Technical report

Creating a 3D-BIM-compliant road design based on IFC alignment originating from an OKSTRA-accordant 2D road design using the TUM Open Infra Platform and the OKSTRA class library

Julian Amann, André Borrmann

Chair of Computational Modeling and Simulation

Leonhard Obermeyer Center

Technische Universität München

12.10.2015

Creating a 3D-BIM-compliant road design based on IFC alignment originating from an OKSTRA-accordant 2D road design using the TUM Open Infra Platform and the OKSTRA class library

Julian Amann, André Borrmann
Chair of Computational Modeling and Simulation
Leonhard Obermeyer Center
Technische Universität München

On behalf of the Federal Highway Research Institute (Bundesanstalt für Straßenwesen), Bergisch Gladbach

Preface

The contents of this analysis has been presented under the title “Open BIM für Infrastruktur – mit OKSTRA und IFC-Alignment zur internationalen Standardisierung des Datenaustauschs” as part of 6th OKSTRA-Symposium (May 20th - 21th 2015) in Cologne (Germany).

Introduction: Building Information Modeling

Building Information Modeling (BIM) stands for the continuous use of high-quality digital data over the entire lifecycle of a building – from the initial design and building construction to its operation and servicing and finally its demolition (Eastman et al. 2008). This avoids breaks in the flow of information and the need to repeatedly re-enter data manually, which is an error prone activity. The core of BIM is the digital building model, which serves as a comprehensive digital representation of the real building and contains, besides the detailed 3D geometry, alphanumeric information such as material types, building types or costs.

BIM yields considerable advantages in many areas of planning and execution (Borrmann et al. 2015). Working with a 3D model ensures that views and sections generated from the model are consistent with one another. BIM improves coordination within the various maintenance groups and helps the planner detect and prevent collisions early on. Quantities calculated from the digital building information model offer a reliable basis for tendering, bidding and associated cost controlling. In addition, the creation of a 3D BIM model can be combined with data on construction sequence to produce a 4D BIM model that can be used to verify processes as well as to plan and manage construction site logistics. The digital building model can be handed over to the building owner after completion of the building process, for direct use for facility management.

The adoption of BIM within the field of structural engineering is relatively advanced. Large public clients in the USA such as the General Service Administration or the US Army Corps of Engineers have been using model-based planning for some time. The same also applies to the Scandinavian countries, notably Finland and Norway. Great Britain is going to make BIM-based planning for all public construction projects mandatory starting from 1 April 2016.

In Germany, too, interest is growing among private and public planners in using BIM for executing building projects. Interest in the public sector focuses primarily on infrastructure buildings. Speaking in April 2014, Alexander Dobrindt (Federal Minister for Transport and Digital Infrastructure in Germany) remarked that “the digitization of building processes offers opportunities for large building projects to be realized on time and on budget.” According to Dobrindt an improved data basis increases transparency and networking among the

participants in a building project. Furthermore, this helps to estimate time schedules, costs and risks earlier and more precisely. “Building the modern way,” he said, “means to build first in the virtual environment and then in the real world.” Since then, the Federal Ministry of Transport and Digital Infrastructure has put the first pilot projects into action.

An important aspect for the success of BIM is the availability of open standards for the lossless exchange of high-quality building information models between software applications from different manufacturers. The Industry Foundation Classes (IFC), drawn up by the international organization buildingSMART, represents a standardized data model that meets these requirements and is now supported by many BIM applications (Borrmann et al. 2015). The data format was standardized as ISO 16739 and will be soon adopted by European standardization organizations, among them the German standardization institute DIN. If non-proprietary, open data formats are used for the execution of BIM projects, we speak of “Open BIM”. Up to the current version (4.0), IFC only supports structural engineering and ignores civil engineering. But, due to the rapidly increasing importance of “BIM for Infrastructure” around the world, the next big release, IFC 5, is scheduled to include a comprehensive civil engineering building extension that will make it possible to describe elements such as roads, railways, bridges, and tunnels.

The first step towards developing an IFC Infrastructure was made between April 2013 and March 2015 as part of the IFC Alignment project, which aimed to develop an alignment extension. This project involved researchers from the Technische Universität München and the French institute CSTB, as well as companies such as AEC3 Germany and Bentley Systems. The project was funded by the public organizations Trafikverket from Sweden and Rijkswaterstaat from the Netherlands. To ensure worldwide acceptance of the future standard, an international expert panel was set up that accompanied the elaboration of the data model in a series of workshops. In February 2015, the data model was promoted to “Candidate Standard” status and passed successfully through the public review process. After the formal adoption by the buildingSMART Standard Commitee the data model will be raised to the state of an official standard (final standard).

The IFC alignment data model will be the basis for many other infrastructure-related data models such as IFC Road, IFC Bridge and IFC Tunnel (Figure 1). Currently, buildingSMART is still looking for funding for these important standardization works, which should be started soon.

To ensure compatibility between the OKSTRA data format standard, a widely used data format within German speaking countries, and the future international IFC Alignment standard, the Chair of Computational Modeling and Simulation from the TU München has been commissioned by the Federal Highway Research Institute (Bundesanstalt für Straßenwesen, BASt) to develop a conversion method between the two formats. This work presents the formats in detail, introduces the differences and specific characteristics and describes an approach for developing conversion functionalities.

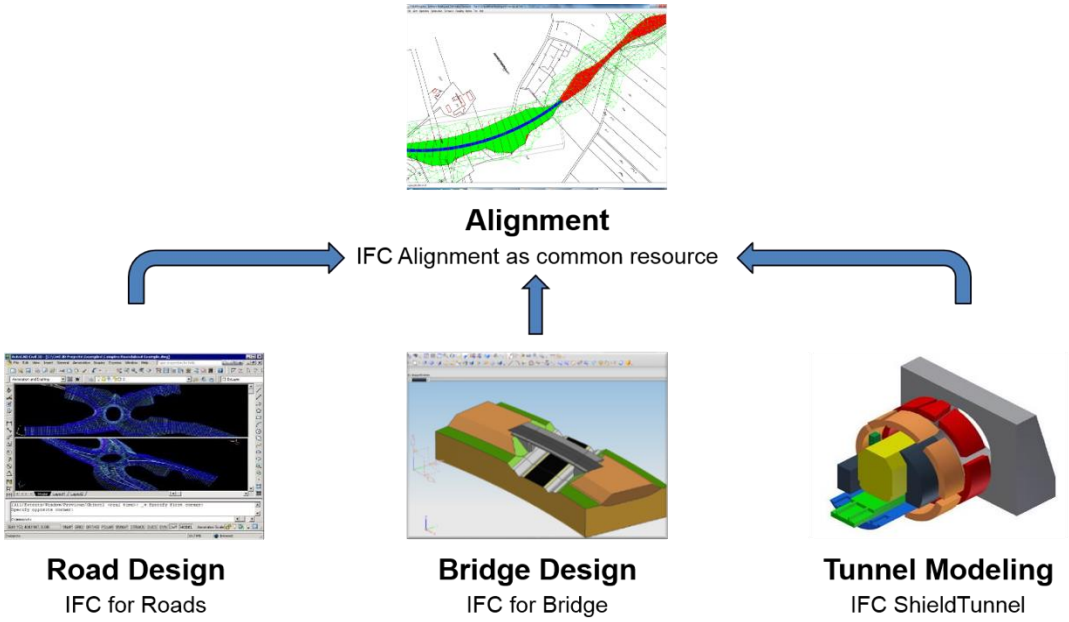

Figure 1: IFC Alignment describes the alignment and can serve as a basis for the development of future standards such as IFC Road, IFC Bridge and IFC Tunnel.

