OGC® DOCUMENT: 23-042

External identifier of this OGC® document: http://www.opengis.net/doc/PER/T19-D001

OGC TESTBED-19 — NON-TERRESTRIAL GEOSPATIAL ENGINEERING REPORT

ENGINEERING REPORT

PUBLISHED

Submission Date: 2024-01-29
Approval Date: 2024-03-28
Publication Date: 2024-07-05
Editor: Sina Taghavikish, Rob Smith, Martin Desruisseaux, Jérôme Jacovella-St-Louis, Carl Stephen Smyth, Charles Heazel

CONTENTS

LIST OF TABLES

LIST OF FIGURES

I EXECUTIVE SUMMARY

Testbed-18 explored the potential use of OGC Standards for non-terrestrial applications and was scoped as a paper study. Validation of the Testbed-18 recommendations has been left for Testbed-19. This OGC Engineering Report (ER) documents recommended changes to OGC Standards and the implementation experience to justify those changes.

The use of OGC Standards include geospatial applications for non-Earth planets as well as interplanetary spatiotemporal applications. Two Standards emerged as key: ISO 19111 (OGC Abstract Specification 2: Referencing by coordinates) and OGC 21-056r11 (OGC GeoPose 1.0 Data Exchange Standard). Extensions to ISO 19111 were identified which would support the representation of non-terrestrial planetary spatial reference systems as well as interplanetary spatiotemporal reference systems.

The GeoPose Standard (GeoPose) was explored as a mechanism to integrate the large number of reference systems and transformations needed to model the geometry of interplanetary spacetime.

In the context of the Double Asteroid Redirection Test (DART) scenario, positions and orientations in different coordinate reference systems and associated attributes such as velocities of non-terrestrial objects were encoded using two different approaches: as sequences of extended GeoPoses, and as OGC Moving Features JSON (MF-JSON). These encoded data were then used as the basis for a 3D visualization demonstration.

This work is not intended to replace the existing standards already used in astronomy such as the World Coordinate System (WCS). The recommendations provided in this ER are rather intended to improve interoperability by specifying how to export a subset of a WCS description as OGC/ISO data structures for consumption by GIS software or other geospatial technology applications.

Testbed-18 also investigated how GeoPose could be integrated with mobile location-aware devices such as smartphones. Engineering Report OGC 22-016r3 (Testbed-18: Moving Features) concluded that GeoPose could enrich data with location and orientation information synchronized to video and other sensors and identified two suitable road network use cases for study using WebVMT in Testbed-19.

II KEYWORDS

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

ogcdoc, ogc, referencing, coordinate operation, space, extraterrestrial, relativity, Minkowski, spacetime, ISO 19111, GeoPose, testbed, DART, OS Bubble

III

CONTRIBUTORS

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

NAME	ORGANIZATION	ROLE
Rob Smith	Away Team	Contributor
Sina Taghavikish	Open Geospatial Consortium	Editor
Charles Heazel	HeazelTech	Contributor
Jérôme Jacovella-St-Louis	Ecere	Contributor
Carl Stephen Smyth	Open Site Plan	Contributor
Martin Desruisseaux	Geomatys	Contributor

1

INTRODUCTION

1 INTRODUCTION

OGC Testbed-18 evaluated the potential use of existing OGC/ISO Standards for applications on celestial bodies other than planet Earth. These applications include data for other planetary bodies, for objects in orbit around those planetary bodies, and for objects in free flight within our solar system. Testbed-18's research on non-terrestrial bodies and standards was largely a paper exercise. This research included an evaluation of the published standards to determine what should be possible. Validation of the conclusions through prototype implementations was left for Testbed-19. This Engineering Report (ER) documents the results of those Testbed-19 prototype implementation efforts.

The following five observations were central to the organization of the Testbed-19 effort.

1. Most of the changes required to support non-terrestrial missions are in the coordinate reference system definitions (part of ISO 19111).
2. The coordinate transformation framework of ISO 19111 is applicable with no change. However, its GML encoding is not widely supported in software.
3. GeoPose can be an alternative (not necessarily easier) encoding for associating non-terrestrial coordinate reference systems and coordinate transforms.
4. The 4D spacetime coordinate systems required for interplanetary space are also useful for terrestrial applications. Examples include satellite communications and solar radiation storm forecasts.
5. Further work on Moving Features and Sensor Integration will play an essential role in enabling non-terrestrial spatial infrastructures.

Agreeing on these five observations allowed Testbed-19 work to proceed in three threads, each of them described in its own chapter. The first thread focused on coordinate reference systems and coordinate transformations. The second thread focused on the use of 4D GeoPose (NeoPose) for highly dynamic non-terrestrial and terrestrial applications. The third thread advanced the work on Moving Features and Sensor Integration with an eye toward incorporation of the work from the first two threads.

1.1. Thread One — ISO 19111+

The first chapter of this ER addresses extensions to ISO 19111 (ISO 19111+). These extensions will allow ISO 19111 to describe both non-terrestrial planetary coordinate reference systems and non-planetary coordinate reference systems in free space. These extensions are documented using UML diagrams, XSD schemas, and Java/Python interfaces which were implemented in Open Source software. Links to source codes on GitHub are given in Annex B (XML) and Annex C (Java).

1.2. Thread Two — NeoPose

The second chapter of this ER describes adaptations of GeoPose (NeoPose) for non-terrestrial applications. Once off the surface of the Earth, there is no single overarching coordinate reference system. A measurement may be expressed in the local (proper) reference system or in that of an observing entity. The value of a measurement may be expressed in an unlimited number of reference systems which calls for a framework to support coordinate transformations between arbitrary coordinate reference systems. ISO 19111 is one such framework, but it is constrained by the above-cited lack of GML support in popular software. With a few modifications, GeoPose would also be well suited for that purpose. This chapter explores the specific modifications to GeoPose needed to create a NeoPose Standard.

1.3. Prototyping ISO 19111+ and NeoPose

The third and fourth chapters describe the two prototype environments developed for Testbed-19 as well as the performance of the ISO 19111+ and NeoPose products in those environments.

Chapter three describes exercises based on the NASA Double Asteroid Redirection Test (DART) mission. DART was a NASA mission which sought to deflect a moonlet in orbit around an asteroid. This scenario is interesting in that it involves high positional accuracy for multiple moving bodies, moving at high velocities, in deep space. It is a good proxy for a deep space tracking and rendezvous mission.

Chapter four describes exercises using the Hillyfields Bubble. The Hillyfields Bubble is a multipurpose dataset captured in support of the development of Augmented Reality/Metaverse architectures and infrastructure prototypes. These data were collected over the UK Ordnance Survey Headquarters in Southampton, UK on the 25th and 26th of April, 2023. The large number of moving objects involved, as well as the number and variety of moving sensors, makes this a good proxy for non-terrestrial planetary missions.

1.4. Thread Three — Moving Features and Sensor Integration

Chapter five returns to the Earth and addresses issues involving data synchronization from multiple sources. This chapter advances work on Testbed-18 Moving Features and Sensor Integration through prototypes and implementations of a set of Road Network use cases. The geotagged video use cases were selected from those documented in the Testbed-18 Moving Features Engineering Report (OGC 22-016r3) and derived from discussions with UK national road network authorities. Concurrent video data were captured with GeoPose by multiple cameras in a collaborative effort hosted by Ordnance Survey in April 2023. The captured data were encoded based on the OGC GeoPose Standard. These data included many moving objects observed by static and moving devices to study how geotagged video and sensor data could be enriched by synchronization and aggregation over time.

2

TERMS AND DEFINITIONS

2 TERMS AND DEFINITIONS

This document uses the terms defined in OGC Policy Directive 49, 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 document and OGC documents do not use the equivalent phrases in the ISO/IEC Directives, Part 2.

This document also uses terms defined in the OGC Standard for Modular specifications (OGC 08-131r3), also known as the ‘ModSpec’. The definitions of terms such as standard, specification, requirement, and conformance test are provided in the ModSpec.

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

Some of the concepts encountered in this Engineering Report are not commonly encountered in the Geospatial field. Others, while they may look familiar, take on a slightly different meaning. Those terms are defined in this section. For more complex cases, a more detailed description in provided in Annex E. The definitions in this section provide links to associated descriptions where they are appropriate.

2.1. Barycenter

the center of mass of two or more bodies that orbit one another

NOTE A barycenter is a dynamical point, not a physical object.

[**SOURCE:** Wikipedia]

2.2. Coordinate

one of a sequence of numbers designating the *position* (Clause 2.16) of a *point*

NOTE In a spatial *coordinate reference system* (Clause 2.3), the *coordinate* numbers are qualified by units.

NOTE See also Clause 2.5.

[**SOURCE:** ISO 19111]

2.3. Coordinate Reference System

coordinate system that is related to an *object* (Clause 2.15) by a *datum* (Clause 2.6)

NOTE For geodetic and vertical *datums* (Clause 2.6), the *object* (Clause 2.15) will be a celestial body such as Earth.

[**SOURCE:** ISO 19111]

2.4. Coordinate System

set of mathematical rules for specifying how *coordinates* (Clause 2.2) are to be assigned to *points*

[**SOURCE:** ISO 19111]

2.5. Coordinate tuple

tuple composed of coordinates (Clause 2.2)

NOTE The number of coordinates in the coordinate tuple equals the dimension of the coordinate system; the order of coordinates in the coordinate tuple is identical to the order of the axes of the coordinate system.

[**SOURCE:** ISO 19111]

2.6. Datum

parameter or set of parameters that realize the *position* (Clause 2.16) of the origin, the scale, and the orientation of a *coordinate system* (Clause 2.4)

[**SOURCE:** ISO 19111]

2.7. Domain

well-defined set

NOTE	*Domains* are used to define the *domain* set and range set of *attributes*, operators, and functions.

[**SOURCE:** ISO 19109, Clause 4.8]

2.8. Domain <general vocabulary>

distinct area of human knowledge to which a terminological entry is assigned

NOTE	Within a database or other terminology collection, a set of *domains* (Clause 2.7) will generally be defined. More than one *domain* (Clause 2.7) can be associated with a given *concept*.

[**SOURCE:** ISO 19104, Clause 4.11]

2.9. Geodesic

a geodesic is a curve representing in some sense the shortest path between two points in a surface

NOTE	It is a generalization of the notion of a “straight line”.

[**SOURCE:** Wikipedia]

2.10. Global Reference System

Coordinate Reference System of the trajectory of the reference point

NOTE	See also Annex E.4.

[**SOURCE:** ISO 19141]

2.11. Grid

covering of a multi-dimensional region using quadrilateral shapes (in the 2D case) or their *n*-dimensional generation (in the nD case) with no overlaps and gaps

[**SOURCE:** ISO 19123]

2.12. Inertial Reference System

frame of reference where a body that is subject to no net force moves with constant velocity in a straight line, at least locally

SOURCE	Matthew Baring, “Lecture Notes for PHYS 532”, Rice University, Spring 2023.
NOTE	The “body” may be the barycenter (Clause 2.1) of a group of bodies.
NOTE	See also Annex E.3.

2.13. Local Reference System

local engineering coordinate system established using the object reference point as its origin

NOTE	See also Annex E.4.

[**SOURCE:** ISO 19141]

2.14. Location

particular *place* or *position* (Clause 2.16)

NOTE	*Locations* are physical *places*, typically on the surface of the Earth, although *locations* can be relative to other, non-earth centric coordinate reference systems.
NOTE	*Locations* can be a single *point*, a centroid, a minimum bounding rectangle, or a set of vectors.

NOTE A *location* should be persistent over time and does not change.

[**SOURCE:** ISO 19112, Clause 3.1.3]

2.15. Object

entity with a well defined boundary and identity that encapsulates state and behavior

NOTE This term was first used in this way in the general theory of object oriented programming, and later adopted for use in this same sense in UML. An *object* is an instance of a *class*. *Attributes* and relationships represent state. *Operations*, methods, and state machines represent behavior.

[**SOURCE:** ISO 19103]

2.16. Position

data type that describes a *point* or *geometry* potentially occupied by an *object* (Clause 2.15) or person

NOTE A *direct position* is a semantic subtype of *position*. *Direct positions* as described can only define a *point*, and therefore not all *positions* can be represented by a *direct position*. That is consistent with the "is type of" relation. An ISO 19107 geometry is also a *position*, but not a *direct position*.

[**SOURCE:** ISO 19133]

2.17. Reification

process by which an abstract idea about a computer program is turned into an explicit data model or other object created in a programming language

[**SOURCE:** Wikipedia]

2.18. Reference Frame

parameter or set of parameters that realize the position of the origin, the scale, and the orientation of a coordinate system

[**SOURCE:** ISO 19111]

2.19. Series expansion

technique that expresses a function as an infinite sum, or series, of simpler functions

NOTE	It is a method for calculating a function that cannot be expressed by just elementary operators (addition, subtraction, multiplication, or division).
NOTE	The resulting so-called series can often be limited to a finite number of terms, thus yielding an approximation of the function.

[**SOURCE:** Wikipedia]

2.20. Serialization

process of translating a data structure or object state into a format that can be stored (e.g., files in secondary storage devices, data buffers in primary storage devices) or transmitted (e.g., data streams over computer networks) and reconstructed later (possibly in a different computer environment)

[**SOURCE:** Wikipedia]

2.21. World Reference System

Spatial Coordinate Reference System object which relates the positions of features in the real world.

NOTE	See also Annex E.4.

[**SOURCE:** OGC 19-045r3]

3

NON-TERRESTRIAL REFERENCE SYSTEMS

3 NON-TERRESTRIAL REFERENCE SYSTEMS

A Coordinate Reference System (CRS) is a framework that uses coordinates to precisely measure locations in space and time. The ISO 19111 Standard, also available as OGC Topic 2, provides an abstract model for this framework, together with a framework for coordinate conversions and transformations. While this model was initially designed for coordinates on the surface of the Earth, it also defines structures, such as `EngineeringCRS`, that can support locations which are not defined relative to the Earth. Those structures are already capable of providing *some* CRS definitions for use in free space. The Testbed-18: Reference Frame Transformation Engineering Report (OGC 22-038r2) proposed extensions to those structures for more complete and accurate definitions of non-terrestrial Coordinate Reference Systems.

ISO 19111 is an abstract model defined in the Unified Modeling Language (UML). It is not intended to be implemented directly. Rather, Implementation Standards are generated from the abstract model. These Implementation Standards provide a machine-readable representation of the abstract model, tailored for a specific implementation domain. These “implementations” are similar to, but not the same as the “implementations” concept from Model Driven Architecture (MDA). To distinguish them, this Engineering Report uses the reification term (Clause 2.17). Some reifications cover only a subset of ISO 19111 for simplicity reasons.

Table 1 — Reifications of ISO 19111 abstract model

STANDARD OR PRODUCT	TYPE	REIFICATION OF	COVER ALL THE UML?
EPSG geodetic dataset	Dataset	ISO 19111:2019	No
ISO 19162:2019 — WKT 2	Encoding	ISO 19111:2019	No
ISO 19136:2007 — GML	Encoding	ISO 19111:2007	Yes
OGC 09-083r4 — GeoAPI 3.0	API	ISO 19111:2007	Yes
PROJ-JSON (community)	Encoding	ISO 19111:2019	To be verified

The following software packages are examples of implementations of above-cited reifications.

Table 2 — Some known implementations of reifications of ISO 19111 abstract model

EXTERNAL PROJECT	IMPLEMENTATION OF	LICENSE	SOURCE CODE
Apache SIS	OGC 09-083 — GeoAPI 3.0.2	Apache 2	GitHub
PROJ-JNI	OGC 09-083 — GeoAPI 3.0.2	MIT	GitHub

OGC Testbed-18 participants evaluated ISO 19111 for use on non-Earth planets and in free space. The results of that analysis were documented in OGC 22-038r2. This Testbed-19 Engineering Report (ER) proposes extensions to the following Standards with programmatic definitions of concepts introduced in Testbed-18.

- ISO 19111, extended with UML diagrams as presented in this section.
- ISO 19136, extended with XSD schemas as presented in Annex B.1.
- GeoAPI 3.0, extended with Java and Python interfaces as presented in Annex C.

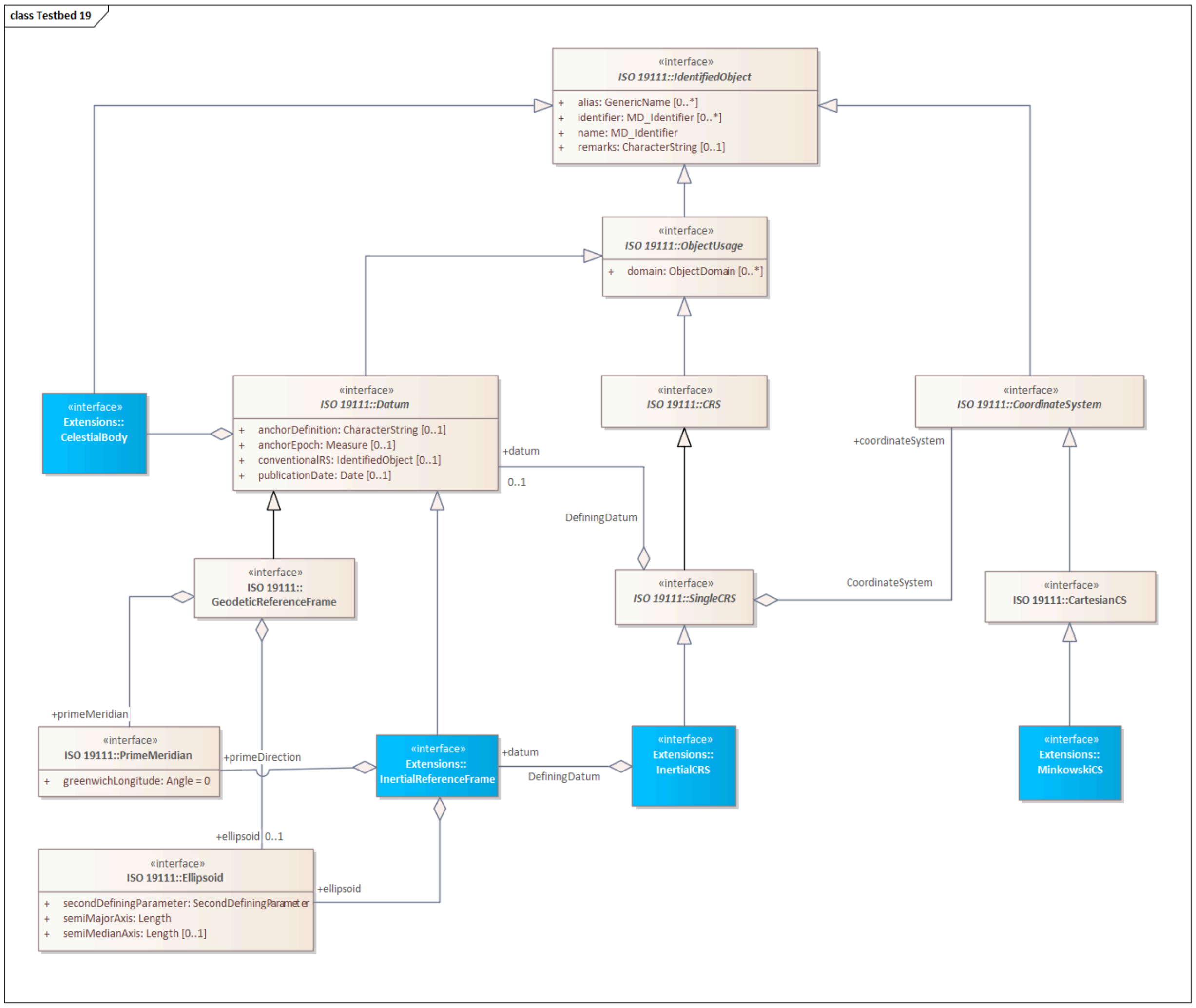
The extensions to each of those standards should be assigned to an OGC working group for discussion or standardization as follows.

- The CRS Standards Working Group (SWG) can evaluate the UML diagrams for eventual incorporation in OGC Topic 2 and ISO 19111.
- The Planetary Domain Working Group (DWG) can evaluate the XML schema for eventual publication on http://schemas.opengis.net/gsp/.
- The GeoAPI SWG can evaluate the new interfaces for eventual incorporation in the GeoAPI main module.

As part of the Testbed-19 work, these extensions were further validated through implementation in a branch of the Apache SIS implementation of the OGC GeoAPI 3.0.2 reification.

3.1. Abstract model overview

The ISO 19111 abstract model contains two main parts: Classes for the definition of coordinate reference systems (`CRS` and related classes such as `Datum`), and classes for the definition of coordinate operations (`CoordinateOperation` and related classes such as parameters). The latter part will be discussed in the Coordinate transformations section of this ER. The following section concerns the first part, the Coordinate Reference System (CRS). The following UML represents some ISO 19111 classes relevant to this section, together with proposed extensions. The tan boxes are classes defined in ISO 19111, while the blue boxes are proposed extensions.



class Testbed 19
«interface» ISO 19111::IdentifiedObject
+ alias: GenericName [0..*]
+ identifier: MD_Identifier [0..*]
+ name: MD_Identifier
+ remarks: CharacterString [0..1]
«interface» ISO 19111::ObjectUsage
+ domain: ObjectDomain [0..*]
«interface» Extensions:: CelestialBody
«interface» ISO 19111::Datum
+ anchorDefinition: CharacterString [0..1]
+ anchorEpoch: Measure [0..1]
+ conventionalRS: IdentifiedObject [0..1]
+ publicationDate: Date [0..1]
«interface» ISO 19111::CRS
«interface» ISO 19111::CoordinateSystem
+coordinateSystem
+datum
0..1
DefiningDatum
«interface» ISO 19111:: GeodeticReferenceFrame
«interface» ISO 19111::SingleCRS
CoordinateSystem
«interface» ISO 19111::CartesianCS
+primeMeridian
«interface» ISO 19111::PrimeMeridian
+ greenwichLongitude: Angle = 0
+primeDirection
«interface» Extensions:: InertialReferenceFrame
+datum
DefiningDatum
«interface» Extensions:: InertialCRS
«interface» Extensions:: MinkowskiCS
+ellipsoid
0..1
«interface» ISO 19111::Ellipsoid
+ secondDefiningParameter: SecondDefiningParameter
+ semiMajorAxis: Length
+ semiMedianAxis: Length [0..1]
+ellipsoid

Figure 1 — Extensions to ISO 19111

`MinkowskiCS` is introduced as a spatiotemporal equivalent of `CartesianCS`. Note that `MinkowskiCS` is defined as a coordinate System (CS), not a coordinate reference System (CRS). Expectations for coordinate systems are described in the next section. Note also that `MinkowskiCS` is defined as a subtype of `CartesianCS` since the temporal axis is considered orthogonal to the spatial axes and the unit of measurement is the same (see Clause 3.2.1.4). A consequence of this hierarchy is that `MinkowskiCS` can be associated to `GeodeticCRS` and `EngineeringCRS`. By contrast, new CS classes would not be defined as direct subtypes of `CoordinateSystem`.

3.1.1. Discussion on keeping the Coordinate Systems simple

An ISO 19111 coordinate system is only a set of axes at no particular location, together with geometry formulas that are implied by the coordinate system *class* (not to be confused with the coordinate system *unions*, which are not covered in this section). For example, the use of a `CartesianCS` implies that angles, distances, surfaces, volumes, *etc.* are computed by formulas derived from Euclidean geometry. The use of a `CylindricalCS` or `SphericalCS` implies different sets of formulas for computing geometric properties. But all `CoordinateSystem` classes defined by ISO 19111 imply relatively simple and well-known formulas. This simplicity is necessary for performing tasks such as computing intersections or solving integral equations for computing surfaces. The most complex coordinate system class defined by ISO 19111 is the `EllipsoidalCS`. In that latter case, solving the integral equations requires series expansions (Clause 2.19), as done in map projections or geodesic (Clause 2.9) calculations. Extensions to ISO 19111 such as `MinkowskiCS` should not be much more complex.

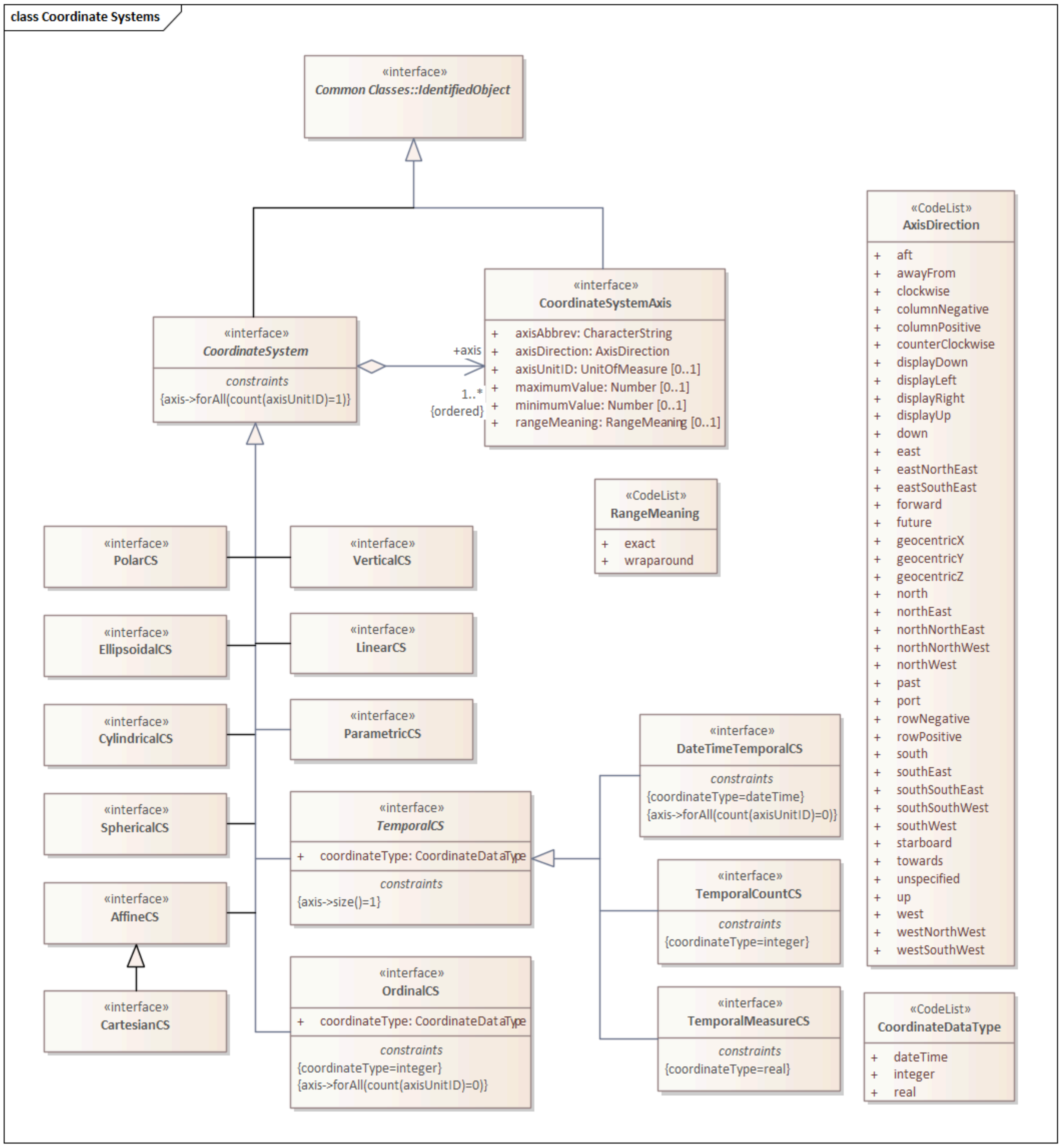


Figure 2 — ISO 19111 Coordinate Systems

The simplicity mandated by the approach defined in the above paragraph may seem restrictive, but is an unavoidable step applied even with complex topographies. Complexity is usually managed by splitting a surface into small cells. Inside each cell, the “simple” mathematics of a coordinate system is assumed to be a good approximation. For example, the figure below zooms in on a single cell of a complex surface. If a software package computes the shortest path (the geodesic) between two points inside that cell by applying linear interpolations between (x_1, y_1, z_1) and (x_2, y_2, z_2) coordinate tuples, then that software is implicitly using a Cartesian coordinate system. The cells size (grid resolution) is chosen in such a way that approximating the real surface by a flat surface inside the cell area is considered an acceptable error.

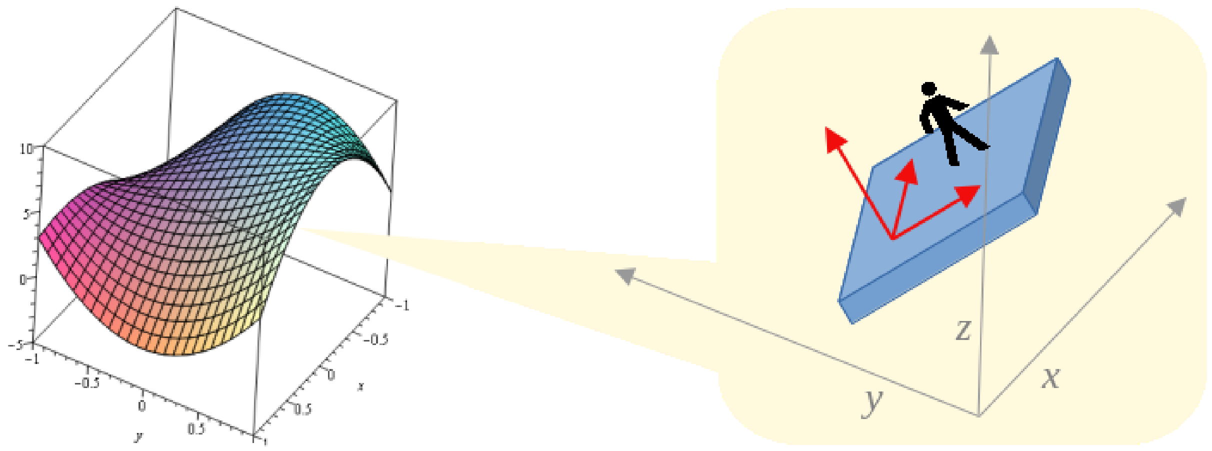


Figure 3 — Surface with local Cartesian CS in each cell

In the above example, the shortest path inside a cell can be computed using the global (Clause 2.10) Cartesian coordinate system for the whole grid (gray axes), or a Cartesian coordinate system which is local (Clause 2.13) to the cell (red axes). In the example, both coordinate systems produce the same results, but the local CS is simpler because it can be reduced to a two-dimensional coordinate system (i.e., the perpendicular axis can be omitted) if all points lie on the surface. Coordinate transformations between those two systems are not discussed in this section, as these transformations require Coordinate Reference Systems (CRS) to provide context for the Coordinate Systems (CS).

The local coordinate system does not need to be of the same class as the global one. For example, the local CS could be an `EllipsoidalCS` if that is a better approximation of the real geometry inside the area of that cell, as illustrated in the figure below. In this case, doing geodesic calculations in the local CS produces different results than the cases illustrated in Figure 3.

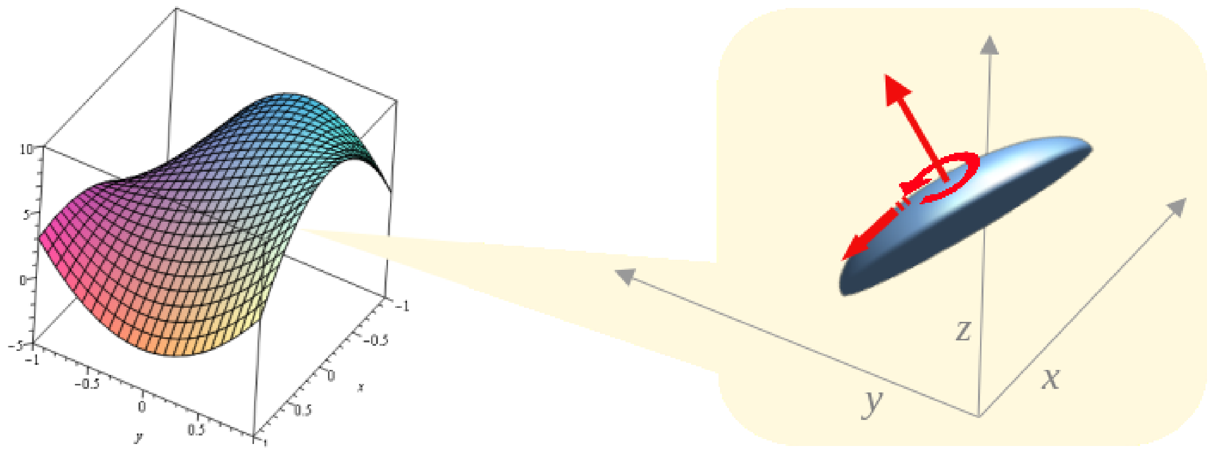


Figure 4 — Surface with local ellipsoidal CS in each cell

In the above examples, calculations must be limited to a single cell. If the feature of interest spans more than one cell, then calculations must be performed separately in each cell and the results combined in some way. ISO 19111 does not specify how multi-cell calculations should be done.

An application can choose to use the same `CoordinateSystem` (CS) for all cells, even if axis directions vary slightly between different cells. Using a common CS means that all cells use the same geometric formulas with axes having the same units of measurement and the same orientations relative to the cell. The precise orientations of the axes are encapsulated in the Reference Frame, which is not part of a Coordinate System (it is part of a Coordinate *Reference* System). Using a uniform Coordinate System makes combining the results of adjacent cells easier, for example, by stating that all cells use a `CartesianCS` with measurements in meters. Therefore, volumes or areas computed in adjacent cells can be summed without unit conversions. This is true even for `EllipsoidalCS` using ellipsoids of different sizes and shapes. In the ISO 19111 model, the shape of the ellipsoid is not a property of the Coordinate System. Rather, it is a property of the Geodetic Reference Frame which, together with the `EllipsoidalCS`, forms a Geodetic Coordinate Reference System.

3.1.2. Discussion on CRS domain of validity versus CS simplicity

An ISO 19111 Coordinate *Reference* System (CRS) is associated with exactly one Coordinate System (CS), except Compound CRS, which is omitted for the following discussion. A CRS is also associated, through subclasses of the `datum` class, with information about the CRS origin and “absolute” axis orientations (e.g., relative to stars). Consequently each cell in a grid such as Figure 3 or Figure 4 has its own local `CRS` because each cell has a different reference frame for different origin and “absolute” axis directions.

Coordinates can be transformed from the global CRS to the local CRS of a particular cell before computing geometric properties in a given cell. Note that in this scenario, the complexity of the irregular surface appears in the process of *transforming* coordinates from the global CRS to a local CRS. This complexity does not appear in any coordinate system before or after the transformation. In ISO 19111 modeling, the complexity is in `CoordinateOperation`, not in `CoordinateSystem`.

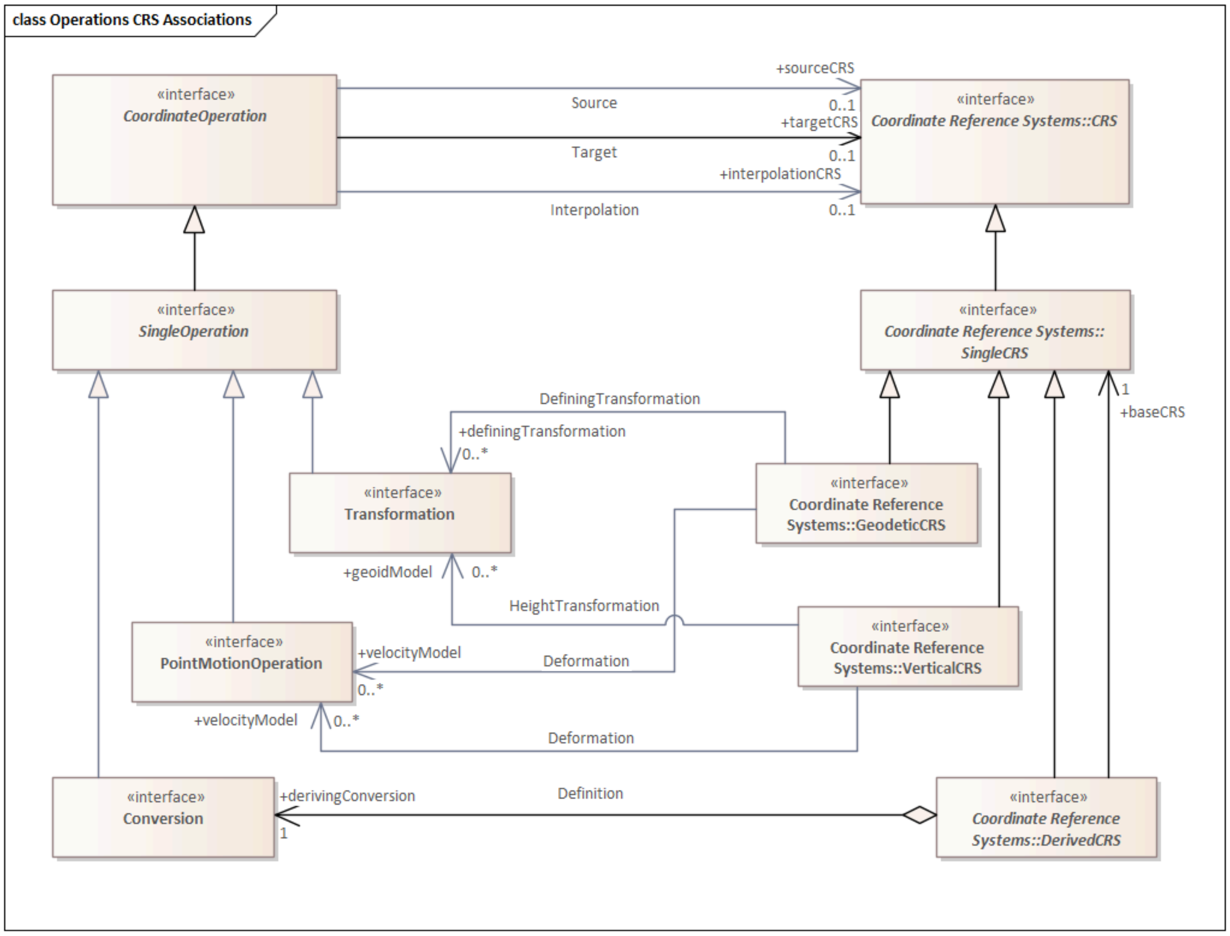

class Operations CRS Associations
«interface» CoordinateOperation
«interface» Coordinate Reference Systems::CRS
+sourceCRS
Source
0..1
+targetCRS
Target
0..1
+interpolationCRS
Interpolation
0..1
«interface» SingleOperation
«interface» Coordinate Reference Systems:: SingleCRS
1
+baseCRS
DefiningTransformation
+definingTransformation
0..*
«interface» Transformation
«interface» Coordinate Reference Systems::GeodeticCRS
+geoidModel
0..*
HeightTransformation
«interface» PointMotionOperation
+velocityModel
Deformation
0..*
«interface» Coordinate Reference Systems::VerticalCRS
+velocityModel
0..*
Deformation
«interface» Conversion
+derivingConversion
1
Definition
«interface» Coordinate Reference Systems::DerivedCRS

Figure 5 — ISO 19111 Operations and Coordinate Reference Systems

Given the restriction that each ISO 19111 CRS (excluding `CompoundCRS`) shall be associated with exactly one CS, and given that a CS should be mathematically simple, the CRS in Figure 3 and Figure 4 can be based on one of the following choices.

1. The global three-dimensional Cartesian coordinate system, in which case the complex surface plays no role. That surface is a feature *inside* the CS domain, but does not define the CS itself and is not involved in any calculation between pairs of (*x*, *y*, *z*) coordinate tuples.

2. A global two-dimensional coordinate system resulting from a curve-fitting of the complex surface. The result can be the surface of a plane, cylinder, sphere, ellipsoid, or something else defined outside ISO 19111. That approximation can be very good in some cases (e.g., Earth surface approximated by an ellipsoid). This option is nevertheless not discussed further in this Engineering Report.

3. A local two-dimensional coordinate system on the complex surface, i.e., the CS of one of the cells. That coordinate system may be Cartesian or ellipsoidal (among

others) as illustrated in the two above examples, depending on the desired compromise between accuracy and mathematical simplicity.

Each alternative has pros and cons. Option #1 allows the referencing of any point anywhere in the domain covered by the global CRS. However, it does not allow accurate geodesic calculations (e.g., shortest path on the surface) between pairs of points. Conversely, option #3 is limited to a small area, but allows much more accurate geodesic calculations in that area.

Note also that complex numerical models, such as those based on fluid mechanic (meteorological and oceanographical models) or based on the equations of Einstein's General Relativity (Annex E.2), do not change the fundamental approach. Those numerical models still need to split a large space into a grid (Clause 2.11) of cells small enough to allow the use of "simple" mathematics inside each cell with acceptable errors. The grids may be more sophisticated than grids of equilateral cells, but they are still grids with cells delimited by analytic functions.

The above discussion implies that if geometric properties or geodesics paths are required with high accuracy, then the associated calculations may require a CRS restricted to a local area. However, the area can be made larger by a choice of coordinate system class, or by a choice of kind of acceptable errors. This is partially illustrated for one-dimensional coordinate systems by the figure below. However, the discussion should be thought of as if the coordinate systems were two- or three-dimensional. The left side shows the case when Cartesian coordinate systems are wanted. No `CartesianCS` can be a good fit for the whole mountain, so the calculations must be restricted to smaller areas. The calculations can only be done between pairs of points on the same side of the mountain. If a calculation needs to be done between two points that are on each side of the mountain, it needs to be split into two parts, computed independently on each side, then combined somehow. In the ISO 19111 model, those two computations are done in two different Coordinate Reference Systems. However, if the coordinate system is based on a quadratic curve instead of straight lines, as shown in the right side of the figure below, then it becomes possible to do computation on the whole mountain with a single CRS. Therfore, computation between points on different sides of the mountain is no longer a problem.

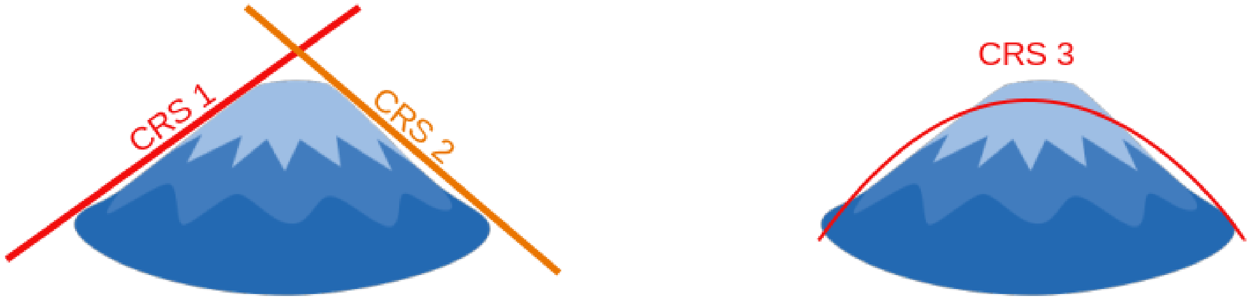


Figure 6 — More complex coordinate system covering larger area

In a two-dimensional coordinate system, the use of `EllipsoidalCS` instead of `CartesianCS` allows covering a larger surface with a continuous function in a way similar to Figure 6. In the case of the Earth, using `EllipsoidalCS` is often an approximation good enough for covering the whole planet with a single CRS, such as WGS84. However, it does not change the general principle that `EllipsoidalCS` is still an approximation. On some irregular bodies, that approximation may not be good enough to support the use of a single `EllipsoidalCS` for the whole body with the desired accuracy.

3.1.3. Coordinate System versus Coordinate Transformation roles in accuracy

A coordinate system based on a cubic curve would be more accurate or would cover a yet larger area than the examples shown in Figure 6, but at the cost of more mathematical complexity. Because the equations for intersections, surfaces, *etc.* can quickly become unsolvable, that mathematical complexity cannot be increased indefinitely. For example, ellipsoidal equations are only of degree 2, and already very difficult to solve. To cover a larger area, an alternative to increasing mathematical complexity is to keep a simpler coordinate system, such as `CartesianCS`, but accept some errors. For example, it is possible to favor accuracy in surface calculations at the expanse of accuracy in angles, or conversely. Controlling those errors is the art of map projections.

The overall accuracy does not depend only on the accuracy of formulas in the chosen coordinate system. Overall accuracy also depends on the accuracy of the transformation used for expressing points in that coordinate system. For example, Figure 7 below illustrates the results of a numerical experiment done at a point located at 45°N on Earth. The following test was repeated at distances varying from 1 meter to 1000 kilometers around the above-cited point.

- A circle is drawn around the center point (0°E 45°N) with a radius equal to the distance to test. The circle is determined by computing the locations of 360 points (each point is separated by 1°) using geodesic formulas on an ellipsoid published by Karney (2013).
- Distances and azimuths (horizontal angles from north) are computed for paths from the center point to the above-cited 360 points using the following methods.
  - Using `SphericalCS` formulas (distances are the arc lengths):
    - On an authalic sphere (sphere of the same surface as the ellipsoid) using the geographic coordinates as-is; and
    - On the same authalic sphere, but with geographic coordinates converted from the geodetic latitudes on the flattened sphere (Earth) to the geocentric latitudes on the authalic sphere. Those latitudes differ by about 21 kilometers. This operation is identified as a “datum shift” in Figure 7.
  - Using `CartesianCS` formulas (distances computed by the Pythagorean formula):
    - After projection to “Mercator (variant B)” with standard parallel at 45°; and
    - After projection to “Lambert Conic Conformal (2SP)” with the two standard parallels at 45°.
- The above calculations are compared with the expected distances and azimuths, which are the values used for drawing the circle in the first step.

The *Y* axis below shows the relative error, as a percentage of the distance shown on the *X* axis. For example, an error of 0.25% for a distance of 10 kilometers is an error of 25 meters. The error bars correspond to one standard deviation.

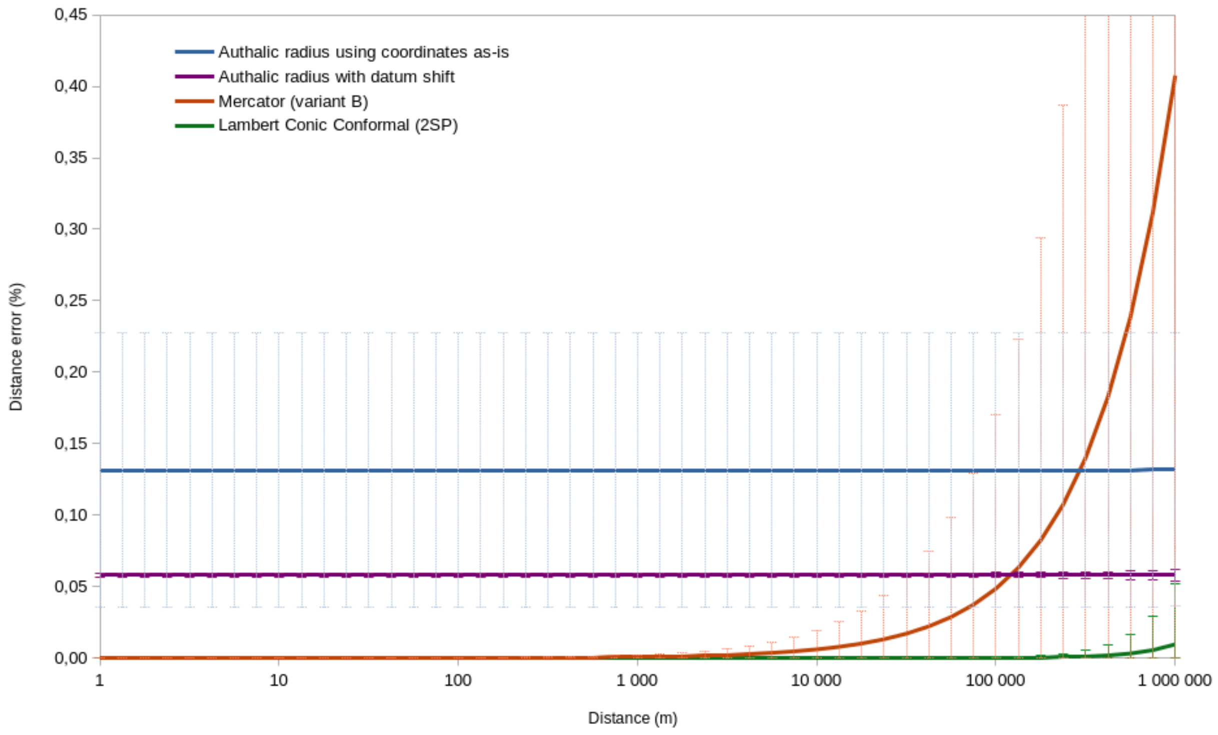


Figure 7 — Relative errors of distances around 0°E 45°N computed with different methods

An observation is that naive application of spherical formulas on Earth, with WGS84 latitudes and longitudes used directly, is not very good compared to other approaches. The relative errors are constant at around 0.13% over the range of tested distances, and the standard deviation is almost as large. A more sophisticated calculation converting the latitude values before applying spherical formulas gives better results, with relative errors at around 0.06% and negligible standard deviations. However, and maybe counter-intuitively, the use of `CartesianCS` gives better results than `SphericalCS` over small distances. Two factors contribute to that accuracy:

- the use of a “standard parallel” projection parameter configures the map projection for optimal accuracy around the latitude used in this test, namely 45°N; and
- the Mercator and Lambert Conic Conformal map projection formulas take into account the flattened shape of Earth.

On distances small enough, the geometrical distortions caused by using `CartesianCS` are smaller than the errors caused by ignoring Earth flatness in spherical formulas or caused by using a global sphere radius which may not be optimal for the test area. (Note: a sphere with geocentric radius computed at 45°N was also tested, but the results are not shown because they were similar to the authalic sphere case). In this test, the use of “Mercator (variant B)” projection is more accurate than spherical formulas for distances up to 100 kilometers and the use of “Lambert Conic Conformal (2SP)” is still more accurate even at 1000 kilometers.

The results of using “Popular Visualization Pseudo Mercator” are not shown in this graph because the errors are around 41% and would not fit in the *Y* axis scale. Those errors are caused mostly by the scale factor not well suited to 45°N. Other factors contributing to the errors are the facts that this projection ignores Earth flatness and uses the equatorial radius instead of some globally better value such as the authalic radius (radius of the authalic sphere).

3.1.4. Implications for MinkowskiCS

`MinkowskiCS` is a four-dimensional coordinate system developed for use in Special Relativity. It ignores the spacetime curvature caused by gravity according to General Relativity (Annex E.2). However, even in the context of General Relativity, the Minkowski coordinate system can be useful as an approximation good enough at short distances from the object of interest. Such an approximation is like the use of `CartesianCS` in map projections, which are flat approximations of a curved surface considered good enough inside a known region.

