parametric continuity (C^0) on interval I . The position with respect to time is constructed from linear splines that are two-positions interpolating polynomials. Therefore, this interpolation also requires at least two positions, i.e., $N \geq 2$.

$$\forall t \in [t_{i-1}, t_i]: P(t) = C(t) = SC_i(t) = \frac{t - t_i}{t_{i-1} - t_i} P_{i-1} + \frac{t - t_{i-1}}{t_i - t_{i-1}} P_i.$$

"Quadratic"

This method interpolates the position at time t by using a piecewise quadratic spline on each interval $[t_{i-1}, t_i]$ with first-order parametric continuity (C^1). Between consecutive positions, piecewise quadratic splines are constructed from the following parametric equations in terms of the time variable. It means the curve of a temporal trajectory is continuous and has a continuous first derivative at the positions in P except two end positions. For this interpolation, at least three leaves at particular times are required, i.e., $N \geq 3$.

$$\forall t \in [t_{i-1}, t_i]: P(t) = C(t) = SC_i(t) = a_i t^2 + b_i t + c_i,$$

$$SC_i(t_{i-1}) = P_{i-1}, \quad SC_i(t_i) = P_i = SC_{i+1}(t_i),$$

$$P'(t_i) = V(t_i) = SC'_i(t_i) = SC'_{i+1}(t_i), \quad a_0 = 0.$$

"Cubic"

This method interpolates the position at time t by using a Catmull-Rom (cubic) spline on each interval $[t_i, t_{i+1}]$ with first-order parametric continuity (C^1). The Catmull-Rom splines is a method that approximate a set of positions with a smooth polynomial function that is piecewise-defined. This interpolation requires at least four positions, i.e., $N \geq 4$, i.e., position P_{i-1} and P_{i+2} are needed to calculate the spline between positions of P_i and P_{i+1} . In order to promise the same interpolation way, this standard specifies a Catmull-Rom interpolation function by the following parametric equations in terms of the time variable: This matrix representation defines the cubic curve that represents the portion of the total curve between two successive control positions: P_i and P_{i+1} , specifying the tangent $\frac{P_{i+1} - P_{i-1}}{2}$ and $\frac{P_{i+2} - P_i}{2}$ at each control position, respectively.

$$\forall t \in [t_i, t_{i+1}]: P(t) = C(t) = SC_i(t) = SC_i(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} M \begin{bmatrix} P_{i-1} \\ P_i \\ P_{i+1} \\ P_{i+2} \end{bmatrix},$$

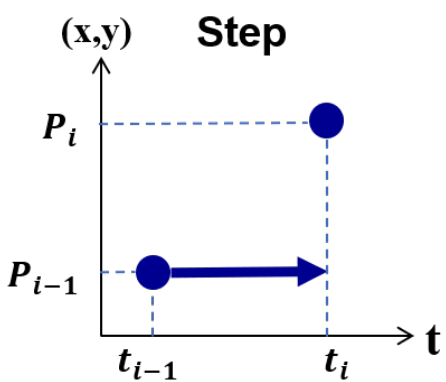
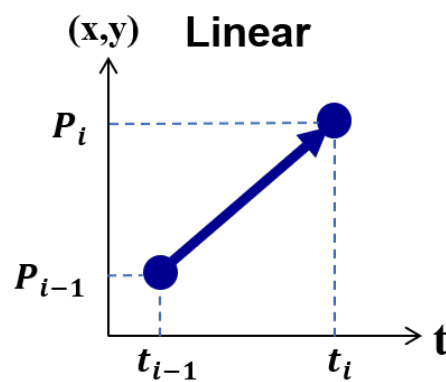
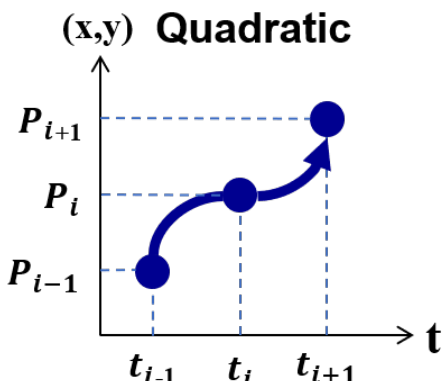
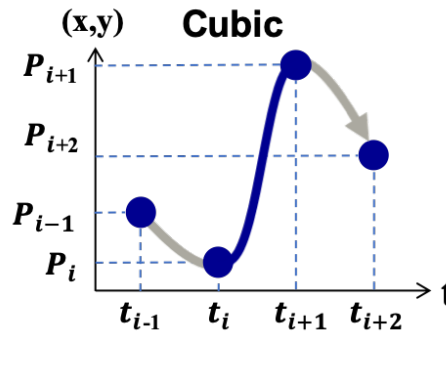
where

$$u = \frac{t - t_i}{t_{i+1} - t_i},$$

$$M = \frac{1}{2} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 2 & -5 & 4 & -1 \\ -1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 0 \end{bmatrix},$$

and

$$P'(t_0) = P_1 - P_0, \quad P'(t_n) = P_n - P_{n-1}.$$

Names	Curves	Names	Curves
Step		Linear	
Quadratic		Cubic	

7.2.10.2 URLs for user-defined parametric curve

If applications need to define their own interpolation methods, the "interpolation" member in the TemporalPrimitiveGeometry object has a URL to address a JSON array of parametric equations defined on a set of intervals of parameter t -value.

Example 12: A TemporalPrimitiveGeometry instance of with a user-defined parametric curve

```

"temporalGeometry": {
  "type": "MovingPoint",
  "datetimes": [...], //T={t_0, t_1, ..., t_n}
  "coordinates": [...], //G={G_0, G_1, ..., G_n}
  "interpolation": "http://www.opengis.net/spec/movingfeatures/json/1.0/prism/example/motioncurve",
  ....
}

```


The URL indicates a JSON object having three members of "crs", "trs", and "equations" as follows.