The data model IFC Alignment

IFC Alignment serves as a description of road and rail alignments. The model is based on the well-established approach of aligning design based on vertical (gradient) and horizontal alignments. The underlying conceptual model (Amann et al. 2014) was developed in cooperation with the OpenGIS Consortium (OGC). IFC Alignment was developed based on this model. In the future, in the context of OGC, the InfraGML standard will be developed that is based on the Geography Markup Language (GML). The jointly developed conceptual model also acts here as a common basis. However, the draft by the OGC also covers further use cases besides IFC Alignment, such as survey or land parcel management. That means that in contrast to IFC Alignment, InfraGML will be used for managing existing assets within GIS environments. Through the cooperation between buildingSMART and OGC, which have developed the common conceptual model on the basis of UML, the harmonization of both standardization projects should be ensured, with a promising connection between the worlds of BIM and GIS as the result.

Figure 2 shows an overview of the newly introduced entities in the IFC 4 schema. A key element is the new *IfcAlignment* class. It references the horizontal alignment (*IfcAlignment2DHorizontal*) and the vertical alignment (*IfcAlignment2DVertical*). The horizontal alignment itself consists of the well-known alignment elements line (IfcLineSegment2D), arc (IfcCircularArcSegment2D) and clothoid (IfcClothoidalArcSegment2D). Besides the clothoid, the standard does not currently contain any other transition curve types. However, new transition curve types such as the Bloss curve will be introduced in future as part of the planned expansions IFC Road and IFC Rail. Line, arc and transition curve share a common subset of data attributes. A common base class named *IfcCurveSegment2D* has therefore been introduced that comprises common data attributes such as start position/direction and segment length. The class *IfcAlignment2DHorizontal* itself consist of an ordered list of *IfcAlignment2DHorizontalSegment* elements, which each reference a concrete alignment element (line, arc or clothoid).

A similar approach was followed for the vertical alignment. Parabolas (*IfcAlignment2DVerSegCircularArc*) and arcs (*IfcAlignment2DVerSegParabolicArc*) for roundings and lines (*IfcAlignment2DVerSegLine*) can be found in the vertical alignment. The

three elements mentioned are, in turn, derived from the class *IfcAlignment2DVerticalSegment*, since they share common properties such as the start gradient of an element. An ordered list of the mentioned alignment elements is managed by the class *IfcAlignment2DVertical*. This class is referenced, in turn, by the class *IfcAlignment*.

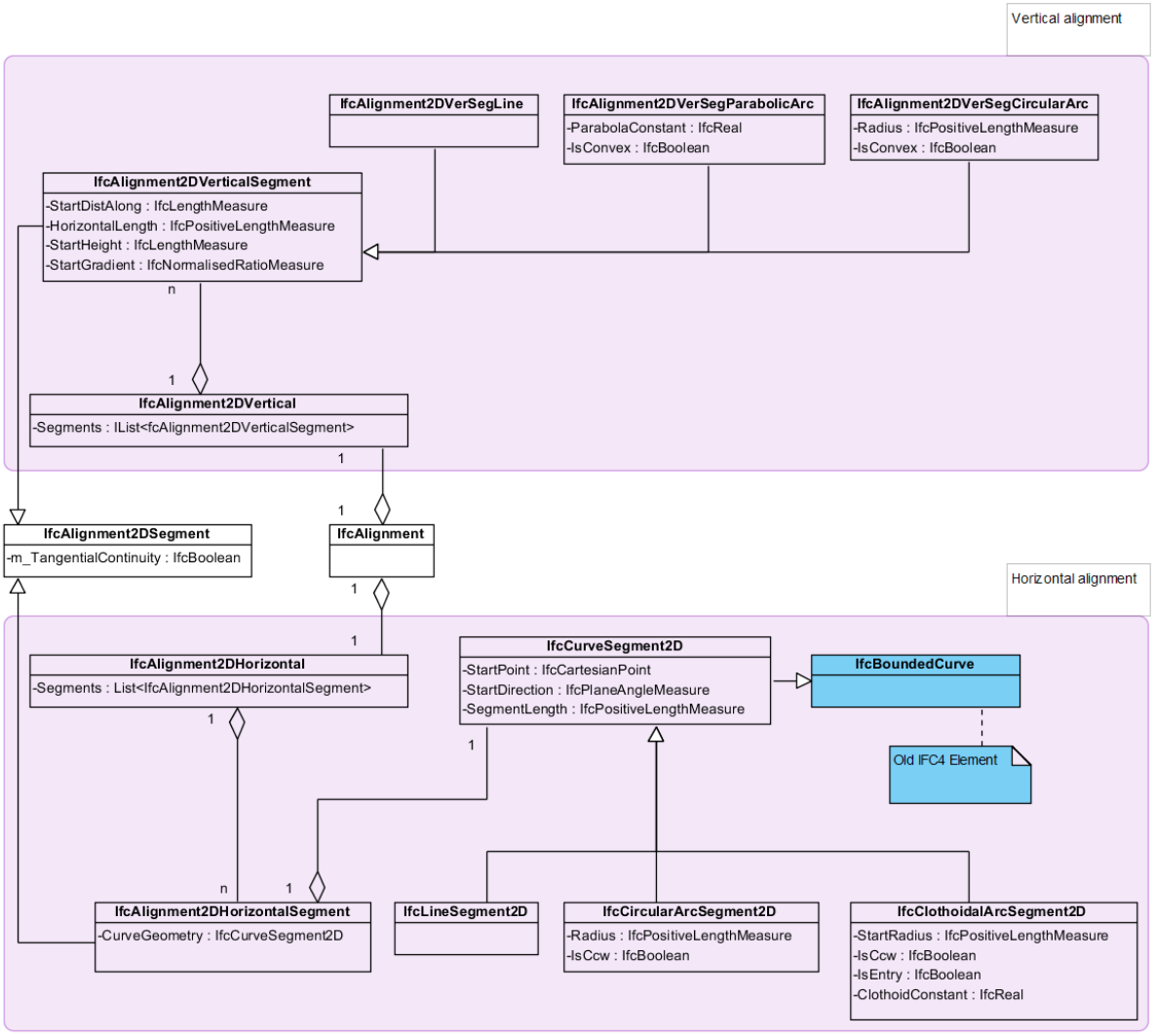

Figure 2: UML class diagram that shows the most important innovations of the IFC Alignment extension

Figure 3 shows an overview of the different parameters of the alignment elements used in the horizontal alignment. As already described, all the alignment elements of the horizontal alignment have a start point (*StartPoint*), a start direction (*StartDirection*) and a length (*SegmentLength*). In addition, the arc has a radius (*Radius*) and an attribute that describes the orientation of the arc (*isCcw*). CCW is short for *counterclockwise* and this attribute is therefore "*true"* if the arc is counterclockwise and "*false*" if not. The clothoid provides a start radius (*StartRadius*) which determines the radius of the clothoid at the start point. If the curvature is 0 at the start point then the start radius is infinity. In this case no value is stored for the start radius. The attribute *isCcw* of the clothoid describes the orientation of the clothoid, as with the arc. The attribute *isEntry* defines if the curvature is increasing or decreasing from the start to the end point of the clothoid. If the curvature is increasing, then the value of *isEntry* is "*true*", if not then "*false*". The last attribute of the clothoid is a clothoid constant (*ClothoidConstant*).