Because Cartesian and Minkowski coordinate systems are flat spacetime approximations, it is tempting to conclude that geodesic (Clause 2.9) calculations performed in those coordinate systems can only be less accurate than geodesic calculations performed in a coordinate system taking the curvatures in account, such as a spherical coordinate system. But as shown in Figure 7, this is not necessarily the case because the overall accuracy depends on two factors: the fitness of the coordinate system to the local geometry and the accuracy of the coordinate operation applied to get there (Clause 3.1.3).

The choice of a coordinate system impacts the complexity of a fundamental object of study in General Relativity. In the context of Einstein’s relativity, spacetime geometry is described by a *metric tensor*. The metric can be represented as a matrix and defines how distances, angles, and other geometry properties are computed. While those metrics are of particular importance in relativity, this concept can be used in classical geometry as well. The Euclidean metric used in `CartesianCS` is the simplest:

$$g_{ij} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

Let *p* be the coordinate tuple of a point in space or space-time, and p_i (or p_j) be one of the coordinates of that tuple. For a small displacement *dp*, the small distance *ds* is given by:

$$ds^2 = \sum_i \sum_j g_{ij} \cdot dp_i \cdot dp_j \tag{2}$$

For the Euclidean metric, the development gives the familiar Pythagorean formula:

$$ds^2 = \begin{pmatrix} +dx \cdot dx + 0 + 0 \\ +0 + dy \cdot dy + 0 \\ +0 + 0 + dz \cdot dz \end{pmatrix} = dx^2 + dy^2 + dz^2 \tag{3}$$

By comparison, the metric at the surface of a `SphericalCS` is as below, assuming a radius of 1, coordinates in (latitude, longitude) order and units in radians. The main thing to note is that this metric is not constant. The matrix varies at every position, because it depends on the latitude φ. Consequently, computing the geodesic distance s^2 on a non-short path requires resolving an integral equation (not done in this ER).

$$g_{ij} = \begin{bmatrix} 1 & 0 \\ 0 & cos(\phi) \end{bmatrix} \tag{4}$$

`MinkowskiCS` can be seen as `CartesianCS` generalized to the spatiotemporal domain. Its metric is shown below, using the (- + + +) sign convention. Coordinates are in (*t*, *x*, *y*, *z*) order with the time coordinate already multiplied by *c*, as required by the `MinkowskiCS` definition proposed in this ER:

$$g_{ij} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

In the same way as the metric on the surface of a `SphericalCS` may be considered of manageable complexity, there are some alternatives to `MinkowskiCS` which are also of manageable complexity. For example, assuming a system with spherical symmetry residing in a vacuum, then the Schwarzschild metric (Annex E.7) may provide a computationally feasible solution for ds^2 in Formula (2). This ER does not propose classes for modeling those alternatives yet, but it is expected that if `MinkowskiCS` is accepted, other classes could follow the path.

`MinkowskiCS` may not provide sufficient accuracy in some situations in the same way as `CartesianCS` on the Earth's surface is not accurate when the area of interest becomes too large. The accuracy on Earth can be improved by "upgrading" `CartesianCS` to `EllipsoidalCS`. Likewise, the accuracy in a relativist context may be improved by "upgrading" `MinkowskiCS` to another relativist coordinate system using a more complex metric. However, those "upgrades" cannot be such that the geometry formulas become unsolvable. When the combination of a large region and high accuracy exceeds what can be solved, splitting the region in smaller cells as illustrated in Figure 3 become unavoidable.

This Engineering Report does not explore how to combine calculations done in adjacent cells. Doing so may require the participation of other OGC working groups such as Discrete Global Grid Systems (DGGS). However, those combinations would probably require coordinate transformations between cells. This is where the complexity of geoid surfaces or gravitational fields are involved. But this complexity is in the *transformation* between two CRSs, not in the CRSs themselves.

3.1.5. Engineering CRS for spacecraft

The coordinate reference system of a spacecraft can be described by an `EngineeringCRS`. This type of CRS was discussed in OGC 22-038r2 and an example is provided with the `DART.xml` file. ISO 19111 restricts the set of CS classes that can be associated to an `EngineeringCRS`. The allowed CS classes are affine, Cartesian, cylindrical, linear, ordinal, polar, and spherical implying that `MinkowskiCS` is also accepted because it is a subtype of `CartesianCS`. However, this is not necessarily the case for other metrics such as Schwarzschild (Annex E.7). Relaxing this restriction on CS type is proposed in Clause 3.2.3.3.

3.2. Recommendations

This section describes the ISO 19111 extensions exercised in this Testbed-19. Each extension is expressed in the following three different ways:

- As Unified Modeling Language (UML) diagrams in this ER.
- As an XML schema extending the Geographic Markup Language (GML) Standard.
- As Java interfaces in a branch of the OGC GeoAPI Standard.

Each of those recommendations can be assigned to an OGC working group for discussion or standardization as follows.

- The CRS SWG can evaluate the UML diagrams for eventual incorporation in OGC Topic 2 and ISO 19111.
- The Planetary DWG can evaluate the XML schema for eventual publication on http://schemas.opengis.net/gsp/.
- The GeoAPI SWG can evaluate the new interfaces for eventual incorporation in the GeoAPI main module.

This section provides recommendations for ISO 19111 only. The recommendations for reifications such as GML and GeoAPI are presented in Annex B.1 and Annex C because they are derivatives of the recommendations for ISO 19111.

3.2.1. Additions to OGC Topic 2/ISO 19111

Most classes listed below are new and do not require modifications to the ISO 19111 Standard, as the classes can be specified as an extension in a separate document. An exception is `CelestialBody` which may need to be included in ISO 19111 if an association from `Datum` to `CelestialBody` is added.

3.2.1.1. InertialCRS

Description: InertialCRS is a 2-, 3-, or 4-dimensional coordinate reference system associated with an Inertial Reference Frame. When associated to a Cartesian coordinate system, axis directions can be `geocentricX`, `geocentricY`, and `geocentricZ`. However, those geocentric axes are not rotating with the celestial body. Instead, these axes are often pointing toward distant celestial objects.

Properties: This class inherits the attributes and associations defined by `SingleCRS` with no addition. However, the `datum` association is restricted to the `InertialReferenceFrame` subtype.

Such restrictions are called *type covariances*. The association is repeated below for specifying this restriction.

Table 3 — InertialCRS properties

PROPERTY	TYPE	OBLIGATION	DESCRIPTION
datum	InertialReferenceFrame	Mandatory	Association to the inertial datum used by this inertial CRS.

NOTE Inertial CRS provides a frame where object motions can be easier to calculate, because objects subject to no net force move in straight lines. It would not be the case in a non-inertial CRS.

NOTE A planet will most likely use a Geodetic Coordinate Reference System. However, Inertial Coordinate Reference Systems can nevertheless be associated to planets as intermediate steps in a chain of coordinate operations from one body to another body. For example, for describing the location of a spacecraft relative to Earth. See Clause 3.4 for a use case with DART data.

3.2.1.2. InertialReferenceFrame

Description: An InertialReferenceFram is an inertial datum attached to a body or barycenter (Clause 2.1), subject to no net force and therefor moving at constant velocity in straight line. The inertial datum MAY be centered on a defined ellipsoid (or sphere). Alternatively, the inertial datum may be associated to a Cartesian (3D case), or to a Minkowski (4D case) coordinate system centered in the ellipsoid (or sphere).

Properties: Inertial Reference Frames have the same properties as Geodetic Reference Frames, except that the ellipsoid and the prime meridian are optional, and the "prime meridian" attribute is named "prime direction." Contrarily to the geodetic case, the prime direction of an Inertial Reference Frame is not defined by the location of a feature on the celestial body. Such a prime meridian would be of little use because its coordinate would be constantly changing with a planet's rotation. Instead, the prime direction is defined relative to a distant feature such as a star. For example, the Ecliptic coordinate reference system defines the primary direction by the March equinox.

Table 4 — InertialReferenceFrame properties

PROPERTY	TYPE	OBLIGATION	DESCRIPTION
celestialBody	CelestialBody	Optional	Association to the celestial body used by this inertial datum.
ellipsoid	Ellipsoid	Optional	Association to the ellipsoid used by this inertial datum.
primeDirection	PrimeMeridian	Optional	Association to the prime meridian used by this inertial datum.

NOTE	Inertial Reference Frames can be used for highly irregular objects such as asteroids. These may not have an ellipsoid nor prime meridian. Planets with a solid surface will most likely use Geodetic Coordinate Reference Frames instead, but Inertial Reference Frames may still be needed as coordinate transformation steps.
NOTE	Inertial Reference Frames may be required for gas planets or stars that are not rotating as a rigid body. For those bodies, it is difficult to define a Geodetic Reference Frame if there is no stable feature for defining a rotation rate. But the ellipsoid attribute is still relevant for those bodies.
NOTE	Inertial Reference Frames are not necessarily centered on a celestial body. The frame may be centered, for example, on the barycenter (Clause 2.1) of a solar system. In the latter case, the frame origin may be located in the vacuum.
NOTE	NASA SPACE uses the “Ephemeris Object” term instead of “Celestial body.”

3.2.1.3. CelestialBody

Description: CelestialBody is a name or identifier of the star, planet, asteroid, or other celestial body for which a reference frame is defined.

Properties: Since `CelestialBody` is an `IdentifiedObject`, it inherits the `name` and `identifiers` properties. The celestial body shall be identified by its name, optionally with aliases, and optionally by one or many identifiers valid in some registries.

3.2.1.4. MinkowskiCS

Description: MinkowskiCS is a 4-dimensional coordinate system. The three spatial dimensions give the position of points relative to orthogonal straight axes. The fourth dimension gives the position in time since an epoch. All axes shall have the same length unit of measure, with time expressed as the distance covered by light in the vacuum during the elapsed time. A `MinkowskiCS` shall have exactly four `axis` property elements.

Properties: No additional properties compared to all other `CoordinateSystem` classes.

3.2.2. Additions to OGC WKT 2 / ISO 19162

This section proposes new keywords for the Well Known Text (WKT) 2 format. Those additions require changes in the definitions provided in Backus-Naur form (BNF). The following table shows a summary of new WKT keywords. Those new keywords should be added (except Minkowski) to the Well-known text representation of coordinate reference systems Standard (OGC 18-010r11) §6.6 — Reserved keywords, with “TBD” replaced by the actual section numbers where the keywords are defined.

Table 5 — Summary of new WKT keywords

KEYWORD	BNF ELEMENT	CLAUSE IN WHICH DEFINED
alias	`<alias keyword>`	TBD
celestialBody	`<celestial body keyword>`	TBD
inertialCRS	`<inertial CRS keyword>`	TBD
Minkowski	`<spatial cs type>`	7.5.1

3.2.2.1. Alias

In OGC 18-010r11 §7.3, rename the section from “Scope, extent, identifier, and remark” to “Scope, extent, identifier, *alias*, and remark”. In the requirement for `<scope extent identifier remark>`, add the following optional item before the identifiers:

```
[ { <wkt separator> <alias> } ] ...
```

Figure 8 — BNF reference to alias

Add a section with the following content:

Requirement: The WKT representation of an `<alias>` shall be:

```
<alias>         ::= <alias keyword> <left delimiter> <alias text> <right
delimiter>
<alias keyword> ::= ALIAS
<alias text>    ::= <quoted Latin text>
```

Figure 9 — BNF definition of alias

3.2.2.2. CelestialBody

In the WKT specification, insert a new section before “8.2.1 Ellipsoid” as follows.

The `<celestialBody>` object is an attribute of `<geodetic reference frame>`. The WKT representation of a celestial body shall be:

```
<celestialBody>          ::= <celestial body keyword> <left delimiter>
                             <celestial body name> <right delimiter>
<celestial body keyword> ::= CELESTIALBODY
<celestial body name>    ::= <quoted Latin text>
```

Figure 10 — BNF definition of celestial body

In section 8.2.3, update the first cell of the requirement table by the following line (the difference compared to the existing standard is the insertion of `[<wkt separator> <celestialBody>]`).

```
<geodetic reference frame keyword> <left delimiter> <datum name> <wkt
separator>
[ <wkt separator> <celestialBody> ] <ellipsoid>  [ <wkt separator> <datum
anchor> ]
[ { <wkt separator> <identifier> } ]…  <right delimiter>
[ { <wkt separator> <prime meridian> } ]
```

Figure 11 — Modified BNF definition of geodetic reference frame

Add the following note below the table, where TBD is the number of the new section described above (the one added before §8.2.1).

- `<celestialBody>` is described in TBD.

3.2.2.3. InertialReferenceFrame

Add a new section using the existing `GeodeticCRS` as a template. The definition of `InertialCRS` is similar to `GeodeticCRS` with `GeodeticReferenceFrame` replaced by `InertialReferenceFrame`.

3.2.3. Changes to ISO 19111

This section describes proposed changes to ISO 19111 abstract model. If accepted, those changes would require the publication of an ISO 19111 amendment and/or an OGC corrigendum.

3.2.3.1. Association to CelestialBody

The association to `CelestialBody` should be defined in the base `Datum` class for allowing the celestial body to be defined for any kind of datum not only inertial, but also geodetic, vertical, parametric, and more. Doing so would require the addition defined in Clause 3.2.1.3 to be included in ISO 19111.

The association is defined as optional for compatibility with existing CRS definitions. If the celestial body is not provided, the default value is unspecified. It may be Earth, or it may be inferred from the context.

3.2.3.2. Remove ObjectUsage for generalization

In the ISO 19111 model, most classes inherit a set of properties from the `IdentifiedObject` parent class. Those properties are *name*, *alias*, *identifier*, and *remarks*. But there is also one additional property, namely *domain*, which is inherited only by the `Datum`, `CRS`, and `CoordinateOperation` classes. ISO 19111:2019 does this restriction by defining those properties in an intermediate class named `ObjectUsage`. ISO 19111:2007 did this restriction by repeating properties in the three above-cited classes, in a way similar to the UML diagram provided in section Clause 3.1.

The following UML diagram shows the ISO 19111:2019 model. The `ObjectUsage` contains only a `domain` property and is inserted between the `IdentifiedObject` parent class and the `Datum`, `CRS`, and `CoordinateOperation` subclasses. This insertion was done for preserving the same restriction as in ISO 19111:2007, where only `Datum`, `CRS`, and `CoordinateOperation` could have a scope and domain of validity.

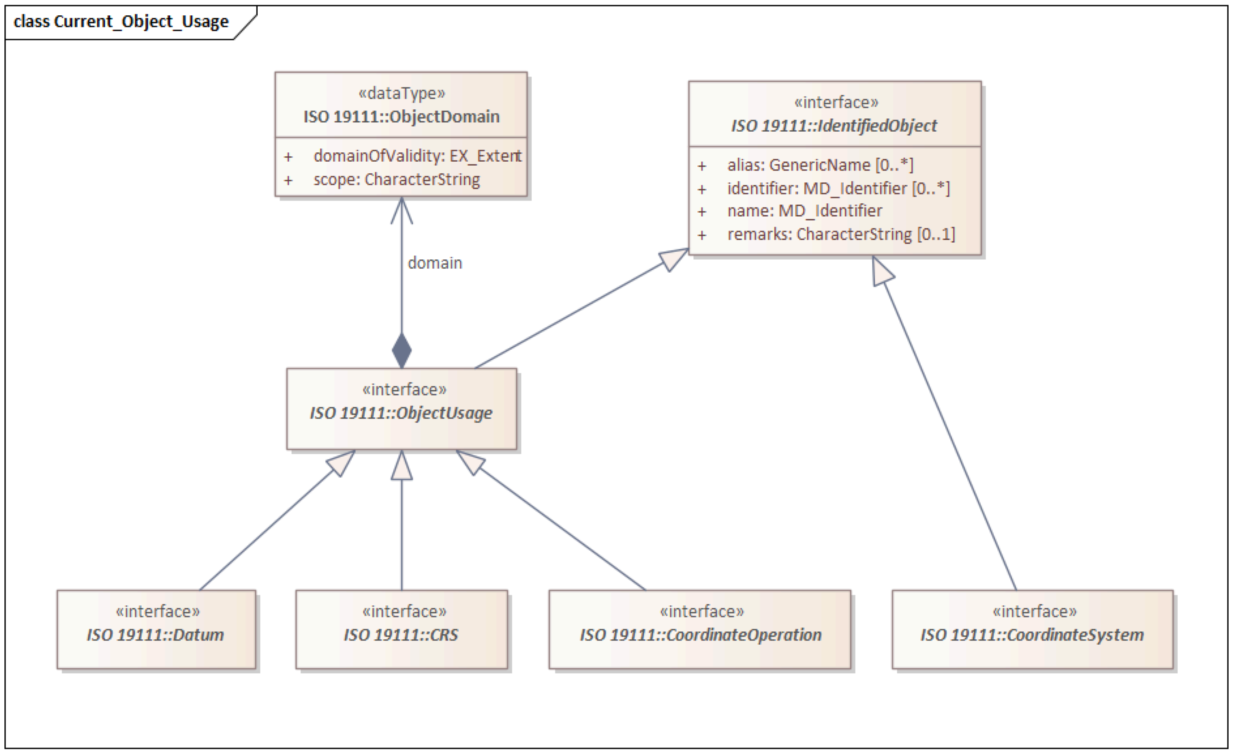

class Current_Object_Usage
«dataType»
ISO 19111::ObjectDomain
+ domainOfValidity: EX_Extent
+ scope: CharacterString
«interface»
ISO 19111::IdentifiedObject
+ alias: GenericName [0..*]
+ identifier: MD_Identifier [0..*]
+ name: MD_Identifier
+ remarks: CharacterString [0..1]
domain
«interface»
ISO 19111::ObjectUsage
«interface»
ISO 19111::Datum
«interface»
ISO 19111::CRS
«interface»
ISO 19111::CoordinateOperation
«interface»
ISO 19111::CoordinateSystem

Figure 12 — Current ISO 19111:2019 model with object usage

Removing the restriction saying that only `Datum`, `CRS`, and `CoordinateOperation` can have a scope or a domain of validity is proposed. There is nothing inherently wrong in specifying a scope on most kinds of CRS-related objects. For example:

- A `MinkowskiCS` definition may want to specify *“for use with special relativity”* as the scope.
- An `Ellipsoid` definition may want to specify a limited geographic domain of validity if the celestial body has an irregular shape with different ellipsoidal approximations for different geographic areas.
- A `PrimeMeridian` definition may want to specify a limited temporal extent if it is determined by an ephemeral feature, for example of the surface of a gas planet.
- A `CoordinateSystemAxis` with direction such as “South along 60°E meridian” may want to specify *”for use at North pole”* as the scope.

The restriction can be removed by deleting the `ObjectUsage` class (not to be confused with the `ObjectDomain` class, which is kept unchanged), and moving the `domain` attribute up to the `IdentifiedObject` parent class.

3.2.3.3. Remove EngineeringCS for generalization

The ISO 19111 Standard contains some coordinate systems (CSs) that are unions instead of classes. The purpose for the CSs is not to imply a new set of geometric formulas, but to restrict the classes of coordinate systems that can be associated to a CRS. One of those unions is `EngineeringCS`, which is a union of all CS classes except the ellipsoidal, vertical, parametric, and temporal coordinate systems. While the use of unions met the intended goal of forbidding the association of an Engineering CRS with, for example, an `EllipsoidalCS`, it may actually forbid associations to any coordinate system that is not in the union list. A closed universe of CSs may be desirable for some CRS types such as `GeodeticCRS`, but `EngineeringCRS` may be a case where an open universe is desirable for allowing associations to user-defined coordinate systems. If an open universe is accepted, then the `EngineeringCS` union should be removed and replaced by a requirement in the text saying that this CRS should not be associated to ellipsoidal, vertical, parametric, or temporal coordinate systems.

This change proposal is not necessary for using `MinkowskiCS` as proposed in this ER. This is because MinkowskiCS is defined as a subtype of the `CartesianCS` class, which is a member of the `EngineeringCS` union. However, that change proposal may become necessary if a user wants to use the Schwarzschild metric (Annex E.7) instead of Minkowski (Annex E.5) with an engineering CRS.

The above issue applies also to `GeodeticCRS`: it may be associated to a `MinkowskiCS` as a specialization of `CartesianCS`, but cannot be associated to other metrics unless the metrics are considered as specializations of some CS classes enumerated in the `GeodeticCS` union. However, the CS restriction in `GeodeticCRS` may be more important than in the `EngineeringCRS` case, so this ER does not propose relaxing that restriction at this stage.

3.2.3.4. Remove temporal origin

The `TemporalDatum` class contains a mandatory `origin` property of type `dateTime`. By default, the origin uses the proleptic Gregorian calendar of the Earth. However, there is no absolute time in the context of special relativity. Specifying a time is done by declaring that an event happened simultaneously with the clock showing some numbers. But two events that are simultaneous according to one observer may not be simultaneous according to another observer.

Time origins were typically used for computing the time offset between two temporal reference systems. For example, when converting from one CRS having its origin in 1970 to another CRS having its origin in 2000, the operation can be done by subtracting 30 years to the temporal coordinate. It may be more robust to provide such value as a coordinate operation defined in an EPSG-like database, like other datum shifts in the spatial domain, instead of inferring the operation from temporal datum properties.

3.3. Coordinate transformations

The ISO 19111 framework for Coordinate Operations is not constrained to the Earth. An ISO 19111 coordinate operation describes the relationship between two CRSs without assumption about the frame types (inertial, geodetic, *etc.*). Different operation subclasses exist, but this hierarchy is for tasks such as describing the kind of errors to expect and for chaining operations together. A class of particular interest is `Transformation`, which contains the following properties (non-exhaustive list).

- `sourceCRS`: The CRS of source coordinate tuples (before transformation).
- `targetCRS`: The CRS of target coordinate tuples (after transformation).
- `interpolationCRS`: The CRS in which coordinates are interpolated.
- `method`: The coordinate transformation formulas which may expect parameters.
- `parameterValue`: Values to assign to the parameters expected by above-cited method.
- `domain`: The scope and the ISO 19115 extent where the transformation is valid.
- `coordinateOperationAccuracy`: Accuracy as an ISO 19157 (data quality) object.

The following UML diagram shows the *spirit* of ISO 19111 `Transformation` and related classes which is not an exact representation of the ISO 19111 model as some classes are omitted and replaced by attributes for simplicity, sometimes in a different location for emphasis. For example, ISO 19111 puts `coordinateOperationAccuracy` in the `CoordinateOperation` parent class, but the UML below shows it in the `Transformation` class in order to highlight that it is of particular relevance there. Note that this framework contains a `transform(CoordinateSet)` operation, which performs the actual task of transforming a set of coordinate tuples.

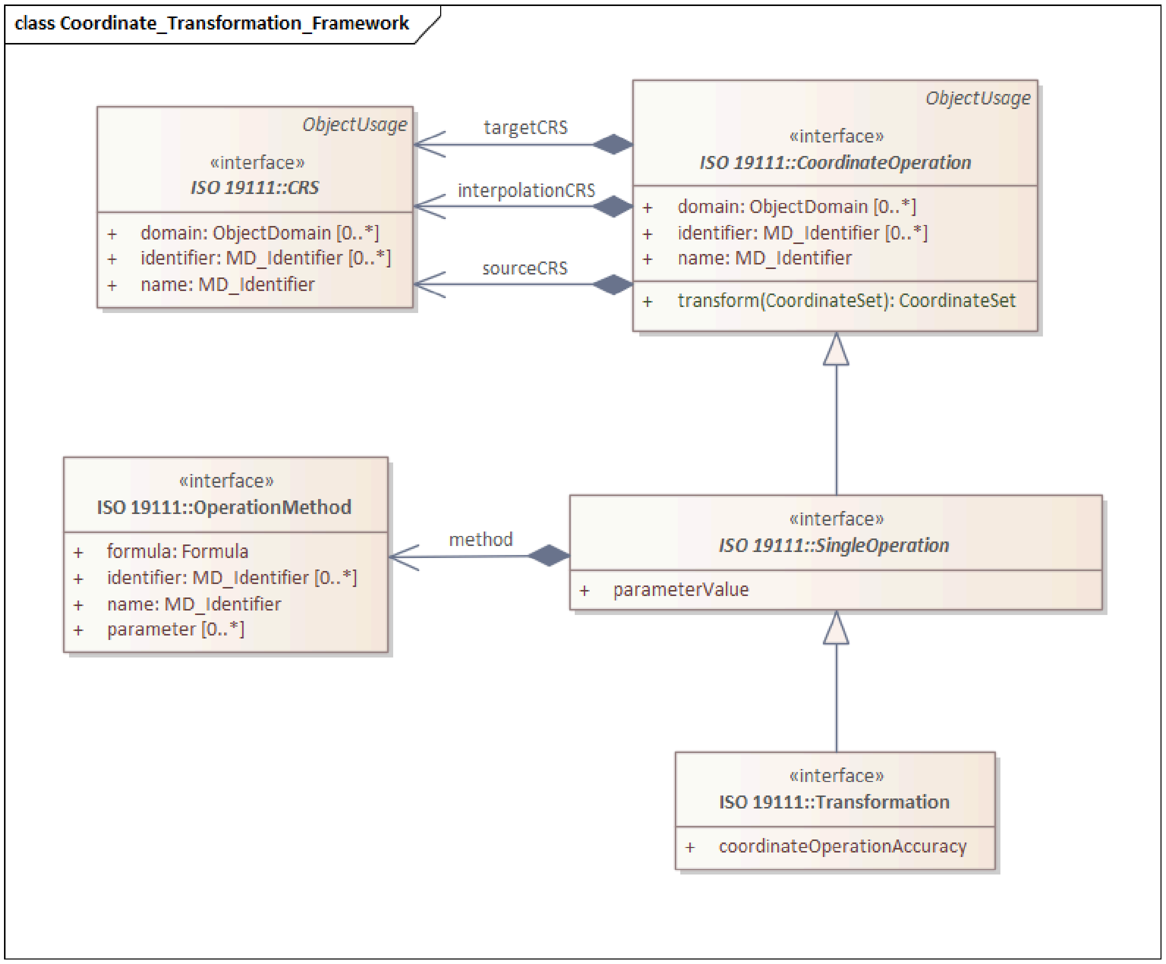

class Coordinate_Transformation_Framework
ObjectUsage
«interface»
ISO 19111::CRS
+ domain: ObjectDomain [0..*]
+ identifier: MD_Identifier [0..*]
+ name: MD_Identifier
targetCRS
interpolationCRS
sourceCRS
ObjectUsage
«interface»
ISO 19111::CoordinateOperation
+ domain: ObjectDomain [0..*]
+ identifier: MD_Identifier [0..*]
+ name: MD_Identifier
+ transform(CoordinateSet): CoordinateSet
«interface»
ISO 19111::OperationMethod
+ formula: Formula
+ identifier: MD_Identifier [0..*]
+ name: MD_Identifier
+ parameter [0..*]
method
«interface»
ISO 19111::SingleOperation
+ parameterValue
«interface»
ISO 19111::Transformation
+ coordinateOperationAccuracy

Figure 13 — Spirit of ISO 19111 coordinate transformation framework

The interpolation CRS is an interesting concept. It is often either the source or target CRS but can also be different. For example, if the source and target CRS are both attached to bodies orbiting around each other or around a larger body, it may be convenient to perform the interpolations in the CRS of the barycenter of the system because the orbiting bodies are subject to accelerations, which can make linear interpolations unsuitable. By contrast, if the barycenter has negligible accelerations, then linear interpolations in the CRS of that barycenter may be better approximations.

The source and target CRS exist in the GML schema but not the interpolation CRS because the concept was introduced in an ISO 19111 revision published after GML was designed. However, this is not a problem for the use cases considered in this ER, because the data to interpolate are encoded as Moving Feature files. Those data shall be expressed in the interpolation CRS (because the data are the values to interpolate), so the natural consequence is that the Moving Feature CRS defines the interpolation CRS.

An operation can be applied to a subset of coordinate tuples. For example, if an operation works only on the spatial coordinates (*x*, *y*, *z*), it does not mean that this operation cannot be used with four-dimensional (*x*, *y*, *z*, *t*) coordinates. ISO 19111 has a concept of “pass-through” operation, which can be combined with any other operation for applying the latter on some coordinates (e.g., the first 3), then pass-through the remaining coordinates (e.g., *t*) unchanged. Thus, `PassThroughOperation` can be combined with existing operations for creating new operations with larger numbers of dimensions. It is possible to create one pass-through operation working on the spatial dimensions of four-dimensional coordinate tuples, another pass-through operation working on the temporal dimension of the same tuples, then chain them together as described in Clause 3.3.1. An example is illustrated below, with ISO 19111 class names in box titles.

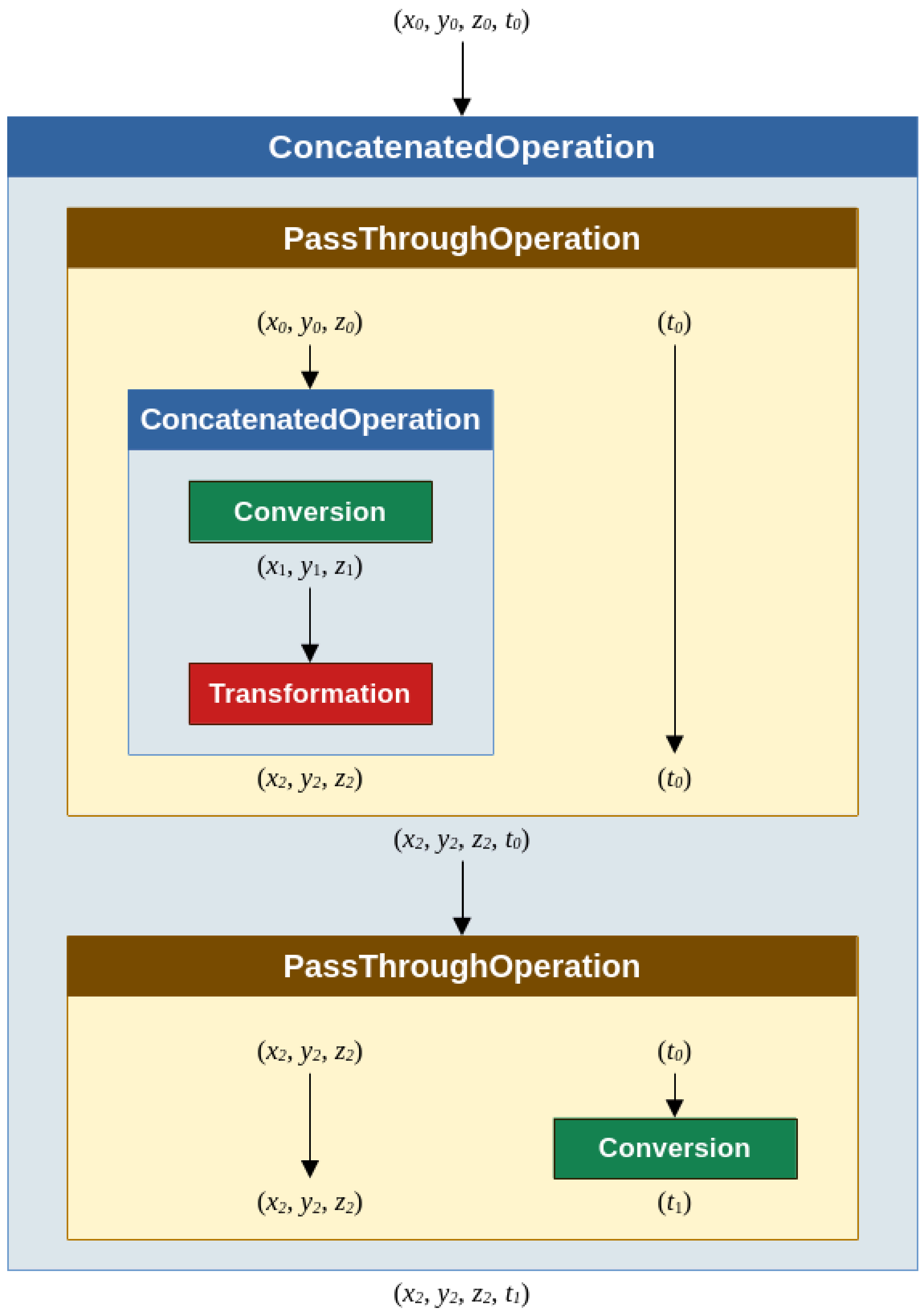


Figure 14 — Example of concatenated pass-through operations

A constraint in the use of pass-through operations as described above is that the time dimension is handled independently of space dimensions, and conversely. For handling dimensions together as in Einstein's relativity, a single operation (not pass-through) working on (*x*, *y*, *z*, *t*) coordinates together must be defined, which can be done with no change to the ISO 19111 coordinate operation framework. This ER has not identified any ISO 19111 limitation for expressing coordinate transformations in space, assuming that the CRS can be described (Clause 3.2) and that required operation methods are defined (Clause 3.3.5).

3.3.1. Chain of operations

ISO 19111 coordinate operations can be chained together in a `ConcatenatedOperation` object, which contains:

- the overall source CRS of the chain;
- the overall target CRS of the chain;
- each step as a reference to a single operation (in GML documents, it can be a `xlink:href` attribute); and
- other attributes common to all operations (identifier, name, scope, domain, and accuracy).

Each step other than pass-through can be qualified as a `Conversion`, `Transformation`, or `PointMotionOperation`. The difference between conversion and transformation was discussed in the Testbed-18 Engineering Report OGC 22-038r2 §5.6, and illustrated in the Testbed-18 3D + Data Space Object Engineering Report OGC 23-011r1 §8 — Importance of datum. The key takeaway is that transformations are the steps where stochastic errors are introduced. Thus, by checking the type of each step, the ISO 19111 model enables users to perform some analysis of the (possibly many) sources of errors. For example in Figure 14, there are stochastic errors in the step represented by a red box. Different chains of operations between the same (source, target) pair of CRS may use different transformation steps with different sources of errors.

A concatenated operation can have as many steps as desired, where each step may use a different method. The main constraint is that the target CRS of a step must be the source CRS of the next step, except for inverse operations where the above-cited CRS can be swapped. Specifically, if a concatenated operation is defined as below:

- Overall source: CRS **A**
- Overall target: CRS **C**
- Step 1: from CRS **B** to CRS **A**
- Step 2: from CRS **B** to CRS **C**

then the software should understand that step 1 needs to be inverted, i.e., it needs to go from CRS **A** to CRS **B**. This is illustrated by Figure 15. Sometimes, the software can derive the inverse operation simply by inverting the sign of parameter values. Sometimes, the inverse operation requires completely different formulas (e.g., map projections). How to derive the inverse operation shall be specified by the operation method documentation. This flexibility avoids the need to define every `OperationMethod` twice: once for the forward direction and once for the reverse direction. Users may define an explicit `OperationMethod` for the reverse direction if that is wished, but it is not required. The figure below is the case where it is not desired.

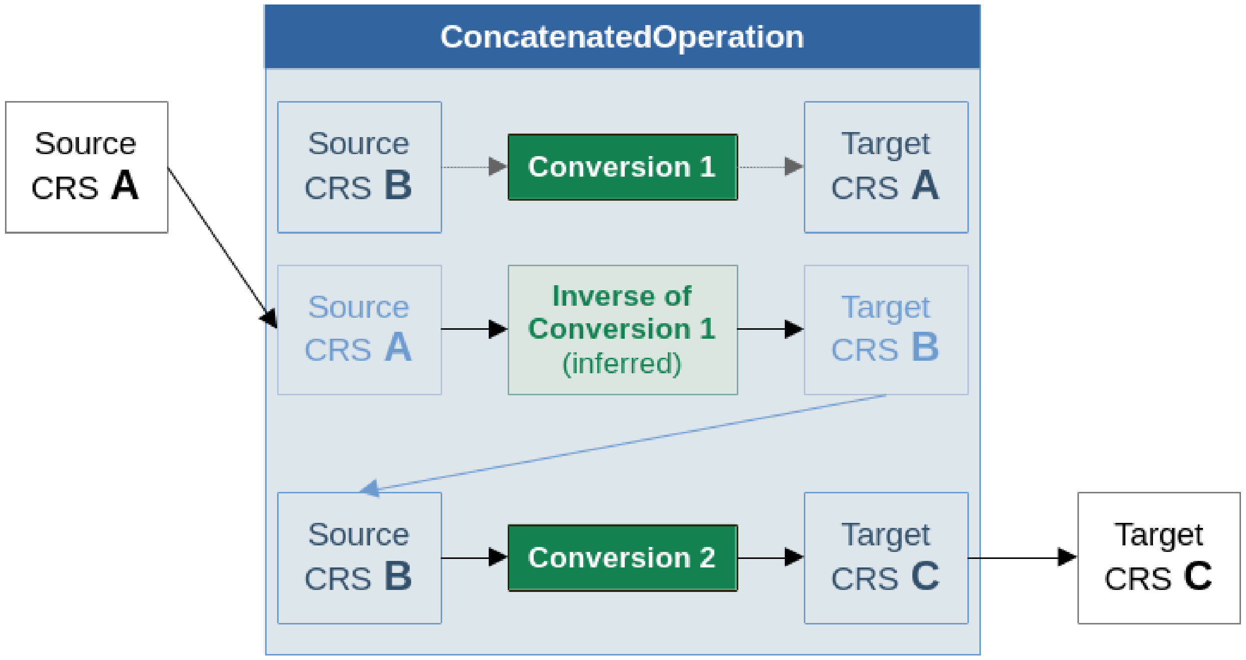


Figure 15 — Example of inverse operation in a concatenated operation

3.3.2. Operation encoding

The Well-Known Text (WKT) 1 format cannot encode coordinate operations, except to some extent via the `PARAM_MT` keyword of the OpenGIS Coordinate Transformation Service Implementation Specification OGC 01-009, but the latter was not a well-known feature. The fact that WKT 1 was practically restricted to Coordinate Reference System encoding may have contributed to the perception that ISO 19111 provides no coordinate operation framework. OGC 18-010r11 (a.k.a. WKT 2) can encode some coordinate operations, but does not cover the full set of ISO 19111 features (e.g., it excludes `PassThroughOperation`) and has no mechanism such as GML `xlink:href` for avoiding repetition (e.g., each `targetCRS` is repeated in the `sourceCRS` of the next step). The Geographic Markup Language (GML) is the most capable encoding for coordinate operations in current OGC Standards, except for the new ISO 19111 classes introduced after 2007. Unfortunately, few open source software applications provide good GML support for coordinate operations. At the time this ER was written, Apache Spatial Information System (SIS) is the only open source project to our knowledge with this capability, but this project is not as widespread, for instance, as GDAL.

NOTE: The lack of well-adopted Coordinate Operation encoding does not mean that the ISO 19111 coordinate operation framework was not used. The EPSG geodetic parameter dataset is an implementation of this framework (EPSG is not only about CRS definitions). EPSG data are extensively used by Apache SIS and PROJ 6+ in what is known as *late-binding* implementations (by contrast, PROJ 4 was an *early-binding* implementation). Actually, the GDAL Coordinate System Barn Raising was largely about supporting the ISO 19111 coordinate operation framework in PROJ and GDAL, both in a new PROJ C++ API and in the the coordinate operation part of EPSG geodetic dataset.

OGC 22-038r2 demonstrated the use of GML for describing a chain of coordinate operations from an imaginary observatory on Earth to an imaginary spacecraft. However, due to technical constraints of the prototype at that time, all steps and their CRSs had to be described in a single GML document, resulting in a XML file about 700 lines long. In this ER, the main components are described in separate GML files, with relationships specified by `xlink:href` attributes in GML. For example, a concatenated operation can be described as below:

```xml
<gml:ConcatenatedOperation gml:id="DimorphosToDART">
  <gml:identifier codeSpace="TB19-D001">DimorphosToDART</gml:identifier>
  <gml:name>Dimorphos to DART</gml:name>
  <gml:scope>Testbed-19 demonstration.</gml:scope>
  <gml:operationVersion>1</gml:operationVersion>
  <gml:sourceCRS xlink:href="BarycenterToDimorphos.xml#DimorphosCRS"/>
  <gml:targetCRS xlink:href="BarycenterToDART.xml#DART"/>
  <!--
    The first step below should actually be "Dimorphos to barycenter", but the
software
    should detect, by inspection of the source and target CRS, that it needs
to use the
    inverse of that operation.
  -->
  <gml:coordOperation xlink:href="BarycenterToDimorphos.xml"/>
  <gml:coordOperation xlink:href="BarycenterToDART.xml"/>
</gml:ConcatenatedOperation>
```

Figure 16 — Concatenated operation example in GML

The use of those files by a prototype is discussed in Clause 3.4.

3.3.3. Finding relationships between CRSs

A measurement may be expressed in the local (proper) reference system (e.g., `GeodeticCRS`) or in that of an observing entity (e.g., `EngineeringCRS`). The value of a measurement may be expressed in an unlimited number of reference systems. ISO 19111 provides a framework to support coordinate transformations between arbitrary coordinate reference systems. The process can proceed as follows.

1. For each pair of Coordinate Reference Systems of interest, it describes the relationship between those two reference systems with a `CoordinateOperation`. An unlimited number of `CoordinateOperation` instances can be created, including many instances for the same pair of reference systems if desired.

2. If chaining some of the above-defined operations is useful, describe those chains with `ConcatenatedOperation` instances. It is not necessary (neither feasible) to enumerate all possible chains. The purpose of explicit `ConcatenatedOperation` instances is to state that, among all possible sequences of operations going from CRS A to CRS C, the path $A \rightarrow B \rightarrow C$ (for example) is preferred.

3. List all above definitions in a database. For example, a relational database, or a directory containing a GML file such as Figure 16 for each coordinate operation.

The EPSG geodetic parameter dataset is an example of a relational database defining a set of coordinate operations (in addition to the CRS definitions).

4. Implement the ISO 19111 `RegisterOperations` class, in particular the following method:

```
findCoordinateOperations(source : CRS, target : CRS) : Set<CoordinateOperation>
```

Figure 17 — Search of Coordinate operations in a register

The above method first searches in the database, or in the directory of GML files, for coordinate operations defined for the given source and target reference systems. If at least one `CoordinateOperation` is found, then all matching operations are returned. Users can inspect the metadata listed in Clause 3.3 (e.g., domain of validity and positional accuracy) for deciding which operation is best suited to the need. Users can also base decisions on an analysis of operations steps, for example as illustrated in Figure 14.

NOTE: If many `CoordinateOperation` instances are defined for the same source and target reference systems, each of those operations should contain at least one `Transformation` step. A logical inconsistency would be to have many operations made of only `Conversion` steps, unless these steps are mathematically equivalent (ignoring rounding errors).

If no predefined operation was found for the given pair of reference systems, then the implementation can try to infer a sequence of operations. For example, if the user requested an operation from *A* to *C* but the database contains only an operation from *A* to *B* and another operation from *B* to *C*, then the implementation can infer the $A \rightarrow B \rightarrow C$ sequence. However, this approach is not always deterministic as the number of possible combinations may grow exponentially with the number of steps, and the implementation may not be able to afford the cost of exploring all possible paths. Providing explicit `ConcatenatedOperation` instances in the database for preferred operation chains is desirable.

3.3.4. Executing coordinate transformation

Once a `CoordinateOperation` has been chosen, coordinates can be changed from being referenced to the `sourceCRS` to being referenced to the `targetCRS`, and/or (in the case of dynamic CRS) from being referenced to one coordinate epoch to being referenced to a second coordinate epoch, by executing the following method.

```
transform(points : CoordinateSet) : CoordinateSet
```

Figure 18 — Coordinate operation execution

A `CoordinateSet` contains coordinate metadata (CRS and data epoch), together with an ordered list of coordinate tuples as `DirectPosition` objects. The `transform` method preserves the tuples ordering.

3.3.5. Standard method proposals

The ISO 19111 framework for coordinate operations can be applied in space with no change. New *operation methods* need to be defined, but this is done using the existing model. For example, seven operation methods were proposed in OGC 22-038r2 annex A:

- four methods for expressing GeoPose's Basic-YPR (yaw, pitch, and roll), Basic-Quaternion, Regular-Series, and Irregular-Series functionalities in terms of ISO 19111 model; and
- three methods for implementing a "Voyager to Observatory" scenario defined in that ER.

This ER focuses on only one operation method but defines that method more extensively for making it more suitable to standardization. The operation method name is "Translation by trajectory" and the proposed code in OGC namespace is http://www.opengis.net/def/method/OGC/1/200 (this URL would also be the place for information on where to publish the definition). The definition is specified as a GML encoding in Annex B.2. This first operation method is very simple. It:

- works on 4-dimensional (*x*, *y*, *z*, *t*) coordinate tuples, but non-relativist;
- assumes that axis directions relative to distant stars do not change; and
- transformations consist of translations provided by a Moving Feature file.

Note that the simplicity of the above characteristics is not due to ISO 19111 limitations. There is no constraint in the number of parameters that an operation method can define. Those parameters can be Booleans, integers, list of integers, measures, list of measures, character strings, filenames, URLs, citations, or geometries. Parameters can also be defined in groups and repeated. The OGC 22-038r2 ER used parameter grouping to demonstrate how a piecewise function can be defined with ISO 19111 framework. In this Testbed-19 ER, the operation method defines only one parameter:

- http://www.opengis.net/def/parameter/OGC/1/200 — "Trajectory file"

The value of that parameter is a filename or URL to a Moving Feature file in a format defined by OGC: JSON, XML, CSV, or netCDF (for this Testbed prototype, only CSV is supported). The Moving Feature file defines the trajectory of the target CRS relative to the source CRS. The target CRS may be, for example, an `EngineeringCRS` attached to a spacecraft and the source CRS may be an `InertialCRS` attached to a planet. The CRS declared in the Moving Feature file for trajectory data is taken as the interpolation CRS. In the simplest case, the interpolation CRS is the same as the source CRS. The algorithm to apply is as follows.

1. For each (*x*, *y* , *z* , *t*) source coordinate tuples, interpolate the position in the trajectory at time *t*. The result is taken as a translation vector in units of the interpolation CRS.
2. If the interpolation CRS is not the same as the source CRS, then:

- get the derivative (Jacobian matrix) of the transform from source CRS to interpolation CRS at the location given by source coordinates;
- inverse the Jacobian matrix; and
- multiply the inverted matrix by the translation vector obtained in step 1. The result is the translation vector converted from interpolation CRS to source CRS.

3. Subtract the translation vector from the source (*x*, *y* , *z*) coordinates and pass the value *t* unchanged.

The algorithm for the reverse operation is below. The condition in step 2 should not mention the source CRS, but in the case where the difference between source and target CRS is only a translation with no scale or rotation, the derivative of the transformation between those two CRSs is an identity matrix everywhere and step 2 can be skipped.

1. For each (*x*, *y*, *z*, *t*) target coordinate tuples, interpolate the position in the trajectory at time *t*. The result is taken as a translation vector in units of the interpolation CRS.
2. If the interpolation CRS is not the same as the source or target CRS, then:
   - get the derivative (Jacobian matrix) of the transform from target CRS to interpolation CRS at the location given by target coordinates;
   - inverse the Jacobian matrix; and
   - multiply the inverted matrix by the translation vector obtained in step 1. The result is the translation vector converted from interpolation CRS to target CRS.
3. Add the translation vector to the target (*x*, *y* , *z*) coordinates and pass the value *t* unchanged.

The use of this operation method is demonstrated by the prototype in Clause 3.4.

3.4. Example using DART data

The extensions to existing OGC and ISO Standards proposed in this Testbed-19 were tested with data from the DART mission. The Double Asteroid Redirection Test (DART) was a mission from the NASA Planetary Defense Coordination Office. The mission measured the changes of Dimorphos orbit around Didymos after an intentional collision of a spacecraft on the Dimorphos surface. DART data are publicly available as text files with a header containing a succinct definition of the inertial reference frame used by the file. Testbed-19 work encapsulated some DART data into ISO 19111 objects using the proposed extensions when necessary, together with trajectory data encoded as Moving Feature files. The results are in the following files available on GitHub:

Table 6 — Example files for DART scenario

FILE	FORMAT	DESCRIPTION
`SolarSystemBarycenter.xml`	GML + TB19 extension	Inertial CRS centered on the solar system barycenter.
`Earth.xml`	GML + TB19 extension	Inertial CRS centered on Earth.
`Didymos.xml`	GML + TB19 extension	Inertial CRS centered on the Didymos main body.
`Dimorphos.xml`	GML + TB19 extension	Inertial CRS centered on the satellite.
`DART.xml`	GML	Engineering CRS centered on the DART spacecraft.
`BarycenterToDART.xml`	GML	Transformation from barycenter CRS to DART CRS.
`BarycenterToDART.csv`	Moving Feature CSV	Trajectory data needed by above transformation.
`BarycenterToDimorphos.xml`	GML	Transformation from barycenter CRS to Dimorphos CRS.
`BarycenterToDimorphos.csv`	Moving Feature CSV	Trajectory data needed by above transformation.
`OperationChain.xml`	GML	Transformation from Dimorphos CRS to DART CRS.

Snippets of above files are provided in Annex B.1.1. Note that contrary to other inertial Coordinate Reference Systems, the Solar System Barycenter CRS does not contain an ellipsoid, as it is not exactly at the Sun's center. Also note that, as said in Clause 1, the coordinate transformation part of ISO 19111 does not need to be extended for supporting operations in space (the standard GML encoding is sufficient).

The target of the coordinate operation is the DART CRS, the same CRS used in creating the GeoTIFF file in OGC 23-028 (OGC Testbed-19 — Extraterrestrial GeoTIFF Engineering Report). This consistency enabled the demonstration to complete the operation chain with a conversion to the GeoTIFF CRS, then to the image pixel coordinates which is useful because a GeoTIFF image using an `EngineeringCRS` contains no information about how this CRS relates to the rest of the universe. This relationship can be encoded as a `CoordinateOperation` in a GML file as shown in this ER, with a source or target CRS matching the GeoTIFF CRS.

3.4.1. Implementation-specific interpretations of OGC Standards

There are two issues that are not formally addressed by OGC standards (decisions left to implementers), but which must be resolved to produce an executable demonstration (Annex B.3). The first issue is in which directory are the CSV files listed in Table 6? This ambiguity is not specific to Testbed-19. For example, the EPSG database also references datum shift files (NADCON, NTv2, *etc.*) in the same way as this ER references the Moving Feature files. Implementations such as Apache SIS or PROJ expect those datum shift files to pre-exist in dedicated directories. But this approach may not be applicable for transformations using user-supplied trajectory files. For the Testbed-19 prototype (Annex B.3), it has been decided that the "Trajectory file" parameter value (e.g., as in Figure B.4) is interpreted as a path relative to the location of the GML file.