Example 13: The members of a user-defined parametric curve

```
{
  "crs": {...},
  "trs": {...},
  "equations": [...]
}
```

- The value of the "crs" member is a spatial CRS object to indicate the order of axes and the position expression.
- The value of the "trs" member is a temporal CRS object to indicate how to use the time instant in the equations to interpolate a new position at a certain time.
- The value of the "equations" is a JSON array composing of a JSON object which has three members of "coefficients", "time", and "enclosed" to represent a parametric equation within a temporal interval. The position $P(t)$ at time position t is derived from a "coefficients" value as a multidimensional array of **polynomials** of $(A_1(t), A_2(t), A_3(t), \dots, A_k(t))$ coordinates during a particular period.

$$[A_0(t) = a_{0, n-1}t^{n-1} + a_{0, n-2}t^{n-2} + \dots + a_{0, 0}t^0]$$

$$[A_1(t) = a_{1, n-1}t^{n-1} + a_{1, n-2}t^{n-2} + \dots + a_{1, 0}t^0]$$

...

$$[A_k(t) = a_{k, n-1}t^{n-1} + a_{k, n-2}t^{n-2} + \dots + a_{k, 0}t^0]$$

If a time position does not belong to any particular period of the elements of equations, there is no interpolation equations at that time. The order of arrays for the interpolation formula of a temporal position SHALL be followed by the order of axes of the CRS. For example, when the CRS is the default spatial CRS (i.e., WGS84), the order of axes is sequentially x (longitude), y (latitude), and z (altitude). The particular periods between any two elements of equations only allows empty or 0-dimensional intersection. The unit of measure of the temporal CRS is applied for computing parametric equations. For example, a time position encoded by IETF RFC 3339 is converted into a signed 64-bit integer(long) value that represents milliseconds, when it refers to the default temporal CRS (i.e., ISO8601).

Example 14: Example of MotionCurve object (<http://www.opengis.net/spec/movingfeatures/json/1.0/prism/example/motioncurve>)

```
{
  "crs" : {
    "type": "Name",
    "properties": {
      "name": "urn:ogc:def:crs:OGC:1.3:CRS84"
    }
  },
  "trs" : {
    "type": "Name",
    "properties": {
      "name": "urn:ogc:data:time:iso8601"
    }
  },
  "equations" :[
    {
      "coefficients": [[1.0, 3.0, 4.1], [2.0, 2.1, 3.0]], // x = 1.0*t^2 + 3.0*t + 4.1 , y = 2.0*t^2 + 2.1*t + 3.0
      "time": ["2011-07-14T22:01:01Z", "2011-07-14T23:01:01Z"],
      "enclosed": [false, true]
    },
    {
      "coefficients": [[4.0, 2.0], [1.0, 2.0]], // x = 4.0*t + 2.0 , y = 1.0*t + 2.0
      "time": ["2011-07-14T23:01:01Z", "2011-07-15T00:01:01Z"],
      "enclosed": [true, true]
    },
    {
      "coefficients": [["sin", "0.0"], ["cos", "0.0"]], // x = sin*t , y = cos*t
      "time": ["2011-07-15T00:01:01Z", "2011-07-16T00:01:01Z"],
      "enclosed": [true, false]
    }
  ]
}
```

Requirement 2.51	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/interpolation The "coordinate" member in the TemporalPrimitiveGeometry object SHALL be interpolated by MotionCurve object. The value of the "interpolation" member in the TemporalPrimitiveGeometry object SHALL be one of string "Discrete", "Step", "Linear", "Quadratic", and "Cubic", or a URL.
Requirement 2.52	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/interpolation/userdefined A URL to address a user-defined parametric curve SHALL address a JSON document containing a JSON object, which has three members with name "crs", "trs", and "equations". The value of the "crs" and "trs" member SHALL be a spatial and temporal CRS object, respectively. The value of the "equations" member SHALL be not empty and be a JSON array whose element is a user-defined parametric segment having three members of "coefficients", "time", and "enclosed".

Requirement 2.53	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/interpolation/userdefined/equations/coefficients The value of "coefficients" member SHALL be a JSON array whose element is an array of coefficients of the interpolation formula of a temporal position. The order of elements in array of the "coefficients" value for the interpolation formula of a temporal position SHALL follow the order of the spatial CRS.
Requirement 2.54	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/interpolation/userdefined/equations/time The value of "time" member SHALL be a JSON array to represent a particular period with two instants of t_s and t_e . The expression of an element in the "time" value SHALL refer to the "trs" .
Requirement 2.55	http://www.opengis.net/spec/movingfeatures/json/1.0/req/prism/tgeometry/interpolation/userdefined/equations/enclosed The value of "enclosed" member SHALL be a JSON array to represent an open (and closed) intervals of the "time" with two Boolean elements.

Chapter 8. Media Types

The MIME media type for MF-JSON text is "application/json" [\[RFC 4627\]](#), "application/json-seq" [\[RFC 7464\]](#), "application/geo+json" [\[RFC 7946\]](#), or "application/geo+json-seq" [\[RFC 8142\]](#). The default encoding is UTF-8.

Annex A: Conformance Class Abstract Test Suite (Normative)



Ensure that there is a conformance class for each requirements class and a test for each requirement (identified by requirement name and number)

A.1. Conformance Test Class: MF-JSON Trajectory Encoding

A.1.1. MF-JSON Trajectory file

A MF-JSON Trajectory file SHALL contain a LinearTrajectory object or a set of LinearTrajectory objects having a compliance with the IETF GeoJSON specification.

Test id:	conf/trajectory
Requirement:	req/trajectory/GeoJSON
Test purpose:	Verify that this requirement is satisfied.
Test method:	Inspect the document to verify the above.

A.1.2. LinearTrajectory object

A LinearTrajectory object SHALL be a GeoJSON Feature object that has two MANDATORY members of "geometry" and "properties". The value of the "geometry" member SHALL be a LineString Geometry object, having "type" = "LineString". The value of the "properties" member SHALL be a GeoJSON object that has at least a member with the named "datetimes". The value of the "datetimes" member is a JSON array. If a Feature has any variable attribute by time and its value, the attribute can be a member inside the "properties" member with the value of a JSON array, the size of its array SHALL be same as 1 or N-1 or N. Note that N is the number of elements in the array of the "coordinates" value. The number of elements in the array of the "coordinates" value in the Geometry object SHALL more than two positions.