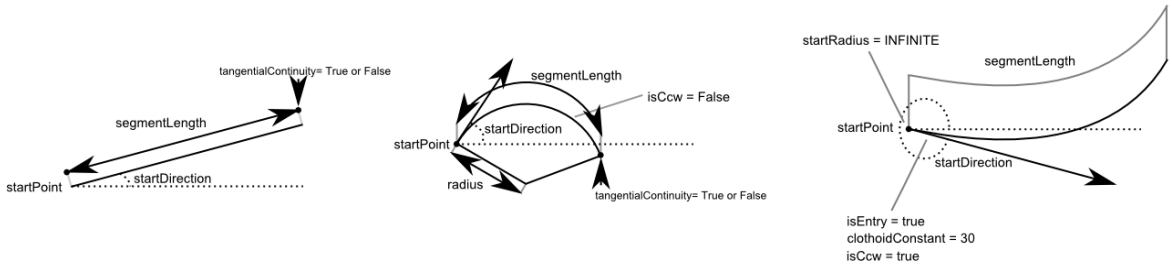

Figure 3: Overview of the different parameters of the alignment elements used in the horizontal alignment. From left to right: IfcLineSegment2D, IfcCircularArcSegment2D and IfcClothoidalArcSegment2D

An alignment (*IfcAlignment*) is always gap free. To describe a gap between two alignments, two separate *IfcAlignment* elements have to be used. This means that an *IfcAlignment* element always consists of a connected sequence of horizontal or vertical alignment elements (horizontal: line, arc, clothoid; vertical: line, parabola, arc). The connectivity between the continuous horizontal and vertical segments does not necessarily have to be tangential. Figure 4 shows an alignment where tangential continuity is not fulfilled. The class *IfcAlignment2DSegment* has an attribute *TangentialContinuity* that defines whether two segments are tangential (*true*) or not (*false*). The tangential continuity flag makes it possible to check whether the calculated end direction of the previous segment matches the provided start direction of the current segment. *IfcAlignment2DHorizontalSegment* and *IfcAlignment2DVerticalSegment* inherit the attribute from this class and can modify the corresponding property of the element.

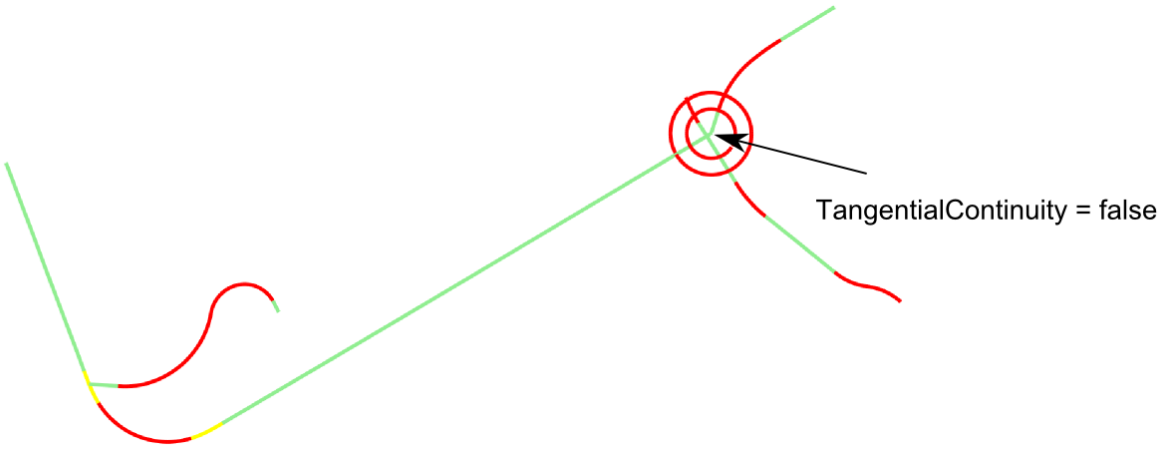

Figure 4: The connectivity between the continuous alignment segments does not necessarily have to be tangential

Figure 5 shows an example of a vertical alignment. The common basis class of all vertical alignments is the class *IfcAlignment2DVerticalSegment*. It has different attributes that are used to describe the individual segments of the vertical alignment. The attribute *StartDistAlong* defines the start station of the corresponding alignment element. In the figure, the line (*IfcAlignment2DVerSegLine*) starts at point AA6 and ends at point AA8. The start gradient is described by the attribute *StartGradient* and corresponds to the slope of the line through the points AE6 and AA8 in the illustrated case. The attribute *StartHeight* defines the height of the start point of the corresponding vertical alignment element. The horizontal length of a vertical alignment is described by the attribute *HorizontalLength*. This is not to be confused with the

ordinary real length of the segment; it describes the horizontal length in the vertical alignment that relates to the length in the horizontal alignment.

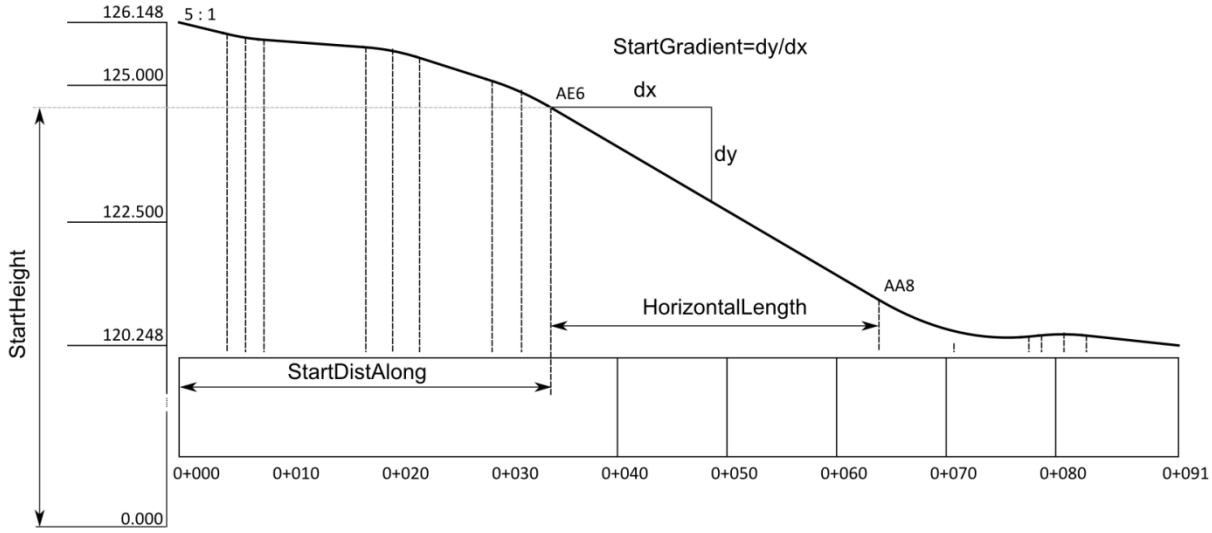

Figure 5: Common data attributes of vertical alignment

More details to the IFC Alignment data model can be found in (Liebich 2014).