The second issue is how to resolve the CRS identifiers in Moving Feature files. The various encodings for Moving Feature (MF) do not include full CRS definitions, but only use identifiers (e.g., Figure B.5). But because of the space context, those identifiers are not EPSG identifiers. In Testbed-19, the identifiers are rather like `urn:ogc:def:crs:JPL::120065803` (note: the use of "JPL" in this ER does not mean an endorsement of the Jet Propulsion Laboratory. It is only an illustration of what it may look like if JPL was publishing those definitions). Since there is no CRS registry for the "JPL" authority, where should users look for this information? The prototype developed and documented in this ER takes advantage of the fact that full CRS definitions are provided in GML documents, as part of the ISO 19111 `CoordinateOperation` definition. For example, the `BarycenterToDimorphos.csv` MF file (Figure B.5) is referenced by the `BarycenterToDimorphos.xml` GML file (Figure B.4). The latter itself contains `xlink:href` attributes to full CRS definitions like Figure B.2, which contains the following line:

```
<gml:identifier codeSpace="JPL:HORIZONS">urn:ogc:def:crs:JPL::120065803</gml:
identifier>
```

Figure 19 — Fragment of Didymos CRS definition

The Apache SIS library uses this context for tracing `urn:ogc:def:crs:JPL::120065803` back to its full CRS definition. Note that there is no need to explicitly create a project-specific registry in some file. Instead, a "virtual registry" is created in memory only and populated automatically by the library at GML parsing time.

3.5. Future work

Some topics not covered in depth in this ER are the definition of an authoritative registry for extra-terrestrial CRSs and the handling of General Relativity.

3.5.1. CRS registry

There is currently no Celestial Reference System (CRS) such as BCRS (Barycentric Celestial Reference System) in the OGC registry. OGC maintains an http://www.opengis.net/def/crs/

IAU/ namespace for astronomical bodies, but this namespace currently contains only Coordinate Reference Systems for planetology based on the work of the IAU's Cartographic Coordinates & Rotational Elements working group. Allocating identifiers for CRSs from other sources is not straightforward, as the OGC Planetary DWG bases the identifiers on NAIF identifiers for uniqueness and semantic reasons.

However, the absence of Celestial Reference Systems in the OGC registry is not necessarily a blocking issue. Clause 3.4 demonstrates how celestial reference systems can nevertheless be used, with CRS definitions in GML files referenced by chains of coordinate operations defined in other GML files. Formats that do not include full CRS definitions, such as Moving Features (MF) datasets, can nevertheless reference those definitions by their identifiers when the MF files are read in the larger context of reading the GML files. Therefor, having inertial CRS definitions in a authoritative registry is a convenience but is not mandatory, as the same CRSs can be defined outside any authoritative registry and still be found by the software.

3.5.2. General relativity

General Relativity may be handled with a spatiotemporal geometry modeled like terrain. It may be possible to leverage the techniques developed for the 3D geoid for 4D spacetime. However, at the time this report was written, the geoid data formats and tools are disparate. The first attempt to provide a unified format as an international standard is the Gridded Geodetic data eXchange Format (GGXF). Testbed-19 participants did not experiment with the possibility of using GGXF because the GGXF Standard is new and its support in software is still in development.

4

GEOPOSE

4 GEOPOSE

While the OGC GeoPose 1.0 Standard is described in terms of traditional Earth-based coordinate systems, adaptation to non-terrestrial use cases is straightforward. This is true even for distances, relative speeds, and accelerations where the absolute Galilean space and time of Newtonian mechanics does not apply. The key is that the two components of a pose can be generalized to support a positioned entity that is oriented to another entity within its frame of reference. Only the details of the transformation and orientation are sensitive to the details of the space, space plus, or space-time.

One of the goals in Testbed-19 (TB-19) is to outline a framework for extension of GeoPose 1.0 to a wide range of spaces and to test these extensions in both Euclidean space and absolute time, and in Minkowski flat space-time.

4.1. Use Cases Addressed in TB-19

Experiments in TB-19 tested the suitability of these extensions to support real-time construction of GeoPose graphs connecting multiple moving objects, each with one or more sensors capable of producing GeoPose output. These experiments were carried out under conditions where relativistic effects must be modelled in detail, demonstrating that the GeoPose concept is adaptable to many kinds of physical spaces and space-times.

By using a single core model with added optional properties, it is possible to create a default Basic GeoPose as a default. This default may be extended to fill the roles of the other targets of GeoPose 1.0 enabling the collapse of the composite serialization (Clause 2.20) forms into a single framework where conformance with the OGC GeoPose 1.0 Standard is still supported.

4.2. GeoPose Conceptual Framework

The meaning of the term “pose” in computer graphics refers to the position of an observer and its orientation to another entity. The non-technical or “display” concept of “pose” refers to the orientation of an observed entity to an observer. There is yet a third kind of pose that often occurs in computer vision — the position and orientation of an object as observed by an imaging sensor.

The common concept of a pose is dynamic. This concept reflects the experience of (1) moving to and standing at a more or less fixed position and visually scanning the surroundings until (2) fixing the gaze on a specific object. Interesting objects, perceived by the eye-brain visual system, are seen in detail in the center of the direction of the gaze. This naturally breaks the conceptualization into two parts — the position of the head and the direction of the visual gaze to which is attached the term “pose”.

Natural language use of the common pose can be confusing, since the “posing” is done by the subject of the gaze, rather than the person doing the gazing. The subject is positioned and oriented to optimize the view as observed from another position and a reciprocal orientation. Nevertheless, the technical application of the term “pose” has come to include (1) the position and orientation of an observing sensor whose gaze is directed in some direction, (2) the orientation with respect to the sensor of a subject entity, and (3) the position and orientation of an entity detected within the field of view of an observing sensor. To clarify the meaning, the first can be called an “observer pose,” the second an “observation pose,” and the third an “object pose.” An alternative term for “object pose” is “target pose.” The term “target” speaks to a possible future action with respect to the object and the more neutral “subject” is to be preferred. Likewise, an “observer pose” might be called a “sensor pose,” which is neutral enough to be a reasonable synonym. Object poses are important in visual object recognition while observer poses are common in connection with virtual cameras and lights in computer graphics scene graphs.

It is important to understand that these distinctions between applications of the pose concept are important for communication between humans but lie outside the scope of the data representation of a pose.

Summary

- *Display pose*: The position and orientation in which a person or other object adopts in order to be scanned, photographed, or painted by an observer.
- *Object pose*: The position and body orientation of an imaged or scanned object.
- *Observer pose*: The position and orientation of a sensor or source of illumination.
- *Observation pose*: The position and orientation from an Observer toward a scene or object of interest.

Representations of object, observations, and observer poses are used in the TB-19 GeoPose tasks. Display poses are not considered.

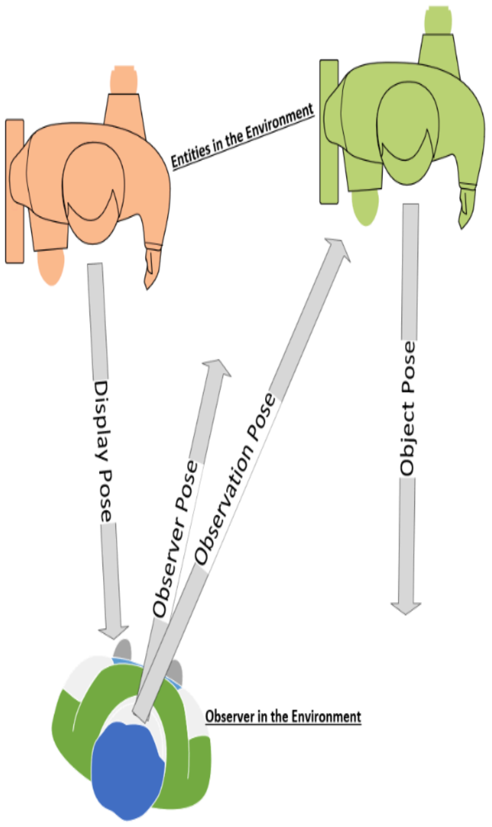
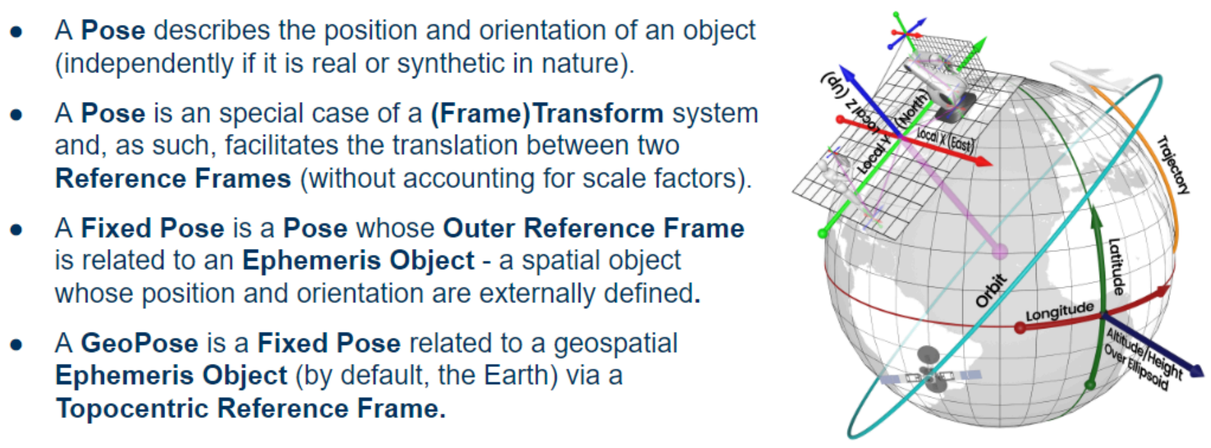


Figure 20 — Display, Observer, and Object Poses

The two concepts are closely related and both are important in spatial computing. An object pose represents the location and orientation of a subject or observation. An observer pose represents the position and orientation of an observer or imaging sensor.

4.3. GeoPose Architecture V 1.0

A pose is a special case of a (Reference Frame) Transform and, as such, facilitates the translation between two spatial or spatiotemporal reference frames. These are called the “outer” and “inner” frames by convention. A fixed pose is a pose whose outer reference frame is related to an Ephemeris Object — a spatial object whose position and orientation are externally defined. A GeoPose is a fixed pose related to a geospatial Ephemeris Object (by default, the Earth) via a topocentric reference frame. The property that makes a pose a GeoPose is that link to an external definition.



- A **Pose** describes the position and orientation of an object (independently if it is real or synthetic in nature).
- A **Pose** is an special case of a **(Frame)Transform** system and, as such, facilitates the translation between two **Reference Frames** (without accounting for scale factors).
- A **Fixed Pose** is a **Pose** whose **Outer Reference Frame** is related to an **Ephemeris Object** - a spatial object whose position and orientation are externally defined**.**
- A **GeoPose** is a **Fixed Pose** related to a geospatial **Ephemeris Object** (by default, the Earth) via a **Topocentric Reference Frame.**

Figure 21 — GeoPose Conceptual Model

4.4. GeoPose Suggestions from TB-18, Public Review, and Implementation

"NeoPose" is a temporary term applied to an experimental follow-on to GeoPose 1.0. Work on NeoPose has the goal of addressing seven GeoPose issues in the context of the Hillyfields Bubble.

4.4.1. Separate Model(s) and Encoding(s)

GeoPose 1.0 has both conceptual and logical models and a corresponding JSON encoding. This packaging came about primarily because the market forces pushing GeoPose standardization expected a JSON realization. The quickest path to a usable standard was to specifically target JSON, and to combine models and encoding in a single document.

Clarity of design and reusability of building blocks within the design is facilitated by separation of models and encoding. Derivation of one or more logical models from a conceptual model or at least a body of well-defined concepts also encourages compatibility and interoperability of the components. NeoPose prototypes a more nuanced approach to a logical implementation of GeoPose concepts and interfaces to supporting elements such as coordinate systems and coordinate reference systems defined in a way that facilitates using existing libraries for implementation of frame transformations.

4.4.2. Fully Defaulted Optional Properties

The extensions to GeoPose in this ER are described and implemented by adding new properties to the GeoPose conceptual model. These new properties potentially add to the size and complexity of GeoPose for the most common applications where a small amount of information is required. This size burden can be avoided by requiring that the corresponding standard specifies well defined default values for each of these new properties. If a computation or other use of a GeoPose requires a property that is not explicit in a serialization, that unique default value is used. Thus a value for each of these is always available, though not necessarily explicitly in a serialized form.

4.4.3. Uncertainty

For this experiment, a simplified approach to uncertainty was used.

Quantitative parameters were given an optional precision property in the same unit of measure as the parameter itself. This precision property is a symmetrical bound on the range of values within a 90% confidence limit.

The precision of the *result* of a coordinate transformation is also important. In some cases it is possible to quantify how imprecision in input parameters results in imprecision in the transformation results. This imprecision in the transformed coordinates is a combination of the transformation itself and imprecision that may be introduced by an approximate method, e.g., linear regression or approximation by series expansion (Clause 2.19). NeoPose does not consider this an important type of error.

Qualitative parameters were given a fractional belief property in the interval [0,1]. A value of 0 indicates no belief that the qualitative category is correct. A value of 1 indicates that the value is certainly correct and intermediate values indicate intermediate belief that the value is correct.

4.4.4. Time

Since the standardization of GeoPose 1.0 in 2022, the importance of time has become more obvious with experience since the standardization of GeoPose 1.0 in 2022. Especially for poses involving moving entities, time is a key element to synchronize poses derived from independent sensor systems. The concept is the simple one of a 1-dimensional time coordinate axis and the location of points — instants — along that axis. A temporal coordinate reference system is needed in order to establish the time coordinate axis, in line with OGC proposals for the modeling of time in OWL time. A time coordinate axis is a one-dimension coordinate system counting time from a starting instant, or epoch.

In TB-19, the time coordinate axes are either UNIX Time or GPS Time. The default value is 0. The default precision is 0.000000000001 second (one picosecond), representing 0.3 mm of light travel time in a vacuum. This value was chosen to be compatible with the precision of terrestrial surveying measurements.

4.4.5. Identity

In most applications, a GeoPose is a property of an entity in the physical world. Associating the entity and GeoPose can be accomplished by modeling the association as a property of an object representing the entity, as a property of the GeoPose, or as an independent bidirectional association. In the case where the association is a property of the GeoPose, there are additional considerations.

First, in GeoPoses representing sensed and detected entities, there can be uncertainty regarding both the existence of the entity and its globally unique identity. For example, a traffic camera may detect a car but initially not detect a license plate. Assigning the detected car a temporary identity with a short “time-to-live” solves the need for a locally unique identifier. This can be used for purposes such as tracking persistence of semantic type through time, as well as temporary hiding of part or all of a detected entity behind a closer entity. It is also useful in resolving changing classifications of a single physical entity through time.

The TB-19 solution was to add an optional unique id and an optional explicit time-to-expire property.

4.4.6. JSON Serialization

Extensions should support a JSON serialization that is as backwards compatible as possible with OGC GeoPose 1.0. Backwards compatibility still allows definition of a logical model that differs from GeoPose 1.0 as long as the JSON encoding still conforms to the JSON-Schema definitions.

4.4.7. Support for Operations

Implementation of the GeoPose Standard with reference frames outside the default 3D WGS-84 geodetic coordinates must make it easy to use existing libraries (e.g., PROJ) or to recognize the frames from parameters (e.g., ISO 19111) or IDs (e.g., EPSG numeric identifiers).

An updated GeoPose logical model should better support implementations of frame transformations defined by existing external standards. A simple serializable structure for Outer and Inner frames, as well as the specific transformation between them, would make it possible for implementers to encapsulate the details. This would better support the use of existing libraries as well as new implementations in a single structure. This framework for modular implementations would not expose the complexity of the multiple forks of some commonly used libraries.

4.5. NeoPose

NeoPose is only an experimental implementation of extensions of the OGC GeoPose 1.0. Any or all details may change if and when the extensions are considered for standardization. Many GeoPose and NeoPose concepts can be mapped to ISO 19111 concepts. A mapping is presented in Clause 4.7.

4.5.1. Architecture

The overall idea is that there is a first transformation that converts the coordinates of an event $e_O = \begin{pmatrix} t \\ x \\ y \\ z \end{pmatrix}$ in the outer frame to corresponding coordinates in an inner frame: $e_I = \begin{pmatrix} t' \\ x' \\ y' \\ z' \end{pmatrix} = T_{OI}\left(e_O\right)$

To be useful, the Outer frame should be attached directly, or via a sequence of transforms, in a linked GeoPose structure to an externally-defined coordinate reference system. This gives the event e_I in the Inner frame context in that external system of time and position. This makes the pose “Geo” if the external reference is tied to the Earth.

The second transform defines a ray cast from the origin of the inner frame. The second transform can take many forms but it is also a transformation of an event $e_R = \begin{pmatrix} t_R \\ x_R \\ y_R \\ z_R \end{pmatrix} = T_R\left(e', s\right)$

, where s is a real number parameter, e.g., distance along the ray in the length coordinates of the Inner frame. A ray may be defined in many ways, some more or less suitable depending on the application context. For example, the ray can be defined as a sequence of rotations, or as a quaternion, or as direction cosines, or as a point on the ray not at the origin of the inner frame CS, or as a rotation matrix, or as another transformation that defines a ray (or geodesic, to be more general) – such as a Lorentz transformation.

These transformations are free to assume default values for unspecified input coordinate values and to ignore, i.e., to project transformed coordinates. For example, time may be ignored, or a 1- or 2-dimensional space may be used.

To summarize, the core structures in the NeoPose areas follows.

- The PoseID which gives a pose a permanent unique ID.
- Pose links which are lists of predecessor and successor poses’ poseIDs.
- A frame transform, including specification of inner and outer frames, as well as a transformation with any needed parameters.
- A further transformation within the inner frame, often a rotation.

The second transformation represents the *orientation* element of the two-part pose paradigm: position plus orientation. When any pose — not only GeoPose — is represented by data, the orientation is defined by the parameters of a coordinate transformation that is often rotation or rotation-like, such as the Lorentz transformation. These parameters are in addition to the coordinates themselves, which are along a ray (or more generally, a geodesic) rooted at the origin of the inner frame coordinate system.

Both transformations take a time coordinate value and three spatial coordinate values (i.e., an event) to a transformed event. The first transform redefines an event in an external CRS with a description of that event in the inner CRS. It is the same event. The second transform provides the ray or geodesic, which is a path within the inner CRS. This is a set of events, parameterized by an independent variable such as distance that generates the ray or geodesic.

Many transformations come from families of transformations, which are identifiable by sets of controlling parameters. Examples include the WGS-84 coordinates in the first transform of each of the GeoPose 1.0 Basic Forms, the quaternion of the Basic-Quaternion form, and the yaw, pitch, and roll angles in the Basic-YPR form.

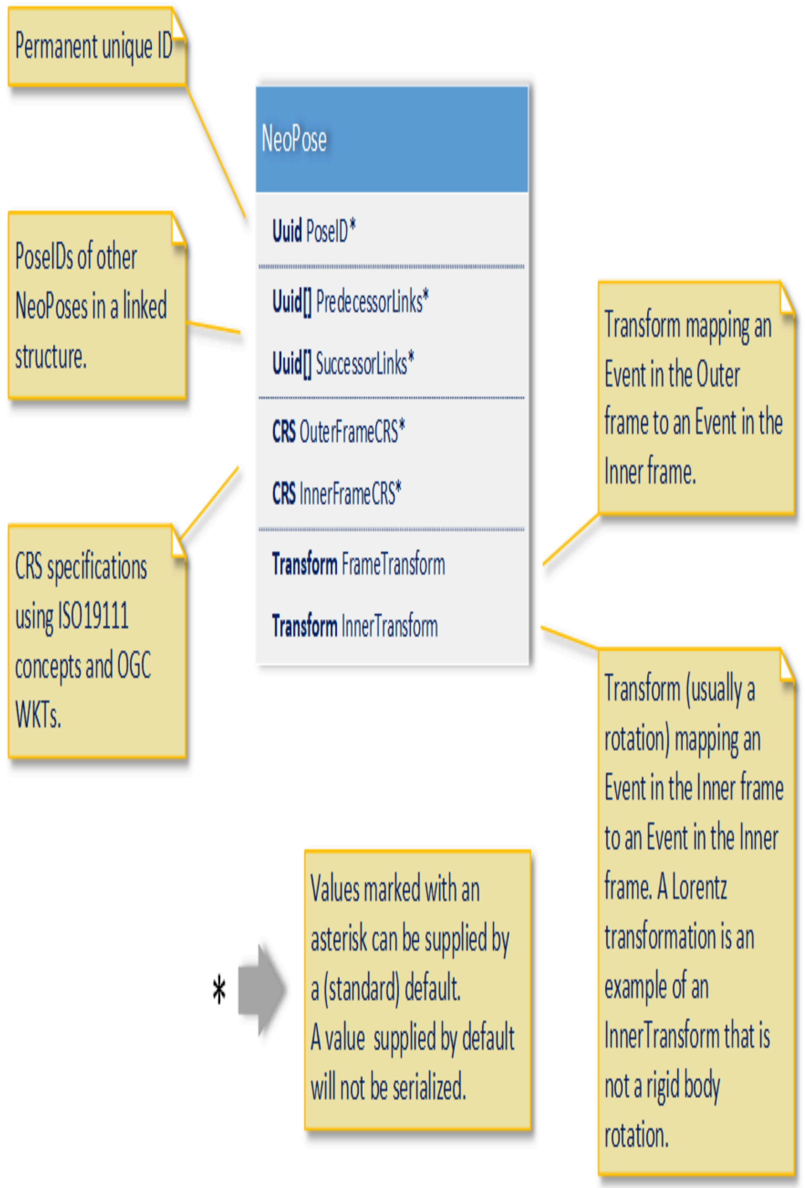

Permanent unique ID
NeoPose
Uuid PoseID*
Uuid[] PredecessorLinks*
Uuid[] SuccessorLinks*
CRS OuterFrameCRS*
CRS InnerFrameCRS*
Transform FrameTransform
Transform InnerTransform
PoseIDs of other NeoPoses in a linked structure.
CRS specifications using ISO19111 concepts and OGC WKTs.
Transform mapping an Event in the Outer frame to an Event in the Inner frame.
Transform (usually a rotation) mapping an Event in the Inner frame to an Event in the Inner frame. A Lorentz transformation is an example of an InnerTransform that is not a rigid body rotation.
*
Values marked with an asterisk can be supplied by a (standard) default. A value supplied by default will not be serialized.

Figure 22 — The NeoPose Core

4.5.1.1. Idiomatic and Templated Transforms

An **event** is a point in space-time. This point has three spatial and one temporal coordinate. Every NeoPose transform maps an event, that is, three space coordinates and one time coordinate, to another event. In many cases events may be projected, with one or more of the coordinates ignored in the transform.

The specification of transforms in a NeoPose can be of two types:

- idiomatic (identified by parameters); and
- templated (identified by an externally standardized scheme).

This division is the NeoPose strategy for dealing with the wide ranges of frame transform specifications across organizations and disciplines. An analogy to the issue of recognition and interpretation is recognition of a cat in an ordinary photo vs in a stylized symbol such as the toothy smile of the Cheshire Cat in Stevenson's *Alice in Wonderland*.

Idiomatic transforms: The key issue in implementing frame transforms is the existence of multiple *idioms* for specifying the controlling parameters. Some are derived from computer graphics, some from mathematics, some from geodesy and cartography, and some from navigation. Most of these idioms are broadly used only within a limited range of use cases. It can be awkward and confusing to force a "natural" expression of a transform using a paradigm from another field. The NeoPose approach is to accept that there are useful *idiomatic* expressions of transforms tied to specific disciplines or use cases.

Templated transforms: In other cases, it is normal to use an organizing scheme (*template*), such as the concepts of ISO 19111, OGC WKTs, by reference to EPSG identifiers, SEDRIS SRM, or NASA SPICE kernels. NeoPose represents such *templated* transform expressions with key-value pairs. The keys are arbitrary but unique, and the values are expressions according to one of the standardized organizing schemes. No attempt is made within the NeoPose representation to parse or interpret these structures. Instead, a form of dynamic or late binding is used. The transformations within the used scheme must itself be able to parse the expressions and use them to implement the described transformation.

There is a separation between transforms that are fully defined by the OGC GeoPose standard and transforms that follow some other system of definition and operational application. There is a mechanism to use *foreign* or external specifications of reference frames and transformations between them. There is also a mechanism to bring the transformations into the OGC standard by defining associated properties as a new idiomatic transform.

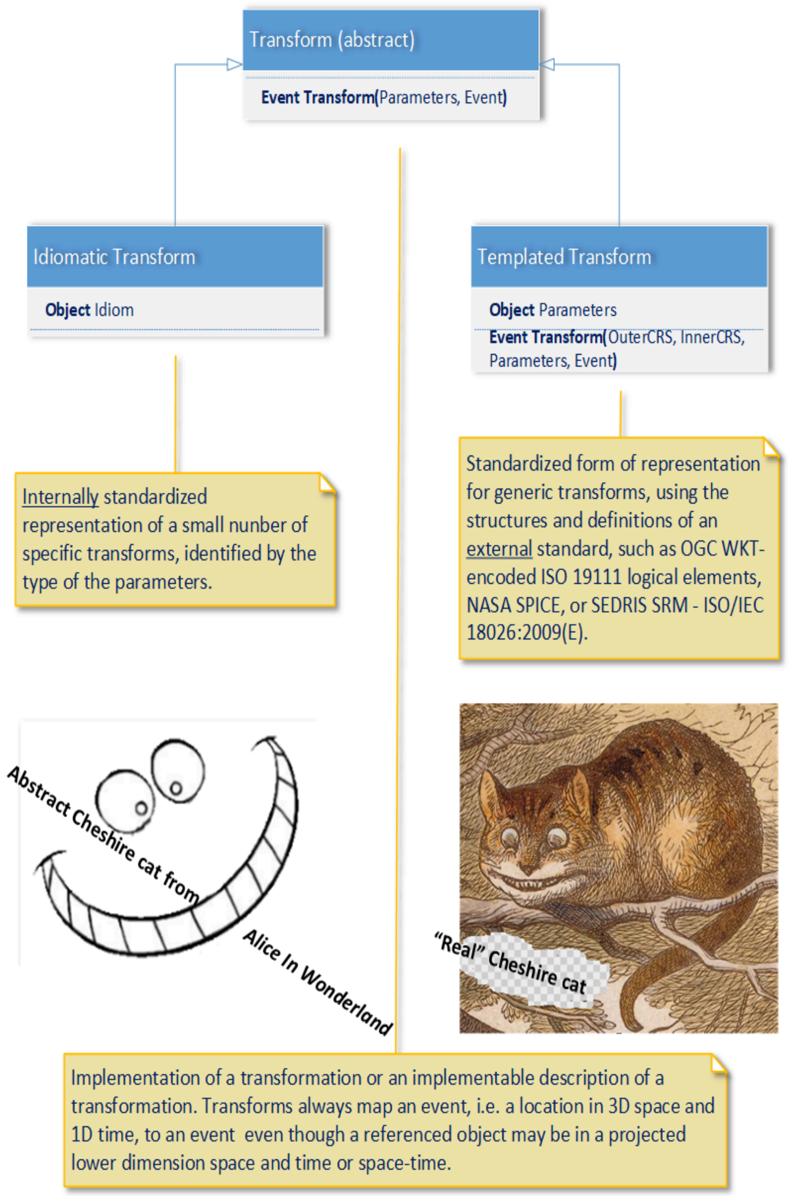

Transform (abstract)
Event Transform(Parameters, Event)
Idiomatic Transform
Object Idiom
Templated Transform
Object Parameters
Event Transform(OuterCRS, InnerCRS, Parameters, Event)
Internally standardized representation of a small nunber of specific transforms, identified by the type of the parameters.
Standardized form of representation for generic transforms, using the structures and definitions of an external standard, such as OGC WKT-encoded ISO 19111 logical elements, NASA SPICE, or SEDRIS SRM - ISO/IEC 18026:2009(E).
Abstract Cheshire cat from Alice In Wonderland
"Real" Cheshire cat
Implementation of a transformation or an implementable description of a transformation. Transforms always map an event, i.e. a location in 3D space and 1D time, to an event even though a referenced object may be in a projected lower dimension space and time or space-time.

Figure 23 — NeoPose Abstract Transform

4.5.1.2. Contracts, Early and Late Binding

There are several strategies for applications, protocols, or systems to achieve interoperability. In TB19, two were considered for GeoPose as follows. First, there may be requirements, (contracts between producers and consumers) that require a specific form to be used to specify transforms. If it is also possible that consumers can parse and interpret a range of forms, then code may be written that expects either specific idiomatic or specific templated frame transforms. This early binding is the most efficient and secure approach at run-time. Second, alternatively, a consumer could attempt to match a received data object against different possible supported forms. This late binding approach can be very flexible but can fail when an unexpected data object is received. Late binding may be supported by additional schema information, such as JSON-Schema definitions of acceptable serialized data objects.

NOTE The above definition of “early-binding” and “late-binding” differ from EPSG definitions, which are about whether the parameters of a coordinate transformation are embedded in the source CRS, or provided in a registry as potentially many transformations associated to a (source CRS, target CRS) pair.

4.5.1.3. Concrete Forms

The concrete idiomatic transforms implemented for TB-19 are shown in the following figure. These include support for the GeoPose 1.0 Basic-YPR and Basic-Quaternion forms as well as a newly proposed “Local” form pair similar to the Basic forms except that the outer frame is a Cartesian 3D coordinate system with lengths in meters and time in picoseconds.

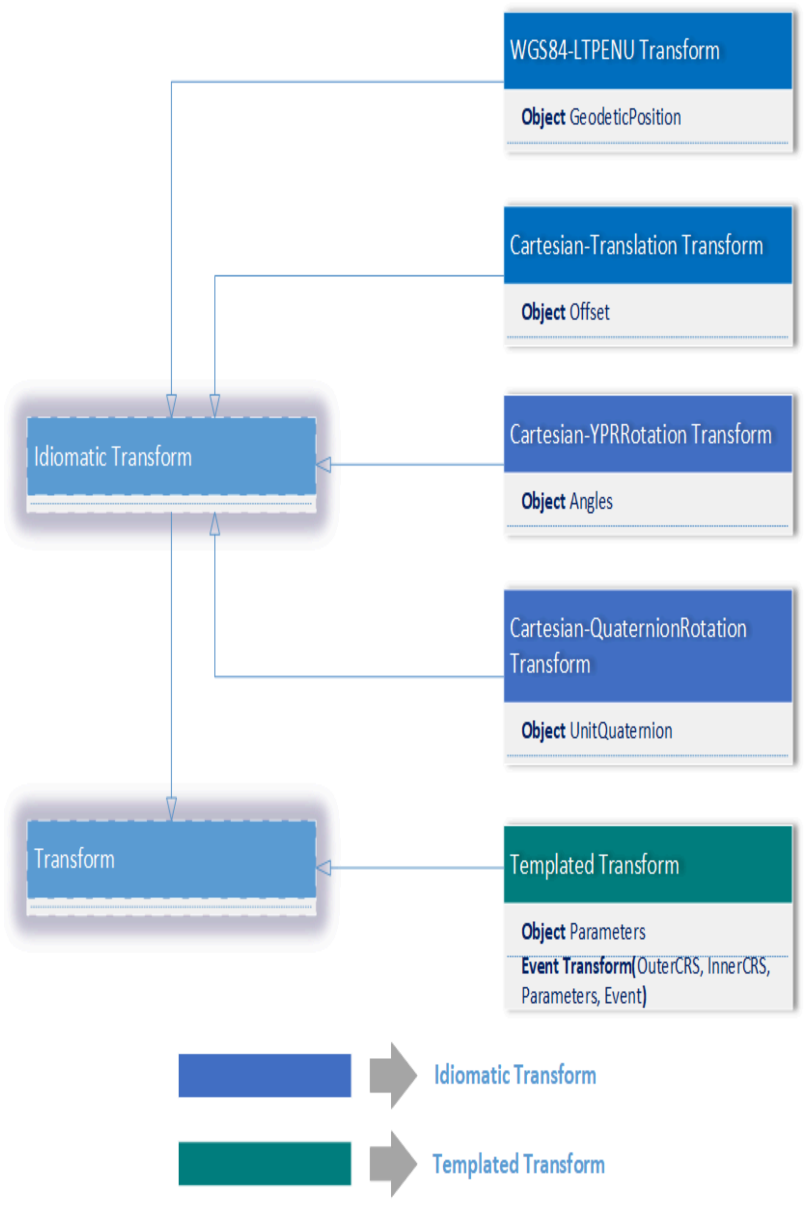
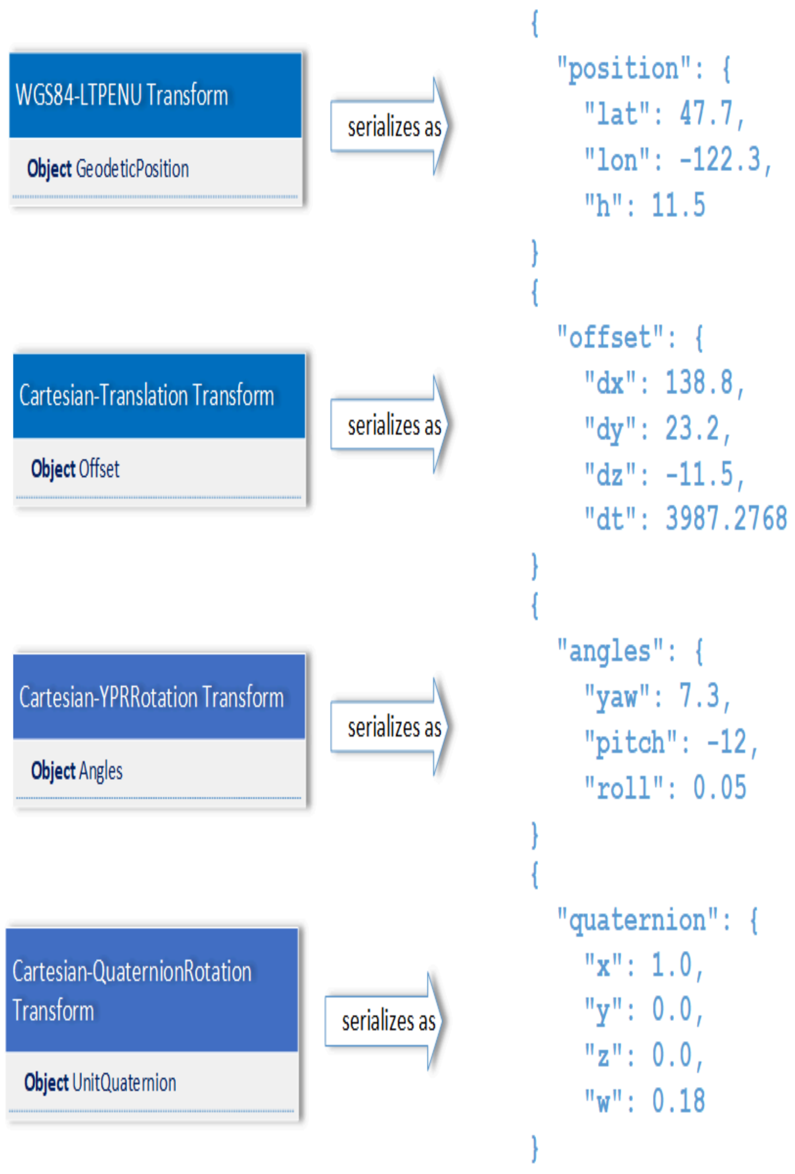


Figure 24 — NeoPose Concrete Transforms

The “GeodeticPosition” object in “WGS84-LTPENU Transform” in the above figure is the “position” object from the GeoPose 1.0 Standard Basic forms. The “YPRRotation” and “QuaternionRotation” objects are the corresponding “angles” and “quaternion” objects. These are examples of the selection of specific transforms from families using transform-specific selection parameters. The following figure has examples of the serialization of the selection parameters.

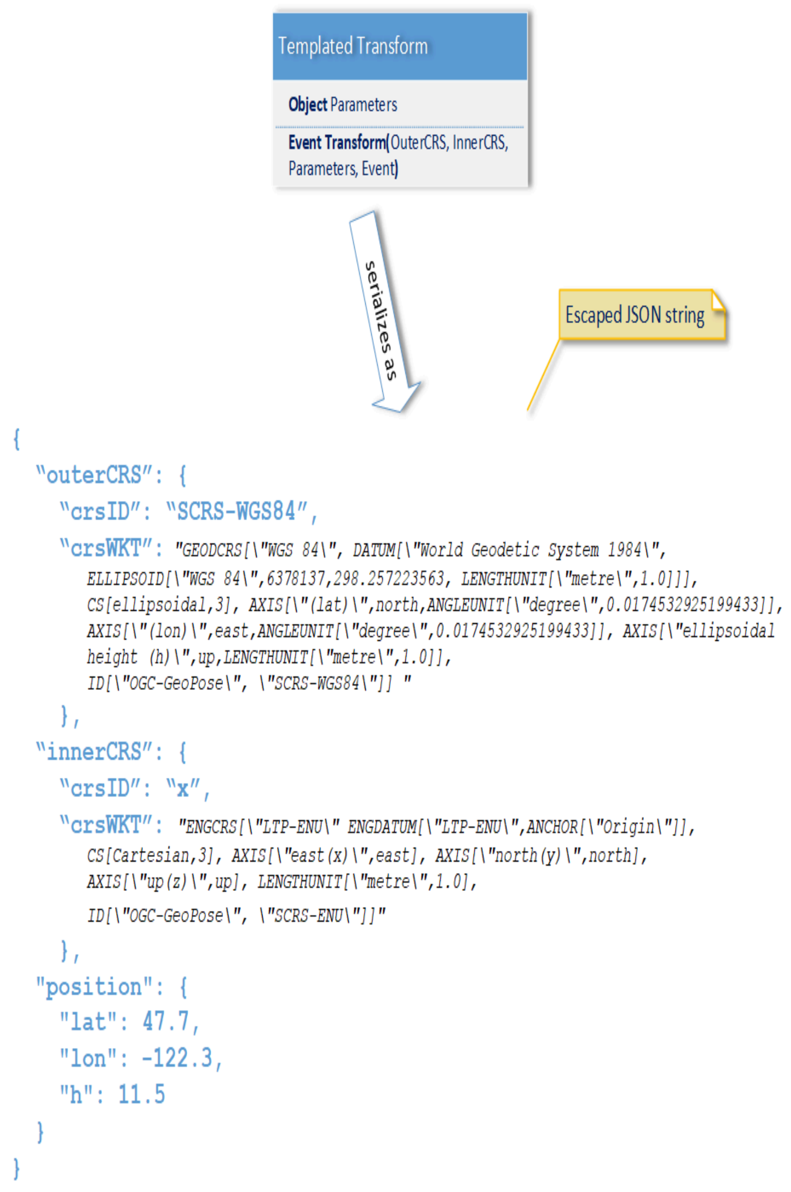
The following figure shows some examples of serialized forms showing the parameters selecting specific transforms from several idioms from GeoPose 1.0 as used in TB-19. Again, the serialized parameters are not input to the implemented event transforms but rather are the method of selection of specific member transforms sharing an idiom.



```
{
  "position": {
    "lat": 47.7,
    "lon": -122.3,
    "h": 11.5
}
{
  "offset": {
    "dx": 138.8,
    "dy": 23.2,
    "dz": -11.5,
    "dt": 3987.2768
}
{
  "angles": {
    "yaw": 7.3,
    "pitch": -12,
    "roll": 0.05
}
{
  "quaternion": {
    "x": 1.0,
    "y": 0.0,
    "z": 0.0,
    "w": 0.18
}
```

Figure 25 — Serialized NeoPose Idiomatic Transforms

Although templated transforms could be implemented with serialization conforming to GeoPose 1.0, this was not done for TB-19. The following is an implementation of the WGS84 to LTP-ENU transform previously shown as an idiomatic transform. Every idiomatic transform can be serialized as a templated transform. Identifying a serialized templated transform as an idiomatic transform is a difficult task and not generally useful. Using the templated form *is* a reasonable means of documenting idiomatic forms.



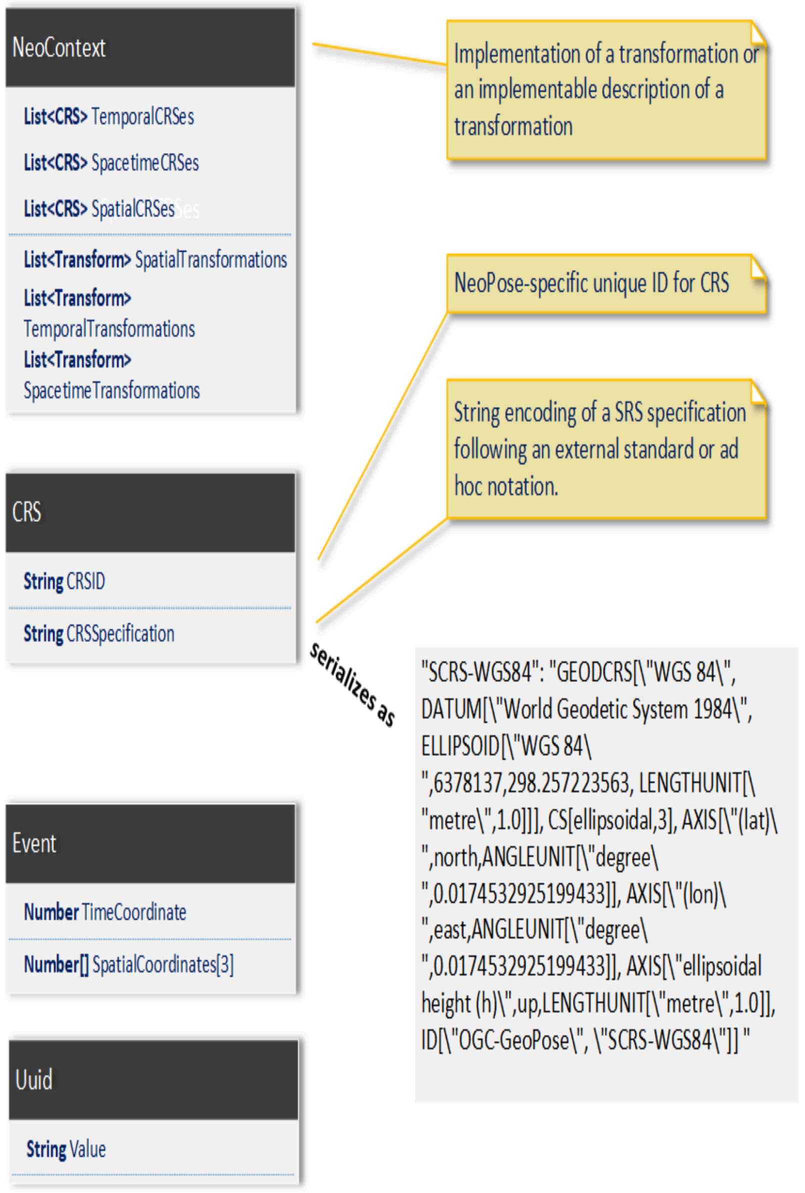
```
{
  "outerCRS": {
    "crsID": "SCRS-WGS84",
    "crsWKT": "GEODCRS[\"WGS 84\", DATUM[\"World Geodetic System 1984\",
      ELLIPSOID[\"WGS 84\",6378137,298.257223563, LENGTHUNIT[\"metre\",1.0]]],
      CS[ellipsoidal,3], AXIS[\"(lat)\",north,ANGLEUNIT[\"degree\",0.0174532925199433]],
      AXIS[\"(lon)\",east,ANGLEUNIT[\"degree\",0.0174532925199433]], AXIS[\"ellipsoidal
      height (h)\",up,LENGTHUNIT[\"metre\",1.0]],
      ID[\"OGC-GeoPose\", \"SCRS-WGS84\"]] "
    },
  "innerCRS": {
    "crsID": "x",
    "crsWKT": "ENGCRS[\"LTP-ENU\" ENGDATUM[\"LTP-ENU\",ANCHOR[\"Origin\"]],
      CS[Cartesian,3], AXIS[\"east(x)\",east], AXIS[\"north(y)\",north],
      AXIS[\"up(z)\",up], LENGTHUNIT[\"metre\",1.0],

      ID[\"OGC-GeoPose\", \"SCRS-ENU\"]]"
    },
  "position": {
    "lat": 47.7,
    "lon": -122.3,
    "h": 11.5
  }
}
```

Figure 26 — NeoPose Templated Transforms

4.5.1.4. NeoPose Support Datatypes

The following are four datatypes that are used in the NeoPose implementation.


NeoContext
List<CRS> TemporalCRSes
List<CRS> SpacetimeCRSes
List<CRS> SpatialCRSes
List<Transform> SpatialTransformations
List<Transform> TemporalTransformations
List<Transform> SpacetimeTransformations
Implementation of a transformation or an implementable description of a transformation
NeoPose-specific unique ID for CRS
String encoding of a SRS specification following an external standard or ad hoc notation.
CRS
String CRSID
String CRSSpecification
serializes as
Event
Number TimeCoordinate
Number[] SpatialCoordinates[3]
Uuid
String Value

```
"SCRS-WGS84": "GEODCRS[\"WGS 84\",
DATUM[\"World Geodetic System 1984\",
ELLIPSOID[\"WGS 84\
",6378137,298.257223563, LENGTHUNIT[\
"metre\",1.0]]], CS[ellipsoidal,3], AXIS[\"(lat)\
",north,ANGLEUNIT[\"degree\
",0.0174532925199433]], AXIS[\"(lon)\
",east,ANGLEUNIT[\"degree\
",0.0174532925199433]], AXIS[\"ellipsoidal
height (h)\",up,LENGTHUNIT[\"metre\",1.0]],
ID[\"OGC-GeoPose\", \"SCRS-WGS84\"]] "
```

Figure 27 — NeoPose Datatypes

The property NeoContext is a convenience element supporting the use of long text strings, such as OGC WKTs defining coordinate reference systems or transformation. The object holds key-value pairs where a short key (local to and defined by an application, system, or protocol) can be used to refer to a long, complex string. For TB-19, a table of 14 entries were implemented as a WKT JSON “virtual registry”.

4.5.1.5. Implementations of Predefined Transformations

Four event coordinate reference systems are implemented. The first two are Galilean combinations of space and time. The third is a flat Minkowski space-time and the fourth is a flat Minkowski space-time with a light speed artificially reduced by a factor of 100 000 to exaggerate relativistic effects.

Event Coordinate Reference Systems (ECRS) as combined TCRS and SCRS:

Table 7 — Predefined OGC WKTs Defining CRSs and Coordinate Operations Used in the Hillyfields Bubble

ID	Description	Source
OGC-GeoPose.SCRS-WGS84	WGS 84	SCRS-WGS84
OGC-GeoPose.SCRS-ENU	LTP-ENU	SCRS-ENU
OGC-GeoPose.SCRS-C3	Local	SCRS-C3
OGC-GeoPose.TCRS-UNIX-T	UNIX Time	TCRS-UNIX-T
OGC-GeoPose.TCRS-SGPS-T	GPS Scaled Time	TCRS-SGPS-T
OGC-GeoPose.CCRS-C3G-SPT	Local Cartesian plus GPS Time	CCRS-C3G-SPT
OGC-GeoPose.CCRS-C3SG-ST	Minkowski Spacetime	CCRS-C3SG-ST
EPSG.15594	EPSG topocentric example A	15594
OGC-GeoPose.T-USG	UNIX Time to scaled GPS Time	T-USG
OGC-GeoPose.T-SGU	Scaled GPS Time to UNIX Time	T-SGU
OGC-GeoPose.T-C3T	Local Cartesian to Local Cartesian via Translation	T-C3T
OGC-GeoPose.T-C3YPRR	Local Cartesian to Local Cartesian via YPR Rotation	T-C3YPRR
OGC-GeoPose.T-C3QR	Local Cartesian to Local Cartesian via quaternion Rotation	T-C3QR
OGC-GeoPose.T-ML	Minkowski Spacetime to Spacetime via Lorentz Transformation	T-ML

NOTE: These WKT definitions and the corresponding JSON "virtual registry" were developed for experimental use during Testbed-19. They may be useful as a starting point for consideration of normative text. However, they have not been reviewed for correctness and conformance to relevant standards. They have the following known limitations:

- *"Minkowski Spacetime"* actually provides a Galilean definition: a compound CRS with time dimension separated from spatial dimensions, each in their own independent coordinate system and without multiplying the time coordinate by the speed of light.

A standard definition should rather group the four dimensions together in a single Minkowski coordinate system (Clause 3.2.1.4).

- `COORDINATEOPERATION` in items like *"UNIX Time to scaled GPS Time"* and *"Local Cartesian to Local Cartesian via Translation"* are missing the `SOURCECRS` and `TARGETCRS` elements.
- `METHOD` names `"LIGO"`, `"Translation"`, `"Angular rotations"`, `"Quaternion rotation"` and `"Lorentz transformation"` are not yet accompanied by a description of their formulas and parameters (e.g., Annex B.2).

The above deviations exist for reasons of compatibility with existing libraries. For example, the introduction of a Minkowski coordinate system or new operation methods require software modifications, which were experimented only in projects where developers were among the Testbed-19 participants.

Transformation code can either validate parameters before calculation or use runtime error handling (e.g., exceptions) to handle errors during calculations. The NeoPose model defines value for non-present NeoPose properties. This supports simple NeoPoses (Basic-YPR, Basic-Quaternion, Local-YPR, and Local-Quaternion) with explicit specification of event coordinate and transformations.

For example, in the Basic-Quaternion form,

```
{
  "position": {
    "lat": 47.7,
    "lon": -122.3,
    "h": 11.5
  },
  "quaternion": {
    "x": 0.2303923729814588,
    "y": -0.03749706362805052,
    "z": 0.15620176329098345,
    "w": -0.9597470155390087
  }
}
```

Figure 28 — Example of a coordinate tuple in quaternion form

The outer event CRS has no explicit time value and expects to see a "location" object with WGS84 lat, lon, and h values. The inner event CRS also has no time value and 3 Cartesian LTP-ENU coordinates. The frame transformation is the WGS-84 to LTP-ENU transform. The non-present values are given an explicit default value of 0, and the transformation code must accept a default value as valid.

In general, NeoPose depends on run-time type determinations, i.e., late or dynamic binding. To the extent possible, an implementation must correctly interpret any possible parameters.

The inner frame or "rotation" transformation also must make a dynamic determination of the type of the parameter object and proceed according to the YPR angle type or quaternion type. Dynamic typing makes it possible to add additional types (e.g., swing and twist or look-at) but the burden is on the transform to deal with them.

Serialization in JSON for the Basic NeoPose is identical to GeoPose 1.0. The Advanced form can also be supported but is effectively superseded by the NeoPose extensions.