Test id:	conf/trajectory/lineartrajectory
Requirement:	<ul style="list-style-type: none">• req/trajectory/lineartrajectory• req/trajectory/geometry• req/trajectory/propeties
Test purpose:	Verify that this requirement is satisfied.

Test method:	Inspect the document to verify the above.
---------------------	---

A.1.3. Datetimes

Each element in the array of the "datetimes" value SHALL be a instant object. An instant object is only a JSON string encoded by ISO 8601 field-based formats using Z or the number of milliseconds since midnight (00:00 a.m.) on January 1, 1970, in UTC. The array of the "datetimes" value SHALL be a monotonic increasing sequence. There is no instant object that has the same value as any other element.

Test id:	conf/trajectory/datetimes
Requirement:	<ul style="list-style-type: none"> • req/trajectory/datetimes • req/trajectory/datetimes/monotonic
Test purpose:	Verify that this requirement is satisfied.
Test method:	Inspect the document to verify the above.

A.1.4. Constraints

The number of elements in both arrays of the "coordinates" value and the "datetimes" value SHALL be equal.

Test id:	conf/trajectory/constraints
Requirement:	req/trajectory/constraints
Test purpose:	Verify that this requirement is satisfied.
Test method:	Inspect the document to verify the above.

A.2. Conformance Test Class: MF-JSON Prism Encoding

A.2.1. MF-JSON Prism file

A MF-JSON Prism file SHALL be compatible with the IETF GeoJSON Format. A JSON object in an MF-JSON Prism file SHALL be encoded by one of the Object types defined in this standard, i.e., TemporalGeometry, TemporalProperties, CoordinateReferenceSystem, MovingFeature, MovingFeatureCollection, LifeSpan, BoundingBox, Geometry, Properties, and MotionCurve.

Test id:	conf/prism
Requirement:	<ul style="list-style-type: none">• req/prism/GeoJSON• req/prism/object
Test purpose:	Verify that this requirement is satisfied.
Test method:	Inspect the document to verify the above.

A.2.2. Conflict

Two encodings of MF-JSON Prism and MF-JSON Trajectory SHALL be not able to coexist with each other in a GeoJSON Feature object.

Test id:	conf/prism/conflict
Requirement:	req/prism/conflict
Test purpose:	Verify that this requirement is satisfied.
Test method:	Inspect the document to verify the above.

A.2.3. TemporalGeometry object

A TemporalGeometry object SHALL have at least a mandatory member with the named "type" and its value SHALL be one of string "MovingPoint", "MovingLineString", "MovingPolygon", "MovingPointCloud", and "MovingGeometryCollection". The value of the "crs" and "trs" member SHALL be a JSON object to represent a CoordinateReferenceSystem (CRS) object. If no CRS instance can be so acquired from upper-level braces of a JSON object, the default spatial CRS (and temporal CRS) SHALL be applied.

Test id:	conf/prism/tgeometry
-----------------	----------------------

Requirement:	<ul style="list-style-type: none"> • req/prism/tgeometry • req/prism/tgeometry/crs
Test purpose:	Verify that this requirement is satisfied.
Test method:	Inspect the document to verify the above.

A.2.4. TemporalPrimitiveGeometry object

A TemporalPrimitiveGeometry object SHALL have at least three mandatory members of "type", "datetimes", and "coordinates". The value of the "type" member SHALL be one of string "MovingPoint", "MovingLineString", "MovingPolygon", and "MovingPointCloud". The value of the "datetimes" member SHALL be a JSON array of a sequence of monotonic increasing instants, having at least one element that is not null. The value of the "coordinates" member SHALL be a JSON array of a sequence of leaf geometries of a temporal geometry, having at least one element that is not null and its expression is depending on the "type". The value of the "interpolation" member SHALL be a MotionCurve object. The value of the "crs" and "trs" member SHALL be a JSON object to represent a CoordinateReferenceSystem (CRS) object. If the TemporalPrimitiveGeometry object has no member with the name "interpolation", the default MotionCurve object ("Linear") SHALL be applied. If no CRS instance can be so acquired from upper-level braces of a JSON object, the default spatial CRS (and temporal CRS) SHALL be applied. The number of elements in both arrays of the "coordinates" value and the "datetimes" value SHALL be equal. The number of elements in both arrays of the "orientations" value and the "datetimes" value SHALL be equal.

Test id:	conf/prism/tgeometry/primitive
Requirement:	<ul style="list-style-type: none"> • req/prism/tgeometry/primitive • req/prism/tgeometry/primitive/interpolation • req/prism/tgeometry/primitive/crs • req/prism/tgeometry/primitive/constraint
Test purpose:	Verify that this requirement is satisfied.
Test method:	Inspect the document to verify the above.

A.2.5. TemporalPrimitiveGeometry object type

A MovingPoint object SHALL have the value of the "type" = "MovingPoint" and the value of the "coordinates" member SHALL be a list of Point coordinates to construct a 0D leaf geometry (point) corresponding to each instant in order. A MovingLineString object SHALL have the value of the "type" = "MovingLineString" and the value of the "coordinates" member SHALL be a list of LineString coordinate arrays to construct a 1D leaf geometry (linestring) corresponding to each instant in order. A MovingPolygon object SHALL have the value of the "type" = "MovingPolygon" and the value of the "coordinates" member SHALL be a list of Polygon coordinate arrays to construct a

2D leaf geometry (polygon) corresponding to each instant in order. A MovingPointCloud object SHALL have the value of the "type" = "MovingPointCloud" and the value of the "coordinates" member SHALL be a list of MultiPoint coordinate arrays to construct a set of points as a leaf geometry corresponding to each instant in order.

Test id:	conf/prism/tgeometry/primitive/type
Requirement:	<ul style="list-style-type: none"> • req/prism/tgeometry/primitive/movingpoint • req/prism/tgeometry/primitive/movinglinestring • req/prism/tgeometry/primitive/movingpolygon • req/prism/tgeometry/primitive/movingpointcloud
Test purpose:	Verify that this requirement is satisfied.
Test method:	Inspect the document to verify the above.