The digital elevation model is mapped on the geometry level to an *IfcTriangulatedFaceSet* and is linked on the semantic level with the element *IfcGeographicElement*. The *IfcGeographicElement* entity is used to tag terrain data.

The OKSTRA data model

The OKSTRA data model describes an alignment by using the *Trasse* class (German word for alignment); see Figure 6.

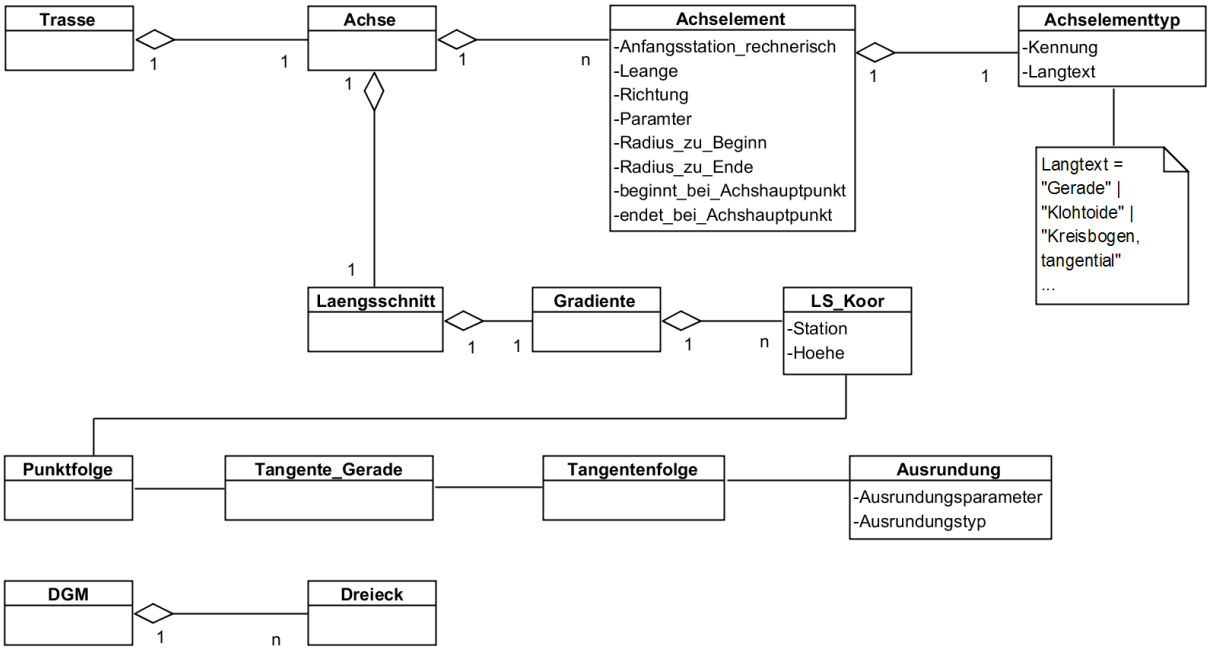

Figure 6: OKSTRA data model for alignment data

The *Trasse* (Alignment) class references an element of type *Achse* (Axis). The *Achse (Axis)* class references an ordered set of axis elements of type *Achselement* (axis element), which

describe horizontal alignment elements. Using those elements an axis element type can be defined e.g. “*Gerade*” (line), “*Klothoide*” (clothoid) or “*Kreisbogen, tangential*” (arc, tangential).

The OKSTRA standard describes the vertical alignment using a vertical point of intersection approach (VPI approach) instead of the segment-based approach used by the IFC Alignment standard. The segment-based approach stores data about the individual segments, e.g. the start and end point is stored for each line and every parabola. The VPI approach stores only the points of vertical intersection and the radius of curves. The start and end points of the individual segments have to be computed explicitly here. Figure 7 shows the VPI approach in comparison to the segment-based approach. Each of the two approaches can be converted to the other. The segment-based approach used for the IFC Alignment project was chosen after intensive discussion within the international community, primarily because the horizontal alignment also uses a segment-based approach.