4.5.2. Implementing Selected Suggestions

The extensions developed in TB-19 are intended to adopt some of the suggestions from TB-18 and to demonstrate that these changes also provide support for GeoPose in non-Euclidean spaces. The name “NeoPose” is used for the purpose of distinguishing the extended GeoPose from the version specified in the OGC 1.0 Standard.

In possible future work, some or all of the NeoPose extensions may also be standardized.

4.5.2.1. Broadening the Formalism

The first step is to refocus on the frame transform as the core concept. To be more general, the transformation in NeoPose is from an *event* in an Outer frame to an *event* in an Inner frame. An event is a unique point in space and time in a single reference frame, independent of the CRS describing that frame.

4.5.2.2. Support External Coordinate Reference Systems

NeoPose support for spatial and temporal coordinate reference systems is via a flexible framework that allows different styles of specification matching different defining organizations.

4.5.2.3. Support External Development of Reference Frame Transformations

One of the limitations of GeoPose 1.0 was the lack of guidance on the definition and implementation of frame transformations.

4.5.2.4. GeoPose in Galilean and Minkowski Spacetimes

How can GeoPose be extended to work in non-Euclidean spaces? What about the class of space-time coordinates with three spatial coordinates and a temporal coordinate identifying events? In terms of the representation of a space using four coordinates, there is an obvious extension to simply considering the GeoPose to support the transformation of event coordinates from one reference frame to another. If the scope is restricted to flat spacetime with a Minkowski metric, then it is precisely the details of the transformations that must be specified. This approach would apply with some additional extensions to curved spacetimes and other metrics. Again, only the details of the transformations are sensitive to the metric structure of spacetime.

In the following sections, only the extension of the transformation to Minkowski space are considered. This is an important first step but eliminates cases where there may be multiple geodesics connecting a sensor with events.

4.5.2.4.1. Coordinates and Sensor Reality

When the speed of causality, or message transmission, or particle motion is finite, the perception of reality becomes ambiguous. By reality, the state of something at an instant of time might be meant. But in a Minkowski spacetime, such a reality can only agree with a static set of coordinates within a single frame of reference (a "rest" frame).

Colloquially, one could say that in a Minkowski spacetime, one cannot believe one's eyes if more than one inertial frame is observed. This has led to some erroneous descriptions of the observable effects of special relativity, such as the visualization of a length contraction or time dilation. For example, one might observe an object in another inertial frame, moving with a large relative velocity. If the coordinates in the observed frame are computed in the observer's frame using the Lorentz transformation, and those coordinates are visualized, that visualization will show a shortening in the direction of relative motion. As a visualization, this would be inaccurate, because the true view using photons scattered off an object in the observed frame would be distorted in the opposite direction by the finite speed of those photons. There is no truly correct view, but only (1) views at an instant in the observer frame of an object in the observed frame and (2) the coordinates of events in the observed frame transformed to the observer's frame. In practice, knowledge of another frame will come from sensor measurements dependent on motion of something (e.g., photons) from the observed frame to the observer frame.

In other words, event coordinates in an observed frame are never observed. This means that the useful transformations in a GeoPose extension to Minkowski space must be of type (1) — based on paths between events in the observed frame to the observer frame all arriving at the same proper time in the observer frame. This is a key result from this examination of the extension of GeoPose to non-Euclidean frames: the orientation is specified in the direction of a geodesic connecting two events. In Minkowski space-time, i.e., flat spacetime, geodesics are straight lines and allow for a linear parametrization in terms of a single independent variable. The work reported in this Engineering Report centers on two examples of such visualization.

The first is the visualization of multiple objects in different frames moving (at least temporarily) in uniform relative motion at a large fraction of the speed of light. The second is the determination of the direction to precisely point an image in low earth orbit at a moving vehicle on the earth's surface. The technique for computing the results in both cases is relativistic ray tracing.

4.5.2.4.2. Extended Bodies

The finite speed of light has consequences even within a single extended physical object or even multiple objects at rest in the same frame because time can be measured at any point within the frame, but communication of time within the frame can only be done, at most, at light speed.

This has consequences for GeoPose since the pose is based on the origin of the Inner reference frame. Time synchronization with GeoPoses anchored at a different location within the same Inner frame can be achieved by calculating the time to travel the Euclidean distance between the origins of any pair of GeoPoses within the same Inner frame.

4.5.2.4.3. Clock Synchronization and Coordination

Within a rest frame, clocks can be synchronized by exchanging messages via light or radio using Einstein clock synchronization. But time can only be synchronized at a single shared event in flat spacetime. The only unambiguous temporal relationships are between events in the same frame. The important point for coordinate calculations is that every spatial location within a specific frame has an associated clock with a time offset from every other point in the frame.

If there is an event shared by the two frames, it is possible to coordinate clocks in two frames in uniform relative motion. For example, the paths of the spatial origins of two frames may coincide at some instant. At that time, clocks at the origins can be set to a known value, e.g., 0. From that event, the time kept by each clock may be Lorentz transformed to the other frame. In TB-19, this idea is used to coordinate the clocks of the various moving frames by a two-step process. First, a spatial point on one frame is identified as the location of the origin of the other frame. The light travel time within the first frame is used to set the clock at the selected point in the first frame according to the clock at the origin of the first frame. The time at that point common to both frames at the instant of coincidence is used to set the clock to a time coordinated with the first frame.

4.5.2.4.4. Effects of Relative Motion

The core concept of the GeoPose is the transformation between Outer and Inner frames of reference. While in Euclidean 3D space, the transformation is static, in relativistic space-time the transformation is dependent on both the geometric structure of that space-time and the relative motion between Inner and Outer frames.

Separation of effects of uniform relative motion, both relativistic and classical, characterizes several named effects. These effects are notable and under the right condition perceptible only because the effects conflict with the naive human metal model of space and time as absolute, and the associated vision based on instantaneous travel of photons from source to observer. Since poses represent the essential spatiotemporal relationships, poses must be computed in accordance with the finite speed of light. Given proper implementation of the frame transformation, the extended NeoPose accounts for all the relativistic and non-relativistic aspects of a 4D space-time.

Non-relativistic effects include the Classical Doppler Effect, Aberration, and Light Travel Time.

4.5.2.4.5. Classical Doppler Effect

A repetitive signal from a source in relative motion will be shifted in frequency due to the changing travel distance between repetitions of the event. In the classical Doppler effect, the signal is transmitted via a medium and the relative velocity of the medium is responsible for the frequency shift: $f' = f\left(\frac{c \pm v^{O}}{c \mp v}\right)$

Note that the Doppler Effect, either classic or relativistic, has no direct bearing on the orientation of a pose. But if the principle of pose representation is that the orientation is in the direction of sensed interactions between source and observer, then in the presence of a medium, propagation speed in the medium, and motion of the medium must be accounted for. These considerations are outside the scope of experiments in TB-19.

4.5.2.4.6. Aberration

Aberration is the effect of relative motion on the angle at which a photon approaches an observer. Since the measurement of the direction from which a photon arrives determines the apparent position of a source, aberration causes an apparent shift in position. While this effect can be modelled in isolation, a more general approach is to focus on the determination of photon paths between source and observer.

4.5.2.4.7. Light Travel Time

Again, this is considered an effect because of the difference in the observation of frames either at rest or in relative motion where the speeds or distances make the finite travel time of a photon from source to observer significant. Fully modeling the geodesic paths of photons will always produce the correct spatiotemporal relationships.

4.5.2.4.8. Relativistic Effects

There is a difference between how things are modeled or measured, and how things appear visually (or more generally via remote observation, for example, by scattering photons or other particles on them). The results of synchronous measurements and how extended objects appear visually depend on independent definitions and physics.

One definition of what is real, from the point of view of one inertial reference frame when observing objects in another frame in uniform relative motion, are the transformed coordinates of physical objects in the observed frame at a single time at the origin of the frame of the observer.

Another is the projection of arriving photons scattered from physical objects in the observed frame. In the first case, calculation of the transformed distances would show a length contraction in the direction of relative motion. In the second, purely visual effects such as Terrell-Penrose rotation would reflect the paths of the arriving photons, giving the impression of a rotation. Both views of reality are valid and it is only necessary to precisely define the nature of the observation.

While length contraction can be calculated, it is not visually observable because it is compensated for by the constant finite speed of light. As a practical matter, with current technology, only simulations of relativistic visual effects are possible.

Relativistic considerations appear in both the task of aiming an imaging sensor from low earth orbit accurately at the Street Drone and in animated rendering of the environment and various objects moving during the data capture in the Hillyfields Bubble. The first depends on straightforward determination of the apparent rotation of coordinate axes via the Lorentz Transformation. The second is a visual effect that depends on the Lorentz-transformed coordinates of light scattering points in the observed frame at events at the corresponding light travel time in the past. The photons across the image arrive simultaneously in the observer frame but departed at different times from points in the observed frame.

The following sections consider the cause and potential impact of specific relativistic effects on spatiotemporal representations in general and (Geo)poses specifically. These effects are human descriptions of real or potentially real -experiences- and relate to either visual effects, i.e., scattered photons arriving at a sensor or other causal interactions between events. In the final analysis, all these effects can be handled uniformly with ray tracing along geodesics.

4.5.2.4.9. Length (Lorentz) Contraction

The contraction of an object as calculated for its coordinates from a frame in uniform relative motion can be computed from the point of view of an event in an observer's frame. This effect is not directly observable by sensors because finite light travel time counteracts the visual (i.e., sensing of photons at an instant by a sensor) observation of the effect.

4.5.2.4.10. Time Dilation

The apparent slowing of time for an object observed from a frame in uniform relative motion can be computed from the point of view of an event in an observer's frame. This effect is also not directly observable by sensors because finite light travel time counteracts the visual (i.e., sensing of photons at an instant by a sensor) observation of the effect.

4.5.2.4.11. Relativistic Doppler Effect

In the relativistic Doppler effect, the relative velocity of the source of the signal: $f_o = f\sqrt{\frac{1-\beta}{1+\beta}}$, where $\beta = \{\frac{v}{c}\}$, v is the speed of the signal source, c is the speed of signal propagation through the medium, and v is the speed of the source through the medium.

4.5.2.4.12. Pointing along geodesics

What does it mean to point at something in a frame in relative motion? One natural answer is to use paths of least or greatest action between a source event and an observer event. In cases where relativistic effects are important, e.g., where $\beta > 0.9$, this is a geodesic (Clause 2.9) connecting the two events. The direction from the source to the observer is then the 3D space geometrical direction of the geodesic as it leaves the source. Likewise, the direction from the observer to the source is the 3D space geometrical direction of the geodesic at the observer event.

4.6. Implications and Recommendations for the OGC GeoPose Standard

The most important result is the development of an experimental framework for gaining experience with spatial, temporal, and event reference frames. This framework supports computational modules for performing transformations between reference frames while preserving backwards compatibility with the OGC GeoPose 1.0 Basic GeoPose concept.

If one selects the view of simultaneously arriving parcels of information, e.g., photons projected back along geodesics to earlier times, then the rotational inner frame transform of the GeoPose is aligned with the direction of one of the geodesics. In Galilean relativity, this is the Euclidean shortest distance. In Minkowski space-time, this is a straight line, and in curved space-time, this is generally a curved line. In all cases, the direction is the direction of a geodesic, and the computational details may be encapsulated in the frame transformation. The computations involve projecting a ray toward the observed object, transforming the ray to the frame of the observed object with a Lorentz transformation, and the intersection of the ray with the observed object. All the expected relativistic effects will be supported.

4.7. Relationship with ISO 19111

Clause 4.5.1.1 distinguishes two types of transforms: idiomatic and templated. Templated transforms can be seen as the most general case, and idiomatic transforms as shortcuts for commonly used transforms in some specific disciplines. ISO 19111 supports the general case but does not formally distinguish the idiomatic case through class hierarchy. However, the fact that a coordinate operation is an idiomatic case can still be expressed, at least at the destination of human readers, by providing an explanatory text in the `CoordinateOperation.domain.scope` property. The following table maps NeoPose concepts from Figure 23 to ISO 19111 classes (Figure 13):

Table 8 — Partial mapping from NeoPose to ISO 19111

NEOPOSE CLASS	ISO 19111 CLASS
`IdiomaticTransform`	`CoordinateOperation` with explanation in `domain.scope`
`TemplatedTransform`	`CoordinateOperation`
`TemplatedTransform.outerCRS`	`CoordinateOperation.sourceCRS`
`TemplatedTransform.innerCRS`	`CoordinateOperation.targetCRS`

NEOPOSE CLASS	ISO 19111 CLASS
`TemplatedTransform.Parameters`	`SingleOperation.parameterValue`
`Event`	`DirectPosition`
`TemplatedTransform.transform(…, Event)`	`CoordinateOperation.transform(CoordinateSet)`

NeoPose idiomatic transforms omit the `sourceCRS` and `targetCRS` properties, which contribute to making them shorter. The same could be done in ISO 19111 model since those parameters are optional and sometime inferred from the context. The context could be that, when the ISO 19111 `OperationMethod` identifies a NeoPose idiomatic transform, then *by definition* the source and target CRS are fixed by the operation method formula. While technically possible, it may not be recommendable in ISO 19111 spirit.

4.7.1. Class hierarchy

One way to compare NeoPose and ISO 19111 models may be that NeoPose sub-typing of `Transform` (with the idiomatic and templated transform subclasses — Figure 23) addresses encoding brevity aspects. ISO 19111 sub-typing of `CoordinateOperation` (with conversion and transformation subclasses among others — Figure 5) addresses more conceptual aspects about the nature of coordinate operations (e.g., whether the parameter values are "by definition" or stochastic). The representation of a transform in ISO 19111 model provides rich metadata but without trying to be compact. Conversely, the representation of the same transform in NeoPose model is more compact but loses some metadata.

4.7.2. Transform parameters encoding

NeoPose uses parameter names directly in a JSON serialization. For example in Figure 25, the "Cartesian-Translation Transform" uses "dx," "dy," "dz," and "dt" parameter labels, while the "Cartesian-YPRRotation Transform" uses "yaw," "pitch," and "roll" parameter labels. By comparison, ISO 19111 parameters are also named, but in an indirect way. Instead of:

```
{
    "lat": 47.7,
    "lon": -122.3,
}
```

Figure 29 — Snippet of NeoPose idiomatic transform parameters

The ISO equivalent (for illustration only, as there is no OGC Standard yet for JSON encoding of ISO 19111):

```
{
    "parameterValue": {
        "parameter": "lat",
        "value": 47.7
    },
    "parameterValue": {
        "parameter": "lon",
        "value": -122.3
    }
}
```

Figure 30 — Snippet of ISO 19111 equivalence (non-standard)

The description, unit of measurement, range of values, *etc.* of “lat” and “lon” parameters are defined in a separated class. That separated class can be seen as describing the template. This topic was covered in more detail, together with examples of GeoPose transform definitions, in OGC 22-038r2 annex A.3.1.

4.7.3. Transforming positions or events

NeoPose provides the following method (Figure 23):

```
TemplatedTransform.transform(outerCRS : CRS, innerCRS : CRS, Parameters,
Event) : Event
```

Figure 31 — NeoPose transform method

By contrast, ISO 19111 separates the process into the following two steps (Clause 3.3.3).

```
RegisterOperations.findCoordinateOperations(source : CRS, target : CRS) : Set
<CoordinateOperation>

CoordinateOperation.transform(points : CoordinateSet) : CoordinateSet
```

Figure 32 — ISO 19111 transform method

The first step (`findCoordinateOperations`) is typically much more costly than the second step (`transform`), because it may require a search for all coordinate operations registered in a database for the given pair of source and target CRS. By contrast, the second step may be as simple as subtraction. Because the first step typically needs to be executed only once before to transform thousands of points, there is a big performance advantage in separating those two steps. It allows executing the costly part outside a loop, then to loop only on the cheapest part. Another reason for this separation is that `findCoordinateOperations` may return many transforms, so the user needs to choose one before to moving to the next step.

However, the performance argument is to be nuanced. The NeoPose `transform` method expects a `Parameters` argument, which is absent from the ISO 19111 methods because those parameters are to be found by `findCoordinateOperations` in the database. Providing the parameters may reduce the needs for NeoPose to perform database queries. An inconvenience is that it puts the burden on users for finding the parameter values. An advantage is that it is easier for users to change those parameter values if desired.

4.7.4. Contextual registry

NeoPose defines a `NeoContext` class used as a contextual registry (Clause 4.5.1.4). This registry associates CRS definitions to identifiers using *key-value* pairs in a file. This file must contain all CRSs referenced by NeoPose transforms (Table 7).

By contrast, the prototype based on the ISO 19111 model takes a different approach (Clause 3.4.1 and Annex B.3). It also has a contextual registry but built automatically in memory. There is no file listing explicitly all *key-value* pairs. The prototype takes advantage of the fact that ISO 19111 definitions of `CoordinateOperation` can contain full CRS definitions, even if indirectly through `xlink:href` attributes. The Apache SIS library used in the prototype automatically builds an in-memory virtual registry populated with all CRS definitions encountered during GML parsing and uses that registry for resolving CRS identifiers in Moving Feature files referenced by the `CoordinateOperation`. However, this strategy is an implementation choice. The OGC/ISO standards make no recommendation.

4.7.5. NeoPose features not covered by ISO 19111

Both NeoPose and ISO 19111 can associate identifiers to reference systems. However, NeoPose has the concept of temporary identifiers with an optional explicit time-to-expire property. There is no standard equivalence in ISO 19115 (the metadata standard used by ISO 19111 for identifiers). However, implementation-specific equivalences are possible by creating subclasses of `MD_Identifier`.

NeoPose transforms not only positions, but also orientations. ISO 19111 is only about positions, at least in the text of that Standard. However, all classes and methods operate on coordinates that are only identified by their position in coordinate tuples. It is technically possible to use 6-dimensional coordinate tuples (or 7-dimensional if including time) with position in the first 3 coordinates and orientation in the last 3 coordinates, then to use those tuples in coordinate operations. Whether this would be stretching too much the spirit of ISO 19111 has not been debated.

4.7.6. ISO 19111 features not covered by NeoPose

ISO 19111 provides more metadata than NeoPose. ISO 19111 uses ISO 19115 (metadata) data structures for expressing citation, identifiers, spatial, vertical, and temporal extents, and ISO 19157 (data quality) for expressing uncertainty. Each of those Standards go far in expressing, for example, how the uncertainty was estimated. Then ISO 19111 itself has a rich set of classes for expressing the semantic of coordinate operations (e.g., *conversion* versus *transformation*) and how the operations are chained together (e.g., Figure 14). NeoPose does not yet go as far, except for the orientation aspect which is not addressed by ISO 19111. However, ISO 19111’s richness makes it more difficult to encode.

5

DART SCENARIO

5 DART SCENARIO

The Double Asteroid Redirect Test (“DART”) was a NASA mission designed to demonstrate asteroid deflection using kinetic impact. In the context of this scenario, participants experimented with relative geospatial referencing, tracking and visualization of non-terrestrial objects. The primary subject of the scenario is the DART impactor spacecraft, traveling from Earth to impact Dimorphos, a natural satellite of the near-Earth asteroid 65803 Didymos. An additional object part of the scenario is the LICIACube (“Light Italian CubeSat for Imaging of Asteroids”), a miniaturized satellite sent to capture information about the effect of the impact on the orbit of Dimorphos around Didymos. LICIACube separated from the DART spacecraft 15 days before the latter made impact.

A video of the DART impact is available here on YouTube.

5.1. Data configuration and retrieval

Source data for the DART scenario was retrieved from the NASA Jet Propulsion Laboratory Horizons systems for the four scenario objects:

- The DART impactor spacecraft
- The 65803 Didymos near-Earth asteroid
- Dimorphos, the natural satellite of Didymos
- The LICIACube spacecraft

The data are in four different Coordinate Reference Systems (CRSs), all based on the International Celestial Reference Frame (ICRF), each with a different reference center as follows.

- The barycenter (Clause 2.1) of the Solar System — this CRS is understood by participants to be at least roughly equivalent to the Barycentric Celestial Reference System (BCRS).
- The Earth center — this CRS is understood by participants to be equivalent to the Geocentric Celestial Reference System (GCRS).
- The barycenter of the combined Dimorphos / Didymos system
- The DART impactor spacecraft

The following steps were used to configure the Horizons systems data retrieval:

- access the Web application at https://ssd.jpl.nasa.gov/horizons/app.html ;
- Step 1: Select “Vector Table” for the ephemeris type;

- Step 2: Select one of the four scenario target bodies by searching for “DART”, “LICIACube”, “Didymos” (choosing “Primary Body”), or “Dimorphos”;
- Step 3: Select one of the four reference centers by searching for “@SSB” (Solar System Barycenter), “@Earth” (choosing “Geocentric”), “@Didymos” choosing “Primary Center,” or “System Barycenter” (“@Dimorphos” could also be chosen to use Dimorphos as the reference center instead);
- Step 4: Select the temporal interval and resolution, in Barycentric Dynamical Time (TDB), as follows:
  - for Didymos and Dimorphos, from 2021-Nov-24 00:00:00.0000 to 2022-Sep-30 00:00:00.0000 at a 5 minute step size;
  - For DART, from 2021-Nov-24 07:17:53.0000 to 2022-Sep-30 00:00:00.0000 at a 5 minute step size; or
  - For LICIACube, from 2022-Sep-11 23:15:10.8790 to 2022-Sep-30 00:00:00.0000 at a 5 minute step size.
- Step 5: Configure the Vector Table Settings as follows:
  - Reference frame: *ICRF*
  - Reference plane: *x-y axes of reference frame*
  - Vector correction: *geometric states*
  - Calendar type: *Mixed*
  - Output units: *km and seconds*
  - Vector labels: (checked)
  - Output TDB-UT: (checked)
  - CSV format: (checked)
  - Object summary: (checked);
- Finally, click the “Generate Ephemeris” button to generate the data, then click “Download Results” once ready to download the dataset.

Note that the shortest available time step size is 1 minute.

The full pre-configured dataset retrieved for the experiments can be downloaded from here.

5.2. Moving Features JSON representation

For the TB-19 *D100 — Moving Features Component*, the DART scenario datasets were converted to Moving Features JSON (MF-JSON). MF-JSON is a GeoJSON encoding extension for the OGC Moving Features Standard. MF-JSON defines the encoding of changing geometry over time, or feature geometry whose location changes over time, as JSON objects. In addition to geographic movement, the objects can represent both constant as well as dynamic properties, and spatiotemporal extents of moving features, based on WGS84 and UTC (Coordinated Universal Time) reference systems.

Testbed-19 participant Ecere developed a simple conversion tool taking the NASA JPL Horizons system as input and generating MF-JSON as output. Since no other Testbed participants were implementing a MF-JSON components as part of the *Geospatial in Space* thread, and because participants realized that the default encoding of *OGC API — Moving Features* differs in fact from MF-JSON, no client or server API component were developed as part of this Testbed initiative. In a real-world scenario, such a tool, or the ability to directly export data in a standardized format such as MF-JSON, could be useful for achieving interoperability between two local components aboard a spacecraft developed by distinct organizations. One component may be capturing location information, while another component further processes that information, such as converting the information to another CRS, predicting trajectories, or sending the data from the spacecraft back to Earth for processing readily available in a standardized format. A specific example where coordinates are converted to another CRS is described below in the Technology Integration Experiments section.

For the purpose of converting the Horizons data to MF-JSON, participants followed the *B.2. Sample Data of OGC Moving Features JSON Trajectory encoding* example from the Moving Features Standard. The x, y, and z coordinates of the object trajectories were mapped to the geometry coordinates of a *LineString* feature. Two constant properties are associated with each feature, `name` being a descriptive title of the object, and `JPLHorizonsID` being the object ID on the Horizons system.

The timestamp of each position was included in the `datetimes` temporal array property required by MF-JSON, with one entry corresponding to every point of the LineString along the trajectory. Although date time should have been converted to UTC coordinates, for the purpose of the experiment the time stamps were kept in the original Gregorian calendar TDB temporal reference system. A separate temporal property, `tdbUTDeltas`, specifies the delta between TDB and UT (Universal Time, also UT1) which varies slightly around 69.182 seconds. UTC is synchronized with UT to remain within 0.9 seconds by using leap seconds. The `tdbJulianDays` temporal property also provides the time in fractional Julian days, which is commonly used by astronomy libraries such as the Naval Observatory Vector Astrometry Subroutines (NOVAS).

Additional temporal properties include the `lightTimes` (one-way down-leg Newtonian light-time in seconds), `rangeRates` (radial velocity with regard to the coordinate center in kilometers per second), `ranges` (distance from the coordinate center in kilometers), and `velocities` (the velocity vector in kilometers per second). Not yet existing URIs were used up to identify the CRS by reference in the `crs` field:

- http://www.opengis.net/def/crs/IAU/0/BCRS for BCRS (Solar System Barycenter),

- http://www.opengis.net/def/crs/IAU/0/GCRS for GCRS (Geocentric),
- http://www.opengis.net/def/crs/Testbed19/0/DidymosSystemBarycenterICRFEquatorial for the Didymos/Dimorphos system barycenter,
- http://www.opengis.net/def/crs/Testbed19/0/DidymosPrimaryCenterICRFEquatorial for the Didymos primary center.

A partial encoding for the DART object trajectory in BCRS appears as follows:

```
{
  "type": "FeatureCollection",
  "crs" : { "type" : "name", "properties" :
     { "name" : "http://www.opengis.net/def/crs/IAU/0/BCRS" } },
  "features": [
    {
      "type": "Feature",
      "id": 1,
      "geometry": {
        "type": "LineString",
        "coordinates": [
          [
            68240910.38511497,
            120050514.8471484,
            52064856.98546032
          ],
          ...
      "properties": {
        "name": "DART (spacecraft)",
        "JPLHorizonsID": -135,
        "datetimes": [
          "2021-11-24T07:17:53Z",
          "2021-11-24T07:22:53Z",
...
```

Figure 33 — Partial MF-JSON encoding of the DART impactor spacecraft trajectory in BCRS

Sixteen MF-JSON files were produced for each of the four objects of interest, in each of the four different CRSs. Additionally, four MF-JSON files were produced combining all four objects, for each of the four different CRSs, where:

- DART corresponds to feature ID 1;
- Didymos corresponds to feature ID 2;
- Dimorphos corresponds to feature ID 3; and
- LICIACube corresponds to feature ID 4.

The combined MF-JSON in the BCRS were used for the visualization experiments described below.

The full set of files encoded as MF-JSON can be downloaded from here.

5.2.1. Barycentric Celestial Reference System (BCRS)

The BCRS has its origin at the Solar System’s center of mass, and its axes are aligned with the International Celestial Reference Frame (ICRF). The ICRF is fixed relative to far away extragalactic objects, defined by measuring quasar positions.

The IAU defined it as a system of space-time coordinates for the solar system in the year 2000. It is considered the system appropriate for the basic ephemerides of solar system objects and astrometric reference data.

The use of the BCRS and of the Barycentric Coordinate Time (TCB, from the French *temps coordonnée barycentrique*) provides a single consistent frame of reference minimizing the effects of General Relativity for observations within our Solar System.

The reference plane of the BCRS is the celestial equator, whereas the primary direction (reference longitude) is the March (Vernal) equinox. BCRS coordinates can be expressed in 3D Cartesian form (x, y, and z) in units such as Astronomical Units (au) or kilometers (km), as used in the JPL Horizons input data and the MF-JSON files generated by these experiments. The BCRS coordinates could alternatively be expressed in spherical coordinates, using declination (δ or lat), right ascension (α or lon), and range coordinates.

Figure 34 illustrates the axes names, reference plane, and primary direction for the Geocentric Celestial Reference System (GCRS), which are the same as for the BCRS. For the BCRS, the Earth reference center would simply be replaced by the Solar System Barycenter.

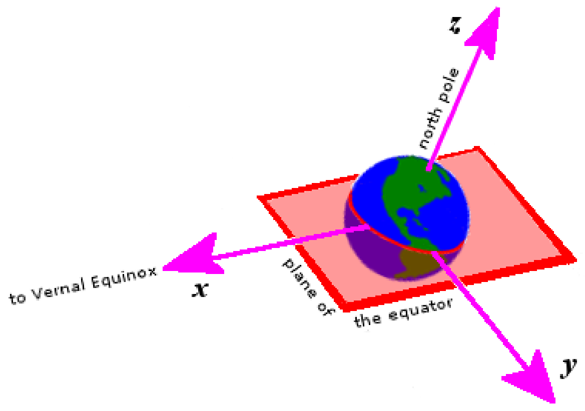


Figure 34 — ICRF Axes for GCRS (source: Wikipedia)

5.2.2. Barycentric Coordinate Time

The Barycentric Coordinate Time (TCB, from the French *temps coordonnée barycentrique*) is the time coordinate complementing BCRS.

TCB is equivalent to the proper time for a clock at rest in the coordinate frame, moving together with the Solar System center of mass, but not influenced by the gravity of solar system objects such as the Sun.

TCB replaces the older Barycentric Dynamical Time (TDB, from the French *temps dynamique barycentrique*).

5.3. Visualization experiments

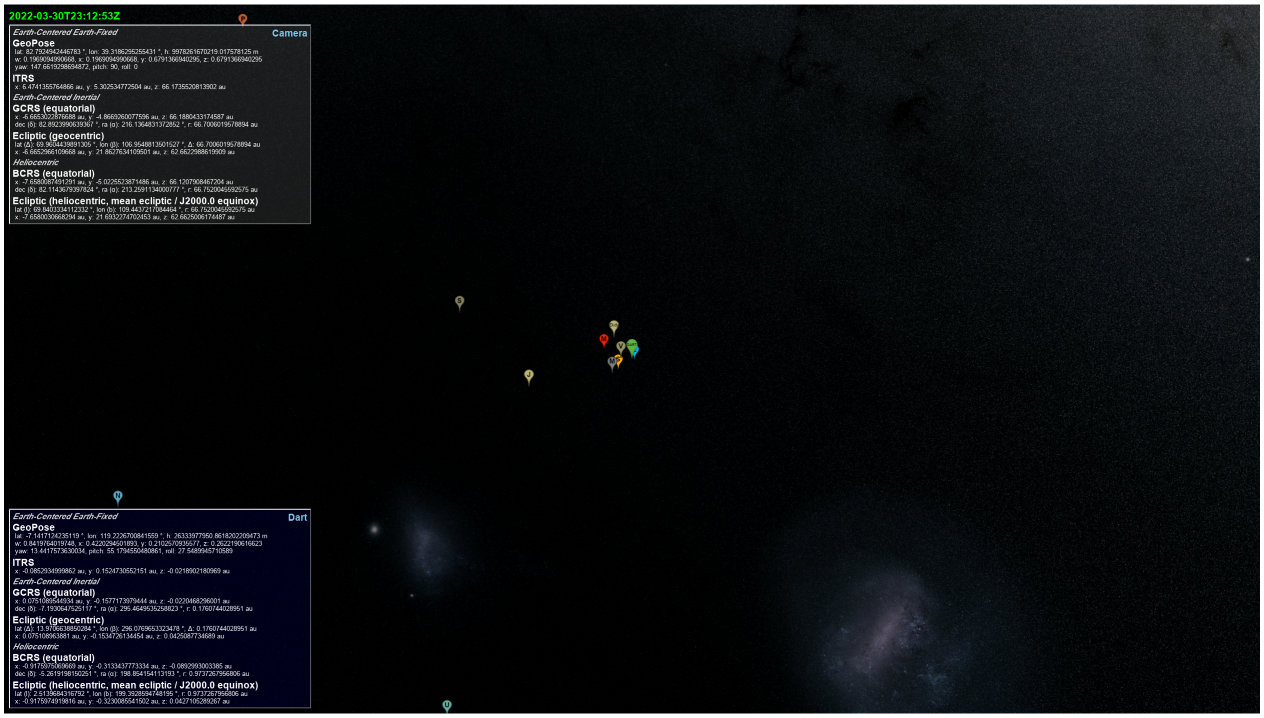
Testbed-19 participant Ecere prototyped a visual demonstration of the DART scenario featuring all four objects (DART impactor spacecraft, Didymos, Dimorphos, and LICIACube), within a simulated Solar System and background map of the stars. Ecere used its GNOSIS SDK and the underlying free and open-source Ecere 3D graphics engine to develop the application, building on previous work from OGC Testbed-18.

The MF-JSON files, converted from the NASA JPL Horizons data, were used as source data for the experiments.

Ecere employed the use of the NOVAS library to instantly provide the positions of all planets in the solar system for simulation, as well as for performing the conversion between inertial and Earth-centered Earth-fixed coordinate reference systems). More details about this use of the NOVAS library can be found in the Testbed-18 ER.

For this DART demonstration, Ecere implemented cubic interpolation of the object positions between the 5 minute steps, using Newton polynomials. This improved interpolation resolved what presented itself as zig-zagging effects of the Voyager spacecraft position in the past Testbed-18 demonstration.

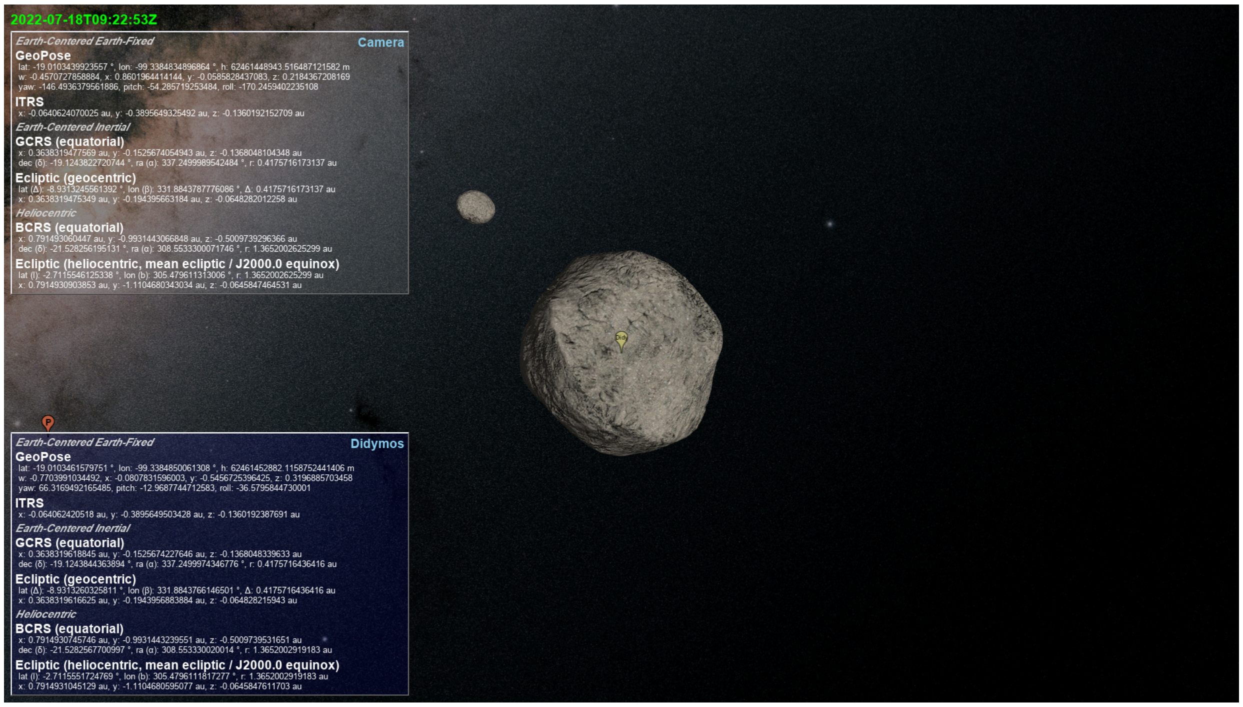
The application uses free 3D models of the spacecrafts, Dimorphos, and Didymos asteroids, as well as the Solar System planets, retrieved primarily in a source glTF format, available from NASA, John Hopkins Applied Physics Laboratory (APL). For the background high-resolution map of the stars from the Milky Way and beyond, the application makes use of the *Gaia Sky in Colour* from European Space Agency, providing an improved situational context.



```
2022-03-30T23:12:53Z
Earth-Centered Earth-Fixed                                   Camera
GeoPose
 lat: 82.7924942446783 °, lon: 39.3186295255431 °, h: 9978261670219.017578125 m
 w: 0.1969094990668, x: 0.1969094990668, y: 0.6791366940295, z: 0.6791366940295
 yaw: 147.6619298694872, pitch: 90, roll: 0
ITRS
 x: 6.4741355764866 au, y: 5.302534772504 au, z: 66.1735520813902 au
Earth-Centered Inertial
GCRS (equatorial)
 x: -6.6653022876688 au, y: -4.8669260077596 au, z: 66.1880433174587 au
 dec (δ): 82.8923990639367 °, ra (α): 216.1364831372852 °, r: 66.7006019578894 au
Ecliptic (geocentric)
 lat (Δ): 69.9604439891305 °, lon (β): 106.9548813501527 °, Δ: 66.7006019578894 au
 x: -6.6652966109668 au, y: 21.8627634109501 au, z: 62.6622988619909 au
Heliocentric
BCRS (equatorial)
 x: -7.6580087491291 au, y: -5.0225523871486 au, z: 66.1207908467204 au
 dec (δ): 82.1143679397824 °, ra (α): 213.2591134000777 °, r: 66.7520045592575 au
Ecliptic (heliocentric, mean ecliptic / J2000.0 equinox)
 lat (l): 69.8403334112332 °, lon (b): 109.4437217084464 °, r: 66.7520045592575 au
 x: -7.6580030668294 au, y: 21.6932274702453 au, z: 62.6625006174487 au
```

```
Earth-Centered Earth-Fixed                                   Dart
GeoPose
 lat: -7.1417124235119 °, lon: 119.2226700841559 °, h: 26333977950.8618202209473 m
 w: 0.8419764019748, x: 0.4220294501893, y: 0.2102570935577, z: 0.2622190616623
 yaw: 13.4417573630034, pitch: 55.1794550480861, roll: 27.5489945710589
ITRS
 x: -0.0852934999862 au, y: 0.1524730552151 au, z: -0.0218902180969 au
Earth-Centered Inertial
GCRS (equatorial)
 x: 0.0751089544934 au, y: -0.1577173979444 au, z: -0.0220468296001 au
 dec (δ): -7.1930647525117 °, ra (α): 295.4649535258823 °, r: 0.1760744028951 au
Ecliptic (geocentric)
 lat (Δ): 13.9706638850284 °, lon (β): 296.0769653323478 °, Δ: 0.1760744028951 au
 x: 0.075108963881 au, y: -0.1534726134454 au, z: 0.0425087734689 au
Heliocentric
BCRS (equatorial)
 x: -0.9175975069669 au, y: -0.3133437773334 au, z: -0.0892993003385 au
 dec (δ): -5.2619198150251 °, ra (α): 198.854154113193 °, r: 0.9737267956806 au
Ecliptic (heliocentric, mean ecliptic / J2000.0 equinox)
 lat (l): 2.5139684316792 °, lon (b): 199.3928594748195 °, r: 0.9737267956806 au
 x: -0.9175974919816 au, y: -0.3230085541502 au, z: 0.0427105289267 au
```

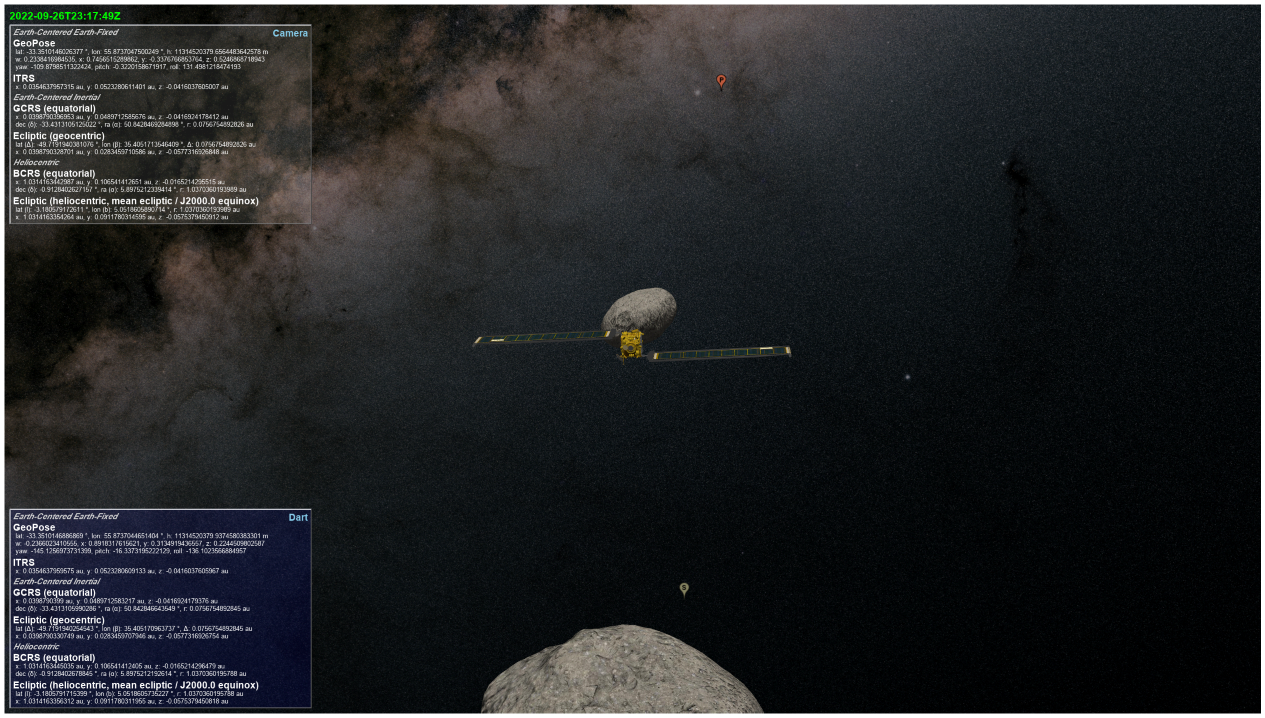
Figure 35 — Visualization of DART Scenario by Ecere (Solar System) — Gaia Sky in Colour from ESA



```
2022-07-18T09:22:53Z
Earth-Centered Earth-Fixed                                   Camera
GeoPose
 lat: -19.0103439923557 °, lon: -99.3384834896864 °, h: 62461448943.516487121582 m
 w: -0.4570727858884, x: 0.8601964414144, y: -0.0585828437083, z: 0.2184367208169
 yaw: -146.4936379561886, pitch: -54.285719253484, roll: -170.2459402235108
ITRS
 x: -0.0640624070025 au, y: -0.3895649325492 au, z: -0.1360192152709 au
Earth-Centered Inertial
GCRS (equatorial)
 x: 0.3638319477569 au, y: -0.1525674054943 au, z: -0.1368048104348 au
 dec (δ): -19.1243822720744 °, ra (α): 337.2499989542484 °, r: 0.4175716173137 au
Ecliptic (geocentric)
 lat (Δ): -8.9313245561392 °, lon (β): 331.8843787776086 °, Δ: 0.4175716173137 au
 x: 0.3638319475349 au, y: -0.194395663184 au, z: -0.0648282012258 au
Heliocentric
BCRS (equatorial)
 x: 0.791493060447 au, y: -0.9931443066848 au, z: -0.5009739296366 au
 dec (δ): -21.528256195131 °, ra (α): 308.5533300071746 °, r: 1.3652002625299 au
Ecliptic (heliocentric, mean ecliptic / J2000.0 equinox)
 lat (l): -2.7115546125338 °, lon (b): 305.479611313006 °, r: 1.3652002625299 au
 x: 0.7914930903853 au, y: -1.1104680343034 au, z: -0.0645847464531 au
```

```
Earth-Centered Earth-Fixed                                   Didymos
GeoPose
 lat: -19.0103461579751 °, lon: -99.3384850061308 °, h: 62461452882.1158752441406 m
 w: -0.7703991034492, x: -0.0807831596003, y: -0.5456725396425, z: 0.3196885703458
 yaw: 66.3169492165485, pitch: -12.9687744712583, roll: -36.5795844730001
ITRS
 x: -0.064062420518 au, y: -0.3895649503428 au, z: -0.1360192387691 au
Earth-Centered Inertial
GCRS (equatorial)
 x: 0.3638319618845 au, y: -0.1525674227646 au, z: -0.1368048339633 au
 dec (δ): -19.1243844363894 °, ra (α): 337.2499974346776 °, r: 0.4175716436416 au
Ecliptic (geocentric)
 lat (Δ): -8.9313260325811 °, lon (β): 331.8843766146501 °, Δ: 0.4175716436416 au
 x: 0.3638319616625 au, y: -0.1943956883884 au, z: -0.064828215943 au
Heliocentric
BCRS (equatorial)
 x: 0.7914930745746 au, y: -0.9931443239551 au, z: -0.5009739531651 au
 dec (δ): -21.5282567700997 °, ra (α): 308.553330020014 °, r: 1.3652002919183 au
Ecliptic (heliocentric, mean ecliptic / J2000.0 equinox)
 lat (l): -2.7115551724769 °, lon (b): 305.4796111817277 °, r: 1.3652002919183 au
 x: 0.7914931045129 au, y: -1.1104680595077 au, z: -0.0645847611703 au
```

Figure 36 — Visualization of DART Scenario by Ecere (Didymos) — Gaia Sky in Colour from ESA, 3D models from NASA / John Hopkins APL



```
2022-09-26T23:17:49Z
Earth-Centered Earth-Fixed                                        Camera
GeoPose
 lat: -33.3510146026377 °, lon: 55.8737047500249 °, h: 11314520379.6564483642578 m
 w: 0.2338416984535, x: 0.7456515289862, y: -0.3376766853764, z: 0.5246868718943
 yaw: -109.8798511322424, pitch: -0.3220158671917, roll: 131.4981218474193
ITRS
 x: 0.0354637957315 au, y: 0.0523280611401 au, z: -0.0416037605007 au
Earth-Centered Inertial
GCRS (equatorial)
 x: 0.0398790396953 au, y: 0.0489712585676 au, z: -0.0416924178412 au
 dec (δ): -33.4313105125022 °, ra (α): 50.8428469284898 °, r: 0.0756754892826 au
Ecliptic (geocentric)
 lat (Δ): -49.7191940381076 °, lon (β): 35.4051713546409 °, Δ: 0.0756754892826 au
 x: 0.0398790328701 au, y: 0.0283459710586 au, z: -0.0577316926848 au
Heliocentric
BCRS (equatorial)
 x: 1.0314163442987 au, y: 0.106541412651 au, z: -0.0165214295515 au
 dec (δ): -0.9128402627157 °, ra (α): 5.8975212339414 °, r: 1.0370360193989 au
Ecliptic (heliocentric, mean ecliptic / J2000.0 equinox)
 lat (l): -3.180579172611 °, lon (b): 5.0518605890714 °, r: 1.0370360193989 au
 x: 1.0314163354264 au, y: 0.0911780314595 au, z: -0.0575379450912 au
```

```
Earth-Centered Earth-Fixed                                        Dart
GeoPose
 lat: -33.3510146886869 °, lon: 55.8737044651404 °, h: 11314520379.9374580383301 m
 w: -0.2366023410555, x: 0.8918317615621, y: 0.3134919436557, z: 0.2244509802587
 yaw: -145.1256973731399, pitch: -16.3373195222129, roll: -136.1023566884957
ITRS
 x: 0.0354637959575 au, y: 0.0523280609133 au, z: -0.0416037605967 au
Earth-Centered Inertial
GCRS (equatorial)
 x: 0.0398790399 au, y: 0.0489712583217 au, z: -0.0416924179376 au
 dec (δ): -33.4313105990286 °, ra (α): 50.842846643549 °, r: 0.0756754892845 au
Ecliptic (geocentric)
 lat (Δ): -49.7191940254543 °, lon (β): 35.405170963737 °, Δ: 0.0756754892845 au
 x: 0.0398790330749 au, y: 0.0283459707946 au, z: -0.0577316926754 au
Heliocentric
BCRS (equatorial)
 x: 1.0314163445035 au, y: 0.106541412405 au, z: -0.0165214296479 au
 dec (δ): -0.9128402678845 °, ra (α): 5.8975212192614 °, r: 1.0370360195788 au
Ecliptic (heliocentric, mean ecliptic / J2000.0 equinox)
 lat (l): -3.1805791715399 °, lon (b): 5.0518605735227 °, r: 1.0370360195788 au
 x: 1.0314163356312 au, y: 0.0911780311955 au, z: -0.0575379450818 au
```

Figure 37 — Visualization of DART Scenario by Ecere (Impactor Spacecraft) — Gaia Sky in Colour from ESA, 3D models from NASA / John Hopkins APL

5.4. GeoPose representation

Building on previous work from Testbed-18, Ecere also prototyped a potential representation of the DART scenario data as a sequence of extended GeoPoses.

In this representation, each element of a `sequence` array is a GeoPose as defined in the GeoPose 1.0 *Basic-Quaternion Permissive* JSON encoding schema, with the exception that the schema is further relaxed to allow for alternative properties for the `position` object to accommodate other CRSs, such as the x, y, and z coordinates of CRSs, like BCRS and GCRS, used in the DART scenario experiments.

The `quaternion` member represents the orientation of the feature. This is a piece of information not included in the MF-JSON source data files for the scenario, but which had to be computed for the purpose of the visualization demonstration. This information can also be expressed in MF-JSON using the optional `orientations` property.

The CRS for all these GeoPoses is mentioned as a top-level property, much like MF-JSON. A `features` property is also used at the top-level to identify multiple features, which can have constant properties such as their name. The `sequence` array of GeoPoses is a property of each feature.

A `validTime` property specifies the time for each pose of the sequence, expressed in UNIX time, as in the GeoPose 1.0 Stream Elements. However, fractional seconds precision, as well as the possibility to use alternate temporal reference systems may be required for space scenarios. In this prototype example, the temporal CRS is still indicated as Barycentric Coordinate Time, included as part of an array of CRS forming a spatiotemporal continuum, indicating that the UNIX time is offset from UTC.