A.2.6. 3D model

The 3D model SHALL be transformed into the fixed local coordinate reference system whose bound is -0.5 to 0.5 for each axis and unit is meter. The coordinate reference system for the 3D model SHALL be a right-handed coordinate system. The value of the "base" member SHALL be a JSON object having two members of "type" and "href". The "type" member has a JSON string to represent a file format, and the "href" member has a web accessible URL to address a 3D model data. If the TemporalPrimitiveGeometry object has a member with the name "base", the value of "type" in the TemporalPrimitiveGeometry object SHALL be a "MovingPoint". The value of the "orientations" member SHALL be a JSON array of a JSON object that has two members of "scales" and "angles". The value of the "scales" member SHALL be a JSON array value of numbers along the X, Y and Z axis in order. The value of the "angles" member SHALL be a JSON array value of numbers of Euler angles along the X, Y and Z axis in order. The value of the "angles" member SHALL define according to the right-hand rule and unit is degree. The number of the element in array of the "orientations" value SHALL be the same as "datetimes". The "orientations" member SHALL be accompanied with the "base" member.

Test id:	conf/prism/tgeometry/primitive/3dmodel
Requirement:	<ul style="list-style-type: none"> • req/prism/tgeometry/primitive/3dmodel • req/prism/tgeometry/primitive/base • req/prism/tgeometry/primitive/orientations • req/prism/tgeometry/primitive/orientations/scales • req/prism/tgeometry/primitive/orientations/angles
Test purpose:	Verify that this requirement is satisfied.

Test method:	Inspect the document to verify the above.
---------------------	---

A.2.7. TemporalComplexGeometry object

A TemporalComplexGeometry object SHALL have at least two mandatory members of "type" and "prisms". The value of the "type" member SHALL be a "MovingGeometryCollection" string. The value of the "prisms" member SHALL be a JSON array of a set of TemporalPrimitiveGeometry instances, having at least one element in the array. The value of the "crs" and "trs" member SHALL be a JSON object to represent a CoordinateReferenceSystem (CRS) object. If no CRS instance can be so acquired from upper-level braces of a JSON object, the default spatial CRS (and temporal CRS) SHALL be applied. A MovingGeometryCollection object SHALL the value of the "type" = "MovingGeometryCollection" and each element of "prisms" SHALL be an TemporalPrimitiveGeometry with one of types of "MovingPoint", "MovingLineString", "MovingPolygon", and "MovingPointCloud". The leaf geometry at a time position must be an instance of type "GeometryCollection" of GeoJSON, which is the union of each leaf of any temporal geometries at the same time.

Test id:	conf/prism/tgeometry/complex
Requirement:	<ul style="list-style-type: none"> • req/prism/tgeometry/complex • req/prism/tgeometry/complex/crs • req/prism/tgeometry/complex/movinggeometrycollection
Test purpose:	Verify that this requirement is satisfied.
Test method:	Inspect the document to verify the above.

A.2.8. TemporalProperties object

A TemporalProperties object SHALL be a JSON array of ParametricValues objects. A ParametricValues object SHALL have at least a mandatory member with the named "datetimes" and more than one member with the name @propertyN, where @propertyN is any string defined by an application as a temporal property.

Test id:	conf/prism/tproperties
Requirement:	<ul style="list-style-type: none"> • req/prism/tproperties • req/prism/tproperties/pvalues
Test purpose:	Verify that this requirement is satisfied.
Test method:	Inspect the document to verify the above.

A.2.9. @propertyN object

A *@propertyN* object SHALL have at least two mandatory members of "type" and "values". The value of the "type" member SHALL be one of string "Measure", "Text", and "Image". The value of the "values" member SHALL be a JSON array whose element is a string (including null, true and false) or numeric value. The value of the "interpolation" member SHALL be only one of string "Discrete", "Step", "Linear", and "Regression" or a URL indicating an InterpolationCode defined in OGC TimeseriesML 1.0 [OGC 15-042r3]. If the *@propertyN* object has no member with the name "interpolation", the default interpolation method ("Discrete") SHALL be applied. The value of the "form" member SHALL be a JSON string as a common code (3 characters) described in the list of Code List Rec 20 by the UN Centre for Trade Facilitation and Electronic Business (UN/CEFACT) or a URI denoting a unit-of-measure defined in a web resource. The number of elements in both arrays of the "datetimes" value and the "values" value SHALL be equal. The available interpolation method SHALL be restricted according to "type" value as follows.

- Measure: Discrete, Step, Linear, Regression
- Text: Discrete, Step
- Image: Discrete, Step

Test id:	conf/prism/tproperties/property
Requirement:	<ul style="list-style-type: none">• req/prism/tproperties/pvalues/property• req/prism/tproperties/pvalues/property/interpolation• req/prism/tproperties/pvalues/property/form• req/prism/tproperties/pvalues/property/constraint• req/prism/tproperties/pvalues/property/interpolation/constraint
Test purpose:	Verify that this requirement is satisfied.
Test method:	Inspect the document to verify the above.

A.2.10. CoordinateReferenceSystem object

A CoordinateReferenceSystem object SHALL have two mandatory members of "type" and "properties". The value of the "type" member SHALL be one of string "Name" and "Link". The value of the "properties" member SHALL be a JSON object with three optional members named "name", "href", and "type" whose value is a JSON string or JSON null value. A named CRS object SHALL have the value of the "type" = "Name" and the value of the "properties" member SHALL be a JSON object containing a "name" member whose value is a string identifying a coordinate reference system. A linked CRS object SHALL have the value of the "type" = "Link" and the value of the "properties" member SHALL be a JSON object containing a "href" member whose value is a dereferenceable URI.

Test id:	conf/prism/crs
-----------------	----------------

Requirement:	<ul style="list-style-type: none"> • req/prism/crs • req/prism/crs/named • req/prism/crs/linked
Test purpose:	Verify that this requirement is satisfied.
Test method:	Inspect the document to verify the above.