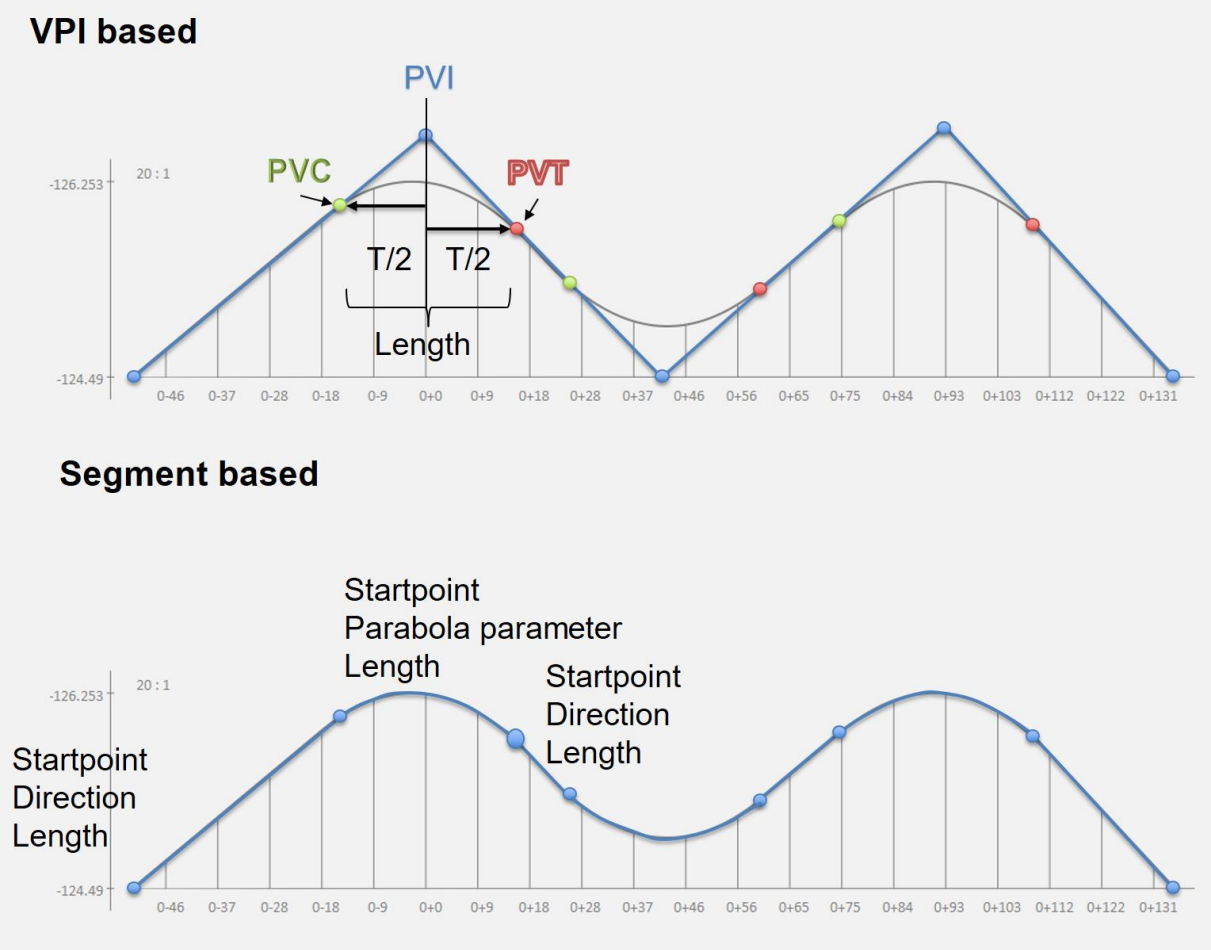

Figure 7: VPI approach compared to the segment-based approach

The vertical alignment is stored in the *Laengsschnitt* (longitudinal section) class and is referenced by the *Achse* (axis) class. The longitudinal in turn references the gradient (of type *Gradiente* = gradient). The gradient consists of points of vertical intersection and optionally curves and curve types. Navigation through a series of associations is needed to access the corresponding attributes (*Punktfolge*, *Tangente, Gerade*, *Tangentenfolge*).

A digital elevation model (DEM) can be stored in the form of a triangle description. The class *DGM* (short for Digitales-Gelände-Modell = digital elevation model) serves this propose. It manages a list of triangles of type *Dreieck* (triangle).

Conversion

To convert horizontal alignment from IFC Alignment to OKSTRA is straightforward since both standards use a segment-based approach for the horizontal alignment and have a relatively similar structure. For instance, the property *beginnt_bei_Achshauptpunkt* can be used to determine the start point of an alignment element for IFC Alignment. Mapping the different attributes of both standards is therefore straightforward. Table 1 shows an example of some conversions:

Table 1: Conversion of the horizontal alignment from IFC Alignment to OKSTRA

IFC Alignment	**OKSTRA**
IfcLineSegment2D.StartPoint	Achselement.beginnt_bei_Achshauptpunkt
IfcLineSegment2D.StartDirection	Achselement.Richtung
IfcLineSegment2D.SegmentLength	= distance(Achselement.beginnt_bei_Achshauptpunkt Achselement.endet_bei_Achshauptpunkt) oder Achselement.Laenge
IfcCircularArcSegment2D.Radius	Achselement.Radius_zu_Beginn
IfcClothoidalArcSegment2D.StartRadius	Achselement.Radius_zu_Beginn
IfcClothoidalArcSegment2D.IsEntry	1/Achselement.Radius_am_Beginn < 1/Achselement.Radius_am_Ende
IfcClothoidalArcSegment2D. ClothoidalConstant	Achselement.Parameter

The conversion of the vertical alignment is a bit more difficult. The corresponding conversion algorithm in pseudo code is shown in Table 2 (this algorithm is written for OKSTRA version 1.014).

Table 2: Algorithm for converting a vertical alignment from OKSTRA to IFC Alignment

1. *Visit each "LS_Koor" of the corresponding gradient*
    a. *Determine the station and height of the point of vertical intersection (read attribute Station and Hoehe)*
    b. *Visit attribute "folgt_auf_LS_Koor" of type "Punktfolge" if it is present*
        i. *Visit Punktfolge (if present) and read attribute "hat_Tangente_Gerade" of type "Tangente_Gerade"*
        ii. *Visit Tangente and read attribute "folgt_auf_Tangente_Gerade" of type "Tangentenfolge" (if present)*
        iii. *Visit Tangentenfolge and read attribute "hat_Ausrundung" of type "Ausrundung" (if present)*
        iv. *Visit Ausrundung and read attribute "Ausrundungsparameter" and "Ausrundungstyp" (if present)*
2. *Compute start and end gradient as well as parameter T/2*
3. *Compute the start and end points of the vertical alignment elements*

In its first pass, the algorithm visits all points of vertical intersection and collects all the available information. This includes the position values (station and height) and optimally the curve parameters. In the second pass, all start and end gradients are computed as well as the parameter $T/2$ (see Figure 7). In the last step, the start and end points of the vertical alignment elements are computed and the IFC Alignment specific alignment segments are created.

OKSTRA version 2.016 introduced several changes to the vertical alignment in comparison to version 1.014. In particular, the object *Laengsschnitt* has been removed and the object *LS_Koor* has been replaced by the object *Grad_Koors*. The amendments make it easier to

access parabola parameters. By way of example, Table 3 list some mappings of the vertical alignment of IFC Alignment elements to the vertical alignment elements of the OKSTRA (2.016) product data model.

Table 3: Conversion of the vertical alignment from IFC Alignment to OKSTRA 2.016

IFC-Alignment	**OKSTRA**
`IfcAlignment2DVerticalSegment .StartDistAlong`	`Gradiente.hat_Grad_Koor.Station`
`IfcAlignment2DVerticalSegment .HorizontalLength`	`Gradiente.hat_Grad_Koor.Station[index+1]` - `Gradiente.hat_Grad_Koor.Station[index]`
`IfcAlignment2DVerticalSegment .StartHeight`	`Gradiente.hat_Grad_Koor.Hoehe`
`IfcAlignment2DVerticalSegment .StartGradient`	$\frac{\text{Gradiente.hat}_{\text{Grad}_{\text{Koor}}}.\text{Hoehe[index}+1] - \text{Gradiente.hat\_Grad\_Koor.Hoehe[index]}}{\text{Gradiente.hat}_{\text{Grad}_{\text{Koor}}}.\text{Station[index}+1] - \text{Gradiente.hat\_Grad\_Koor.Station[index]}}$
`IfcAlignment2DVerSegParabolicArc .ParabolaConstant`	= abs(FocalLength * 2) FocalLength = Distance between focus and apex = 1.0 / (4.0 * a) with a = 0.5 * ((slopeAtEnd – slopeAtEnd) / (endStation - startStation) Start and end station can be determined using `Gradiente.hat_Grad_Koor.Station`. Slope at start: see `IfcAlignment2DVerticalSegment .StartGradient` Slope at end: Analog to `IfcAlignment2DVerticalSegment .StartGradient`
`IfcAlignment2DVerSegParabolicArc .IsConvex`	If FocalLength * 2.0 < 0 then `IsConvex` = false otherwise true.
`IfcAlignment2DVerSegCircularArc .Radius`	Arcs within the vertical alignment are not supported by OKSTRA
`IfcAlignment2DVerSegCircularArc .IsConvex`	Arcs within the vertical alignment are not supported by OKSTRA

In addition to the 2D-based approach using a vertical and horizontal alignment, it is also possible to store a pure 3D-based alignment within IFC Alignment, but the preferred option within IFC Alignment is the 2D-based approach. Furthermore, OKSTRA does not support pure 3D-based alignments.