The additional temporal properties are associated with each of the poses using a `properties` member. This prototyping experiment may be considered together with the work discussed in the NeoPose section of this ER and the previous Testbed-18 results to plan for the next revision of GeoPose. As also discussed for Testbed-18, this experiment highlights the fact that features with an associated sequence of GeoPoses could be an alternative representation for use cases similar to those of MF-JSON. It should also be noted that *OGC API — Moving Features* itself defines another representation for moving features which is not MF-JSON, and that the draft *OGC API — Connected Systems* Standard defines yet another representation (as discussed in this OGC API — Moving Features issue “Clarify the relationship with OGC API — Connected Systems”).

```
{
   "crs": [
      "http://www.opengis.net/def/crs/IAU/0/BCRS",
      "http://www.opengis.net/def/crs/IAU/0/TCB"
   ],
   "features": [
      {
         "id": -135,
         "properties": { "name": "DART (spacecraft)" },
         "sequence": [
            {
               "validTime": 1637756273,
               "position": {
                     "x": 68240910.38511497,
                     "y": 120050514.8471484,
                     "z": 52064856.98546032 },
               "quaternion": {
                     "x": 0.5131, "y": 0.14325,
                     "z": 0.65354, "w": 0.5376739680327
               },
               "properties":
               {
                  "lightTime": 492.2720936600098,
                  "rangeRate": -6.9976361812831,
                  "range": 147579460.9631405,
                  "tdbJulianDay": 2459542.804085648,
                  "tdbUTDelta": 69.182893,
                  "velocity": [
                     -30.6999082420738,
                     5.8733080516535,
                     6.8604559150432 ]
               }
            },
            {
               "validTime": 1637756573,
               "position": {
                     "x": 68231758.62649576,
                     "y": 120052328.7904038,
                     "z": 52066954.64639068
               },
               "quaternion": {
                     "x": 0.5131, "y": 0.14325,
                     "z": 0.65354, "w": 0.5376739680327
               },
               "properties":
               {
                  "lightTime": 492.2653694323749,
                  "rangeRate": -6.4612417416349,
                  "range": 147577445.0904097,
                  "tdbJulianDay": 2459542.80755787,
                  "tdbUTDelta": 69.182894,
                  "velocity": [
                     -30.3272855767727,
                     6.2118990480757,
                     7.1061873410806
                  ]
               }
            },
            {
               "validTime": 1637756873,
               "position": {
                     "x": 68222706.12123032,
                     "y": 120054237.4679789,
                     "z": 52069112.30994629
               },
               "quaternion": {
                     "x": 0.5131, "y": 0.14325,
                     "z": 0.65354, "w": 0.5376739680327
               },
               "properties":
               {
                  "lightTime": 492.2591279187188,
                  "rangeRate": -6.0279800273367,
                  "range": 147575573.9316891,
                  "tdbJulianDay": 2459542.811030093,
                  "tdbUTDelta": 69.182894,
                  "velocity": [
                     -30.0339023211599,
                     6.505482370605,
                     7.2672787646241
                  ]
               }
            },
            ...
```

Figure 38 — Encoding DART spacecraft moving positions in BCRS and associated properties as a prototyped extended GeoPose sequence

5.5. Technology Integration Experiments using OGC GeoAPI

Testbed-19 participants Ecere and Geomatys planned to conduct additional experiments integrating the Apache SIS library, based on the OGC GeoAPI, to perform conversion of coordinates between different CRSs while visualizing the results using Ecere's GNOSIS library. Those experiments were still works in progress at the time this ER was being written and are the subject of on-going and future work.

As a high level description of what these experiments entail, Apache SIS is a native Java library whereas the GNOSIS library is written in the eC language, a superset of C with additional modern programming language features such as object-oriented constructs, reflection, properties, and dynamic modules. The GNOSIS visualization demonstration would invoke the Apache SIS library through the Java Native Interface (JNI), exactly as would be done for code written in the C language.

Since the GNOSIS library is primarily intended for visualizing objects referenced to Earth, the visualization scene graph natively deals with Earth-centered Earth-fixed (ECEF) coordinates expressed in geodetic latitude, longitude, and elevation above the WGS84 ellipsoid, as used in EPSG:4979 and basic GeoPose. The experiments would therefore potentially use the Apache SIS calls to first parse the explicit description of a coordinate reference system from a data source, and then convert coordinates from that source to EPSG:4979 target coordinates. Such conversion would likely require integration of calls to a library for handling celestial coordinate reference systems, such as the NOVAS library. This could be done again using the JNI mechanism in the opposite direction, invoking the NOVAS C library from Java.

Alternatively, the target CRS could be the Barycentric Celestial Reference System, with the GNOSIS demonstration program performing the conversion from BCRS to ECEF using the NOVAS C library, as done in the DART visualization experiments described above, as well as in previous Testbed-18 experiments.

The source coordinates could be in any of the CRSs for which data was retrieved (BCRS, DART-centric, Didymos-centric, Dimorphos/Didymos system-centric), or in another CRS for which sample data can be produced for the purpose of the experiment, such as Dimorphos-centric. It is to be noted that all four of those CRSs follow the orientation of the ICRF relative to the distant stars, and therefore conversion between these CRSs is a simple translation based on the position of the objects used as reference centers, at a given time, for the source and target CRS.

6

HILLYFIELDS BUBBLE EXPERIMENTS

6 HILLYFIELDS BUBBLE EXPERIMENTS

The OGC GeoPose 1.0 Standard is described in terms of traditional Earth-based coordinate systems. Adaptation to non-terrestrial use cases is straightforward, even for distances, relative speeds, and accelerations where the absolute Galilean space and time of Newtonian mechanics do not apply. The key is that the two components of a pose can be generalized to support a positioned entity that is oriented to another entity within its frame of reference. Only the details of the transformation and orientation are sensitive to the details of the space or space-time.

The primary goal for GeoPose in this Testbed was to develop a framework for extension of GeoPose 1.0. These extensions focus on suggestions from work in OGC TB-18 and other implementation experiences gained since publication of the standard. These fall into the following three categories.

- Better integration with existing external standards for defining coordinate reference systems and transformations between coordinate reference systems.
- Linked structures supporting mutual observations of sensors by sensors in a common environment.
- Extensions to a range of spaces, including Minkowski flat space-time, that is, a space of events in a four-dimensional space characterized by a finite and constant speed of causal interaction between distinct events.

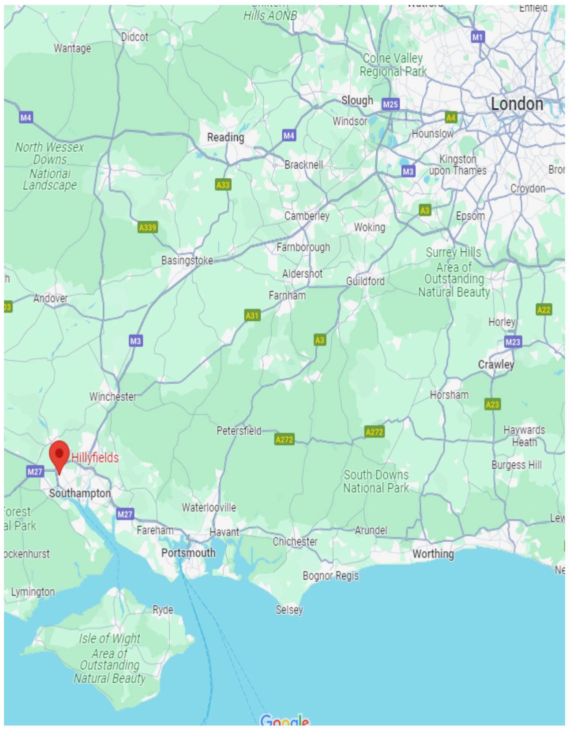


Figure 39 — Hillyfields Location Near Southampton

To support these experiments with real-world sensor observations, a test area was created in a place called *Hillyfields*, surrounding the car park and headquarters building of the UK Ordnance Survey (OS), the UK National Mapping Agency, in Southampton, UK.

6.1. Testing the Experimental Implementation

The experimental NeoPose implementation was tested using data gathered or simulated for use in TB-19.

6.1.1. The Hillyfields Bubble Data Capture

Ultimately, the purpose of many spatial and spatiotemporal representations is to act as a replica of a bounded part of the physical world. A space-time bubble ("bubble") is one type of replica. A bubble replica is a bounded unit of spatiotemporal information. Within the bubble's bounds, there are representations of all physical objects and materials relevant to the intended application of the replica. Paper maps are an example of a bubble. A paper map (or the map in your car's navigation system) has well-defined bounds, a period of temporal validity, mapping of semantics to symbology, and anything important according to its modeling criteria is represented. In addition, the cartographic representation must be interpretable and useful to humans. To generalize a bit, a replica (e.g., a bubble) must:

- represent objects bounded in space and time, i.e., the bubble surface;
- have a semantic representation;
- be complete under specific conditions, i.e., meet a "closed-world" condition (this means that nothing exists but the contents and that the contents completely fill the time and space bounded by the bubble surface); and
- provide access to the contained semantic elements anchored in space and time in a way that supports the retrieval and manipulation requirements of a user experience.

The bubble concept is very useful in consideration of GeoPose objects and especially in composite collections of related GeoPose objects.

One way to define a bubble is with a GeoPose and a set of bounds. To illustrate use of OGC GeoPose 1.0 in complex configurations and to explore extensions of GeoPose to support a completely general pose concept that works in any type of space or space-time, TB-19 used a bubble representing a real location, centered on the headquarters of the UK OS. Using a single well-defined replica that contains a variety of semantic elements whose locations, orientations, semantic categories, and appearance were observed in several timespans. This bubble can serve as a Testbed for developing, testing, and comparing software and systems in interactive applications and experiences, especially those involving complex pose relationships between objects in multiple reference frames.

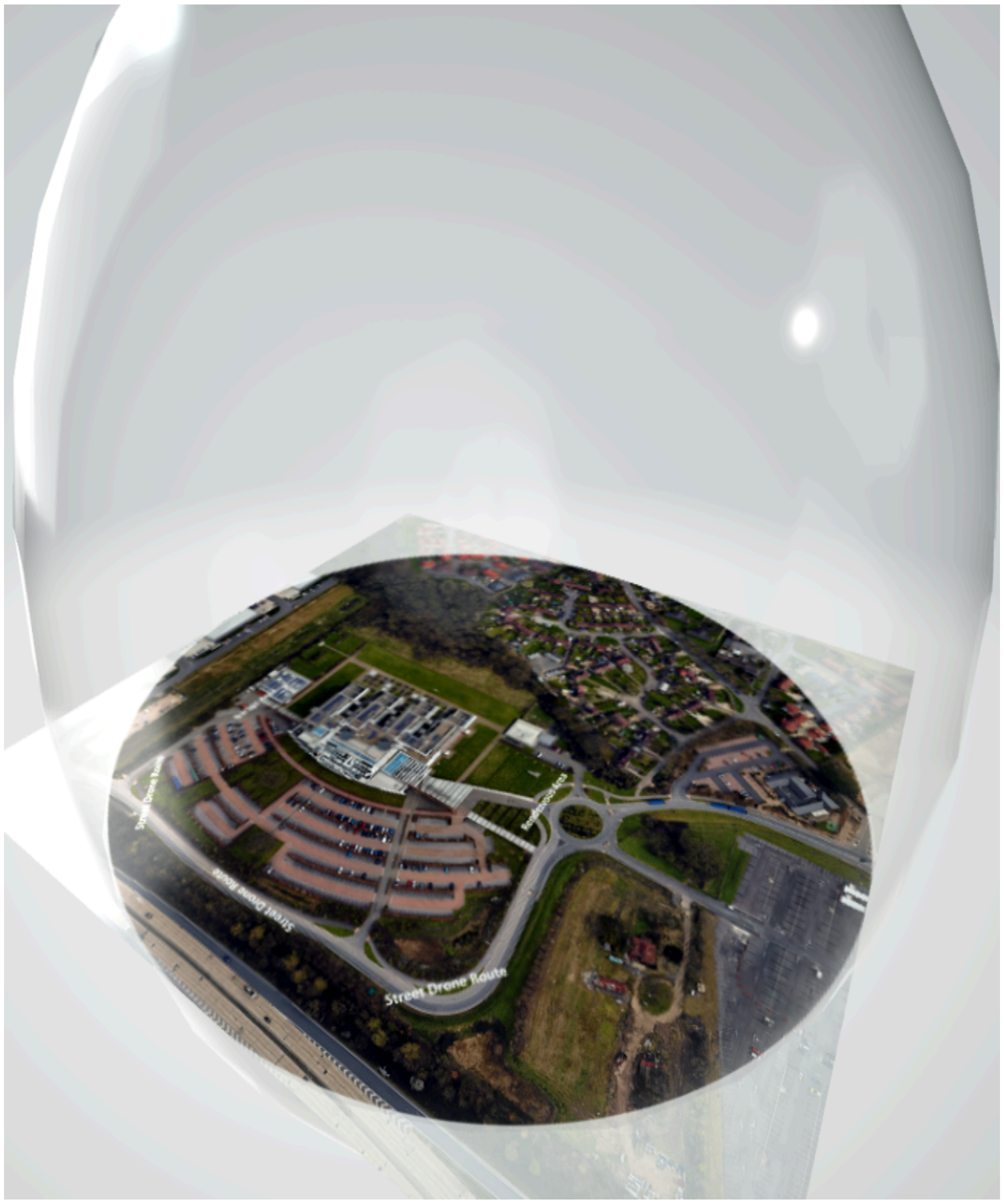


Figure 40 — Hillyfields Bubble, UK Ordnance Survey

The Hillyfields Bubble is a multipurpose dataset captured in support of the development of Augmented Reality/Metaverse architectures and software infrastructure prototypes. Data capture was successfully carried out at the UK OS on April 25th and 26th, 2023. The capture included data from visual and navigation sensors on:

- a Land vehicle [RTK GPS, compass, gyro, accelerometer, 2 video streams, LiDAR];
- two people [GPS, gyro, compass, accelerometer, audio, 5312×2988 video];
- two fixed video cameras recording with Web Video Map Tracks (VMT) annotation; snf
- a drone following car [GPS, compass, gyro,1920×1080 video], 7 runs 25th, 5 runs 26th, 10 Hz pose resolution.

There were eight runs on the 25th and three runs on the 26th.

6.1.2. Simulated Satellite in Low Earth Orbit

Simulated coordinates were calculated for each second from a point overhead Longyearbyen, Norway to a point over Bamako, Mali.

JSON track or KML track with one point per second.

6.1.3. Extension to Minkowski Space-time

Two tasks were defined in an area centered on the car park in front of the headquarters building of the UK Ordnance Survey in Southampton, UK:

- *Pointing task*: Aiming an imaging sensor in low earth orbit at a moving vehicle on the ground; and
- *Minkowski space visualization task*: Visually observing a collection of entities in approximately uniform relative motion at velocities approaching the speed of light.

These tasks were designed to exercise new features in the extended GeoPose 1.0 (NeoPose proposal) model. Data from instrumented moving vehicles, persons, and a fixed background environment were collected on April, 2023. This data capture in a constrained volume of space and time was named the "Hillyfields Bubble".

6.2. Experimental Results

There were three types of experimental results:

- NeoPose representation of moving objects;
- visualization in Minkowski space-time; and
- pointing from low earth orbit with light travel time and relativistic effects.

6.2.1. NeoPose Representation of Objects within the Hillyfields Bubble

Each frame of each imaging sensor attached to each moving object was processed to identify people, cars, cameras, and drones. The lists of observed objects were represented as Local NeoPoses. In some cases, it was possible to identify an external imaging sensor and to establish the identity of an observed object.

6.2.2. Minkowski Space-time Visualization Task

In this task, each of the camera sensors may be the observer and the background environment as well as each of the other moving sensors are in corresponding observed frames. In Galilean relativity, the geometry of all objects in observed frames is viewed as though the speed of light were infinite. In other words, the geometries of the observed objects are independent of their motion relative to the observer. In Minkowski flat space-time, the view of observed objects must be determined by casting rays back from the projected pixels of the sensor. Although one might consider applying a Lorentz transformation to the entire geometry of an observed object in uniform relative motion, this "coordinate" view will not give a correct visual or sensor result for an extended object. What is necessary is, in effect, to project the projected image of the sensor image plane backwards in time to find what objects are intersected. If there are *k* objects in different inertial frames, then *k* versions of each traced ray must be considered individually, and the results combined to show the actual resulting light reaching the observer sensor pixel. The resulting computational cost is then *k* times that of classical raytracing in Galilean relativity. The actual ray casting and intersection computation is not changed:

```
1.  For each pixel $p$ :
    a)  Set the color $C_p$ to black and the number of contributing rays $N_p$ to 0:
        i)  For each moving object $O_k$ :
            A)  cast a ray from the image plane through a camera pinhole;
            B)  Lorentz transform the ray $Rp_k$ to object frame of $O_k$ ;
            C)  find intersection, if any, with object k; and
            D)  if there is an intersection, then using some (any) lighting
                model and object material, compute a color at the
                intersection point and update $C_p$ and $N_p$ .
```

These steps must be repeated for every frame of an animation sequence. In the Hillyfields Bubble, this is a rate of 29.97 frames per second. For an image size of 1920 x 1080, and *k*=5 moving objects, this is 310 million rays per second. Raytracing takes several seconds per frame.

6.2.3. Details for the Lorentz Transformation

Suppose the observed object is moving at a relative velocity v^{ij} with respect to the observer:

$$v^{ij} = |\mathrm{v}^{\mathrm{ij}}|(1) \tag{6}$$

Then compute

$$\beta^{ij} = \frac{v^{ij}}{c}(2) \tag{7}$$

and

$$\gamma^{ij} = \frac{1}{\sqrt{1 - \beta^{ij^2}}} (3) \tag{8}$$

Then for object i

$$e_i = \begin{bmatrix} t^i \\ x^i \\ y^i \\ z^i \end{bmatrix} (4) \tag{9}$$

$$L^{ij} = \begin{bmatrix} \gamma^{ij} & -\gamma^{ij}\beta^{ij}v_x^{ij} & -\gamma^{ij}\beta^{ij}v_y^{ij} & -\gamma^{ij}\beta^{ij}v_z^{ij} \\ -\gamma^{ij}\beta^{ij}v_x^{ij} & 1+(\gamma^{ij}-1)v_x^{ij^2} & 1+(\gamma^{ij}-1)v_x^{ij}v_y^{ij} & 1+(\gamma^{ij}-1)v_x^{ij}v_z^{ij} \\ -\gamma^{ij}\beta^{ij}v_y^{ij} & 1+(\gamma^{ij}-1)v_y^{ij}v_x^{ij} & 1+(\gamma^{ij}-1)v_y^{ij^2} & 1+(\gamma^{ij}-1)v_y^{ij}v_z^{ij} \\ -\gamma^{ij}\beta^{ij}v_z^{ij} & 1+(\gamma^{ij}-1)v_z^{ij}v_x^{ij} & 1+(\gamma^{ij}-1)v_z^{ij}v_y^{ij} & 1+(\gamma^{ij}-1)v_z^{ij^2} \end{bmatrix} (5) \tag{10}$$

The actual transformation is

$$e_j = L^{jk} e_k (6) \tag{11}$$

6.2.4. The Pointing Task

Determination of the departure and arrival directions of a geodesic (Clause 2.9) between pairs of events in distinct inertial frames is a simple definition, but calculation of pointing angles, i.e., the orientation part of a pose, is in general not easy because an observer needs to have a target event in the frame of the source. There must be some source of predictions of the world line of the source in order to compute the geodesic from source to observer.

In TB-19, as the spacecraft orbit is visible above the horizon at the Hillyfields Bubble, the clock is set to 0 and a signal is sent to the Hillyfields Bubble with the location of the spacecraft in its frame using the Hillyfields LTP-ENU coordinate system, as well as a prediction of the future location each second for the next 500 seconds. A copy of the location message can be found here.

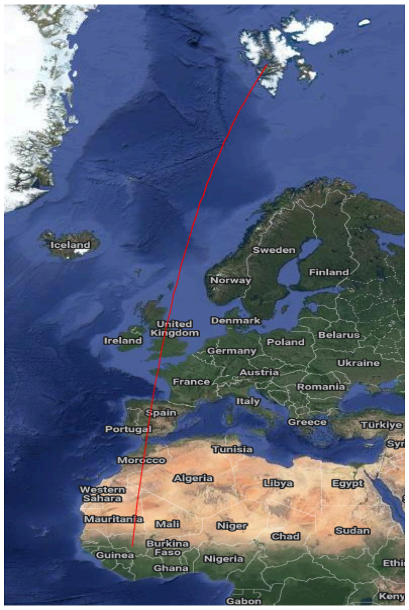


Figure 41 — Satellite Track from Longyearbyen, Svalbard to Bamako, Mali.

In Galilean or Minkowski space-time, geodesics are unique. They are also straight lines in Euclidean space. In general space-time, there may be many curved geodesics linking source and observer. That case is not considered in this ER.

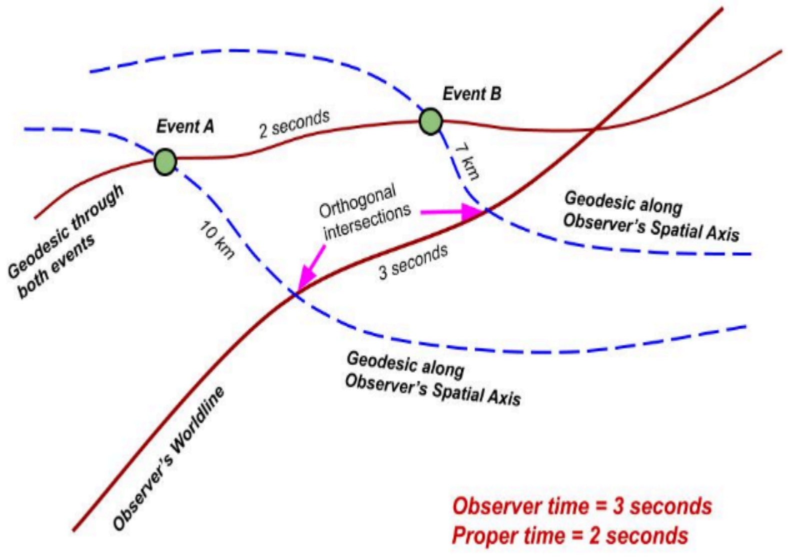

Event A
2 seconds
Event B
7 km
10 km
Orthogonal intersections
Geodesic through both events
Geodesic along Observer's Spatial Axis
3 seconds
Geodesic along Observer's Spatial Axis
Observer's Worldline
Observer time = 3 seconds
Proper time = 2 seconds

Figure 42 — The Pointing Task in Space-time.

- Aim visual sensor on satellite in orbit at 750 000 km above the Hillyfields z = 0 plane at LiDAR unit on Street Drone:
  - about 140 seconds of visibility time;
  - approximate inertial reference frame for this time; and

- compute pointing angles with respect to spacecraft frame.

The computations are the same as for raytracing, except that the rays are cast from the intended target back toward the satellite track. Intersections with the predicted satellite track give a sequence of rays that will intersect the desired object at some time during the satellite's overpass.

7

ROAD NETWORK USE CASES

7 ROAD NETWORK USE CASES

In Testbed-18, Away Team identified two road network use cases that could benefit by integrating WebVMT and GeoPose (OGC 21-056r11) with Moving Features from discussions with the UK National Highways Agency. Road traffic datasets can be enriched by aggregating geotagged video with sensor data observed from multiple sources concurrently, provided that these can be synchronized with sufficient accuracy.

7.1. Overview

In Testbed-19, Away Team proposed a plan to collect and aggregate geotagged video data suitable for analysis of the road network use cases using the following tools developed in previous OGC Testbeds.

- **Testbed-16:** Moving object previsualization was successfully demonstrated using WebVMT integrated with web technologies in a browser.
- **Testbed-17:** A suite of WebVMT tools was developed to track a cyclist from a moving vehicle using video and LiDAR data with GeoPose.
- **Testbed-18:** A 3D compass app was developed to demonstrate how GeoPose can be recorded using a smartphone, which was publicly released free of charge for Android devices.

7.2. Use Cases

Previous discussions with national road network authorities highlighted two use cases of particular interest as follows.

- **Wrong-way traffic** to rapidly identify and locate vehicles traveling in the opposite direction to the correct traffic flow is a safety-critical issue. Being able to interview these elusive drivers can also provide valuable feedback to help reduce incident recurrence with targeted improvements to road junction design.
- **Litter monitoring** to assess accretion rates of objects that can hinder traffic flows by harvesting geotagged dashcam footage from a fleet of vehicles which routinely traverse the road network for maintenance, delivery, or transportation purposes. Metrics can be used to plan and prioritize remedial action.

The essential elements of these use cases can be reduced to the following concept demonstrations.

- **Traffic cameras:** Associate observations of a moving vehicle seen by two static traffic cameras at different locations along a road.
- **Fleet survey:** Associate observations of a static object seen by dashcams from two moving vehicles passing the same location at different times.
- **Multi-view survey:** Associate observations of a static object seen from front- and rear-facing dashcams on a moving vehicle passing a location.

7.3. Data Capture

In April 2023, Open Site Plan and Away Team participated in a collaboration with Ordnance Survey (OS) at the OS Southampton headquarters to capture geotagged video from a selection of location-aware video devices. Footage was recorded from multiple devices concurrently. These devices included smartphones, body-worn cameras, a drone, and a StreetDrone Twizy autonomous vehicle. This collection was required in order to study a number of use cases including the litter monitoring and wrong-way traffic scenarios previously identified in Testbed-18. The aim was to show how accurately data collected by different devices can be synchronized and demonstrate the value added by aggregating this information.

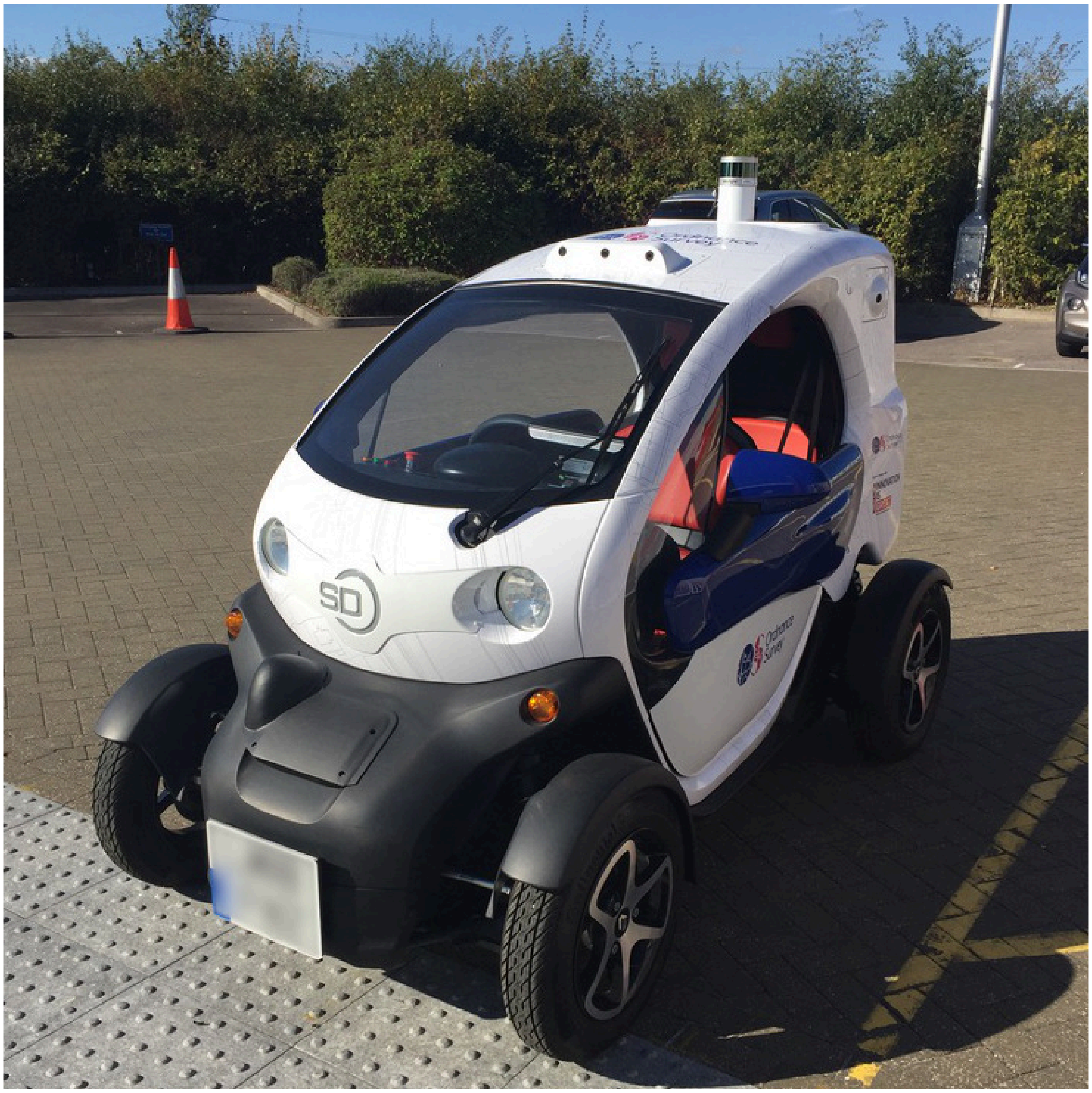


Figure 43 — StreetDrone Vehicle Operated By Ordnance Survey

Geotagged video was recorded by two smartphones running Away Team’s TrkdCam application which were mounted on tripods at the roadside. These acted as static traffic cameras to record details of passing vehicles for the wrong-way traffic use case. The GeoPose of each device was captured using the 3D Compass app from Testbed-18, together with reference photos to verify locations.

Objects were placed at known positions by the roadside to act as targets for the litter monitoring use case, including common litter items such as metal drink cans and plastic bottles. In addition, the UK National Highways Agency supported this collaboration by lending motorway maintenance equipment which included temporary signs, metal A-frame stands for those signs, and traffic cones with lights. The Agency has an operational requirement to quickly identify and collect such items which are accidentally left behind after roadworks are finished as these are valuable assets that can be hazardous to road users when left in the wrong location.

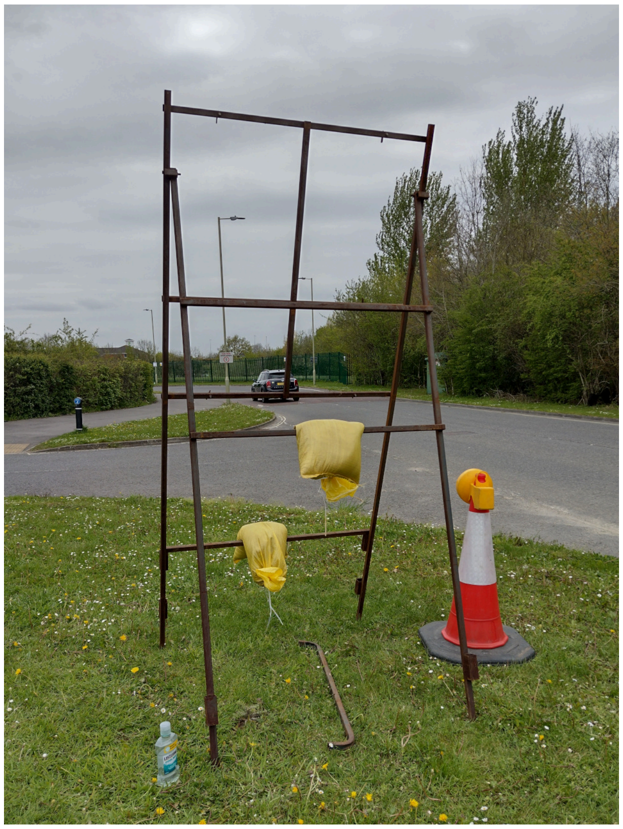


Figure 44 — Road Sign A-Frame, Traffic Cone & Plastic Bottle

7.4. Analysis

Having captured the data, the initial challenge was to track passing vehicles from the roadside footage.

7.4.1. Vehicle Tracking

Moving objects were tracked from video using a process that was designed to take advantage of the oblique viewing angle by using lines of sight. This method is also robust to variations in the pan and tilt of the tripod mounting which inevitably occurred when the camera was removed to download files.

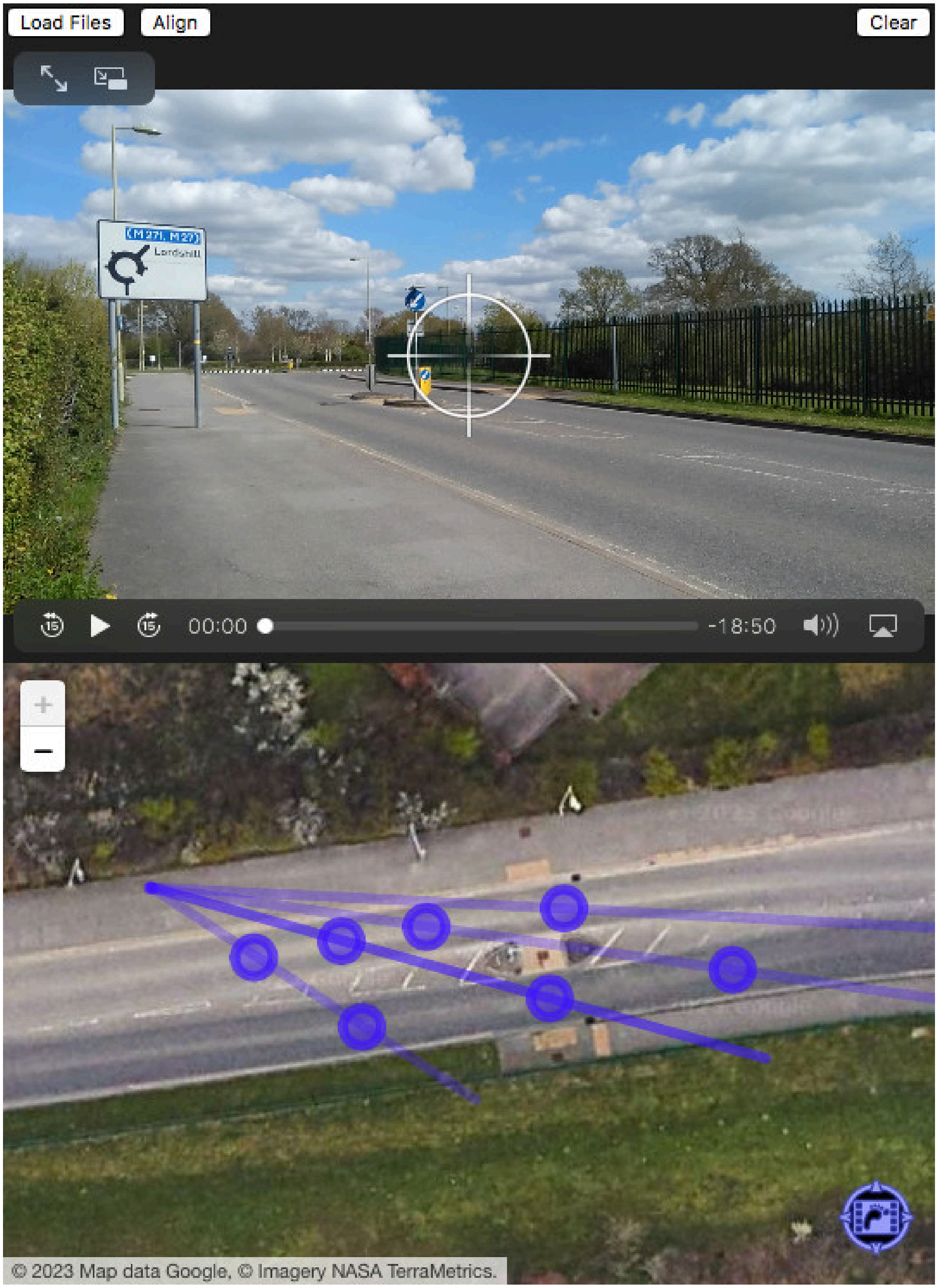


Figure 45 — Lines of Sight With Target Location Zones

The procedure can be summarized as follows:

1. identify suitable lines of sight from the video frames;
2. identify target locations corresponding to those lines;
3. identify video frames with a moving object that matches those target locations;
4. add a path segment to the corresponding WebVMT file; and
5. verify accuracy using previsualization in a web browser.

Figure 46 shows the observed track synchronized with video and a faint line indicating the camera's line of sight.

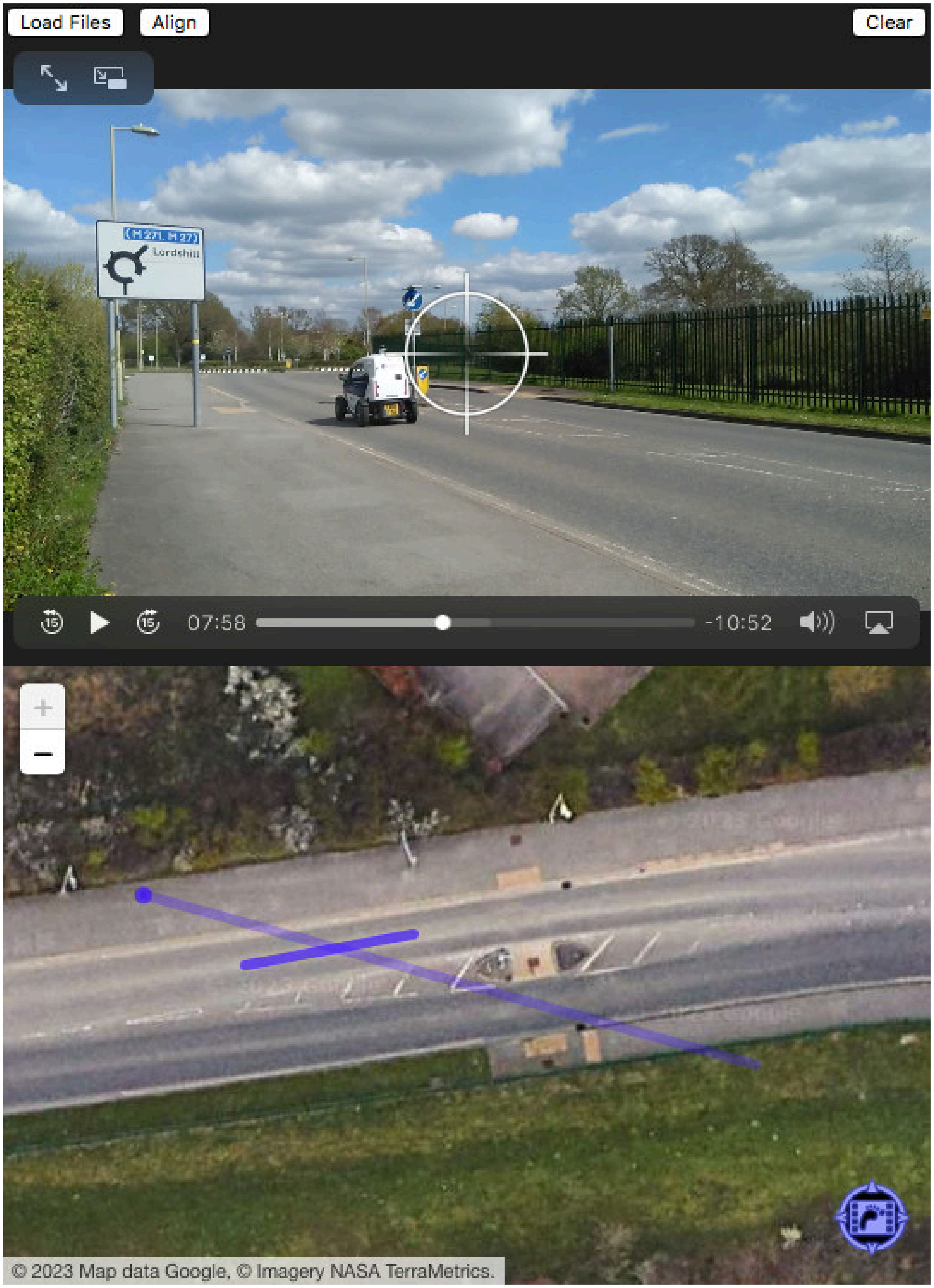


Figure 46 — Vehicle Tracking Previsualization

7.4.2. Synchronization Issues

The next challenge was to accurately synchronize data from different devices with a common timeline — in this case Coordinated Universal Time (UTC). Timing errors hinder the process of correctly associating observations of the same object at different times and locations between different devices. Hence, poor synchronization significantly reduces the value of data aggregation.

The smartphones used to record video from static tripods were linked to the cell network which enables access to accurate UTC time. However, smartphone cameras typically have a significant delay between a user request to start video recording and the first frame of that recording, and no accurate measurement of the video start time could be found. The latency depends on a number of factors including the camera hardware response time, resource pressure from other tasks, and the device's power management strategy. As a result, the estimated start time was only accurate to within a few seconds.

By contrast, the StreetDrone vehicle was able to record timing data accurately to within 10ms and is synchronized with UTC. A process was devised to match the trajectory recorded aboard the vehicle with the trajectory observed by the smartphone cameras to synchronize their video timelines with UTC. Visual analysis is limited to the unit time of a single video frame which is around 33ms at 30 frames per second.

Data were recorded natively in WebVMT by the smartphones and exported to WebVMT from the StreetDrone's inertial measurement unit (IMU) enabling tools developed in Testbed-17 to be reused for analysis and allowed results to be integrated with online maps for verification in a web browser.

A new WebVMT utility was developed to calculate a list of closest approach times to a static location by a moving object. This tool can calculate results to millisecond accuracy and includes a radius parameter that enables multiple approaches to be discriminated correctly. By matching the StreetDrone trajectory data against the target locations used for vehicle tracking by the roadside cameras, an average time offset was calculated for each video file captured by the static cameras. In addition, the standard deviation was calculated to quantify the quality of the synchronization process. Results were used to adjust the WebVMT start time parameter in order to accurately synchronize the path segments observed by the static camera with UTC.

A second utility was added to the existing suite that clips WebVMT content to match video files with revised start times and durations. WebVMT files synchronized with UTC were clipped and merged using this and an updated version of the Testbed-17 tools suite to create previsualizations that demonstrate the accuracy of these results — see Synchronized Previsualization.

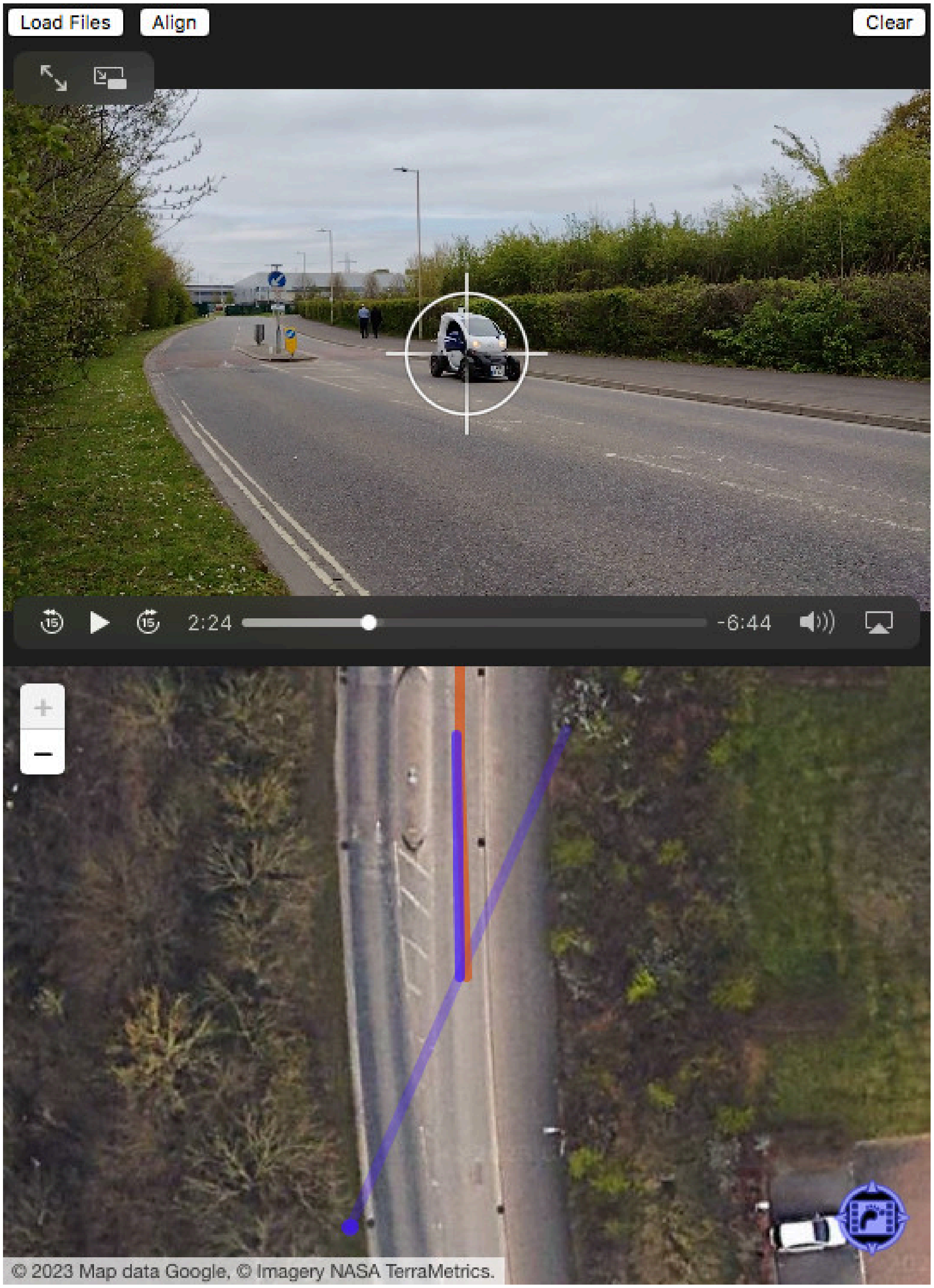


Figure 47 — Synchronized Data Previsualization

7.5. Results

7.5.1. Synchronization Metrics

The geotagged video files recorded had fairly consistent time offsets from UTC for each of the two smartphone devices — one was typically 1.5 seconds ahead and the other was around 2 seconds behind UTC. Quality metrics were good for both cameras since most standard deviations are in the range between 50ms and 200ms which is indicative of the average error.

Table 9 — Nokia 5 Synchronization Metrics

WEBVMT FILE	MEAN TIME OFFSET	STANDARD DEVIATION	SAMPLE SIZE
VID_20230425_145407.vmt	1.280	0.050	4
VID_20230425_151801.vmt	n/a	-	-
VID_20230425_171532.vmt	1.381	0.052	4
VID_20230426_120410.vmt	1.584	0.188	4
VID_20230426_130526.vmt	1.593	0.089	4
VID_20230426_150321.vmt	1.673	0.065	4

Table 10 — Moto G30 Synchronization Metrics

WEBVMT FILE	MEAN TIME OFFSET	STANDARD DEVIATION	SAMPLE SIZE
VID_20230425_145232.vmt	-1.755	-	1
VID_20230425_145348.vmt	-2.263	-	1
VID_20230425_150129.vmt	-1.915	0.154	2
VID_20230425_151132.vmt	-2.107	-	1
VID_20230425_151535.vmt	n/a	-	-
VID_20230425_171521.vmt	-2.278	0.138	4

WEBVMT FILE	MEAN TIME OFFSET	STANDARD DEVIATION	SAMPLE SIZE
VID_20230426_120319.vmt	-1.577	0.384	4
VID_20230426_130701.vmt	-1.959	0.052	4
VID_20230426_150316.vmt	-1.367	0.044	4

The best results represent a statistical error of less than 100ms (3 frames) with a 95% confidence level. In terms of road vehicles, this equates to a distance error of less than 1.3m at 50km/h (30mph) or 3.1m at 110km/h (70mph).

The accuracy of these results is confirmed by the previsualization screenshots shown in Synchronized Previsualization.

7.5.2. Wrong-Way Traffic Use Case

The scheme devised to track passing vehicles from video footage could be easily extended to detect wrong-way traffic.

Target location zones are sufficiently small to allow discrimination of individual lanes on a road. Using three or more zones per lane enables an automated detection routine to reliably determine in which direction a vehicle is traveling, and to discriminate between adjoining lanes which may have traffic flowing in opposite directions.

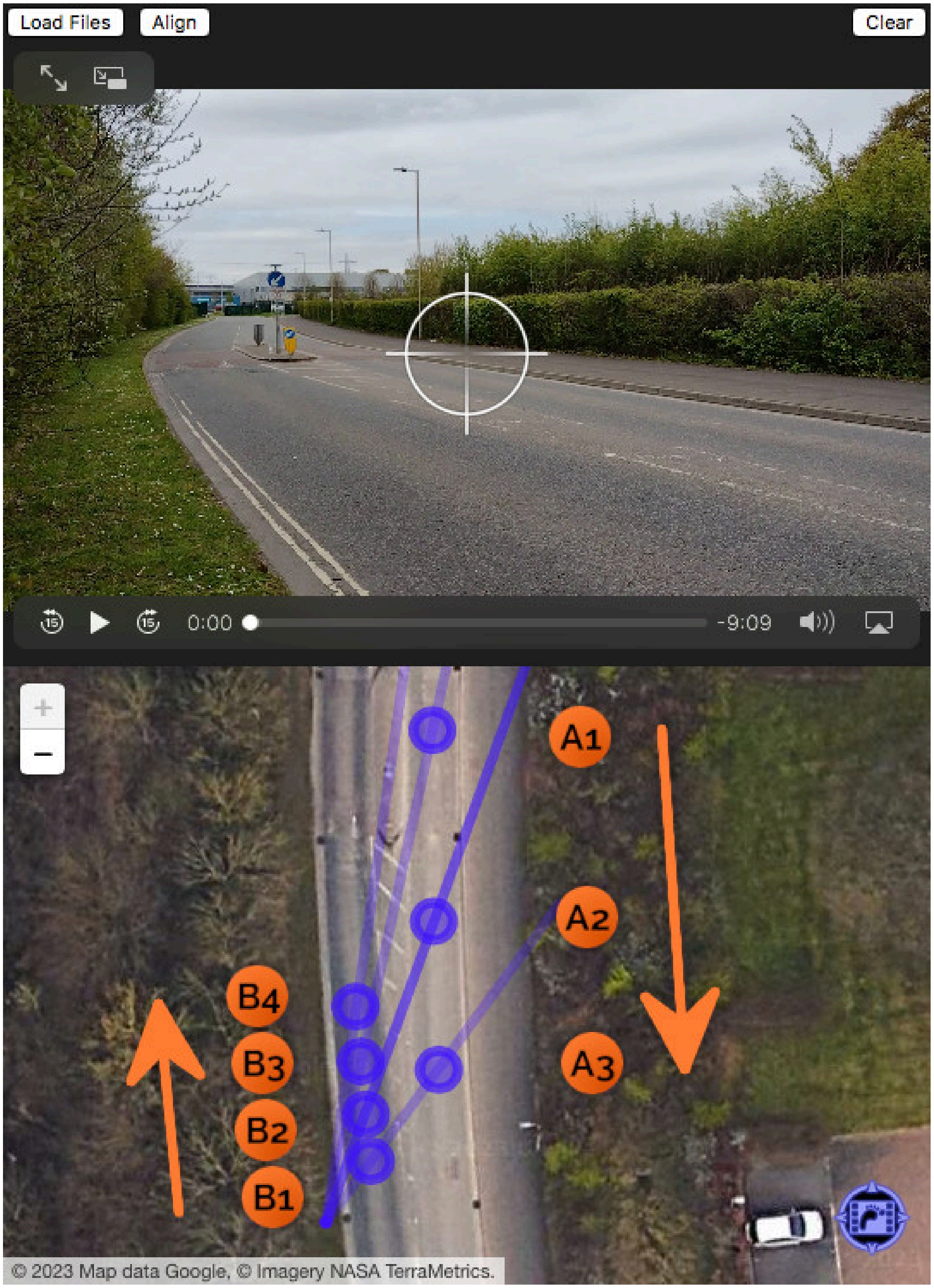


Figure 48 — Target Zones To Detect Wrong-Way Traffic

Figure 48 shows the target location zones and lines of sight in blue. Detection labels for the zones in lanes A and B are shown in orange, and the arrows show the correct direction of traffic flow.