A.2.11. MovingFeature object

A MovingFeature object SHALL be a GeoJSON Feature object that have two mandatory members of "type" and "temporalGeometry". The value of the "type" member SHALL be a "Feature" string. The value of the "temporalGeometry" member SHALL be a JSON object to represent a TemporalGeometry object, not allowing the JSON null. The value of the "temporalProperties" member SHALL be a JSON array of TemporalProperties objects, allowing the JSON null. The value of the "crs" and "trs" member SHALL be a JSON object to represent a CoordinateReferenceSystem (CRS) object. The value of the "bbox" member SHALL be a JSON array to represent a BoundingBox object, allowing the JSON null. The value of the "time" member SHALL be a JSON array to represent a LifeSpan object, allowing the JSON null. The value of the "geometry" member SHALL be any JSON object or a JSON null value for the "geometry" member in a GeoJSON Feature object. The value of the "properties" member SHALL be any JSON object or a JSON null value for the "properties" member in a GeoJSON Feature object. If no CRS instance can be so acquired in a file, the default spatial CRS (and temporal CRS) SHALL be applied to the object.

Test id:	conf/prism/feature
Requirement:	<ul style="list-style-type: none"> • req/prism/feature • req/prism/feature/temporalProperties • req/prism/feature/crs • req/prism/feature/bbox • req/prism/feature/time • req/prism/feature/geometry • req/prism/feature/properties
Test purpose:	Verify that this requirement is satisfied.
Test method:	Inspect the document to verify the above.

A.2.12. MovingFeatureCollection object

A MovingFeatureCollection object SHALL be a GeoJSON FeatureCollection object that have two mandatory members of "type" and "features". The value of the "type" member SHALL be a

"FeatureCollection" string. The value of the "features" member SHALL be a JSON array whose element is a MovingFeature object. The value of the "crs" and "trs" member SHALL be a JSON object to represent a CoordinateReferenceSystem (CRS) object. The value of the "bbox" member SHALL be a JSON array to represent a BoundingBox object, allowing the JSON `null`. The value of the "time" member SHALL be a JSON array to represent a LifeSpan object, allowing the JSON `null`. The value of the "label" member SHALL be a JSON string to indicate an alias of the collection, allowing the JSON `null`. If no CRS instance can be so acquired in a file, the default spatial CRS (and temporal CRS) SHALL be applied to the object. The number of elements in an array of the "features" value SHALL be more than 1.

Test id:	conf/prism/featurecollection
Requirement:	<ul style="list-style-type: none"> • req/prism/featurecollection • req/prism/featurecollection/crs • req/prism/featurecollection/bbox • req/prism/featurecollection/time • req/prism/featurecollection/label • req/prism/featurecollection/constraints
Test purpose:	Verify that this requirement is satisfied.
Test method:	Inspect the document to verify the above.

A.2.13. LifeSpan object

A LifeSpan object SHALL be a JSON array to represent a particular period with two instants of t_s and t_e . Two elements of t_s , t_e SHALL be $t_s \leq t_e$. The expression of an element in the array SHALL refer to the default temporal CRS object (ISO8601).

Test id:	conf/prism/time
Requirement:	<ul style="list-style-type: none"> • req/prism/time • req/prism/time/element • req/prism/time/crs
Test purpose:	Verify that this requirement is satisfied.
Test method:	Inspect the document to verify the above.

A.2.14. BoundingBox object

A BoundingBox object SHALL be a JSON array of length $2 * n$ where n is the number of dimensions represented in the spatial bounding box. The elements in the array SHALL be two coordinates

(lower-bound coordinate and upper-bound coordinate). The order of values SHALL follow the axes order of single position of longitude, latitude, and elevation. The expression of an element in the array SHALL refer to the default spatial CRS object (WGS84).

Test id:	conf/prism/bbox
Requirement:	<ul style="list-style-type: none"> • req/prism/bbox • req/prism/bbox/element • req/prism/bbox/crs
Test purpose:	Verify that this requirement is satisfied.
Test method:	Inspect the document to verify the above.

A.2.15. MotionCurve object

The "**coordinate**" member in the TemporalPrimitiveGeometry object SHALL be interpolated by MotionCurve object. The value of the "**interpolation**" member in the TemporalPrimitiveGeometry object SHALL be one of string "**Discrete**", "**Step**", "**Linear**", "**Quadratic**", and "**Cubic**", or a URL to address a user-defined parametric curve. A user-defined parametric curve SHALL be a JSON object which has three members of "**crs**", "**trs**", and "**equations**". The value of the "**equations**" member SHALL be not empty and be a JSON array whose element is a user-defined parametric segment having two members of "**coefficients**" and "**time**". The value of "**coefficients**" member SHALL be a JSON array whose element is an array of coefficients of the interpolation formula of a temporal position. The order of elements in array of the "**coefficients**" value for the interpolation formula of a temporal position SHALL follow the order of the spatial CRS. The value of "**time**" member SHALL be a JSON array to represent a particular period with two instants of t_s and t_e . The expression of an element in the "**time**" value SHALL refer to the "**trs**". The value of "**enclosed**" member SHALL be a JSON array to represent an open (and closed) intervals of the "**time**" with two Boolean elements.

Test id:	conf/prism/tgeometry/interpolation
Requirement:	<ul style="list-style-type: none"> • req/prism/tgeometry/interpolation • req/prism/tgeometry/interpolation/userdefined • req/prism/tgeometry/interpolation/userdefined/equations/coefficients • req/prism/tgeometry/interpolation/userdefined/equations/time • req/prism/tgeometry/interpolation/userdefined/equations/enclosed
Test purpose:	Verify that this requirement is satisfied.
Test method:	Inspect the document to verify the above.

Annex B: Sample Data of Moving Feature (Informative)

B.1. Sample Data of OGC Moving Features XML and CSV encoding

Figure 10 shows an example for linear trajectories of two moving points A and B. Each trajectory has the start time and the end time. Both points start to move at t=10sec and end at t=19sec. While the movement of B does not change, the movement of A is changed at t=15sec. Example 15 and Example 16 shows how to encode the trajectories shown in Figure 10 by using two OGC Moving Features encoding standards. However, XML and CSV encoding standards have limitation on the representation of prism geometries such as **MF_PrismGeometry** and **MF_RigidTemporalGeometry**. They describe the implementation of only a type of **MF_TemporalTrajectory** with linear interpolation of movement. This MF-JSON standard specifies how to encode the same information by using GeoJSON; Additionally, it defines new JSON members to support applications that need more power expression of the movement of a feature whose geometry may be 0D, 1D, 2D, 3D geometric primitives, or their aggregations and continuously changes over time with various temporal interpolation methods.