The decision to favor 2D representation within IFC Alignment is a product of the fact this is also used to store the original engineering parameters of the alignment (curvature, radius, slope, etc.). This means such information is directly accessible, verifiable and easier to manipulate. If needed, a 3D-based alignment can always be computed from the 2D base alignment. The other way around would only be possible to a very limited extent. That said, it is also possible to store a pure 3D-based alignment using IFC Alignment. This ensures that viewers without alignment processing are able to display an alignment correctly. A limitation in this respect is that the redundant storage of a 2D and 3D alignment can lead to inconsistencies. IFC Alignment will serve as a basis for IFC Bridge, IFC Tunnel, etc. The components defined therein will in any case be modelled primary as 3D geometry.

Cross sections or the pavement design specifications are not covered by IFC Alignment as the purpose of IFC Alignment is purely to describe the alignment in its horizontal and vertical components. However, proposals for handling cross sections already exist such as (Amann et al. 2015) or (Singer et al. 2014). The proposal for extending IFC Alignment described in (Amann et al. 2015) is visualized in Figure 8.

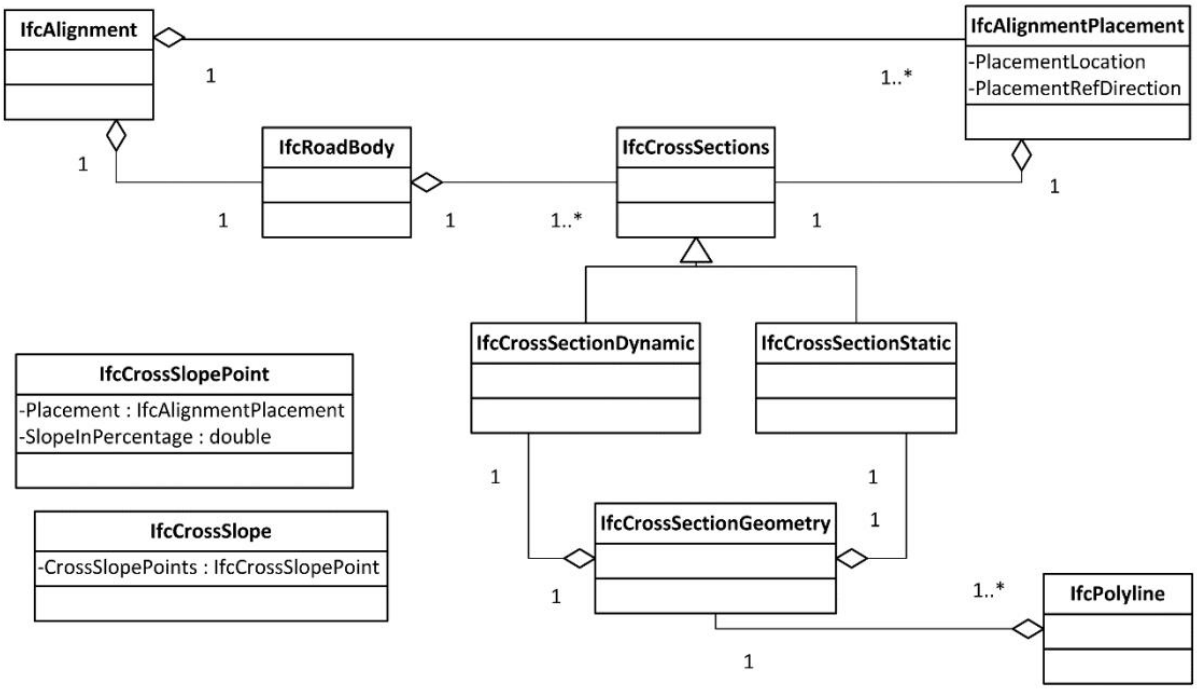

Figure 8: Proposal for extending IFC Alignment with cross sections

More details to this extension can be found in (Amann et al. 2015) and (Singer et al. 2014).

Cross sections and pavement design specifications will be considered in buildingSMART's IFC Road project in the future. As part of buildingSMART's INFRA Room, other possibilities of adding road bodies have also been discussed, such as the Line String approach (see Inframodel.fi 2014) as used by the Finnish road standard. It is currently not clear which approach will prevail in the end or if a hybrid approach will be taken.

The TUM Open Infra Platform

The TUM Open Infra Platform (OIP) has been developed by the Chair of Computational Modeling and Simulation at the Technische Universität München for viewing alignment and digital elevation model data. OIP supports several file formats for alignment and digital elevation data. It has export and import functions for IFC Alignment, LandXML, and OKSTRA. Furthermore, it permits the import of ASCII-XYZ data and laser scan data in LAS 1.1/1.2 format and has partial support for IFC 2x3, IFC 4, and IFC Bridge. Figure 9 shows the different possibilities for file conversion. In particular, OIP can be used to convert OKSTRA data to IFC Alignment data. The other way around also works.

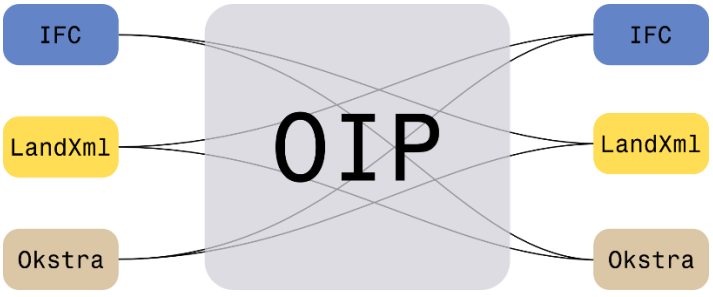

Figure 9: File conversion possibilities within the TUM Open Infra Platform

The program can be used for free and can be downloaded from the website https://www.cms.bgu.tum.de/oip. The website contains more information about the system requirements, installation instructions as well as documentation of the different features. Figure 10 shows a screenshot of the TUM Open Infra Platform.