The expected sequence of detections for traffic traveling in the correct direction is either A1→A2→A3 or B1→B2→B3→B4 in this case — since vehicles drive on the left in the UK. Wrong-way traffic would be detected by either A3→A2→A1 or B4→B3→B2→B1. This system could be easily adapted for use on a multi-lane road such as a motorway but could also be used to police a no-overtaking zone on a two-way road.

An unexpected result was discovered during analysis which demonstrates the strength of this traffic monitoring scheme. The StreetDrone performed a U-turn which was captured by one of the static cameras. Although the algorithm was not designed to identify this maneuver, the incident was detected since the detection zones were triggered in an erroneous order that correctly indicates the event and demonstrates the value of this simple approach.

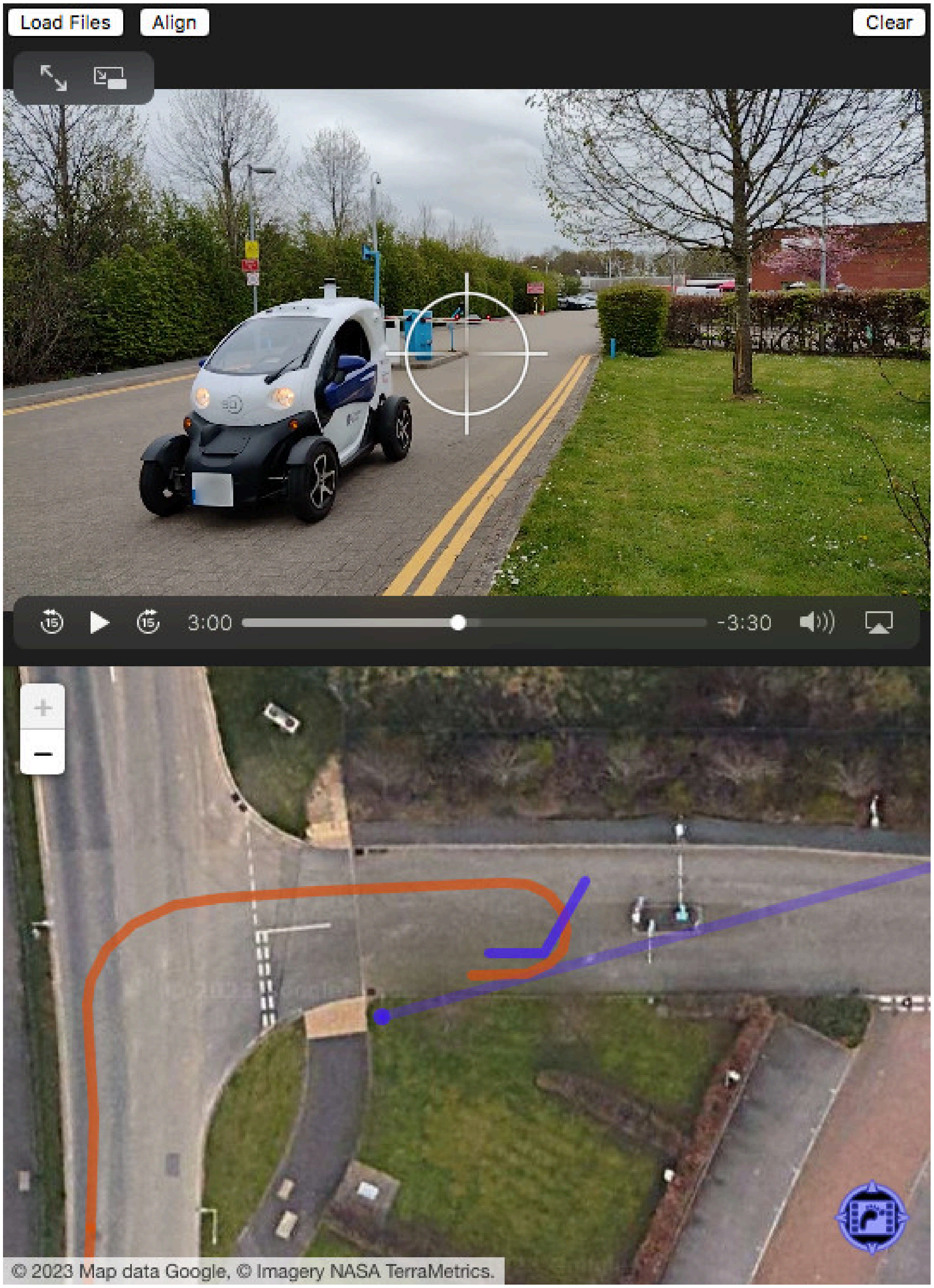


Figure 49 — Unexpected U-Turn Detection

7.5.3. Vehicle Speed Forensics Use Case

Another application for sight line analysis techniques was highlighted by an episode of the BBC Crash Detectives documentary about the Gwent Police Forensics Collision Investigation team — first broadcast on 9 October 2023.

In the first episode of the fourth series, the team investigated a hit and run incident in which a man died on a Welsh mountain road in the early hours of the morning. Although there were no witnesses or CCTV cameras, officers were made aware of video footage from a home security system located about 1km away on the other side of the valley. The cameras had recorded timestamped video which included car headlights traveling along the relevant road in the distance.

Initial police analysis of the night-time footage was unpromising, but then a daytime image was obtained from the same camera which they overlaid to show reference points in the scene. This approach allowed officers to estimate the vehicle's speed at the time of the accident.

The problem seemed well-suited to the techniques developed during this Testbed, so the vehicle's speed was estimated using WebVMT analysis in order to demonstrate how easily this could be applied to an operational police issue.

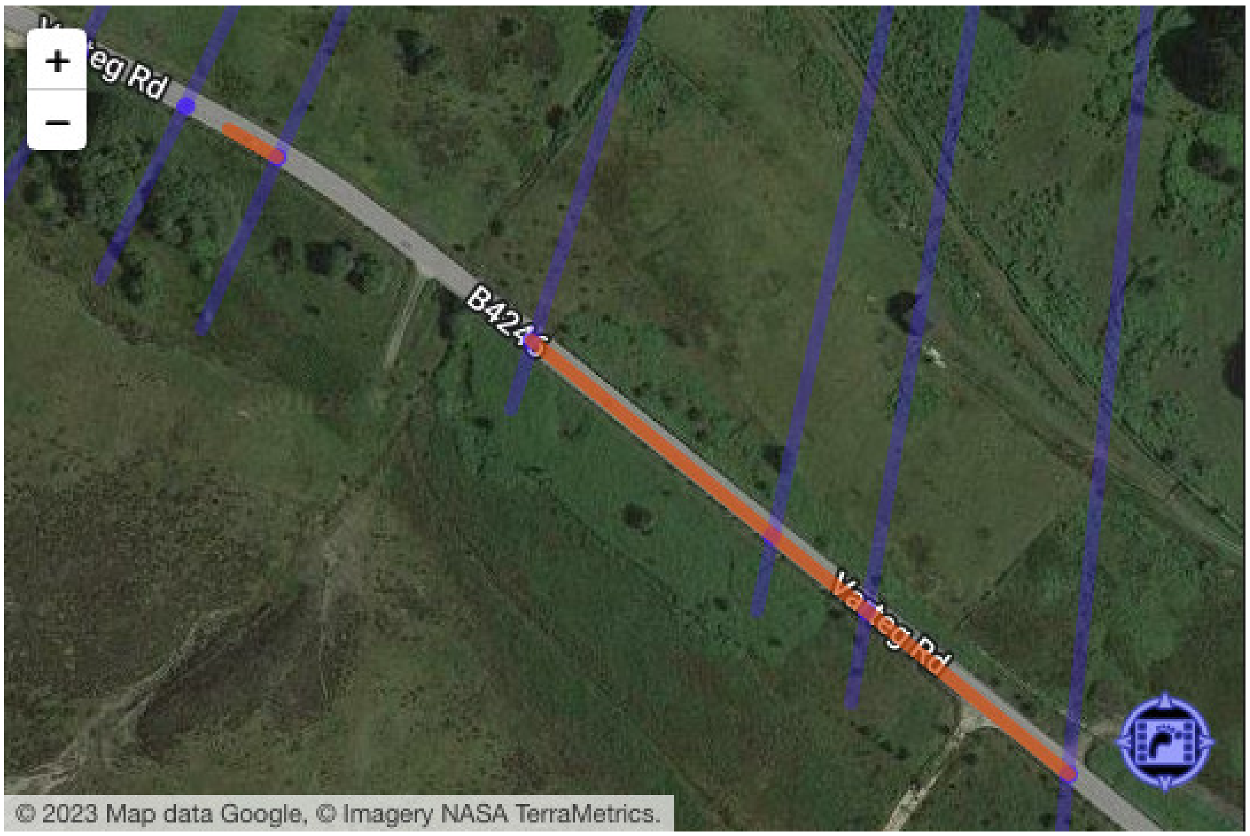


Figure 50 — Crash Detectives WebVMT Analysis

Two full-screen CCTV clips from the show were analyzed. The camera location was accurately identified to within 1m from freely available web maps to establish relevant sight lines and detection zones on the road.

Table 11 — Vehicle Speed Analysis

VIDEO CLIP	ELAPSED TIME	DISTANCE TRAVELLED	AVERAGE SPEED
1	15.232s	222m	52.5km/h, *32.6mph*
2	2.352s	34m	52.0km/h, *32.3mph*

Results were calculated for two segments of the observed journey based on geospatial analysis of WebVMT data using the same sight line techniques previously described.

7.6. Conclusions

Testbed-19 participant Away Team successfully demonstrated how moving objects can be used to synchronize data recorded by different devices with a high level of accuracy substantiated by quantifiable errors. This accurate synchronization enables aggregation of data from low-cost commercial video devices including dashcams, drones, body-worn cameras, and smartphones that are geospatially-aware.

A process was identified that supports automated detection of wrong-way vehicles using roadside traffic cameras, which is an operational requirement for national road network agencies. In addition, passing fleet vehicles can be used to synchronize these observations with UTC in order to aggregate video and enrich other traffic monitoring data.

Sight line techniques were successfully applied to track vehicles and determine the vehicle's speeds from CCTV footage by geotagging video displayed in a web browser demonstrating how footage can be augmented with geospatial data to leverage online technologies through web integration and can streamline operational use cases for police investigators. WebVMT's text-based format has been used to facilitate access to timed video metadata and help guide analysis in a quick and verifiable way.

GeoPose and WebVMT analysis tools developed in previous Testbeds have been refined and enhanced to address these new challenges highlighting the value of geospatial interoperability and the benefits of modular software that can be reused in future studies. Several new WebVMT features were implemented as prototypes in this study which include simplified data interpolation, heading, and negative cue times. This demonstrates the importance of operational use cases and how Testbed innovation helps to ensure that OGC designs meet real world requirements.

There was insufficient time in Testbed-19 to study the litter monitoring use case, though the data have been captured and this analysis could be deferred to Testbed-20 as described in Future Work.

7.7. Future Work

7.7.1. Litter Monitoring Use Case

Away Team captured data for the litter monitoring use case in April 2023, which could be studied in Testbed-20 since there was insufficient time to address this in Testbed-19.

Litter locations were identified and exported to WebVMT which could be matched against video and LiDAR data recorded by the StreetDone. Analysis could be performed using WebVMT tools developed during Testbed-17 for tracking static objects from a moving vehicle to monitor litter locations.

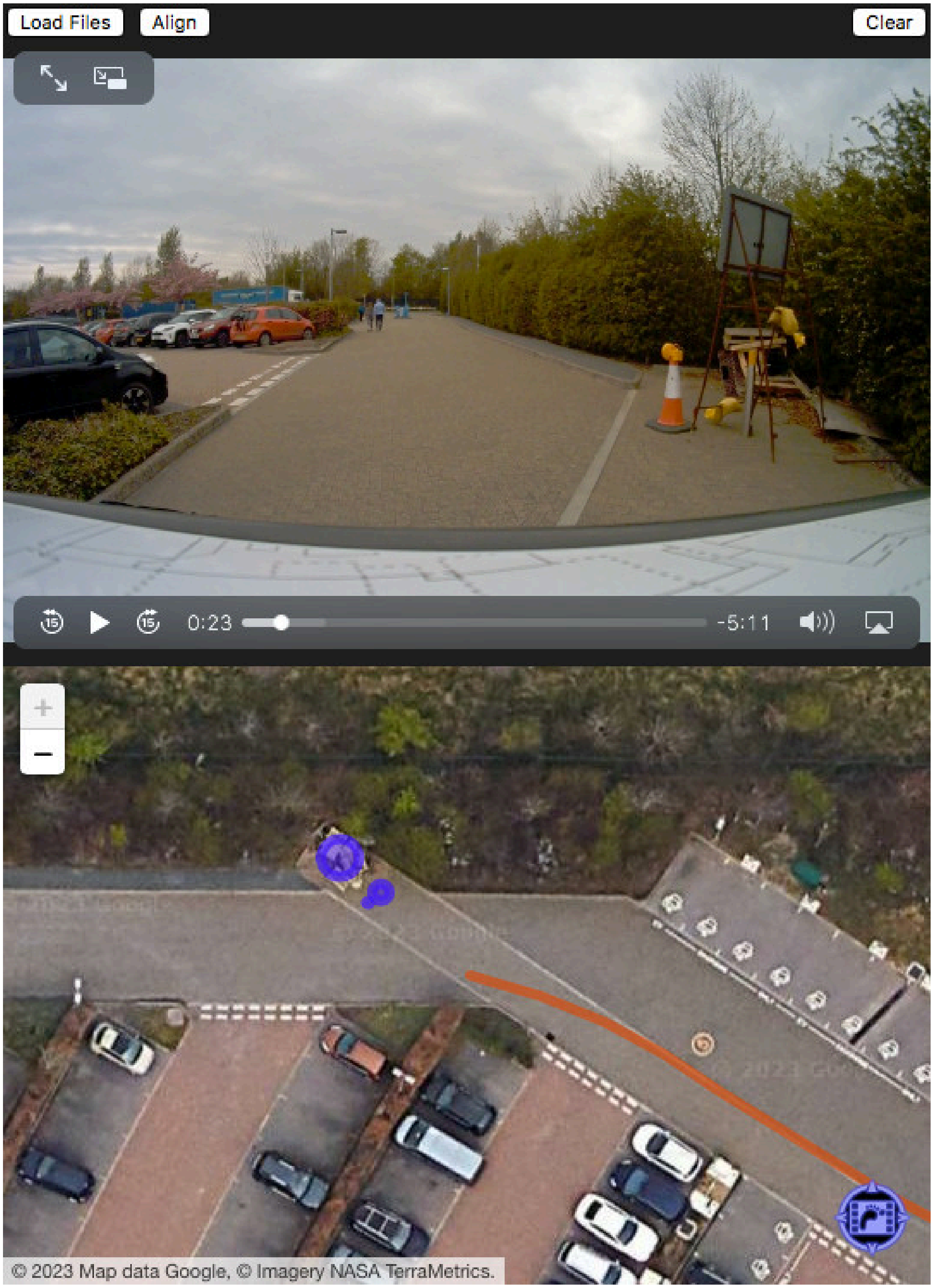


Figure 51 — Identified Locations For Litter Monitoring Use Case

In addition, video was concurrently captured by StreetDrone’s front- and rear-facing cameras to enable multi-view data aggregation. Hence the risk of other road vehicles obstructing the survey vehicle’s view of a target location during a single pass can be mitigated in operational use.

Synchronizing data gathered by different devices and aggregating data from multiple passes of the same target location enables accretion rates to be calculated over time enabling more accurate monitoring and prediction of litter levels so maintenance teams can be deployed more efficiently.

8

CONCLUSIONS

8 CONCLUSIONS

Conclusions of this Engineering Report fall into three categories:

1. Clause 8.1
2. Clause 8.2
3. Clause 8.3

8.1. ISO 19111 Evaluation

OGC Testbed-18 (TB-18) proposed extensions to OGC Standards for use beyond planet Earth. TB-18 also provided an executable prototype, but without testing the proposed extensions. Instead, the prototype “abused” existing structures such as `EngineeringCRS`. This TB-19 ER corrects this shortcoming by formalizing the proposed extensions in code snippets that can be submitted to OGC Standard Working Groups, and by implementing those extensions in prototypes.

A first key result is that most ISO 19111 data structures for CRS definitions are suitable for objects in space, but a few additions and modifications are desirable. Additions such as `InertialCRS` and `MinkowskiCS` (Clause 3.2.1) can be done in a separated standard. Modifications such as associating datum to `CelestialBody` or retrofitting `ObjectUsage` into `IdentifiedObject` (Clause 3.2.3) would require changes to the ISO 19111 standard itself.

A second key result is that ISO 19111 data structures for `CoordinateOperation` are suitable with no change for transforming positions in space. ISO 19111 provides a powerful framework for defining coordinate operations, chaining the operations together, identifying which steps may introduce stochastic errors, expressing domain of validity and uncertainty, and associating (source, target) pairs of CRS to coordinate operations, e.g., using a registry (Clause 3.3). This framework is already extensively used in the EPSG geodetic parameter dataset (the fact that EPSG provides not only CRS definitions, but also coordinate operations relating different CRSs together is not well known) and in software such as implementations of the OGC GeoAPI Standard, Apache SIS, and PROJ 6+.

A third key result is that while WKT is a well-known encoding for the CRS part of ISO 19111, the encoding of the coordinate operation part of ISO 19111 has not been as widely adopted. WKT 2 has some support for coordinate operations. However, this is not as well-known as the CRS part. Further, WKT 2 lacks some ISO 19111 features such as pass-through operations (Figure 14), limits itself to the most important metadata for simplicity reasons, and causes some repetitions because it does not have something equivalent to the GML `xlink:href` attribute (OGC 22-038r2 section 8.7). On the other hand, GML + Annex B.1 have all the capabilities needed for Testbed-19, but few software packages support GML to the extent needed. The open source libraries providing sufficient GML support (e.g., Apache SIS) are not the most popular ones (e.g., PROJ or GDAL).

A fourth key result is that GeoPose provides a coordinate operation framework that differs significantly from the ISO 19111 framework. However, the GeoPose framework can be mapped to a subset of ISO 19111 framework (Clause 4.7) except for two features that are unique to GeoPose: Short-lived temporary identifiers, and transformations of orientations in addition of positions. Except for the orientation concept, the GeoPose framework is closer to compact encoding considerations (e.g., “idiomatic transforms” as shortcuts of “templated transforms”). The ISO 19111 framework is closer to conceptual considerations (e.g., whether the coordinate operation may imply stochastic errors). GeoPose provides a JSON encoding which may be easier to manipulate than GML. However, no comparison with a JSON encoding of ISO 19111, such as PROJ-JSON, were made in Testbed-19.

8.2. GeoPose Extensions

The GeoPose extensions are summarized in the GeoPose section of this report as the “NeoPose” proposal.

These extensions were implemented and integrated into a two-day experimental capture of real-world imagery and navigation data as observed by:

- front and rear of a car;
- two people (the riders) with body-mounted cameras;
- a handheld camera;
- two stationary cameras;
- an autonomous aerial vehicle; and
- a simulated spacecraft in low earth orbit.

The “Hillyfields Bubble” is a spherical volume centered on the headquarters of the UK Ordnance Survey in Southampton, UK. The bubble has a diameter of 1 km. The captured data represent the background environment and activities of moving objects during multiple repetitions of the staged rendezvous of a pedestrian and a car dispatched by a ride-hailing service. The captured observations were incorporated into several investigations, including the experimental implementation of the NeoPose proposal in Testbed-19 and the prototyping of visual positioning and other services by the Metaverse Standards Forum Real/Virtual Domain Working Group.

A significant element of the initial processing of the image data was to build an image processing pipeline to support detection of vehicles and people. Once detected, the data were converted to GeoPoses, using the extensions. Development of secondary processing with the goal of automated construction of pose graphs from image streams, also using the extensions, is ongoing. An important element of the graph construction is recognizing that multiple poses observe the same real-world entity. These poses may be observed by the same sensor at different times or different sensors at the same time. The approach validated in this Testbed depends on the use of time-limited unique identifiers and confidence measures for the belief that pairs of unique identifiers refer to the same underlying real-world entity.

Testbed-19 also provided an opportunity to extend the GeoPose paradigm to a more general model for implementation of the “position plus orientation” model. This extension is based on representing the “position” part as the frame transformation that links an event in an Inner frame to some Outer frame that is defined by an external scheme or standard, as in OGC GeoPose 1.0. The “orientation” part becomes a second “rotation-like” transform within the Inner frame. That this supports observation and pointing compatible with special relativity and space-time with a Minkowski metric was demonstrated.

8.3. Video Synchronization With GeoPose

Away Team successfully demonstrated how moving objects can be used to synchronize data recorded by different devices with a high level of accuracy substantiated by quantifiable errors. This accurate synchronization enables aggregation of data from low-cost commercial video devices including dashcams, drones, body-worn cameras, and smartphones that are geospatially-aware.

A process was identified that supports automated detection of wrong way vehicles using roadside traffic cameras, which is an operational requirement for national road network agencies. In addition, passing fleet vehicles can be used to synchronize these observations with UTC in order to aggregate video and enrich other traffic monitoring data.

Sight line techniques have been successfully applied to track vehicles and determine their speed from CCTV footage by geotagging video displayed in a web browser. This demonstrates how footage can be augmented with geospatial data to leverage online technologies through web integration and can streamline operational use cases for police investigators. WebVMT’s text-based format has been used to facilitate access to timed video metadata and help guide analysis in a quick and verifiable way.

GeoPose and WebVMT analysis tools developed in previous Testbeds have been refined and enhanced to address these new challenges highlighting the value of geospatial interoperability and the benefits of modular software that can be reused in future studies. Several new WebVMT features were implemented as prototypes in this study which include simplified data interpolation, heading, and negative cue times. This demonstrates the importance of operational use cases and how Testbed innovation helps to ensure that OGC designs meet real world requirements.

A

ANNEX A (INFORMATIVE) ABBREVIATIONS/ACRONYMS

A ANNEX A (INFORMATIVE) ABBREVIATIONS/ACRONYMS

1D	One-dimensional
3D	Three-dimensional
4D	Four-dimensional
CRS	Coordinate Reference System
CS	Coordinate System
CSV	Comma Separated Values
ECEF	Earth-Centered — Earth-Fixed
ECI	Earth-Centered Inertial
GML	Geographic Markup Language
ICRF	International Celestial Reference Frame
IMU	Inertial Measurement Unit
ISO	International Organization for Standardization
MF	Moving Features
OGC	Open Geospatial Consortium
UML	Unified Modeling Language
UTC	Universal Time Coordinated
WebVMT	Web Video Map Tracks
WKT	Well-Known Text
NOTE	in some contexts, CRS is the abbreviation of "Celestial Reference System" instead of "Coordinate Reference System."

B

ANNEX B (INFORMATIVE) STANDARD PROPOSALS FOR XML

B ANNEX B (INFORMATIVE) STANDARD PROPOSALS FOR XML

This section submits two files that can be used as starting points for standardization by relevant Working Groups. The first file is an XSD schema for Coordinate Reference System definitions. The second file is an XML file defining a coordinate operation method for simple coordinate transformations using Moving Feature files. Those files, together with examples, are available on GitHub.

B.1. XML schema for CRS

The following XML schema defines a GML extension with the new classes introduced in Clause 3.2.1. All GML extensions are done in the `http://www.opengis.net/gsp/1.0` namespace (non-standardized) and encoded in a single XML schema file named `geospatialInSpace.xsd`. This schema can be used as a starting point for standardization.

This schema also contains one additional class, namely `TriaxialEllipsoid`, which is not listed in Clause 3.2.1 because ISO 19111:2019 already supports triaxial ellipsoids, but the standard GML schema is still based on the older ISO 19111:2007 model. Consequently `gsp:TriaxialEllipsoid` has to be added as an extension of `gml:Ellipsoid`, but only for the XML schema. This approach should not be replicated in abstract models because the way that `TriaxialEllipsoid` is defined below does not follow the Liskov Substitution Principle.

```
<?xml version="1.0"?>
<schema version            = "1.0.0"
        targetNamespace    = "http://www.opengis.net/gsp/1.0"
        xmlns:gsp          = "http://www.opengis.net/gsp/1.0"
        xmlns:gml          = "http://www.opengis.net/gml/3.2"
        xmlns              = "http://www.w3.org/2001/XMLSchema"
        xml:lang           = "en"
        elementFormDefault = "qualified">

  <annotation>
    <appinfo source="urn:ogc:specification:gml:schema-xsd:geospatialInSpace.
xsd:1.0.0">
        geospatialInSpace.xsd
    </appinfo>
    <documentation>
      The Geospatial in Space schema is an extension to the Coordinate
Reference System
      components defined in the GML schema.
      This schema is not an OGC Standard but is part of an OGC Public
Engineering
      Report created as a deliverable in an OGC Interoperability Initiative
and is not an
      official position of the OGC membership. It is distributed for review
and comment.
      It is subject to change without notice and may not be referred to as an
OGC Standard.
    </documentation>
  </annotation>
  <import namespace="http://www.opengis.net/gml/3.2"
      schemaLocation="https://schemas.opengis.net/gml/3.2.1/
coordinateReferenceSystems.xsd"/>


  <!--
    Coordinate System
  -->
  <element name="MinkowskiCS" type="gsp:MinkowskiCSType" substitutionGroup=
"gml:AbstractCoordinateSystem">
    <annotation>
      <documentation>
        gsp:MinkowskiCS is a 4-dimensional coordinate system.
        The three spacial dimensions give the position of points relative to
orthogonal straight axes.
        The fourth dimension gives the position in time since an epoch.
        All axes shall have the same length unit of measure,
        with time expressed as the distance covered by light in the vacuum
during the elapsed time.
        A MinkowskiCS shall have exactly four gml:axis property elements.
      </documentation>
    </annotation>
  </element>
  <complexType name="MinkowskiCSType">
    <complexContent>
      <extension base="gml:AbstractCoordinateSystemType"/>
    </complexContent>
  </complexType>
  <complexType name="MinkowskiCSPropertyType">
    <annotation>
      <documentation>
        gsp:MinkowskiCSPropertyType is a property type for association roles
to a Minkowski coordinate system,
        either referencing or containing the definition of that coordinate
system.
      </documentation>
    </annotation>
    <sequence minOccurs="0">
      <element ref="gsp:MinkowskiCS"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
  </complexType>


  <!--
    Triaxial ellipsoid
  -->
  <element name="TriaxialEllipsoid" type="gsp:TriaxialEllipsoidType"
substitutionGroup="gml:Ellipsoid">
    <annotation>
      <documentation>
        gsp:TriaxialEllipsoid is an ellipsoid described using an additional
semi-median axis attribute.
```

```
        This attribute is for planetary applications and is not used when
describing a bi-axial oblate
        reference ellipsoid model of the Earth. For a triaxial reference
ellipsoid it is usual for the
        secondDefiningParameter to be the ellipsoid's semi-minor axis.
      </documentation>
    </annotation>
  </element>
  <complexType name="TriaxialEllipsoidType">
    <complexContent>
      <extension base="gml:EllipsoidType">
        <sequence>
          <element ref="gsp:semiMedianAxis"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
  <element name="semiMedianAxis" type="gml:MeasureType">
    <annotation>
      <documentation>
        gsp:semiMedianAxis specifies the length of the semi-median axis of the
ellipsoid, with its units
        and uses the MeasureType with the restriction that the unit of measure
referenced by uom must be suitable for a length,
        such as meters or feet.
      </documentation>
    </annotation>
  </element>
  <complexType name="TriaxialEllipsoidPropertyType">
    <annotation>
      <documentation>
        gsp:TriaxialEllipsoidPropertyType is a property type for association
roles to a triaxial ellipsoid,
        either referencing or containing the definition of that ellipsoid.
      </documentation>
    </annotation>
    <sequence minOccurs="0">
      <element ref="gsp:TriaxialEllipsoid"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
  </complexType>


  <!--
    Datum
  -->
  <element name="CelestialBody" type="gsp:CelestialBodyType" substitutionGroup=
"gml:Definition">
    <annotation>
      <documentation>
        A gsp:CelestialBody is a name or identifier of the star, planet,
asteroid, or other celestial body
        for which a reference frame is defined.
      </documentation>
    </annotation>
  </element>
  <complexType name="CelestialBodyType">
    <complexContent>
      <extension base="gml:IdentifiedObjectType">
        <!-- We should put a reference to ISO 19112 here, but there is
apparently no XML schema for it. -->
      </extension>
    </complexContent>
  </complexType>
  <complexType name="CelestialBodyPropertyType">
    <annotation>
      <documentation>
        gsp:CelestialBodyPropertyType is a property type for association roles
to a celestial body,
        either referencing or containing the definition of that celestial body.
      </documentation>
    </annotation>
    <sequence minOccurs="0">
      <element ref="gsp:CelestialBody"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
  </complexType>

  <element name="InertialReferenceFrame" type="gsp:InertialReferenceFrameType"
substitutionGroup="gml:AbstractDatum">
    <annotation>
      <documentation>
        gsp:InertialReferenceFrame is an inertial datum defining the precise
location and orientation
        in 3- or 4-dimensional space of a defined ellipsoid (or sphere), or of
a Cartesian (3D case),
        or of a Minkowski (4D case) coordinate system centered in the
ellipsoid (or sphere).
      </documentation>
    </annotation>
  </element>
  <complexType name="InertialReferenceFrameType">
    <complexContent>
      <extension base="gml:AbstractDatumType">
        <sequence>
          <element ref="gsp:celestialBody"  minOccurs="0"/>
          <element ref="gml:ellipsoid"      minOccurs="0"/>
          <element ref="gsp:primeDirection" minOccurs="0"/>
          <!--
            Departures from gml:GeodetiscDatum:
            - Ellipsoid became optional in ISO 19111:2019.
            - Prime meridian interpreted as prime direction.
            - Elements in different order, from most important to less
relevant (for an inertial CRS).
              This order is also for easier migration if a future GML version
declares celestial body.
          -->
        </sequence>
      </extension>
    </complexContent>
  </complexType>
  <element name="celestialBody" type="gsp:CelestialBodyPropertyType">
    <annotation>
      <documentation>gsp:celestialBody is an association role to the celestial
body used by this inertial datum.</documentation>
    </annotation>
  </element>
  <element name="primeDirection" type="gml:PrimeMeridianPropertyType">
    <annotation>
      <documentation>
        gsp:primeDirection is an association role to the prime meridian of an
inertial datum.
        The prime direction is the direction of the 0° of longitude,
identified by a distant feature such as a star.
      </documentation>
    </annotation>
  </element>
  <complexType name="InertialReferenceFramePropertyType">
    <annotation>
      <documentation>
        gsp:InertialReferenceFramePropertyType is a property type for
association roles to a inertial datum,
        either referencing or containing the definition of that datum.
      </documentation>
    </annotation>
      <sequence minOccurs="0">
        <element ref="gsp:InertialReferenceFrame"/>
      </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
  </complexType>


  <!--
    Coordinate Reference System
  -->
  <element name="InertialCRS" type="gsp:InertialCRSType" substitutionGroup=
"gml:AbstractSingleCRS"/>
  <complexType name="InertialCRSType">
    <annotation>
      <documentation>gsp:InertialCRS is a coordinate reference system with
axes having fixed directions relative to stars.</documentation>
    </annotation>
    <complexContent>
      <extension base="gml:AbstractCRSType">
        <sequence>
          <choice>
            <element ref="gml:ellipsoidalCS"/>
            <element ref="gml:cartesianCS"/>
            <element ref="gml:sphericalCS"/>
            <element ref="gsp:minkowskiCS"/>
          </choice>
          <element ref="gsp:inertialReferenceFrame"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
  <element name="minkowskiCS" type="gsp:MinkowskiCSPropertyType">
    <annotation>
      <documentation>gsp:minkowskiCS is an association role to the Minkowski
coordinate system used by this CRS.</documentation>
    </annotation>
  </element>
  <element name="inertialReferenceFrame" type="gsp:InertialReferenceFrameProper
tyType">
    <annotation>
      <documentation>gsp:inertialReferenceFrame is an association role to the
inertial datum used by this CRS.</documentation>
    </annotation>
  </element>
  <complexType name="InertialCRSPropertyType">
    <annotation>
      <documentation>gsp:InertialCRSPropertyType is a property type
for association roles to a inertial coordinate reference system, either
referencing or containing the definition of that reference system.</
documentation>
    </annotation>
    <sequence minOccurs="0">
      <element ref="gsp:InertialCRS"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
  </complexType>
</schema>
```

Figure B.1 — XML schema proposal for standardization by a SWG

B.1.1. Examples

Testbed 19 participants defined three example files compliant with the above XSD schema: `Didymos.xml`, `Dimorphos.xml`, and `SolarSystemBarycenter.xml`. These are provided in the `examples` sub-directory. The context for those examples is presented in Clause 3.4. Below is the example defining the Dimorphos CRS, with the Testbed-19 proposed extensions in the `gsp` namespace.

```
<?xml version="1.0" encoding="UTF-8"?>
<gsp:InertialCRS xsi:schemaLocation = "http://www.opengis.net/gsp/1.0 ../
geospatialInSpace.xsd"
                 xmlns:gsp          = "http://www.opengis.net/gsp/1.0"
                 xmlns:gml          = "http://www.opengis.net/gml/3.2"
                 xmlns:xsi          = "http://www.w3.org/2001/XMLSchema-
instance"
                 xmlns:xlink        = "http://www.w3.org/1999/xlink"
                 gml:id             = "DimorphosCRS">

  <gml:description>
    This GML file describes an inertial Coordinate Reference System (CRS) for
the Dimorphos moonlet.
    Dimorphos is a satellite of the Didymos asteroid, which has a CRS defined
in a separated file.
    Those two CRSs use the same three-dimensional Cartesian coordinate system,
but with different
    origin (not encoded in this file). The axes orientations are fixed
relative to distant stars.
  </gml:description>
  <gml:identifier codeSpace="JPL:HORIZONS">urn:ogc:def:crs:JPL::120065803</gml:
identifier>
  <gml:name codeSpace="JPL:HORIZONS">Dimorphos satellite (inertial)</gml:name>
  <gml:remarks>
    This CRS definition uses new objects proposed by Testbed-19 with the "gsp"
prefix.
    The JPL and IAU namespaces used in this file are not for official JPL/IAU
codes.
    Those namespaces are used only for demonstrating what the identifiers may
look
    like if JPL and IAU published CRS definitions as GML objects.
  </gml:remarks>
  <gml:scope>For use with the reconstructed trajectory of the Didymos system.</
gml:scope>
  <gml:cartesianCS>
    <gml:CartesianCS gml:id="DimorphosCS">
      <gml:identifier codeSpace="JPL:HORIZONS">urn:ogc:def:cs:JPL::ICRF</gml:
identifier>
      <gml:name>Cartesian 3D CS (km).</gml:name>
      <gml:axis>
        <gml:CoordinateSystemAxis gml:id="axisX" uom="urn:ogc:def:uom:EPSG:
:9036">
          <gml:identifier codeSpace="JPL:HORIZONS">urn:ogc:def:axis:JPL::ICRF-X
</gml:identifier>
          <gml:name>X</gml:name>
          <gml:axisAbbrev>X</gml:axisAbbrev>
          <gml:axisDirection codeSpace="JPL:HORIZONS">geocentricX</gml:
axisDirection>
        </gml:CoordinateSystemAxis>
      </gml:axis>
      <gml:axis>
        <gml:CoordinateSystemAxis gml:id="axisY" uom="urn:ogc:def:uom:EPSG:
:9036">
          <gml:identifier codeSpace="JPL:HORIZONS">urn:ogc:def:axis:JPL::ICRF-Y
</gml:identifier>
          <gml:name>Y</gml:name>
          <gml:axisAbbrev>Y</gml:axisAbbrev>
          <gml:axisDirection codeSpace="JPL:HORIZONS">geocentricY</gml:
axisDirection>
        </gml:CoordinateSystemAxis>
      </gml:axis>
      <gml:axis>
        <gml:CoordinateSystemAxis gml:id="axisZ" uom="urn:ogc:def:uom:EPSG:
:9036">
          <gml:identifier codeSpace="JPL:HORIZONS">urn:ogc:def:axis:JPL::ICRF-Z
</gml:identifier>
          <gml:name>Z</gml:name>
          <gml:axisAbbrev>Z</gml:axisAbbrev>
          <gml:axisDirection codeSpace="JPL:HORIZONS">geocentricZ</gml:
axisDirection>
        </gml:CoordinateSystemAxis>
      </gml:axis>
    </gml:CartesianCS>
  </gml:cartesianCS>
  <gsp:inertialReferenceFrame>
    <gsp:InertialReferenceFrame gml:id="DimorphosDatum">
      <gml:identifier codeSpace="JPL:HORIZONS">urn:ogc:def:datum:JPL::120065803
</gml:identifier>
      <gml:name>Dimorphos inertial reference frame</gml:name>
      <gml:scope>For orbit positionning.</gml:scope>
      <gsp:celestialBody>
        <gsp:CelestialBody gml:id="Dimorphos">
          <gml:identifier codeSpace="JPL:HORIZONS">urn:ogc:def:body:JPL:
:120065803</gml:identifier>
          <gml:name>Dimorphos</gml:name>
        </gsp:CelestialBody>
      </gsp:celestialBody>
      <gml:ellipsoid>
        <gsp:TriaxialEllipsoid gml:id="DimorphosEllipsoid">
          <gml:identifier codeSpace="JPL:HORIZONS">urn:ogc:def:ellipsoid:JPL:
:120065803</gml:identifier>
          <gml:name>Dimorphos ellipsoid</gml:name>
          <gml:semiMajorAxis uom="urn:ogc:def:uom:EPSG::9001">88.5</gml:
semiMajorAxis>
          <gml:secondDefiningParameter>
            <gml:SecondDefiningParameter>
              <gml:semiMinorAxis uom="urn:ogc:def:uom:EPSG::9001">58</gml:
semiMinorAxis>
            </gml:SecondDefiningParameter>
          </gml:secondDefiningParameter>
          <gsp:semiMedianAxis uom="urn:ogc:def:uom:EPSG::9001">87</gsp:
semiMedianAxis>
        </gsp:TriaxialEllipsoid>
      </gml:ellipsoid>
      <!-- Prime meridian is optional for an inertial reference frame. -->
    </gsp:InertialReferenceFrame>
  </gsp:inertialReferenceFrame>
```

```
</gsp:InertialCRS>
```

Figure B.2 — Definition of Didymos inertial CRS in extended GML

B.2. GML for operation methods

The following XML file defines an operation method for coordinate transformations using a trajectory file. Such an operation method, if approved, could be published on http://www.opengis.net/def/method/OGC/0/.

```
<?xml version="1.0" encoding="UTF-8"?>
<gml:OperationMethod xsi:schemaLocation = "http://www.opengis.net/gml/3.2 http:
//schemas.opengis.net/gml/3.2.1/gml.xsd"
                     xmlns:gml          = "http://www.opengis.net/gml/3.2"
                     xmlns:xsi          = "http://www.w3.org/2001/XMLSchema-
instance"
                     xmlns:xlink        = "http://www.w3.org/1999/xlink"
                     gml:id             = "TranslationByTrajectory">

  <gml:description>

   Non-relativist time-varying translations defined by a Moving Feature file.
    This method is applied by coordinate operations transforming coordinate
tuples from one reference frame to
    another reference frame, when the latter is moving along a trajectory with
coordinates expressed relatively
    to the former reference frame. This simple operation method assumes that
the two reference frames share the
    same axis orientations, so no yaw, pitch, or roll is used. The
transformation is applied by subtracting the
    (x,y,z) coordinates interpolated in the file at time t, possibly after
conversion from interpolation CRS.
    The CRS declared in the moving feature file is taken as the interpolation
CRS.
  </gml:description>
  <gml:identifier codeSpace="OGC">http://www.opengis.net/def/method/OGC/1/200</
gml:identifier>
  <gml:name codeSpace="OGC">Translation by trajectory</gml:name>
  <gml:remarks>
    This definition is not yet an OGC standard. It is published for comments.
  </gml:remarks>
  <gml:formula>
    For each (x,y,z,t) source coordinate tuples, interpolate the position in
the trajectory at time t.
    Then, subtract the interpolated position to the source (x,y,z) coordinates
and pass the value of t unchanged.
  </gml:formula>
  <gml:sourceDimensions>4</gml:sourceDimensions>
  <gml:targetDimensions>4</gml:targetDimensions>
  <gml:parameter>
    <gml:OperationParameter gml:id="TrajectoryFile">
      <gml:description>
        Trajectory of the target frame relative to the source frame.
        Shall be a Moving Feature file in a format defined by OGC: JSON, CSV,
XML or netCDF.
      </gml:description>
      <gml:identifier codeSpace="OGC">http://www.opengis.net/def/parameter/OGC/
1/200</gml:identifier>
```

```
      <gml:name>Trajectory file</gml:name>
      <gml:remarks>
        For the Testbed-19 prototype, only Moving Feature CSV encoding is
supported.
      </gml:remarks>
    </gml:OperationParameter>
  </gml:parameter>
</gml:OperationMethod>
```

Figure B.3 — Definition of "translation by trajectory" operation method

The value of the “Trajectory file” parameter is a relative file. The above proposal does not specify in which directory the file is found. The same ambiguity exists for datum shift files in the EPSG database for example. One possible policy, used in Annex B.3, is to decide that the “Trajectory file” parameter is a path relative to the GML file.

B.2.1. Examples

Below is the example defining a coordinate transformation from the solar system barycenter CRS to the Dimorphos CRS (not shown but similar to Figure B.2). Note that contrarily to the CRS definitions in Annex B.1, the following example does not need any extension to the GML schema.

```
<?xml version="1.0" encoding="UTF-8"?>
<gml:Transformation xsi:schemaLocation = "http://www.opengis.net/gml/3.2 http:
//schemas.opengis.net/gml/3.2.1/gml.xsd"
                    xmlns:gml          = "http://www.opengis.net/gml/3.2"
                    xmlns:xsi          = "http://www.w3.org/2001/XMLSchema-
instance"
                    xmlns:xlink        = "http://www.w3.org/1999/xlink"
                    gml:id             = "BarycenterToDimorphos">
  <gml:description>
    This GML file describes the transformation of coordinates from the CRS
    encoded in `SolarSystemBarycenter.xml` to the CRS encoded in `Dimorphos.
xml`.
  </gml:description>
  <gml:identifier codeSpace="TB19-D001">BarycenterToDimorphos</gml:identifier>
  <gml:name>Solar system barycenter to Dimorphos</gml:name>
  <gml:scope>Testbed 19 demonstration.</gml:scope>
  <gml:operationVersion>1</gml:operationVersion>
  <!--
    For this transformation, we need the time coordinate the be appended to
the inertial CRS.
    We could create separated files, but it is convenient to declare the
compound CRS just here.
  -->
  <gml:sourceCRS>
    <gml:CompoundCRS gml:id="SolarSystemCRS">
      <gml:identifier codeSpace="TB19-D001">SolarSystemCRS</gml:identifier>
      <gml:name>Solar system barycenter CRS + time</gml:name>
      <gml:scope>Testbed 19 demonstration.</gml:scope>
      <gml:componentReferenceSystem xlink:href="SolarSystemBarycenter.xml"/>
      <gml:componentReferenceSystem xlink:href="Time.xml"/>
    </gml:CompoundCRS>
  </gml:sourceCRS>
  <gml:targetCRS>
    <gml:CompoundCRS gml:id="DimorphosCRS">
      <gml:identifier codeSpace="TB19-D001">DimorphosCRS</gml:identifier>
      <gml:name>Dimorphos CRS + time</gml:name>
      <gml:scope>Testbed 19 demonstration.</gml:scope>
      <gml:componentReferenceSystem xlink:href="Dimorphos.xml"/>
      <gml:componentReferenceSystem xlink:href="Time.xml"/>
    </gml:CompoundCRS>
  </gml:targetCRS>
  <gml:method xlink:href="../translationByTrajectory.
xml#TranslationByTrajectory"/>
  <gml:parameterValue>
    <gml:ParameterValue>
      <gml:valueFile>BarycenterToDimorphos.csv</gml:valueFile>
      <gml:operationParameter xlink:href="../translationByTrajectory.
xml#TrajectoryFile"/>
    </gml:ParameterValue>
  </gml:parameterValue>
</gml:Transformation>
```

Figure B.4 — Definition of the coordinate transformation from Solar system barycenter CRS to Dimorphos CRS

The above coordinate transformation has only one parameter, the filename of a trajectory file, as defined in the Annex B.2 standard method proposal. The example below shows the first lines of this file. The format is OGC 14-084r2 — Moving Feature Comma Separated Values (CSV). The use of this format is not mandated by the method proposal. Any OGC Moving Feature standard encoding should fit.

```
@stboundedby, urn:ogc:def:crs:JPL::0, 3D, -2.399009211149789E8
-2.600503793378296E8 -1.128053022057934E8, 1.594158811484139E8
2.306403707413973E7 884091.9549210322, 2021-11-24T00:00:00Z, 2022-09-
30T00:00:00Z, minute
@columns, mfidref, trajectory
DART,      0,    500,  -2.399009211149789E+08  -2.197847597866884E+08  -
8.486100409484243E+07
DART,    500,   1000,  -2.395270074034736E+08  -2.200427418443480E+08  -
8.500157096911532E+07
DART,   1000,   1500,  -2.391523435762360E+08  -2.203000257135218E+08  -
8.514186855044459E+07
DART,   1500,   2000,  -2.387769286812965E+08  -2.205566146997555E+08  -
8.528189727361111E+07
DART,   2000,   2500,  -2.384007597117848E+08  -2.208125047865278E+08  -
8.542165512259945E+07
```

Figure B.5 — First lines of trajectory data for the Barycenter to Dimorphos coordinate transformation

A coordinate operation may use many trajectory files like above. Each trajectory file can be used in one coordinate operation step, then many steps can be chained as shown in Figure 16.

B.3. Executable demo

An executable demo using the Apache Spatial Information System (SIS) library and the files shown in above snippets is available on GitHub. For producing an executable prototype, the following improvements were made in the Apache Spatial Information System (SIS) library and are proposed for the next SIS release. The first improvement follows EPSG practice. The remaining improvements are based on project-specific decisions in areas where OGC/ISO standards leave room for interpretation (Clause 3.4.1).

1. Allow a GML document to be divided in an arbitrary number of smaller documents referenced by `xlink:href` attributes. The URI fragments (the part after the `#` character) can be used for targeting a specific element in the referenced document. For example, if a document contains `<gml:datum xlink:href="my-CRS.xml#my-datum"/>` (note that this element is empty), then the implementation first searches a `my-CRS.xml` file in the same directory as the GML document which is using that `xlink:href` attribute, then searches inside `my-CRS.xml` for the element having a `gml:id="my-datum"` attribute. The content of the latter element is then used as the content of the former `<gml:datum/>` element.

2. For each `<gml:ParameterValue>` whose content is `<gml:valueFile>`, if the file is not absolute and does not exist in implementation-specific cache, then the path to the value file is resolved relative to the directory of the GML document. For example, if a GML document defines a coordinate operation with a parameter whose value is `<gml:valueFile>BarycenterToDART.csv</gml:valueFile>`, then the implementation will search for `BarycenterToDART.csv` in the same directory as the GML document declaring that parameter.

3. If an auxiliary file (for example, a Moving Feature file) is loaded as a consequence of decoding the GML document, and if that auxiliary file uses URN for specifying its CRS, then the implementation will search in the GML document for a CRS definition having a `<gml:identifier>` element whose value is the URN.

4. If the target CRS of a `ConcatenatedOperation` is not the same as the target CRS of the last step but the datum is the same, then the implementation will automatically add one final step for doing a change of coordinate system (e.g., from Cartesian to spherical). Inferring that step automatically is convenient because there is not a predefined operation method such as “Cartesian to spherical” for encoding explicitly a change of coordinate system.

Those improvements apply to the DART scenario as below.

- Using improvement 1, GML documents can be kept relatively simple. See, for example, Figure 16 and Figure B.2.

- Using improvement 2, the GML document defining the “Solar system barycenter to DART” operation method and the `BarycenterToDART.csv` Moving Feature file can be located anywhere, as long as they share the same directory. It allows the prototype to be executed without requiring a search path to be configured.

- Using improvement 3, the `BarycenterToDART.csv` file can declare that its CRS is `urn:ogc:def:crs:JPL::0` (barycenter). Because the GML document parsed before the Moving Feature file contains a CRS identified by `<gml:identifier>urn:ogc:def:crs:JPL::0</gml:identifier>`, that CRS will be used in the Moving Feature file. It allows the Moving Feature file to use reference systems that are not yet in any registry, even if the Moving Feature CSV format does not allow embedded CRS definitions.

- Using improvement 4, all transformation steps can be defined fully in a Cartesian coordinate system, and the last step is converted automatically to the spherical coordinate system defined by `DART.xml`.