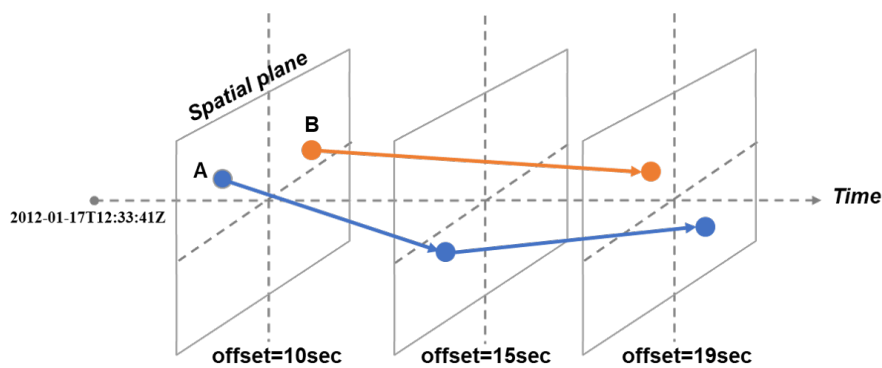


Figure 10. Example for two linear trajectories

Example 15: OGC Moving Features XML Core encoding of linear trajectories.

```
<?xml version="1.0" encoding="UTF-8"?>
<mf:MovingFeatures>
  <mf:sTBoundedBy offset="sec">
    <gml:EnvelopeWithTimePeriod srsName="urn:ogc:def:crs:EPSG:6.6:4326">
      <gml:lowerCorner>50.23 9.23</gml:lowerCorner>
      <gml:upperCorner>50.31 9.27</gml:upperCorner>
      <gml:beginPosition>2012-01-17T12:33:41Z</gml:beginPosition>
      <gml:endPosition>2012-01-17T12:37:00Z </gml:endPosition>
    </gml:EnvelopeWithTimePeriod>
  </mf:sTBoundedBy>
  .....
  <mf:foliation>
    <mf:LinearTrajectory gml:id="LT0001" mfIdRef="A" start="10" end="15">
      <gml:posList>11.0 2.0 12.0 3.0</gml:posList>
      <mf:Attr>walking,1</mf:Attr>
    </mf:LinearTrajectory>
    <mf:LinearTrajectory gml:id="LT0002" mfIdRef="B" start="10" end="19">
      <gml:posList>10.0 2.0 11.0 3.0</gml:posList>
      <mf:Attr>walking,2</mf:Attr>
    </mf:LinearTrajectory>
    <mf:LinearTrajectory gml:id="LT0003" mfIdRef="A" start="15" end="19">
      <gml:posList>12.0 3.0 10.0 3.0</gml:posList>
      <mf:Attr>walking,2</mf:Attr>
    </mf:LinearTrajectory>
  </mf:foliation>
</mf:MovingFeatures>
```

Example 16: OGC Moving Features CSV encoding of linear trajectories.

```
@stboundedby,urn:x-ogc:def:crs:EPSG:6.6:4326,2D,50.23 9.23,50.31 9.27,2012-01-17T12:33:41Z,2012-01-17T12:37:00Z,sec
@columns,mfidref,trajectory,state,xsd:token, type code ,xsd:integer
A,10,15,11.0 2.0 12.0 3.0,walking,1
B,10,19,10.0 2.0 11.0 3.0,walking,2
A,15,19,12.0 3.0 10.0 3.0,walking,2
```


B.2. Sample Data of OGC Moving Features JSON

Trajectory encoding

[Example 17](#) illustrates an example of the MF-JSON Trajectory encoding corresponding to two moving points, as shown in [Figure 10](#) to be replaceable with [Example 15](#) and [Example 16](#).

Example 17: Example of the MF-JSON Trajectory encoding

```
{
  "type": "FeatureCollection",
  "features": [
    {
      "type": "Feature",
      "id": "A",
      "geometry": {
        "type": "LineString",
        "coordinates": [[11.0,2.0], [12.0,3.0], [10.0,3.0]]
      },
      "properties": {
        "datetimes": ["2012-01-17T12:33:51Z", "2012-01-17T12:33:56Z", "2012-01-17T12:34:00Z"],
        "state": ["walking", "walking"],
        "typecode": [1, 2]
      }
    },
    {
      "type": "Feature",
      "id": "B",
      "geometry": {
        "type": "LineString",
        "coordinates": [[10.0,2.0], [11.0,3.0]]
      },
      "properties": {
        "datetimes": ["2012-01-17T12:33:51Z", "2012-01-17T12:34:00Z"],
        "state": ["walking"],
        "typecode": [2]
      }
    }
  ]
}
```

B.3. Sample Data of OGC Moving Features JSON Prism encoding

Figure 11 shows the movement of a car which has a rigid 3D solid, and Example 18 is an example of moving-feature instances encoded by the MF-JSON Prism format.