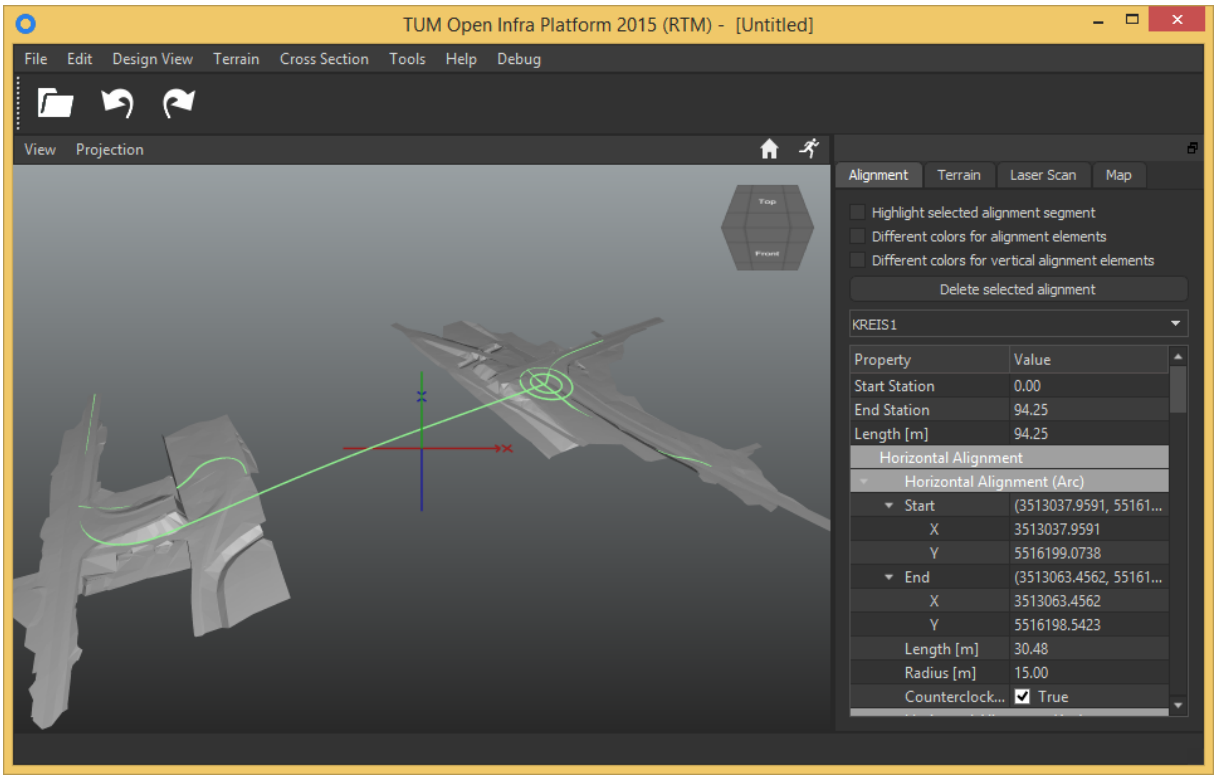

Figure 10: Screenshot of TUM Open Infra Platform

Figure 11 shows an import of digital elevation model (Figure 11, left) data in the XYZ format. Additionally, an import of laser scan data in the LAS format is shown (Figure 11, right).

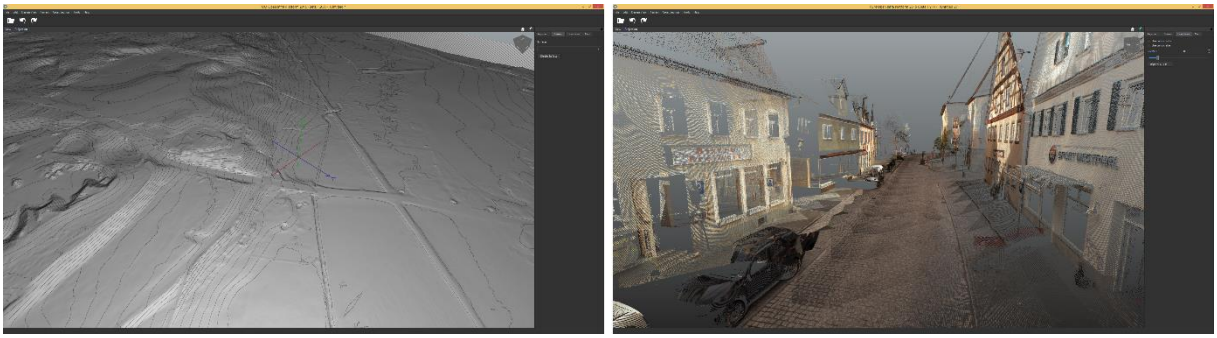

Figure 11: Left: XYZ import. Right: LAS import

Tests performed

In our software test, we created 50 test files in the IFC Alignment data format. Using the TUM Open Infra Platform we then converted these files to the OKSTRA data format and did a visual check. In the same manner we also converted the OKSTRA data file to the IFC alignment data format and checked them visually. Figure 12 shows three examples of our visual tests. The first column shows IFC alignment files and the second one shows the corresponding OKSTRA files. The generated OKSTRA files were also validated using the OKSTRA tool (Werkzeug), Version 1.3.0.25.