C

ANNEX C (INFORMATIVE) GEOAPI INTERFACES

C ANNEX C (INFORMATIVE) GEOAPI INTERFACES

This annex lists the new GeoAPI interfaces introduced in Testbed 19. Those interfaces have been derived from the abstract model described in Clause 3. The snippets shown below omit the license headers, some Javadoc tags, package and import statements for brevity, but are available in their full version in a branch on GitHub. Those snippets are built on top of the following existing interfaces:

- `CoordinateSystem`, `Datum`, `Ellipsoid`, `IdentifiedObject`, `PrimeMeridian` and `SingleCRS` are GeoAPI 3.0 interfaces derived from ISO 19111.
- `Location` is an interface derived from ISO 19112 and currently in GeoAPI-pending (not yet released as a standard).

```
/**
 * Identifies the star, planet, asteroid or other celestial body for which a
reference frame is defined.
 */
public interface CelestialBody extends IdentifiedObject {
    /**
     * Returns the location of this body in reference system by identifiers.
     *
     * @see ReferenceSystemUsingIdentifiers
     */
    default Optional<Location> getLocation() {
        return Optional.empty();
    }
}
```

Figure C.1 — Celestial Body interface proposal for the GeoAPI Standard (Java version)

In the `Datum` parent interface (not shown here), an optional `getCelestialBody()` method is added. Then:

```
/**
 * Origin and orientation of an inertial CRS.
 * An inertial reference frame defines the precise location and orientation in
3- or 4-dimensional space
 * of a defined ellipsoid (or sphere), or of a Cartesian (3D case) or
Minkowski (4D case) coordinate system
 * centered in this ellipsoid (or sphere).
 */
public interface InertialReferenceFrame extends Datum {
    /**
     * Returns the ellipsoid.
     */
    default Optional<Ellipsoid> getEllipsoid() {
        return Optional.empty();
    }

    /**
     * Returns the prime direction.
     * The prime direction is defined relative to a distant feature such as a
star.
     * For example, the Ecliptic coordinate reference system defines the
primary direction by the March equinox.
     *
     * @return the prime direction.
     */
    default Optional<PrimeMeridian> getPrimeDirection() {
        return Optional.empty();
    }
}
```

Figure C.2 — Inertial Reference Frame interface proposal for the GeoAPI Standard (Java version)

Used by:

```
/**
 * A 2-, 3- or 4-dimensional coordinate reference system with axes at fixed
position relative to stars.
 *
 * <p>This type of CRS can be used with coordinate systems of type
 * {@link org.opengis.referencing.cs.EllipsoidalCS},
 * {@link org.opengis.referencing.cs.SphericalCS},
 * {@link org.opengis.referencing.cs.CartesianCS} or
 * {@link org.opengis.referencing.cs.MinkowskiCS}.</p>
 */
public interface InertialCRS extends SingleCRS {
    /**
     * Returns the datum, which shall be inertial.
     */
    @Override
    InertialReferenceFrame getDatum();
}
```

Figure C.3 — Inertial CRS interface proposal for the GeoAPI Standard (Java version)

May use:

```
/**
 * A 4-dimensional spatio-temporal coordinate system with time expressed as
lengths.
 * The three spacial dimensions give the position of points relative to
orthogonal straight axes.
 * The fourth dimension gives the position in time since an epoch.
 * All axes shall have the same length unit of measure,
 * with time expressed as the distance covered by light in the vacuum during
the elapsed time.
 *
 * <p>This type of CS can be used by coordinate reference systems of type
 * {@link org.opengis.referencing.crs.InertialCRS} or
 * {@link org.opengis.referencing.crs.EngineeringCRS}.</p>
 */
public interface MinkowskiCS extends CoordinateSystem {
}
```

Figure C.4 — Minkowski CS interface proposal for the GeoAPI Standard (Java version)

An implementation of the GeoAPI branch is available on a branch of Apache SIS.

D

ANNEX D (INFORMATIVE) SYNCHRONIZED PREVISUALIZATION

ANNEX D (INFORMATIVE) SYNCHRONIZED PREVISUALIZATION

These screenshots show data synchronized using WebVMT in a web browser. The trajectory of the StreetDrone recorded on board is shown in orange, synchronized with its trajectory observed by the static cameras in blue. A crosshair marks the frame center and this line of sight is shown on the map as a faint blue line with the camera marked as a blue dot.

Three examples are included in this report to demonstrate the accuracy of the synchronization process.

D.1. Example 17:15-17:21, 25th April 2023

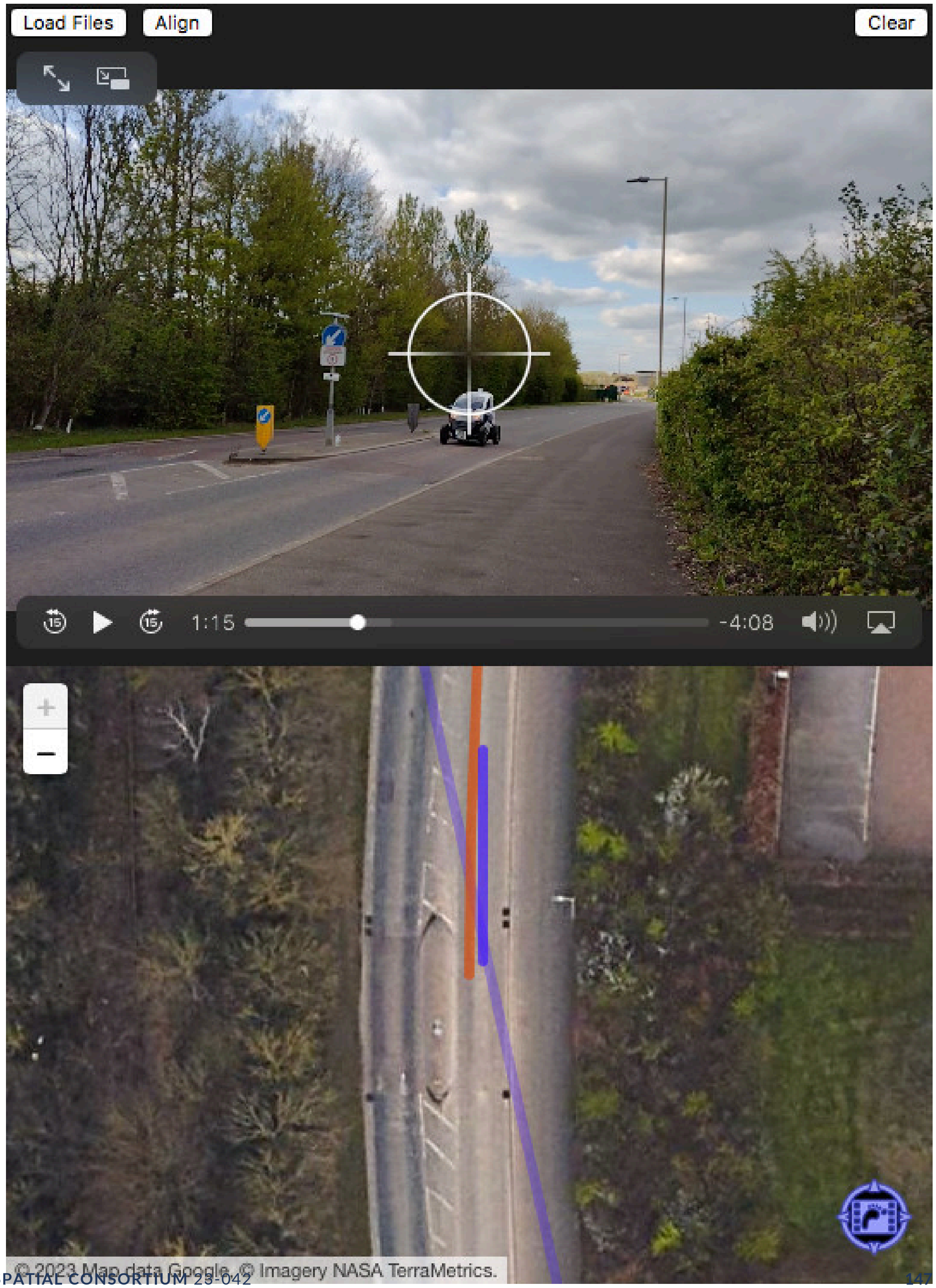


Figure D.1 — Synchronized Previsualization - Moto G30, 17:15-17:21 25 April, Pass 1

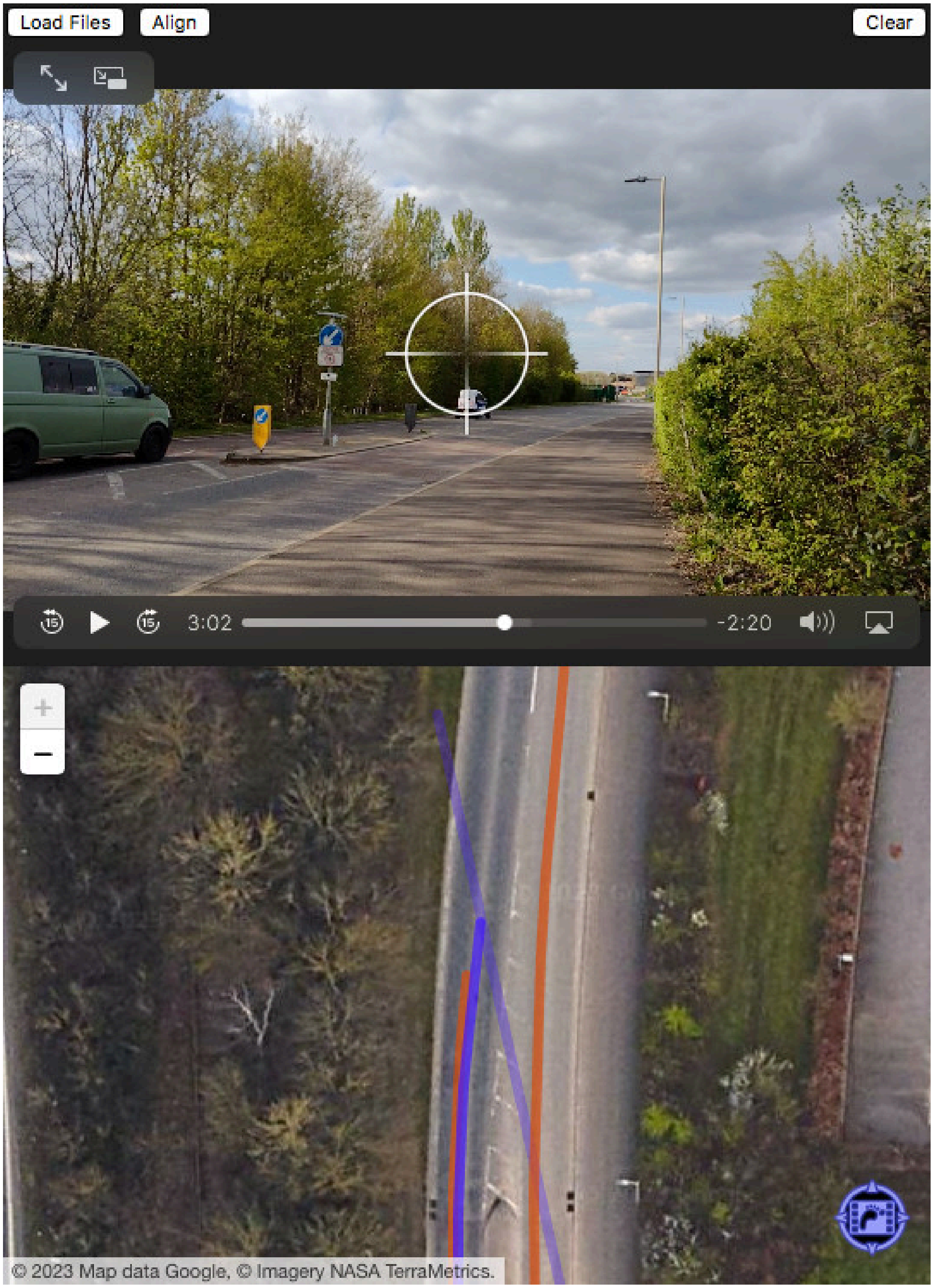


Figure D.2 — Synchronized Previsualization - Moto G30, 17:15-17:21 25 April, Pass 2

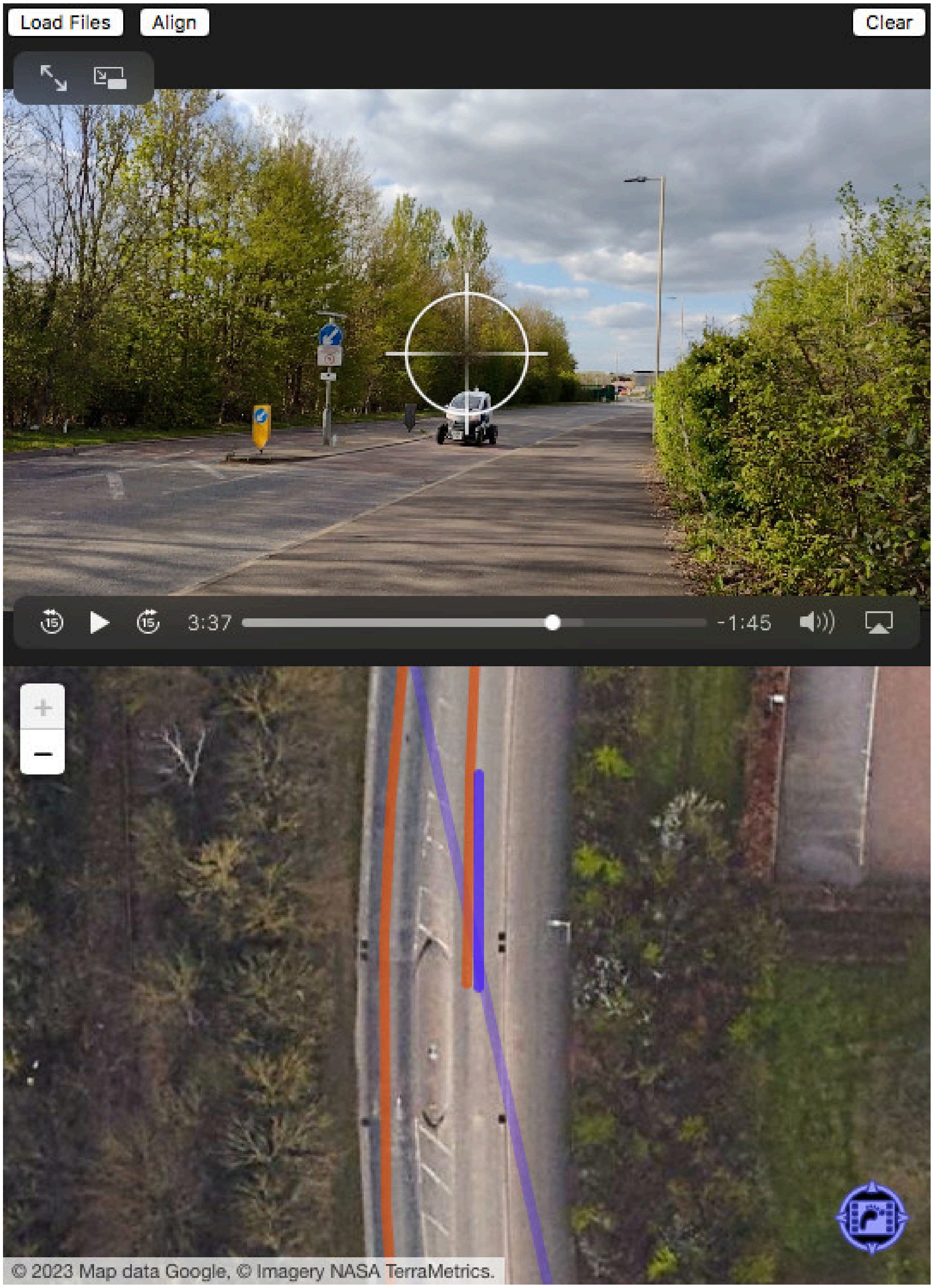


Figure D.3 — Synchronized Previsualization - Moto G30, 17:15-17:21 25 April, Pass 3

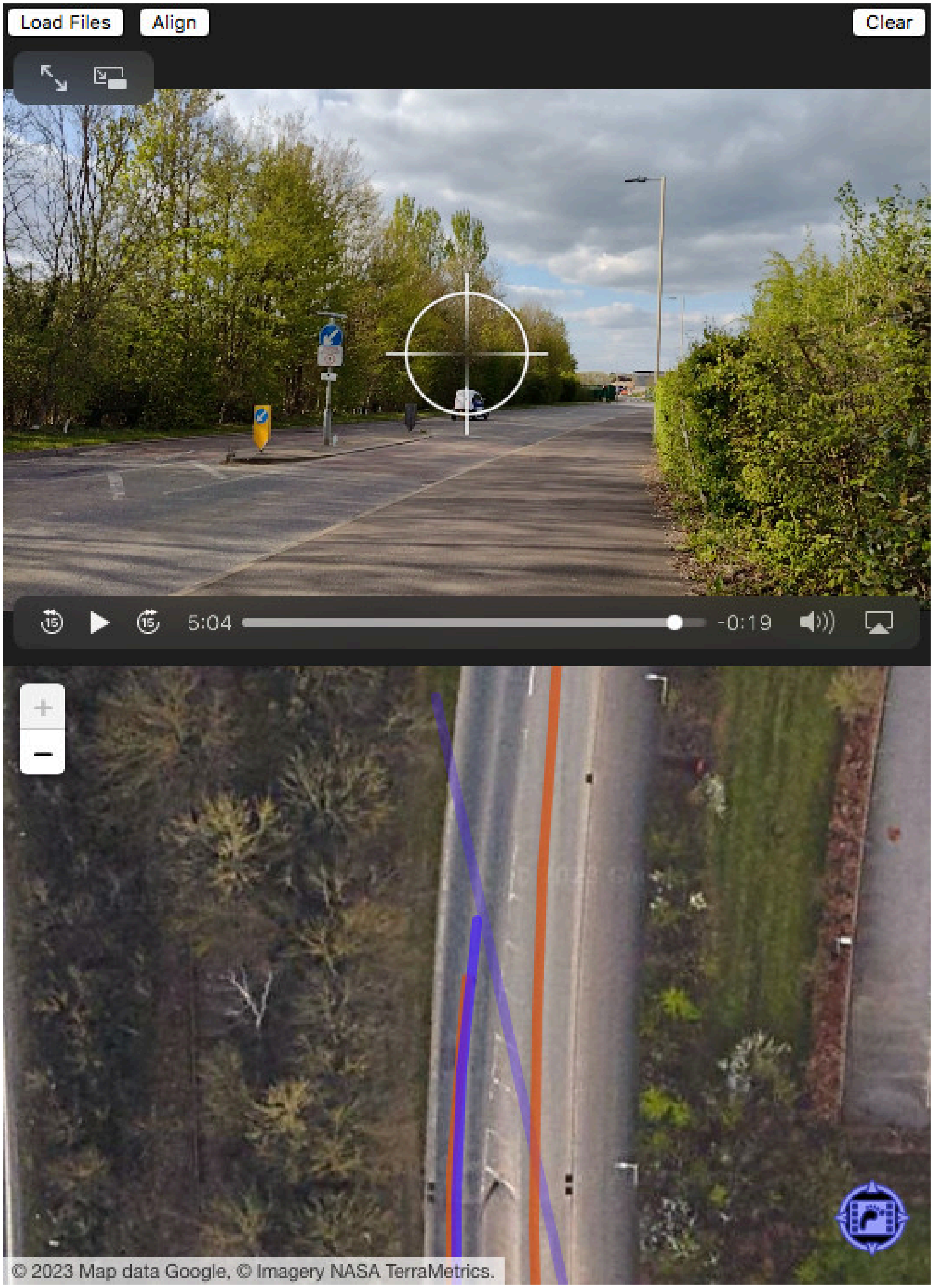


Figure D.4 — Synchronized Previsualization - Moto G30, 17:15-17:21 25 April, Pass 4

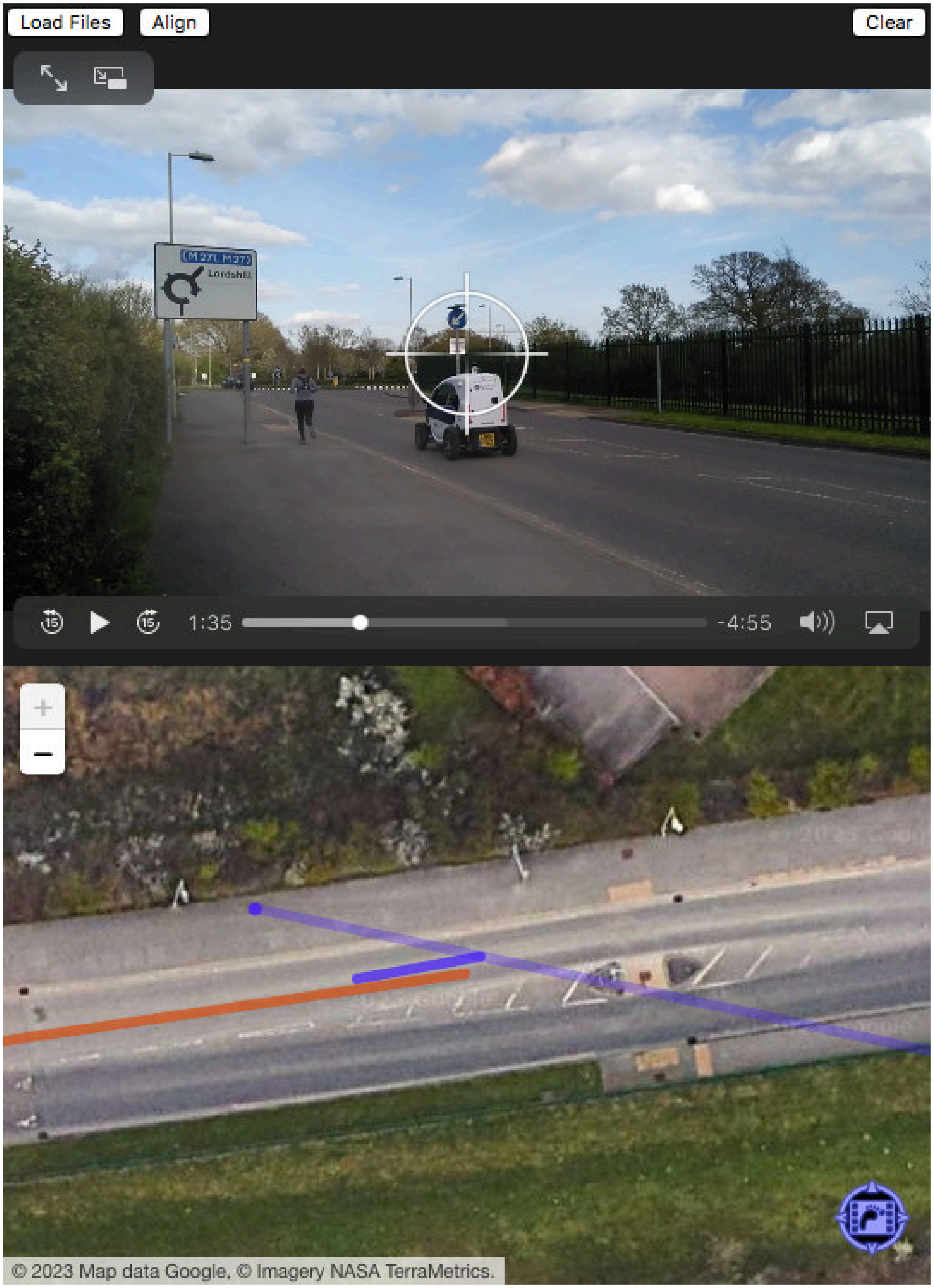


Figure D.5 — Synchronized Previsualization - Nokia 5, 17:15-17:21 25 April, Pass 1

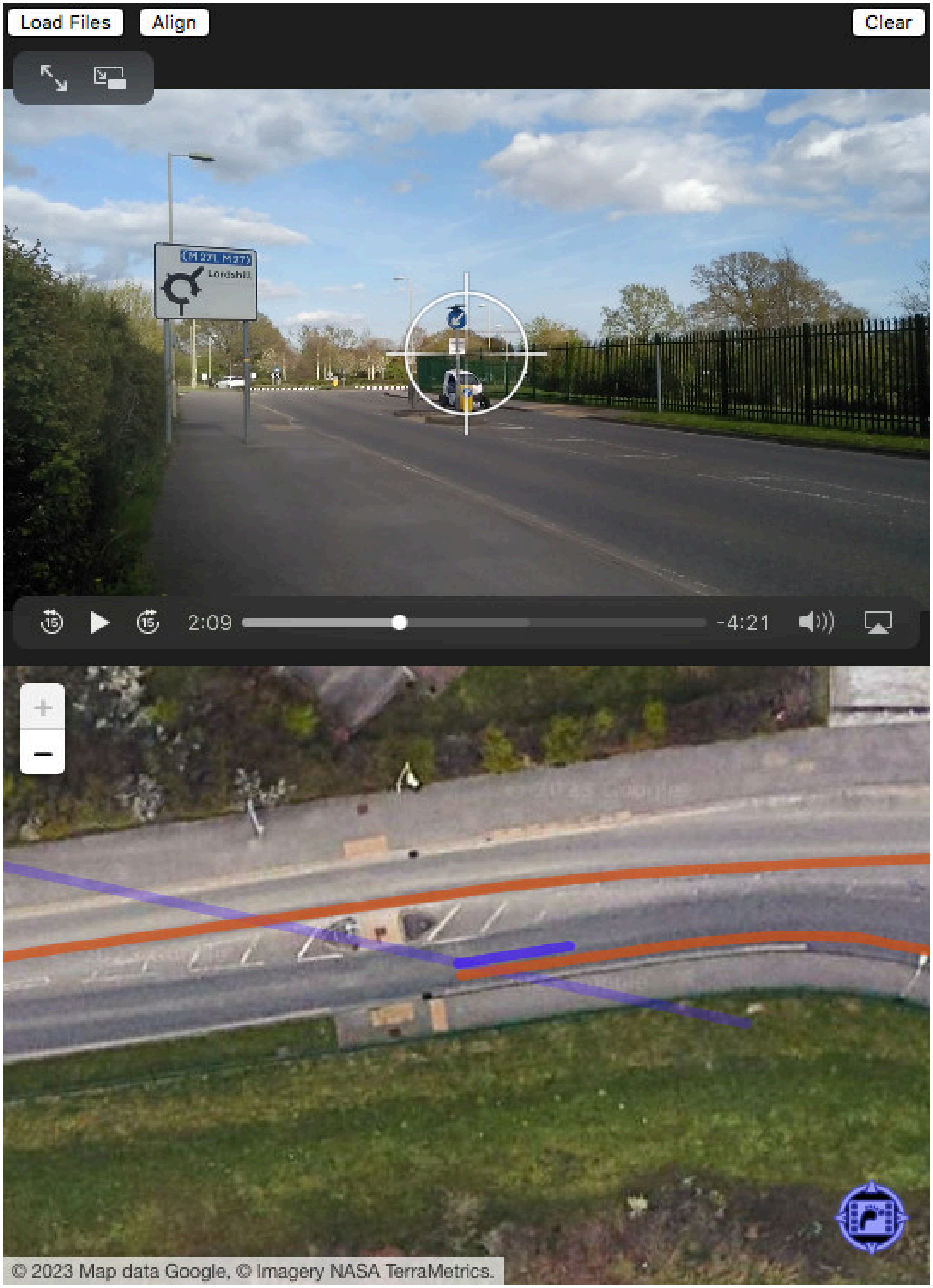


Figure D.6 — Synchronized Previsualization - Nokia 5, 17:15-17:21 25 April, Pass 2

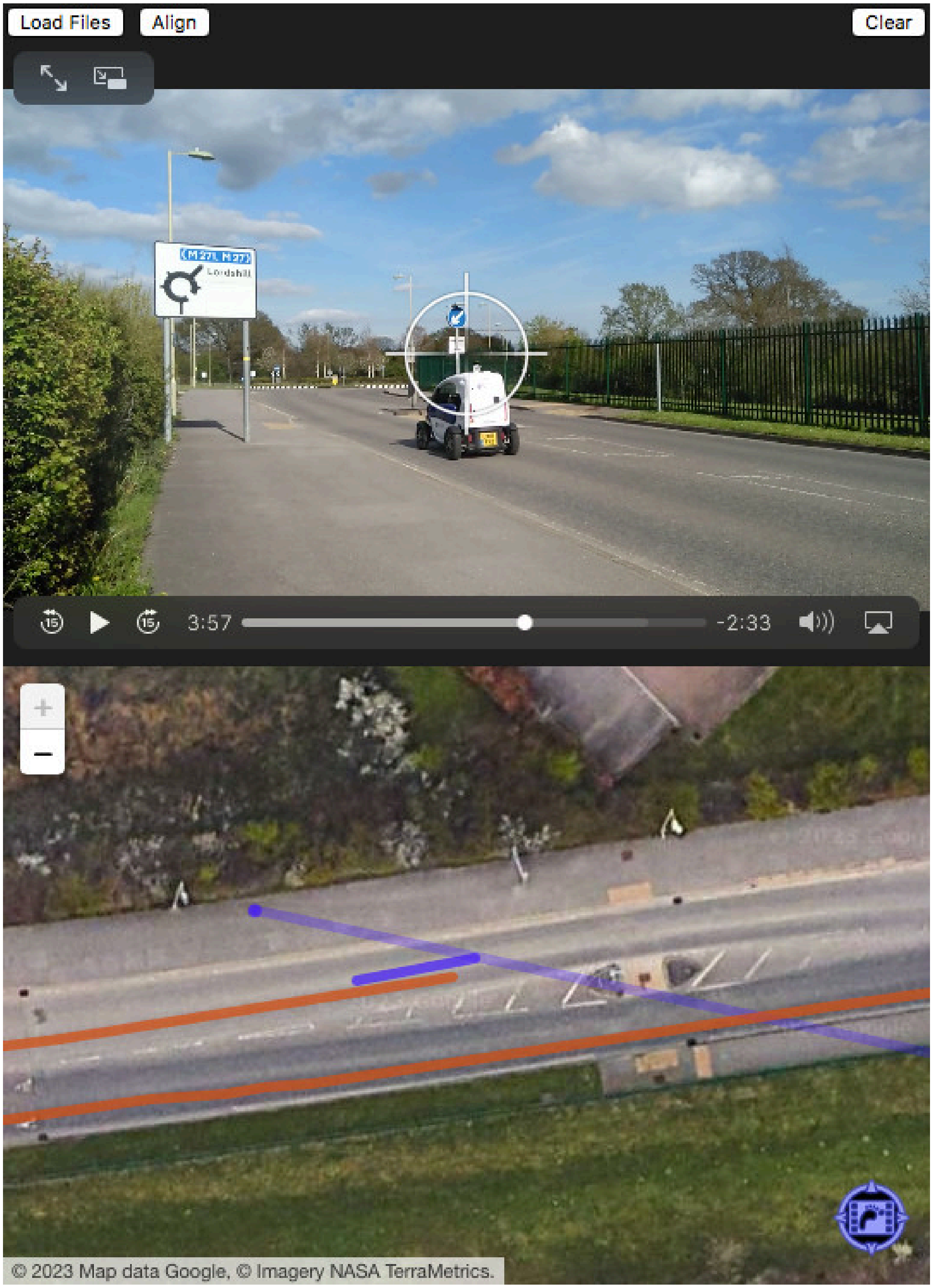


Figure D.7 — Synchronized Previsualization - Nokia 5, 17:15-17:21 25 April, Pass 3

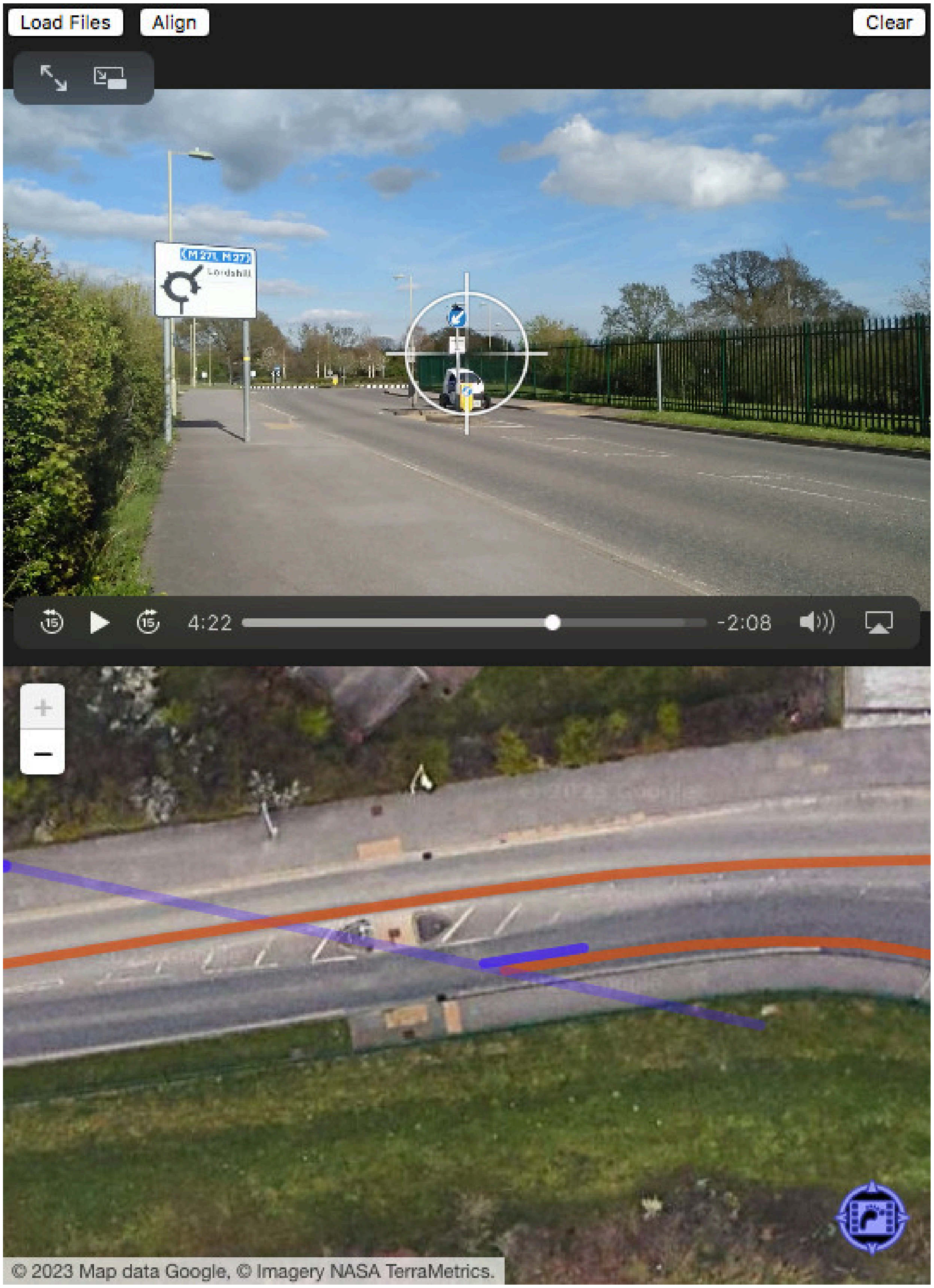


Figure D.8 — Synchronized Previsualization - Nokia 5, 17:15-17:21 25 April, Pass 4

D.2. Example 12:02-12:09, 26th April 2023

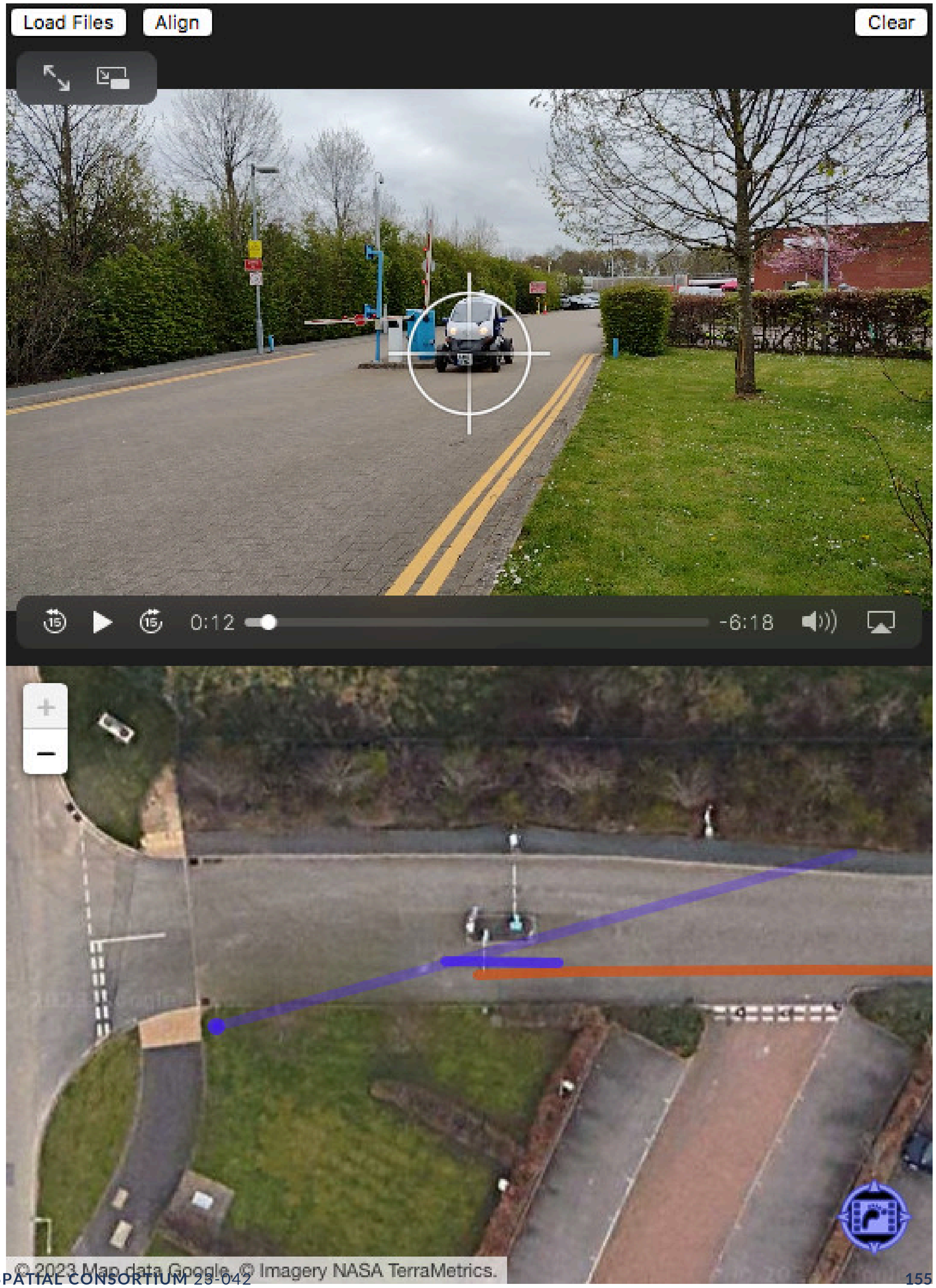


Figure D.9 — Synchronized Previsualization - Moto G30, 12:02-12:09 26 April, Pass 1

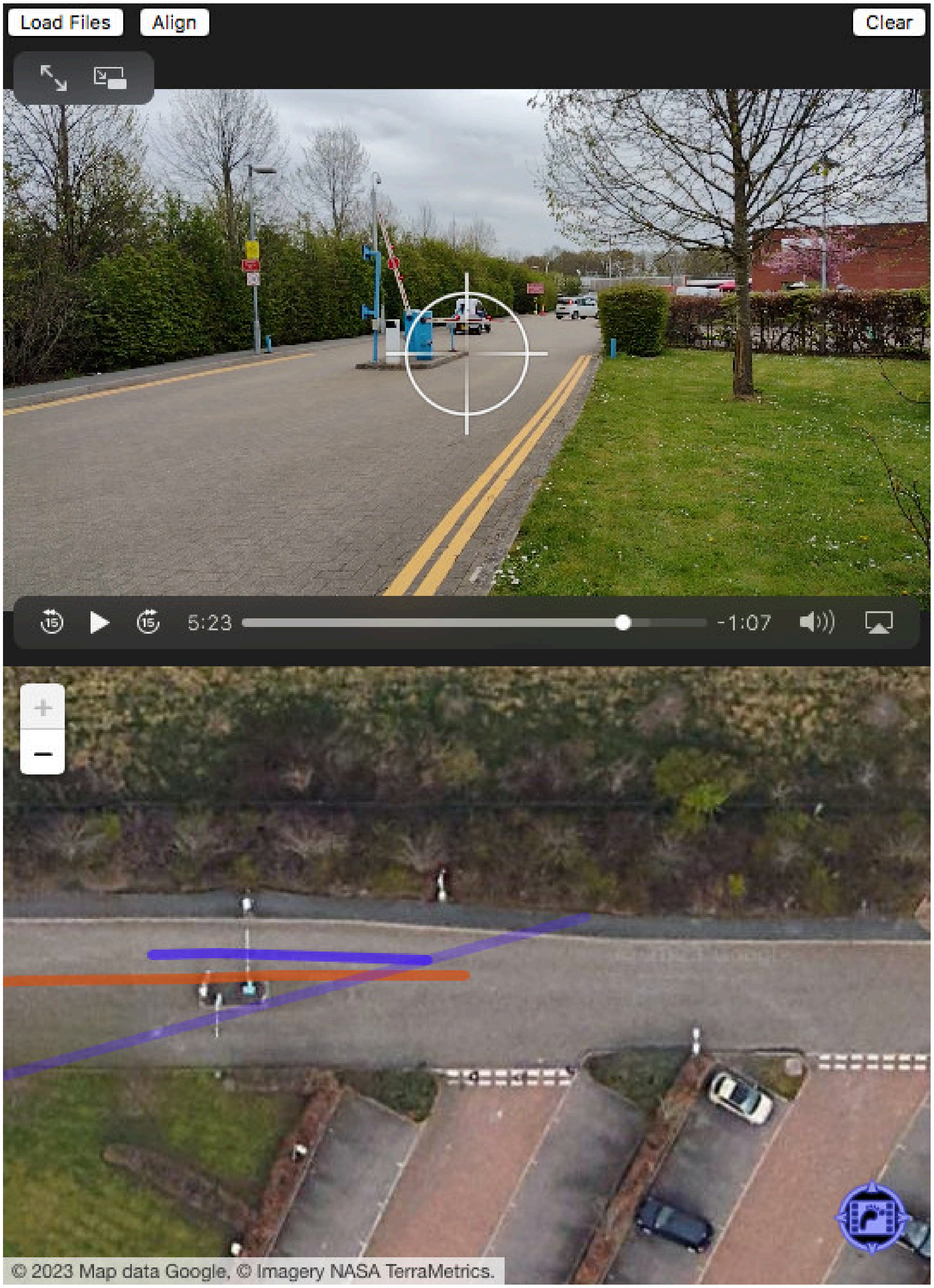


Figure D.10 — Synchronized Previsualization - Moto G30, 12:02-12:09 26 April, Pass 2

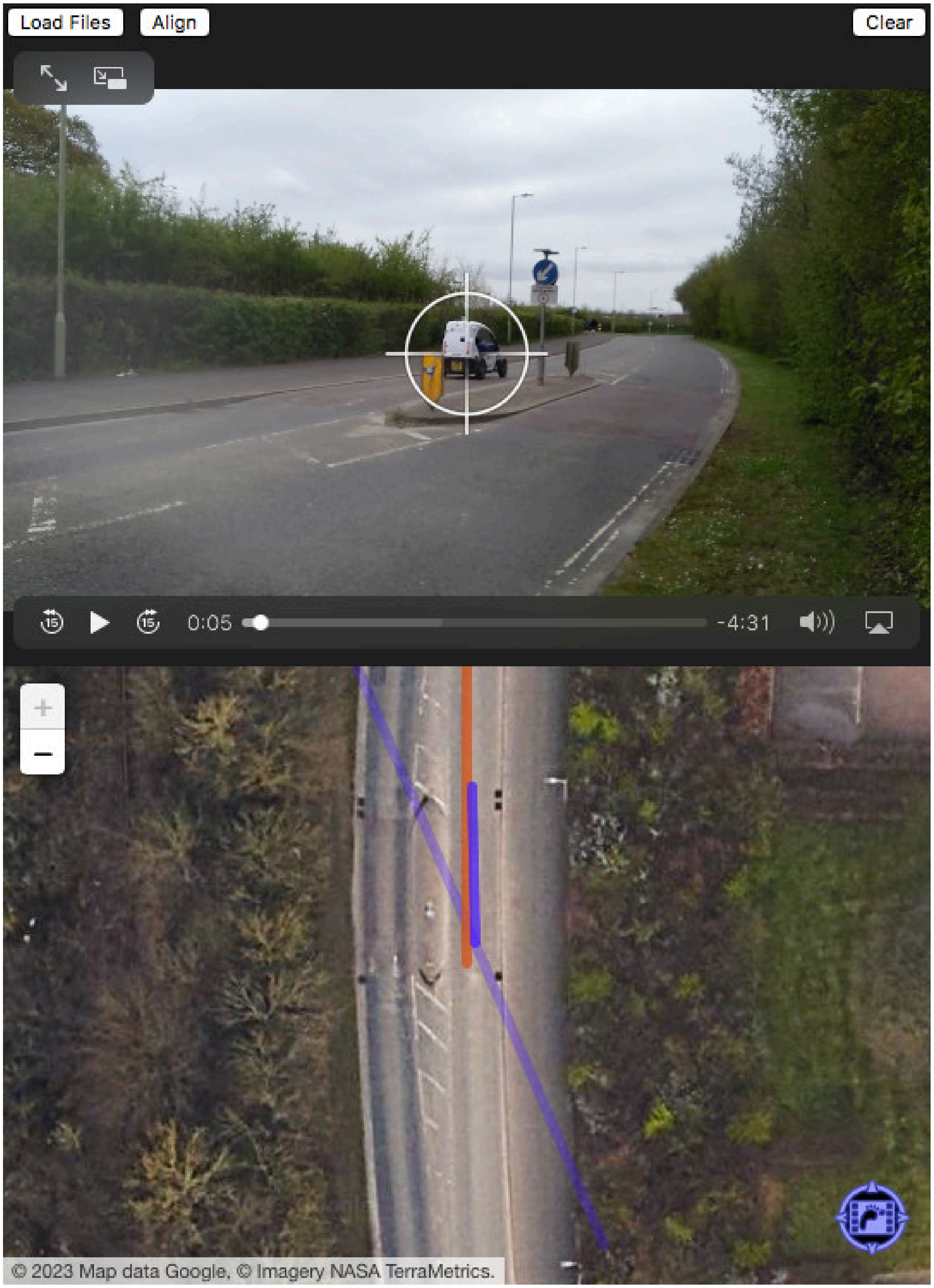


Figure D.11 — Synchronized Previsualization - Nokia 5, 12:02-12:09 26 April, Pass 1

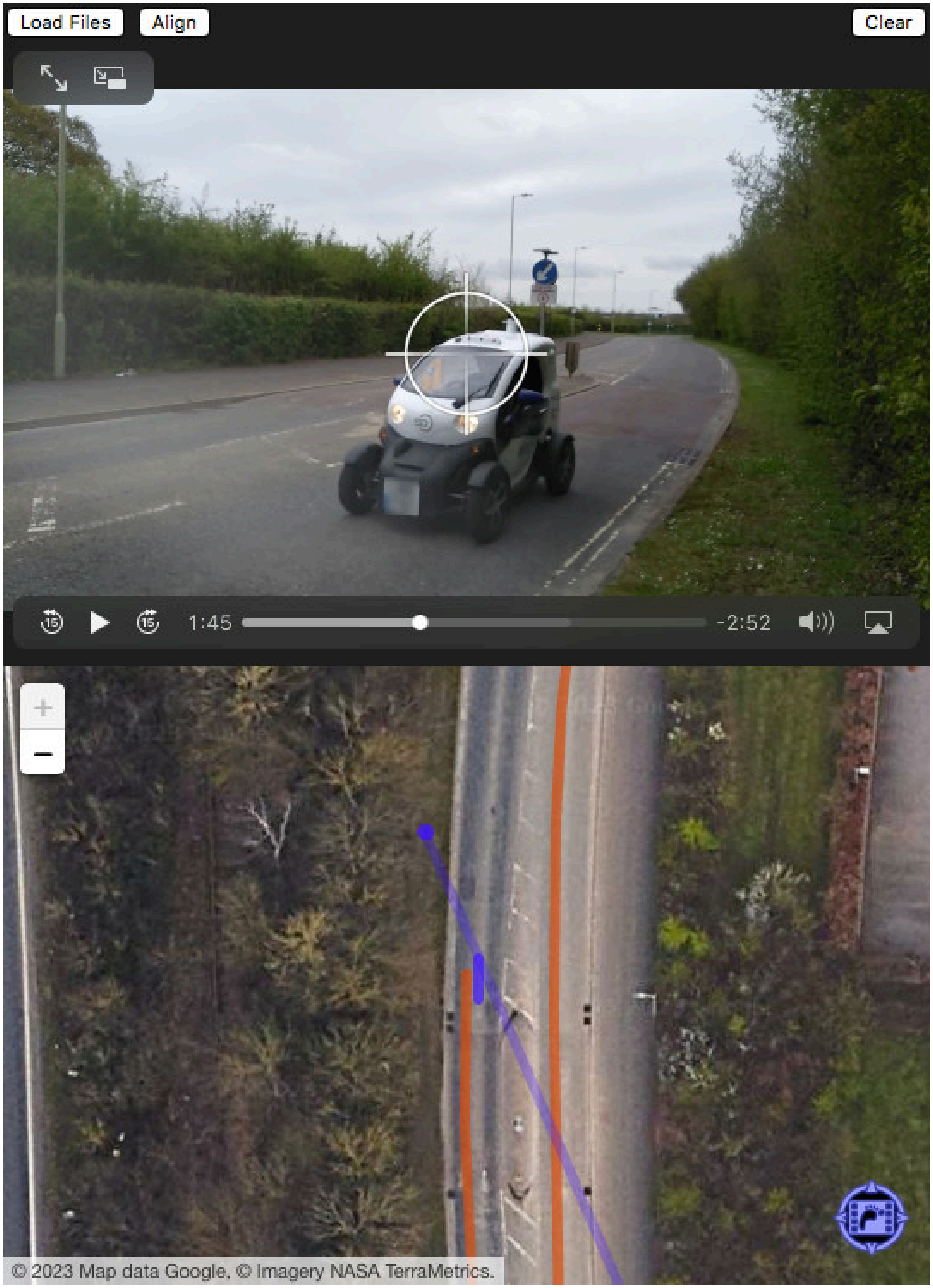


Figure D.12 — Synchronized Previsualization - Nokia 5, 12:02-12:09 26 April, Pass 2