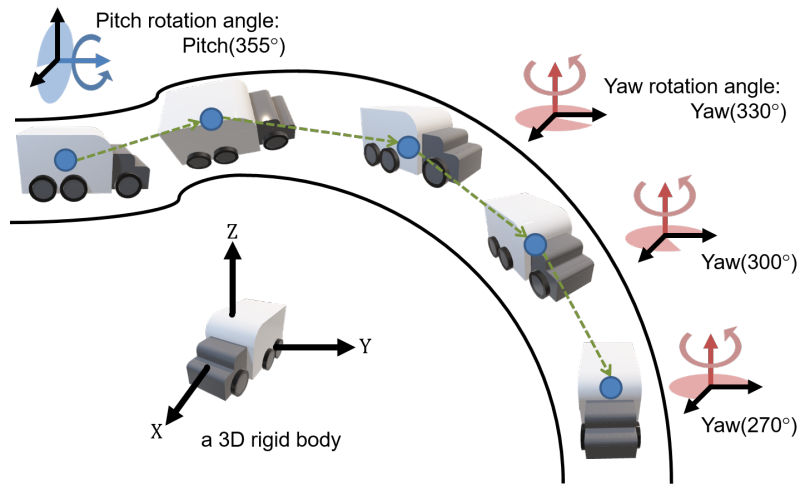


Figure 11. Example of the movement of a car which has a rigid 3D solid

Example 18: Example of the MF-JSON Prism encoding

```
{
  "type": "Feature", //(MANDATORY)
  "crs" : { //(DEFAULT) If there is no "crs" member, the default crs is "WGS84" with longitude and latitude units of
    decimal degrees
    "type": "Name",
    "properties": {"name": "urn:ogc:def:crs:OGC:1.3:CRS84"}
  },
  "trs" : { //(DEFAULT) If there is no "trs", the default trs is "Gregorian".
    "type": "Link",
    "properties": {
      "type": "OGCDEF",
      "href": "http://www.opengis.net/def/uom/ISO-8601/0/Gregorian",
    }
  },
  "temporalGeometry": { //(MANDATORY) one parameter set of geometries of a moving feature
    "type": "MovingPoint",
    "datetimes": [ "2011-07-14T22:01:01Z", "2011-07-14T22:01:02Z", "2011-07-14T22:01:03Z", "2011-07-14T22:01:04Z",
      "2011-07-14T22:01:05Z"],
    "coordinates": [ [139.757083, 35.627701, 0.5], [139.757399, 35.627701, 2.0], [139.757555, 35.627688, 4.0],
      [139.757651, 35.627596, 4.0], [139.757716, 35.627483, 4.0] ],
    "interpolation": "Linear",
    "base": {
      "type": "glTF",
      "href": "http://www.opengis.net/spec/movingfeatures/json/1.0/prism/example/car3dmodel.gltf"
    },
    "orientations": [
      {"scales": [1,1,1], "angles": [0,0,0]},
      {"scales": [1,1,1], "angles": [0,355,0]},
      {"scales": [1,1,1], "angles": [0,0,330]},
      {"scales": [1,1,1], "angles": [0,0,300]},
      {"scales": [1,1,1], "angles": [0,0,270]}
    ]
  },
  "temporalProperties": [ //(OPTIONAL) dynamic non-spatial attributes, extended from 'properties'
    { // a group of temporal properties that are measured at the same times
      "datetimes": [ "2011-07-14T22:01:01.450Z", "2011-07-14T23:01:01.450Z", "2011-07-15T00:01:01.450Z"],
    }
  ]
}
```

```

    "length": {
      "type": "Measure",
      "form": "http://www.qudt.org/qudt/owl/1.0.0/quantity/Length", // a URI denoting a unit-of-measure
      "values": [1.0, 2.4, 1.0], // the array of values for "length", with the same number of elements as
"datetimes"
      "interpolation": "Linear",
      "description": "description1" //(OPTIONAL)
    },
    "discharge" : {
      "type" : "Measure",
      "form" : "MQS", // a symbol for m^3/s(a cubic metre per second)
      "values" : [3.0, 4.0, 5.0],
      "interpolation": "Step"
    }
  },{
    "datetimes" : [1465621816590, 1465711526300],
    "camera" : {
      "type" : "Image",
      "values" : ["http://www.opengis.net/spec/movingfeatures/json/1.0/prism/example/image1",
"iVBORw0KGgoAAAANSUHE....."],
      "interpolation": "Discrete"
    },
    "labels":{
      "type": "Text", // a predefined unit for a string value
      "values": ["car", "human"], // the array of values for "labels", with the same number of elements as
"datetimes"
      "interpolation": "Discrete",
      "description": "description2" //(OPTIONAL)
    }
  }
],
"geometry": { //(OPTIONAL) GeoJSON IETF RFC 7946 geometry object
  "type": "LineString",
  "coordinates": [ [139.757083, 35.627701, 0.5], [139.757399, 35.627701, 2.0], [139.757555, 35.627688, 4.0],
[139.757651, 35.627596, 4.0], [139.757716, 35.627483, 4.0] ]
},
"properties": { //(OPTIONAL)
  "name": "car1",
  "state": "test1",
  "video": "http://www.opengis.net/spec/movingfeatures/json/1.0/prism/example/video.mpeg",
  "description": "Example of the MF-JSON Prism encoding"
},
"bbox": [139.757083, 35.627483, 0.0, 139.757716, 35.627701, 4.5], //(OPTIONAL) 2*n array with the lowest values for
all axes followed by the highest values
"time": ["2011-07-14T22:01:01Z", "2011-07-15T01:11:22Z"], //(OPTIONAL) min and max time instant to cover all temporal
positions in ``temporalGeometry`` and ``temporalProperties``
"id" : "A"
}

```

Annex C: Deformation of Rigid Body (Informative)

If an application defines a wrong interpolation of motion, it brings unintended consequences. For example, the 2D triangle with a rigid body in [Figure 12](#) has translational and rotational motion at the same time. It means when the application exports data with MF-JSON Prism, it needs to define the `"base"` and `"orientations"` member as well as `"interpolation"` for a temporal geometry. If the application exports data without `"base"` and `"orientations"` value like [Example 19](#), the intermediate positions of the moving feature are interpolated with the linear interpolation function, as shown in [Figure 12](#). Even though the object has a rigid body, its boundary is deformed. [Figure 13](#) also shows the same phenomena that even the 2D polygon becomes 0D point in the middle time.