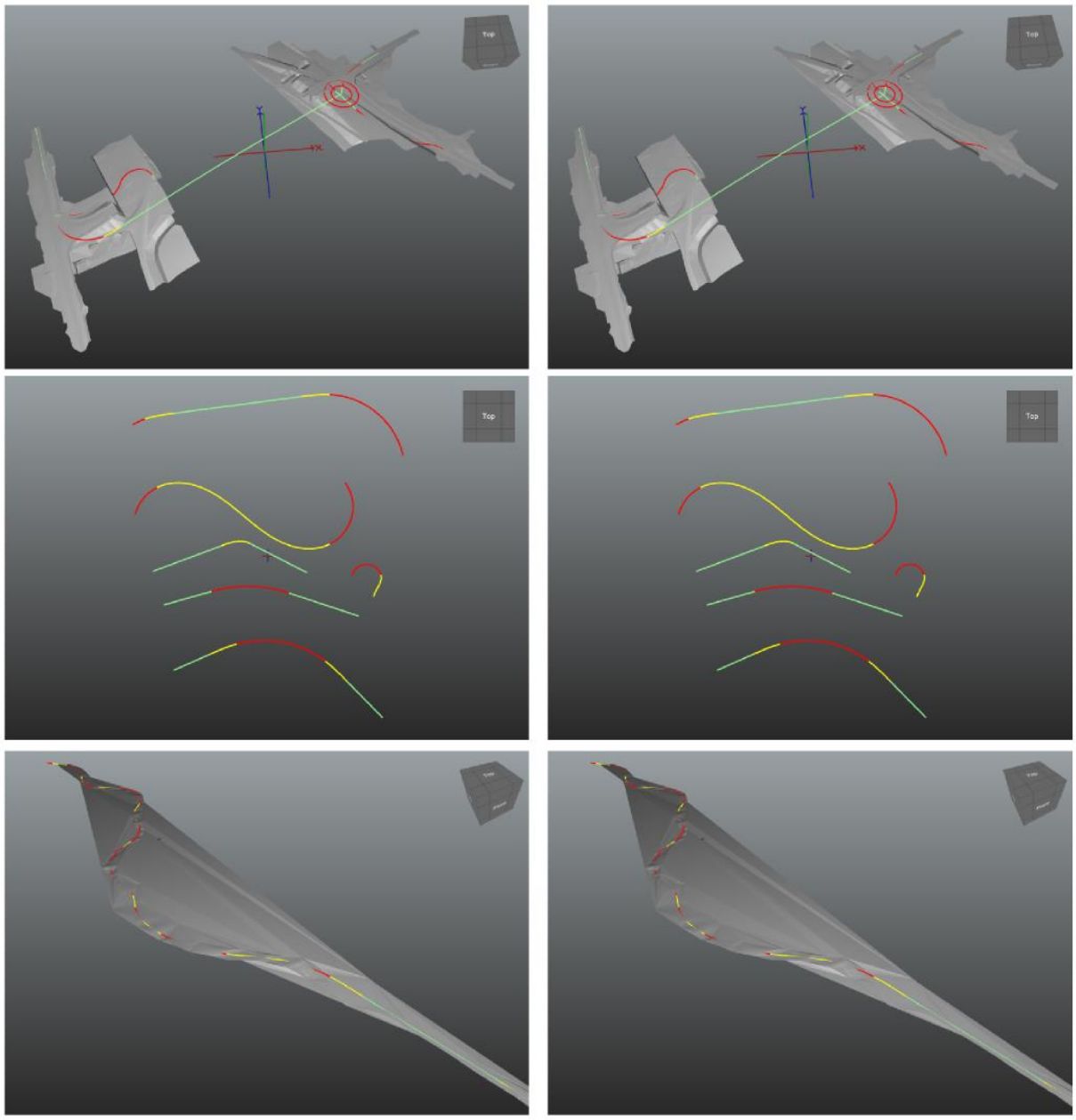

Figure 12: Visual checking of converted files

The TUM Open Infra Platform uses floating point numbers with single precision (Float32) and double precision (Float64) according to IEEE 754. Those provide only a limited accuracy as demonstrated in Code Snippet 1.

```
#include <iostream>

int main() {
	if (0.362 * 100.0 != 36.2)
		std::cout << "different" << std::endl;

	if (0.123 * 100.0 / 100.0 != 0.123)
		std::cout << "also different" << std::endl;

	return 0;
}
```

Code Snippet 1: A C++ program that returns the output "different" and "also different"

Since the conversion from IFC Alignment to OKSTRA uses also arithmetic operations based on double precision, the conversion can lead to small differences.

To assess the differences that can occur when converting an IFC Alignment file to OKSTRA and vice versa, we implemented a simple test. This test converts an IFC Alignment file (orginal.ifc) to an OKSTRA file. Subsequently the OKSTRA file is then converted back to an IFC Alignment file. This process corresponds to one conversion cycle in our test. We repeated this conversion cycle 10 times and examined the resulting differences as shown in Figure 13. The difference in the end curvature amounts to 6.0×10^{-18} and is therefore nearly 0. The start and end positions have greater differences, but are also within acceptable limits.

Figure 13: Differences after multiple conversions

The numerical stability of our conversion algorithm can, of course, be optimized, but the resulting differences are acceptable.

Outlook and areas for further research

The IFC Alignment extension represents the first step towards supporting infrastructure projects in the IFC world. The IFC data model still lacks the possibility to describe road cross sections or tunnel and bridge buildings, and the corresponding elements are likewise lacking for railway design. However, several proposals have already been made (Yabuki et al. 2007, Yabuki et al. 2008, Lebegue et al 2012, Amann et al. 2013, Amann et al. 2015). On the basis of these proposals, official buildingSMART projects should emerge in the near future to promote the international standardization process, and to establish a strong standard for the complete description of infrastructure buildings at an international level.

At present, the following buildingSMART International (bSI) projects are currently in preparation: IFC Rail, IFC Road, and IFC Bridge. All three projects will use IFC Alignment as a common foundation. Furthermore, an extension project of IFC Alignment is planned which will be named IFC Alignment 1.1.

In the context of IFC Alignment 1.1, new transition curves will be introduced, such as Bloss-curves, as well as a uniform way of referencing an alignment. This will, for example, make it possible to bind cross sections of a road to certain station values of the corresponding alignment. Finally, offset alignments are also planned as part of the IFC Alignment 1.1 project. The financial funding of the IFC Alignment 1.1 project has not yet been fully clarified. Support

from the previous funding bodies (Trafikverket and Rijkswaterstaat) seems likely, but it would be desirable to bring other governmental organizations and countries into the project.

German institutes should, where possible, participate in this and other infrastructure extensions of the IFC standard, through their own research projects and as consultative partners in order to guarantee that German interests are taken into account in the creation of European and international standards.

This also applies to the IFC Road project, which aims to extend IFC 4 for road design. Substantial preparatory work has been done in this field already (significantly from Korea and China). We strongly advise German authorities and software manufacturers to take part in the IFC road projects, as the extensive experience that they have gained through the development of OKSTRA can contribute to international standardization. At the same time, it would be good to ensure maximum compatibility between the OKSTRA Standard and the infrastructure extensions to IFC.

References

Amann, J.; Borrmann, A,; Hegemann, F.; Jubierre, J.R.; Flurl, M.; Koch, C.; König, M.: A Refined Product Model for Shield Tunnels Based on a Generalized Approach for Alignment Representation, In: Proc. of the ICCBEI 2013, Tokyo, Japan, 2013

Amann, J., Borrmann, A., Chipman, T., Lebegue, E., Liebich, T., Scarponcini, P., P6 IFC Alignment – Conceptual Model, http://www.buildingsmart-tech.org/downloads/ifc/ifc5-extension-projects/ifc-alignment/ifcalignment-conceptualmodel-cs, 2014, buildingSMART

Amann, J.; Singer, D.; Borrmann, A.: Extension of the upcoming IFC alignment standard with cross sections for road design, In: Proc. of the ICCBEI 2015, Tokyo, Japan, 2015

Borrmann, A., König, M., Koch, C., Beetz, J. (Hrsg.): Building Information Modeling – Technologische Grundlagen und industrielle Praxis, Springer Vieweg, 2015

Eastman, C., Teicholz, P., Sacks, R., Liston, K., BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers and Contractors, Wiley Publishing, 2008

Inframodel.fi (2014). Application schema, version 3.0.1, http://www.inframodel.fi/en/

Lebegue, E., Fiês, B. and Gual, J., (2012). IFC-Bridge V3 Data Model – IFC 4, Edition R1

Liebich, T., IFC Alignment Schema, http://www.buildingsmart-tech.org/downloads/ifc/ifc5-extension-projects/ifc-alignment/ifcalignment-ifcextension-cs, 2014, buildingSMART

Singer, D.; Amann, J.: Erweiterung von IFC Alignment um Straßenquerschnitte In: Proc. of the 26th Forum Bauinformatik, Darmstadt, Deutschland, 2014

Yabuki, N., Azumaya, Y., Akiyama, M., Kawanai Y., Miya, T. (2007): Fundamental Study on Development of a Shield Tunnel Product Model. Journal of Civil Engineering Information Application Technology 16, pp. 261-268

Yabuki, N. (2008). Representation of caves in a shield tunnel product modeling. In Proc. of the 7th European Conference on product and Process Modeling. Sophia Antipolis, France.