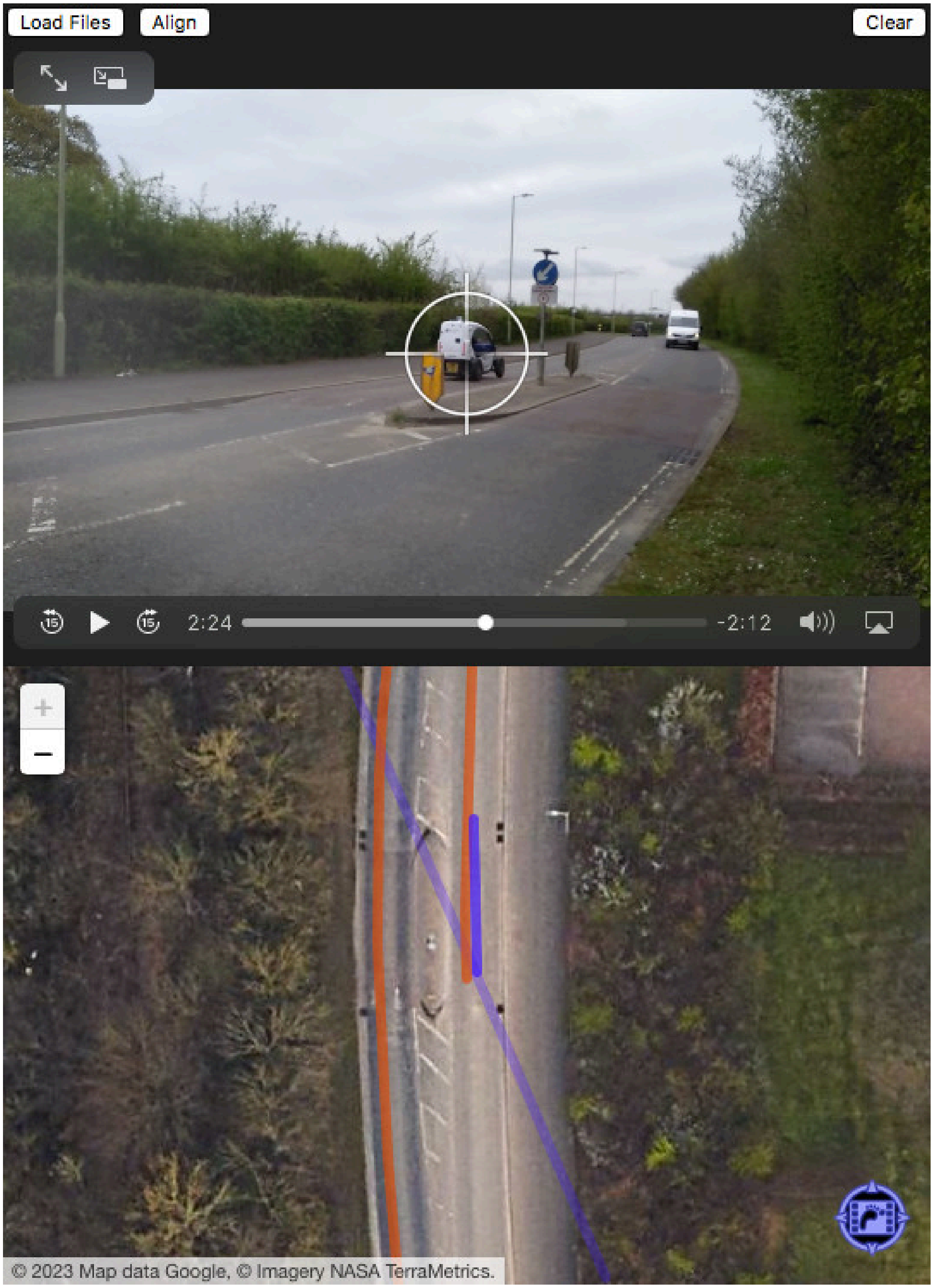


Figure D.13 — Synchronized Previsualization - Nokia 5, 12:02-12:09 26 April, Pass 3

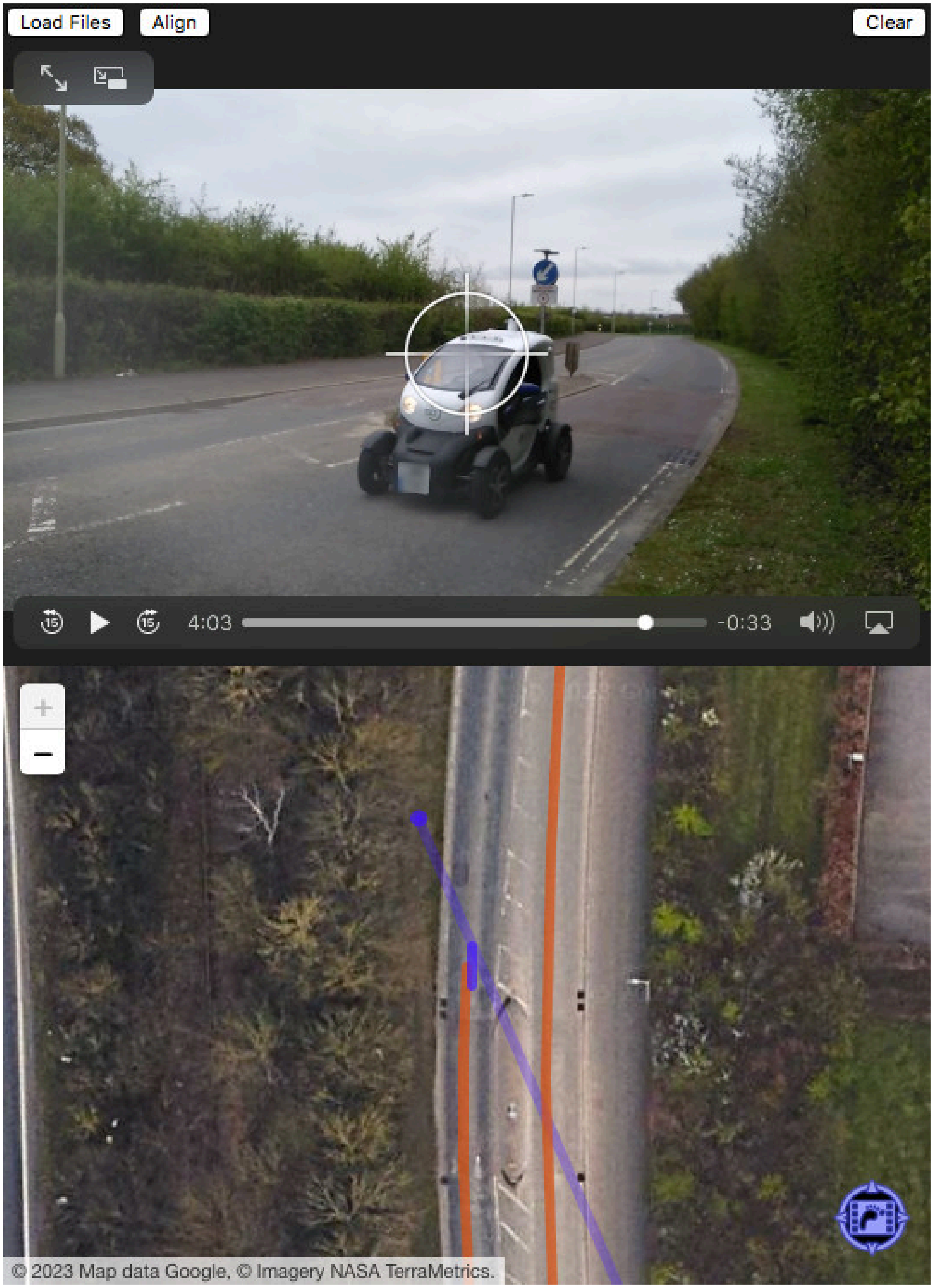


Figure D.14 — Synchronized Previsualization - Nokia 5, 12:02-12:09 26 April, Pass 4

D.3. Example 13:08-13:14, 26th April 2023

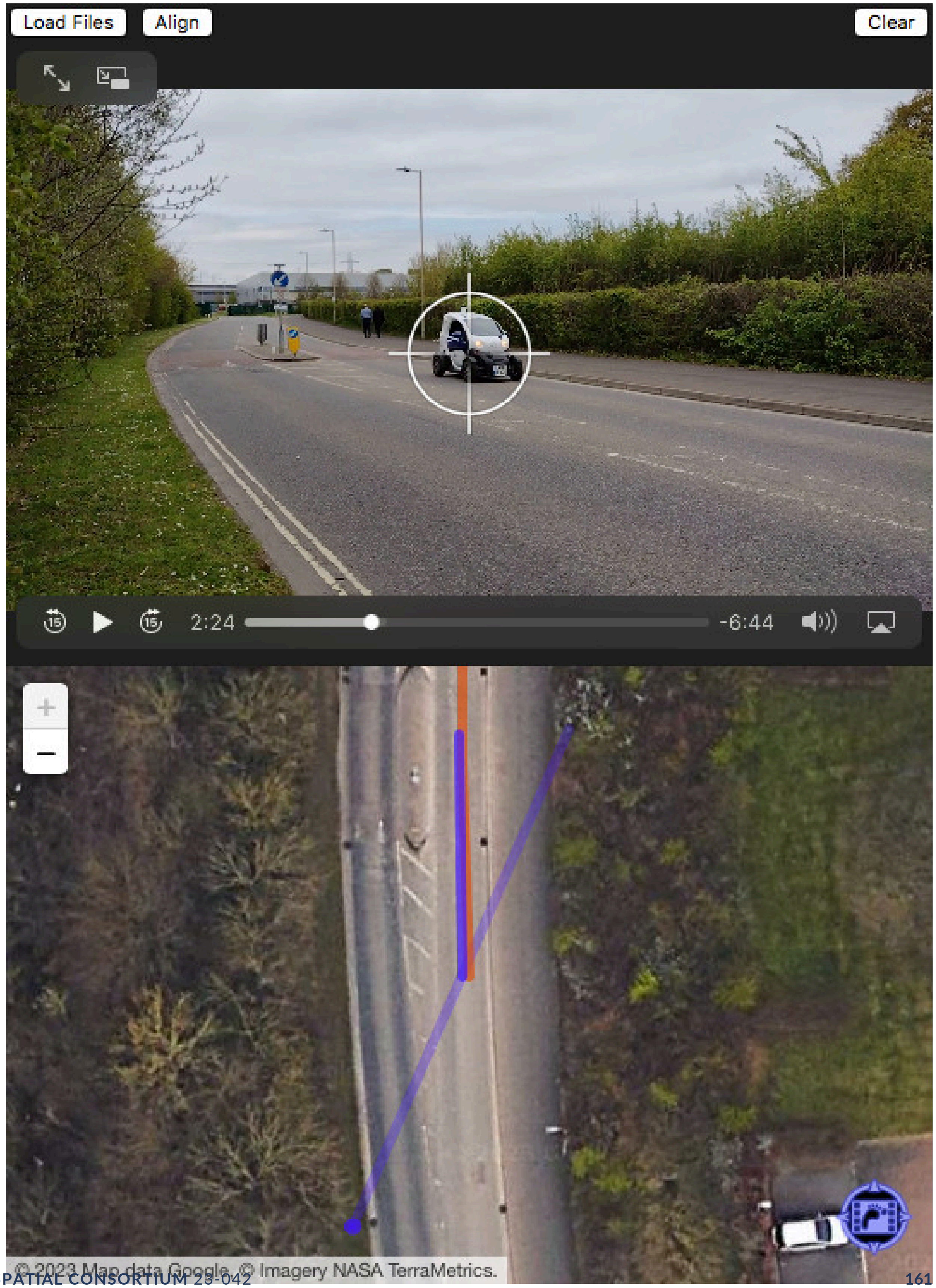


Figure D.15 — Synchronized Previsualization - Moto G30, 13:08-13:14 26 April, Pass 1

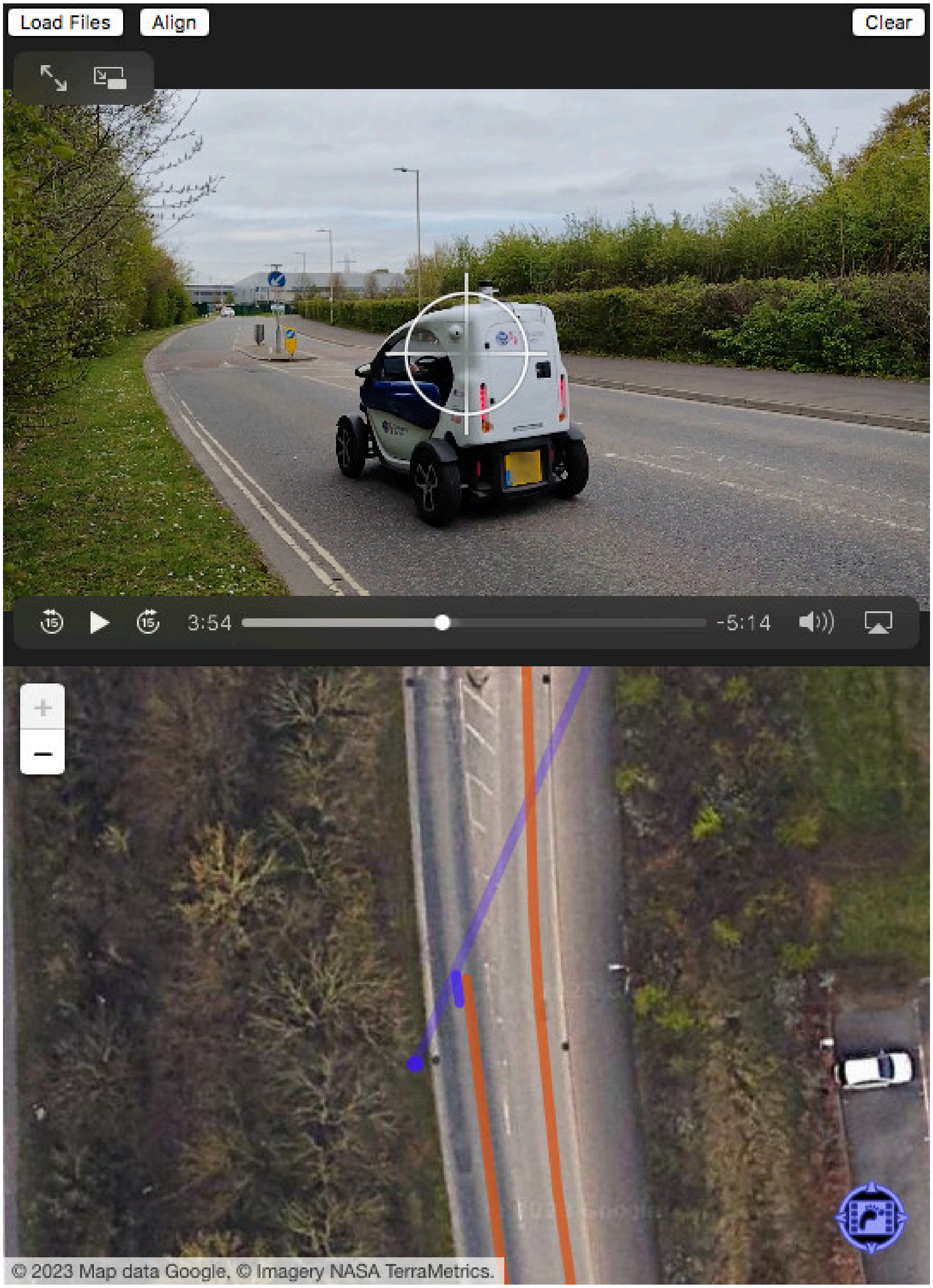


Figure D.16 — Synchronized Previsualization - Moto G30, 13:08-13:14 26 April, Pass 2

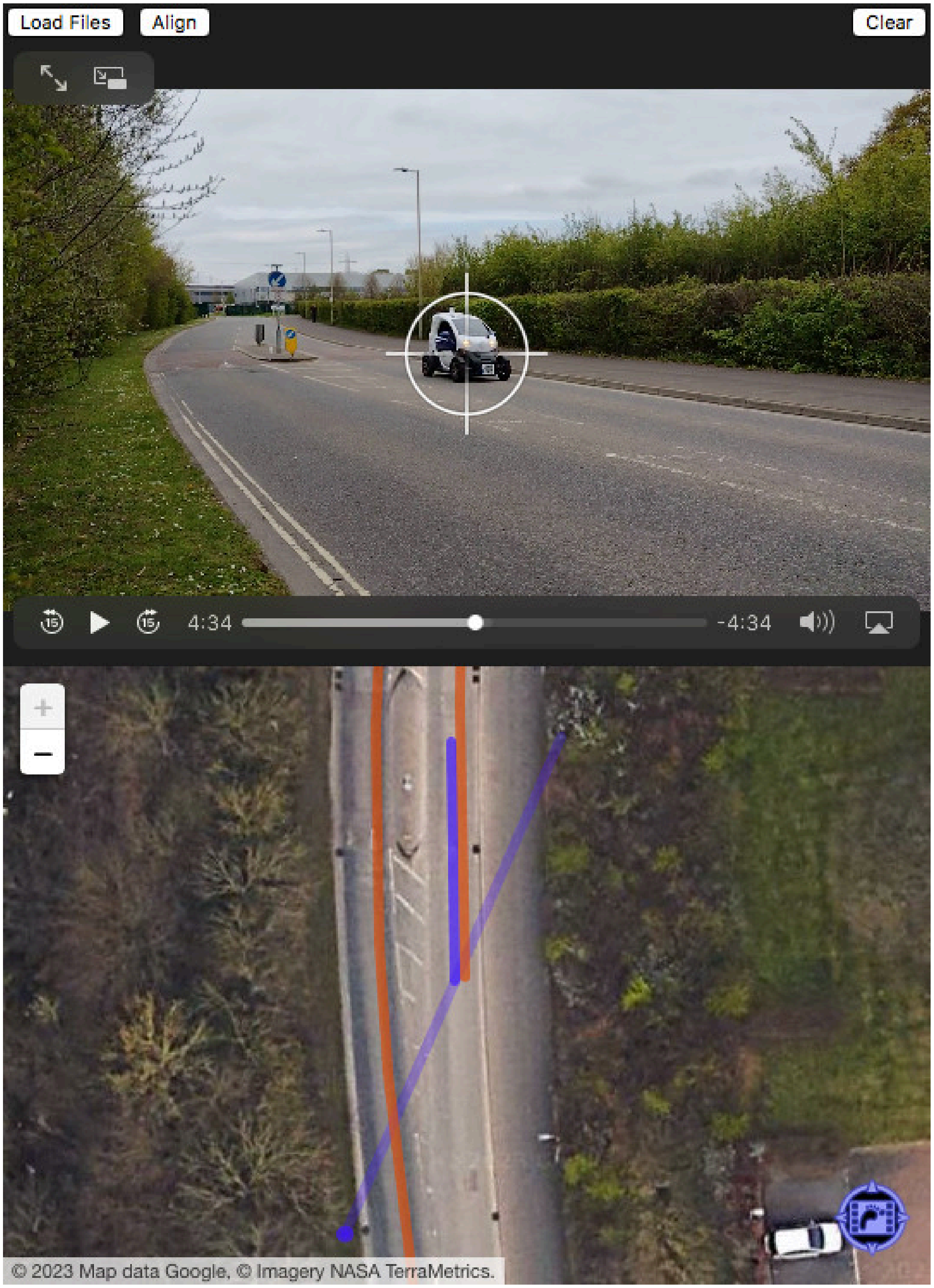


Figure D.17 — Synchronized Previsualization - Moto G30, 13:08-13:14 26 April, Pass 3

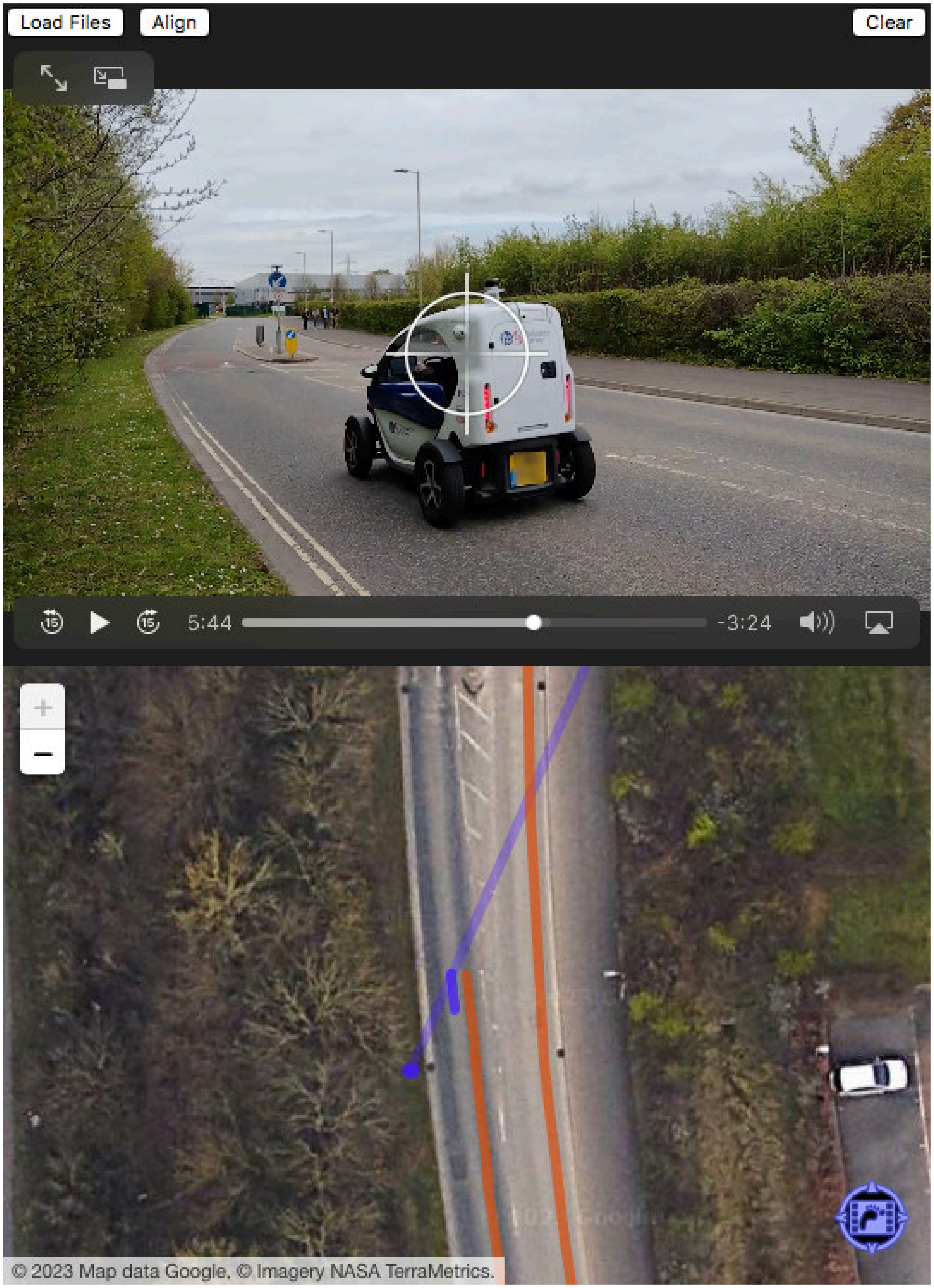


Figure D.18 — Synchronized Previsualization - Moto G30, 13:08-13:14 26 April, Pass 4

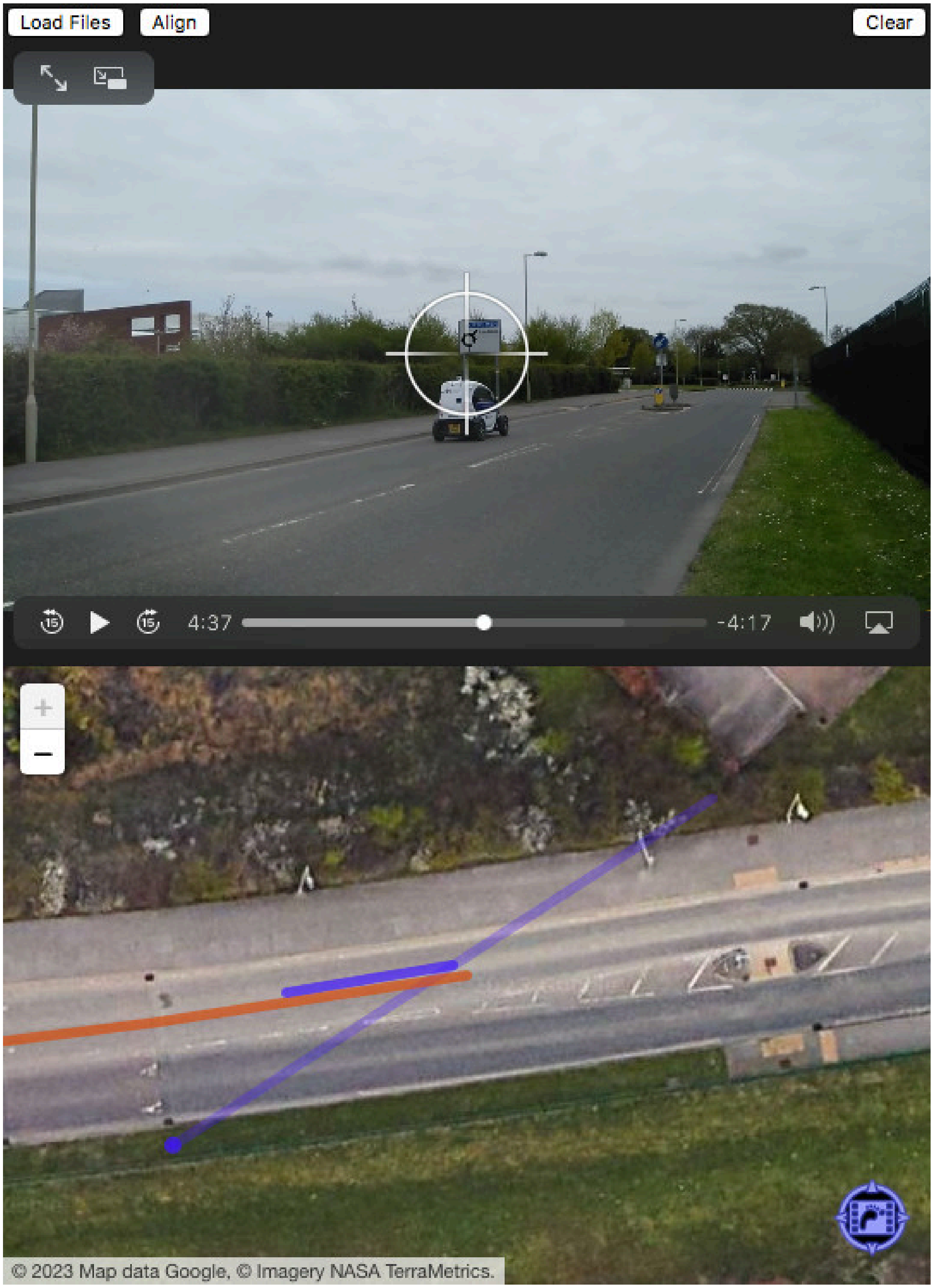


Figure D.19 — Synchronized Previsualization - Nokia 5, 13:08-13:14 26 April, Pass 1

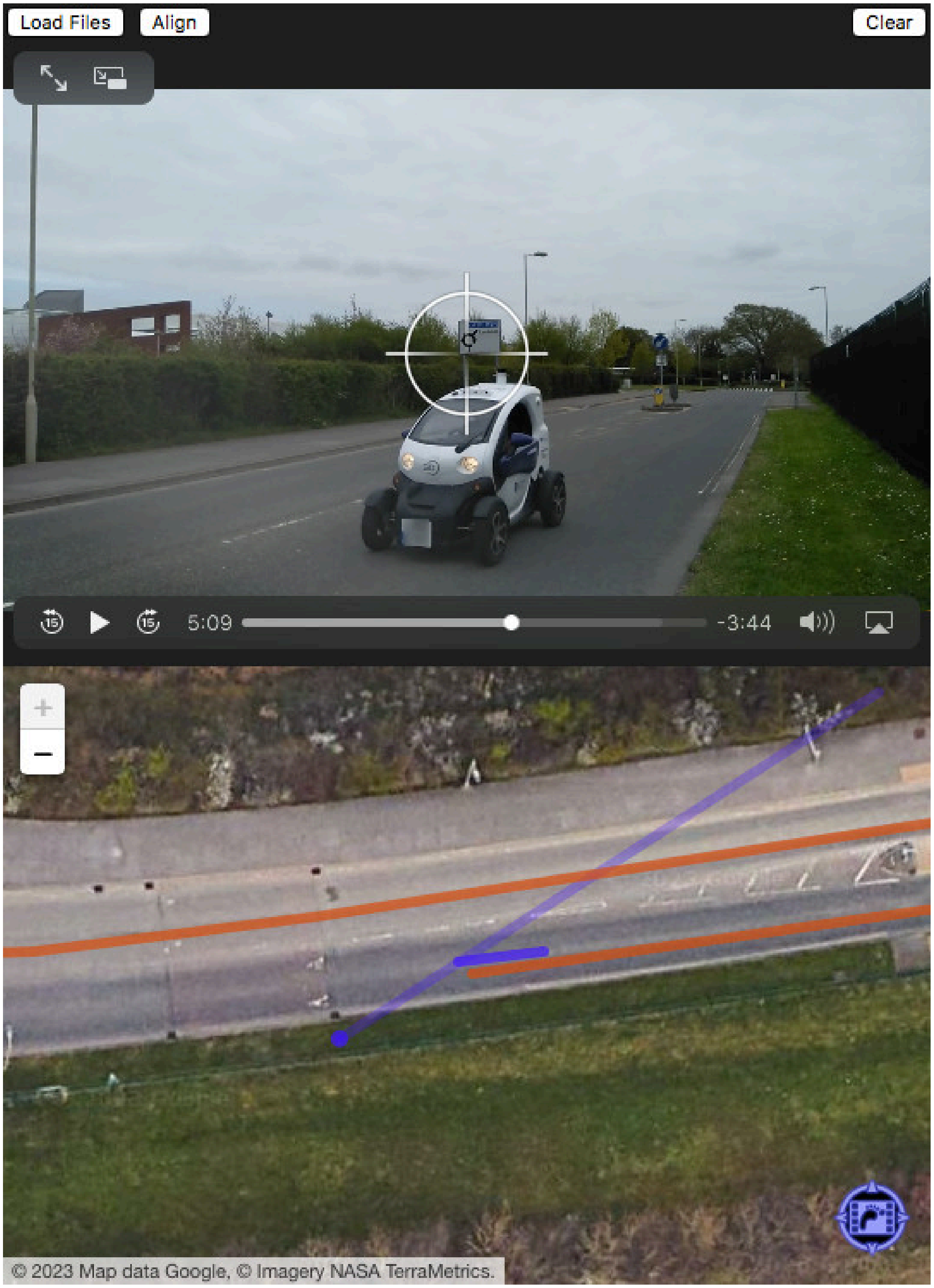


Figure D.20 — Synchronized Previsualization - Nokia 5, 13:08-13:14 26 April, Pass 2

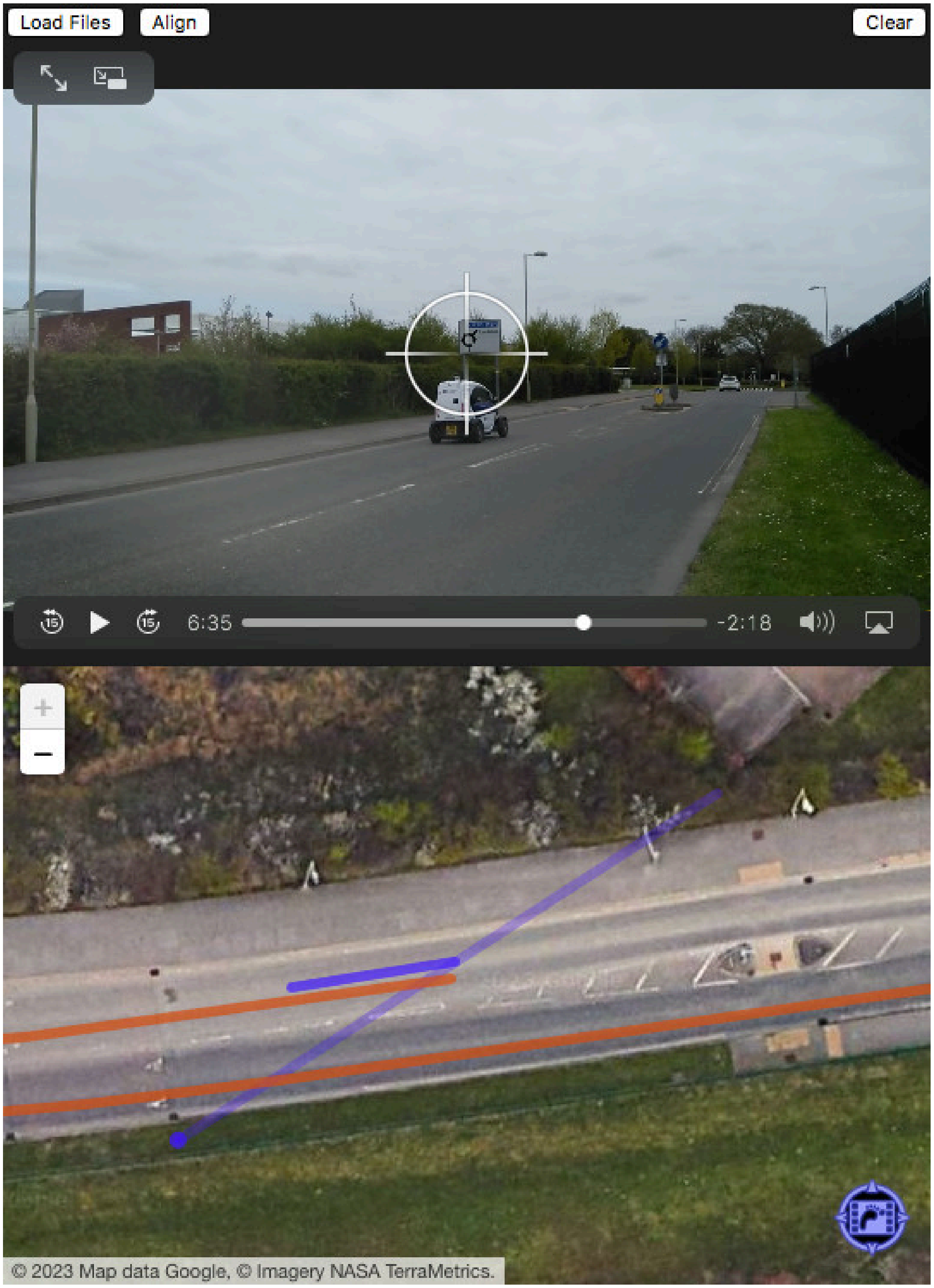


Figure D.21 — Synchronized Previsualization - Nokia 5, 13:08-13:14 26 April, Pass 3

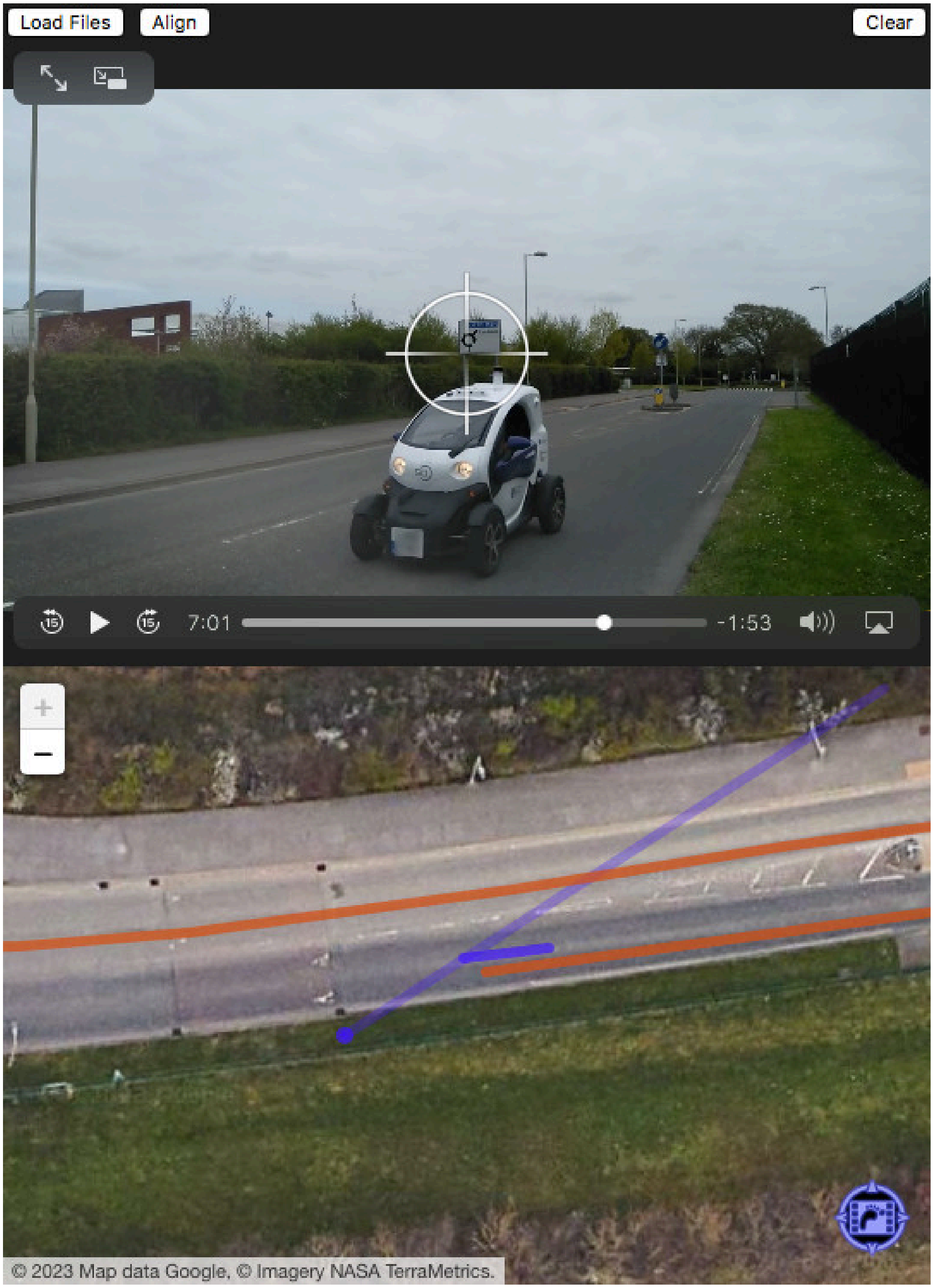


Figure D.22 — Synchronized Previsualization - Nokia 5, 13:08-13:14 26 April, Pass 4

E

ANNEX E (INFORMATIVE) FUNDAMENTAL CONCEPTS

E ANNEX E (INFORMATIVE) FUNDAMENTAL CONCEPTS

As the Geospatial field expands into new disciplines, it is important that existing accepted concepts are not re-invented or re-defined. This section summarizes some of the existing concepts from Space related disciplines that are relevant to this Engineering Report.

E.1. Euclidean Geometry

Euclidean geometry is a mathematical system attributed to ancient Greek mathematician Euclid, which is in his textbook on geometry, Elements. Euclid’s approach consists in assuming a small set of intuitively appealing axioms (postulates) and deducing many other propositions (theorems) from these. Although many of Euclid’s results had been stated earlier, Euclid was the first to organize these propositions into a logical system in which each result is proved from axioms and previously proved theorems.

The Elements begins with plane geometry, still taught in secondary school (high school) as the first axiomatic system and the first examples of mathematical proofs. It goes on to the solid geometry of three dimensions. Much of the Elements states results of what are now called algebra and number theory, explained in geometrical language. [Wikipedia]

E.2. General Relativity

Both Newtonian Physics and Special Relativity assume Euclidean geometries. However, reality is not so well behaved. General Relativity postulates that gravity is not a force but a consequence of the shape of spacetime. Reality is non-Euclidean. Spacetime is distorted by mass. Furthermore, mass, velocity, and energy are interrelated. Frames of Reference become very complex. Too complex to address in this Engineering Report.

However, the impact of General Relativity cannot be ignored. The effects are significant enough that GPS must account for both special relativity and general relativity to deliver positions at 1-meter levels and time at 100-nanosecond levels to its users.

E.3. Inertial Frame of Reference

Most of the Frames of Reference described in this Engineering Report are Inertial Frames of Reference. Inertial Frames of Reference are characterized by the following constraints:

- Newton's laws apply within the Frame of Reference;
- Newton's first law applies to the Frame or Reference itself; and
- the Frame of Reference itself is not subject to any accelerating force.

Both Newtonian Physics and Special Relativity work with Inertial Frames of Reference. Non-inertial Frames of Reference come into play with General Relativity.

E.4. Local, Global, and World Reference Systems

The OGC Moving Features Standard defies three types of Reference System: Local, Global, and World. The Local reference system is the reference system within which a Feature and its Properties reside. All measurements of those properties are "proper" measurements. The World reference system is the spatiotemporal Frame of Reference within which all features and operations exist. Typically, the universe, solar system, or planet. The Global Reference System is an intermediary between the World and Local reference systems. It serves to capture the location and orientation of the trajectory (movement vector) of the Local reference system relative to the World reference system. Note that the Local reference system is not always aligned with the direction of motion.

E.5. Minkowski Spacetime

Minkowski spacetime integrates space and time into a single, four-dimensional, Cartesian coordinate system. It was developed to support Special Relativity where space and time are tightly coupled concepts. However, Minkowski spacetime should also prove useful to represent the geometry of any moving object. An object in motion does not have a fixed location. Rather, its location is a function of time. Likewise, time is a function of the objects location. Therefore, an objects position can be represented as a point (event) in spacetime. Likewise, its trajectory is a curve (line string) in spacetime. There is no longer a need to correlate separate spatial and temporal geometries.

E.6. Newtonian Physics

Newtonian Physics is the physics of our everyday world. It can be summarized by Newtons Laws of Motion:

1. a body remains at rest, or in motion at a constant speed in a straight line, unless acted upon by a force;
2. when a body is acted upon by a net force, the body's acceleration multiplied by its mass is equal to the net force; and
3. if two bodies exert forces on each other, these forces have the same magnitude but opposite directions.Thorton and Marion (2024)

OGC Standards typically assume that Frames of Reference are static. The relative displacement and orientation between two Frames of Reference does not change over time. In those cases where there is relative motion, the relative velocity is low. Newtonian Physics is sufficient for use in this context.

E.7. Schwarzschild Metric

In Einstein's theory of general relativity, the Schwarzschild metric (also known as the Schwarzschild solution) is an exact solution to the Einstein field equations that describes the gravitational field outside a spherical mass on the assumption that the electric charge of the mass, angular momentum of the mass, and universal cosmological constant are all zero. The solution is a useful approximation for describing slowly rotating astronomical objects such as many stars and planets, including Earth and the Sun. [Wikipedia].

E.8. Special Relativity

Once off the surface of the Earth, the distances and velocities between objects become very large. Measurable differences between local and remote observations become common. These differences are due to the speed of light which is the maximum speed in which information can travel from the observed Frame of Reference to the observing Frame of Reference. A remote observer can know what the value of a property was but can never know what the value is.

Two key concepts arise out of this:

- Proper measurements: Those observations performed in the same Frame of Reference as the property being measured.

- Observed measurements: Those observations performed in a Frame of Reference which is in motion relative to the Frame of Reference of the property being measured.

Lorentz transformations are used to convert Observed Measurements into Proper Measurements.

E.9. World Line

The world line (or worldline) of an object is the path that an object traces in 4-dimensional spacetime. This is an important concept in modern physics and particularly, theoretical physics. [Wikipedia]

Consider a real-world object. At any specific time, the object has a location in space. The combination of a location in space and the time at which the object is at that location is an Event in spacetime, aka a unique location in space and time. The World Line of an object is the sequence of Events for that object from creation to destruction, everywhere and at every time that the object existed. World Lines are typically used with Minkowski spacetime to provide the mathematical tools for Special Relativity.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Roger Lott: OGC 18-005r5, *Topic 2 — Referencing by coordinates Corrigendum*. Open Geospatial Consortium (2021). http://www.opengis.net/doc/AS/topic-2/5.0.1.

[2] Martin Daly: OGC 01-009, *Coordinate Transformation Services — OLE/COM*. Open Geospatial Consortium (2001).

[3] Roger Lott: OGC 18-010r11, *Geographic information — Well-known text representation of coordinate reference systems*. Open Geospatial Consortium (2023). http://www.opengis.net/doc/is/crs-wkt/2.1.11.

[4] Martin Desruisseaux, Logan Stark: OGC 23-011r1, *Testbed-18: 3D+ Data Space Object Engineering Report*. Open Geospatial Consortium (2023). http://www.opengis.net/doc/PER/T18-D024.

[5] Brittany Eaton: OGC 22-016r3, *Testbed-18: Moving Features Engineering Report*. Open Geospatial Consortium (2023). http://www.opengis.net/doc/PER/T18-D020.

[6] Brittany Eaton: OGC 22-016r3, *Testbed-18: Moving Features Engineering Report*. Open Geospatial Consortium (2023). http://www.opengis.net/doc/PER/T18-D020.

[7] Martin Desruisseaux: OGC 22-038r2, *Testbed-18: Reference Frame Transformation Engineering Report*. Open Geospatial Consortium (2023). http://www.opengis.net/doc/PER/T18-D025.

[8] Adrian Custer: OGC 09-083r4, *GeoAPI 3.0 Implementation Standard with corrigendum*. Open Geospatial Consortium (2018).

[9] Carl Stephen Smyth: OGC 21-056r11, *OGC GeoPose 1.0 Data Exchange Standard*. Open Geospatial Consortium (2023). http://www.opengis.net/doc/IS/geopose/1.0.0.

[10] Kyoung-Sook KIM, Nobuhiro ISHIMARU: OGC 19-045r3, *OGC Moving Features Encoding Extension — JSON*. Open Geospatial Consortium (2020). http://www.opengis.net/doc/IS/mf-json/1.0.0.

[11] Emeric Beaufays, C.J. Stanbridge, Rob Smith: OGC 20-036, *OGC Testbed-16: Full Motion Video to Moving Features Engineering Report*. Open Geospatial Consortium (2021). http://www.opengis.net/doc/PER/t16-D021.

[12] Guy Schumann: OGC 21-036, *OGC Testbed-17: Moving Features ER*. Open Geospatial Consortium (2022). http://www.opengis.net/doc/PER/t17-D020.

[13] Akinori Asahara, Ryosuke Shibasaki, Nobuhiro Ishimaru, David Burggraf: OGC 14-084r2, *OGC® Moving Features Encoding Extension: Simple Comma Separated Values (CSV)*. Open Geospatial Consortium (2015). http://www.opengis.net/doc/IS/movingfeatures/csv-extension/1.0.0.

[14] ISO: ISO 18026, *Information technology — Spatial Reference Model (SRM). International Organization for Standardization, Geneva (2009)*. ISO

[15] ISO: ISO 19103, *Geographic information — Conceptual schema language*. International Organization for Standardization, Geneva https://www.iso.org/standard/56734.html.

[16] ISO: ISO 19104, *Geographic information — Terminology*. International Organization for Standardization, Geneva https://www.iso.org/standard/63541.html.

[17] ISO: ISO 19109, *Geographic information — Rules for application schema*. International Organization for Standardization, Geneva https://www.iso.org/standard/59193.html.

[18] ISO: ISO 19111, *Geographic information — Referencing by coordinates*. International Organization for Standardization, Geneva https://www.iso.org/standard/74039.html.

[19] ISO: ISO 19112, *Geographic information — Spatial referencing by geographic identifiers*. International Organization for Standardization, Geneva https://www.iso.org/standard/70742.html.

[20] ISO: ISO 19115, *Geographic information — Metadata*. International Organization for Standardization, Geneva https://www.iso.org/standard/26020.html.

[21] ISO: ISO 19123, *Geographic information — Schema for coverage geometry and functions*. International Organization for Standardization, Geneva https://www.iso.org/standard/40121.html.

[22] ISO: ISO 19133, *Geographic information — Location-based services — Tracking and navigation*. International Organization for Standardization, Geneva https://www.iso.org/standard/32551.html.

[23] ISO: ISO 19136, *Geographic information — Geography Markup Language (GML)*. International Organization for Standardization, Geneva https://www.iso.org/standard/32554.html.

[24] ISO: ISO 19141, *Geographic information — Schema for moving features*. International Organization for Standardization, Geneva https://www.iso.org/standard/41445.html.

[25] ISO: ISO 19157, *Geographic information — Data quality*. International Organization for Standardization, Geneva https://www.iso.org/standard/32575.html.

[26] ISO: ISO 19162, *Geographic information — Well-known text representation of coordinate reference systems*. International Organization for Standardization, Geneva https://www.iso.org/standard/76496.html.

[27] OGC: *OGC 23-028: Extraterrestrial GeoTIFF Engineering Report*, 2024.

[28] Carroll, Lewis, 1832-1898. *Alice's Adventures in Wonderland.* Peterborough, Ont. :Broadview Press, 2000.

[29] NASA. *Double Asteroid Redirection Test (DART).* https://science.nasa.gov/mission/dart

[30] IOGP: *EPSG Geodetic Parameter Dataset*. https://epsg.org

[31] Macdonald, A. (1983), *Clock synchronization, a universal light speed, and the terrestrial red-shift experiment*. American Journal of Physics, 51 (9): 795–797

[32] Kaplan, George H. *The IAU Resolutions on Astronomical Reference Systems, Time Scales, and Earth Rotation Models*, p. 1., 2006. https://arxiv.org/abs/astro-ph/0602086

[33] Time Ontology in OWL https://www.w3.org/TR/2022/CRD-owl-time-20221115/

[34] R. Penrose, *The Apparent Shape of a Relativistically Moving Sphere*. Proc. Camb. Phil. Soc., vol 55 Jul 1958.

[35] NASA NAIF: An Overview of SPICE (2023). https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/Tutorials/pdf/individual_docs/03_spice_overview.pdf

[36] WebVMT W3C Group Note (OGC 23-037): https://www.w3.org/TR/webvmt/

[37] Wikipedia: *Barycenter (astronomy)*. Retrieved January 11, 2024. https://en.wikipedia.org/wiki/Barycenter_(astronomy)

[38] Wikipedia: *Geodesic*. Retrieved January 11, 2024. https://en.wikipedia.org/wiki/Geodesic

[39] Wikipedia: *Reification (computer science)*. Retrieved January 11, 2024. https://en.wikipedia.org/wiki/Reification_(computer_science)

[40] Wikipedia: *Series expansion*. Retrieved February 22, 2024. https://en.wikipedia.org/wiki/Series_expansion

[41] Wikipedia: *Serialization*. Retrieved February 22, 2024. https://en.wikipedia.org/wiki/Serialization

[42] Karney, C.F.F: Algorithms for geodesics. J Geod 87, 43–55 (2013). https://doi.org/10.1007/s00190-012-0578-z

[43] Thornton, Stephen T.; Marion, Jerry B. *Classical Dynamics of Particles and Systems (5th ed.)*. Brook Cole. p. 49. ISBN 0-534-40896-6 (2004).