Example 19: Example of a wrong encoding

```
{
  "type": "Feature", //(MANDATORY)
  "temporalGeometry": { //(MANDATORY) one parameter set of geometries of a moving feature
    "type": "MovingPolygon",
    "datetimes": ["2011-07-14T22:01:01Z", "2011-07-14T22:01:05Z"],
    "coordinates": [
      [[
        139.77431058883667,
        35.621948192128826
      ],
      [
        139.77530837059018,
        35.62197435581956
      ],
      [
        139.7748255729675,
        35.62327380835506
      ],
      [
        139.77431058883667,
        35.621948192128826
      ]
    ]],
      [[
        139.77635979652405,
        35.62244966137756
      ],
      [
        139.77626860141754,
        35.621734521667385
      ],
      [
        139.77767407894135,
        35.62201796195175
      ],
      [
        139.77635979652405,
        35.62244966137756
      ]
    ]
    ],
    "interpolation": "Linear"
  },
  ...
}
```

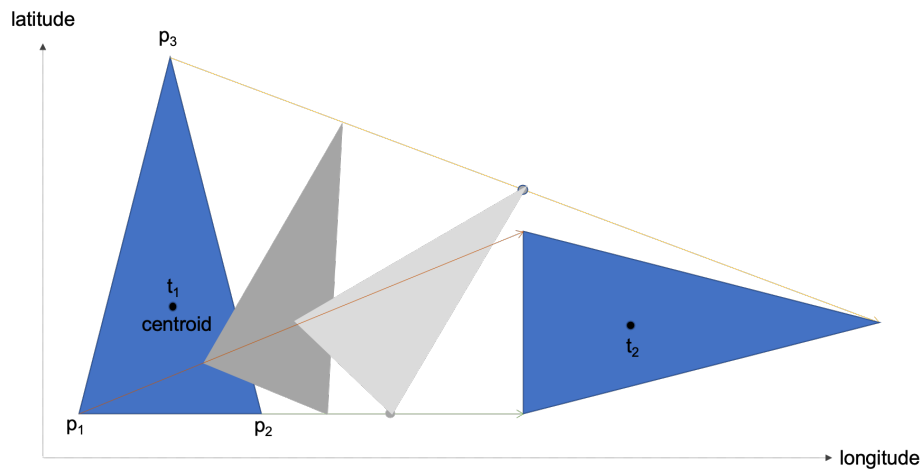


Figure 12. Example of deformation of rigid body with linear interpolation

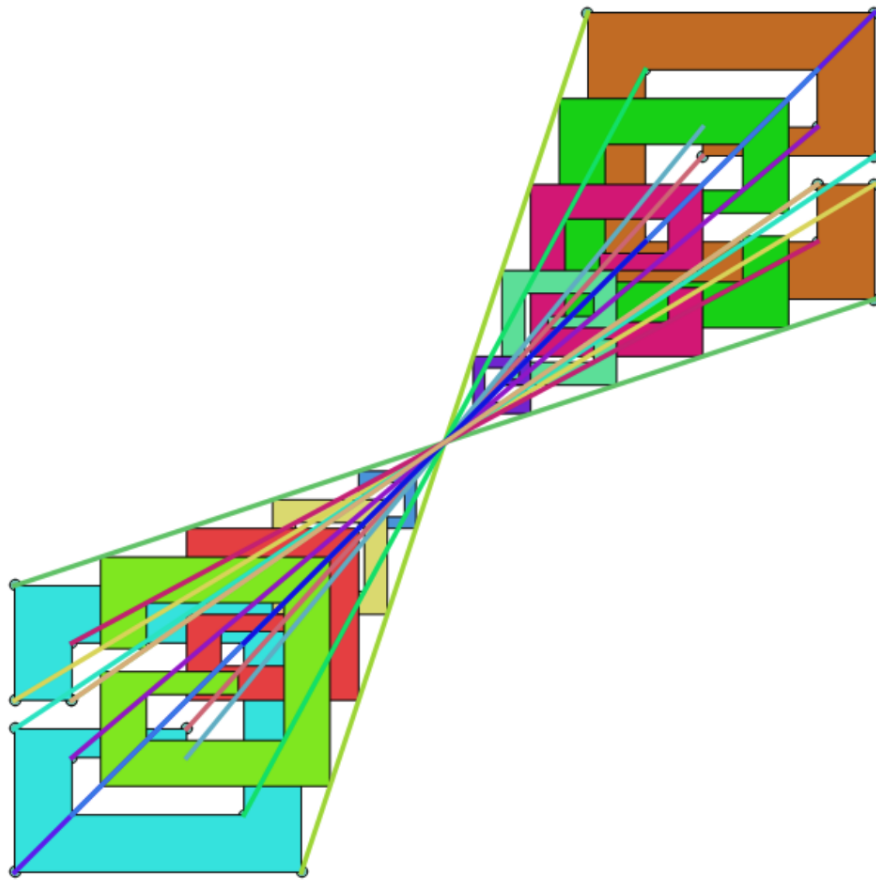


Figure 13. Example of dimensional deformation of a 2D MovingPolygon feature with linear interpolation

Annex D: Revision History

Date	Release	Editor	Primary clauses modified	Description
2018-12-01	1.0	Kyoung-Sook Kim	all	initial version
2018-01-25	1.0	Kyoung-Sook Kim, Taehoon Kim	all	first draft
2019-04-21	1.0	Kyoung-Sook Kim	trajectory	revision
2019-06-20	1.0	Kyoung-Sook Kim	prism	revision
2019-07-27	1.0	Kyoung-Sook Kim, Taehoon Kim	requirement, conformance	addition
2019-12-20	1.0	Kyoung-Sook Kim	revised by public comments	revision

Annex E: Bibliography

- [1] National Imagery and Mapping Agency, "Department of Defense World Geodetic System 1984: Its Definition and Relationships with Local Geodetic Systems", Third Edition, 1984.
- [2] Rec 20 – Codes for Units of Measure Used in International Trade, UN Centre for Trade Facilitation and Electronic Business (UN/CEFACT), <https://www.unece.org/unecefact/codelistrecs.html>
- [3] EPSG Geodetic Parameter Dataset, <http://www.epsg.org>
- [4] The GeoJSON Format Specification. 2008, <https://geojson.org>
- [5] Rick Parent, Chapter 3 - Interpolating Values, Editor(s): Rick Parent, Computer Animation (Third Edition), Morgan Kaufmann, 2012, Pages 61-109, <https://doi.org/10.1016/B978-0-12-415842-9.00003-4>.
- [6] Standard Triangulated Language (STL), http://www.fabbers.com/tech/STL_Format
- [7] Wavefront OBJ, https://en.wikipedia.org/wiki/Wavefront_.obj_file
- [8] PLY - Polygon File Format, <http://paulbourke.net/dataformats/ply/>
- [9] glTF™ (GL Transmission Format), <https://github.com/KhronosGroup/glTF>