Integration of Movement Data into 3D GIS

Joie Lim[1,2], Filip Biljecki[1,3], Rudi Stouffs[1]

[1]Department of Architecture, [2]Department of the Built Environment, [3]Department of Real Estate
National University of Singapore, Singapore
(joie.lim, filip, stouffs)@nus.edu.sg

Keywords: Movement trajectories, OD data, CityGML, Dynamizer, SensorThings, OGC.

Abstract

With the rise in usage of digital twins in the field of urban planning, the integration of data sources such as sensor data in 3D city models is a challenge that is often brought up. This is even more so with movement data, as they contain a geographical or geometrical aspect that is often overlooked. This paper looks at existing methods to integrate sensor data with 3D city models, and especially at the Dynamizer module in CityGML that supports the integration of dynamic data into the models directly. We look at how the Dynamizer module works with movement data and compare it with another method, SensorThings API. The comprehensive investigation we conduct involved different types of movement data, at different spatial and temporal scales, e.g. origin-destination public transport data in Singapore and London, migration routes in the USA, and highly detailed customer movement in a supermarket in China. While the Dynamizer is more flexible in representing data, it has less support and it is more restrictive when accessing the data. On the other hand, the FROST-Server implementation of the SensorThings API is more restrictive in representing data, however, accessing the data is more flexible.

1. Introduction

The concept of Digital Twins was initially proposed as a virtual model of a physical product to enable testing, optimization and simulations in manufacturing (Grieves, 2014). It was later adapted to be an integrated multi-scale simulation of physical entities to mirror the life of its twin for a better understanding of them (Glaessgen and Stargel, 2012). Over time, the concept of Digital Twins expanded and diverged as new industries adopted it and it is applied to new case studies (Guo and Lv, 2022; El Saddik, 2018; Kaiblinger and Woschank, 2022). Some examples include the manufacturing industry (Tao et al., 2019; Parris et al., 2016; Qi et al., 2018; Jwo et al., 2022), the medical field (Karakra et al., 2018), the aerospace industry (Glaessgen and Stargel, 2012; Reifsnider and Majumdar, 2013), and business (Agalianos et al., 2020).

Digital twins have become popular in the field of urban planning and in conjunction with smart cities. Many cities such as Incheon, Helsinki and Singapore already have existing digital twins (Milner, 2021; Ruohomäki et al., 2018; Kobie, 2022), while others such as Las Vegas and Los Angeles are in the process of developing their own (Plautz, 2022; Goldin, 2022). They vary in implementation and usage, but refer to virtual city models of the built environment. They are based on 3D models with geometric and semantic information, and are able to incorporate real-time (dynamic) data to enable analysis, simulation and prediction to inform intervention decisions in the real world, in the areas of urban design, planning and systems operation (Dembski et al., 2020). Some examples of use include being used to facilitate public participation, city management and road life management (Bouzguenda et al., 2019; Yun and Lee, 2019; Svítek et al., 2020; Yu et al., 2021).

Of the many challenges that one faces when attempting to implement an Urban Digital Twin (UDT), integration is one of the major challenges that is often brought up in studies (Lei et al., 2023). This process includes both the integration of data sources, such as various 2D and 3D sources for geometric data and various real-time sensors, as well as the integration of systems, such as differently structured databases and standards (Ramu et al., 2022; Lu et al., 2020; Li et al., 2020).

To support such an integration, a number of methods and technologies have been investigated. Among these are the utilization of existing technology such as the Internet of Things (Rivera et al., 2020; Duan et al., 2020) and standards such as CityGML, Observations and Measurements, Sensor Observation Service and SensorThings. While these are not standards developed specifically for digital twins, they are commonly used in various implementations as they cover areas such as 3D city models, sensor data management and retrieval of data.

In digital twin implementations, CityGML is often used as the standard for the 3D city model as it supports both geometric and semantic information. CityGML has finalized its conceptual model for CityGML 3.0 (Kolbe et al., 2021). Among the changes made since CityGML 2.0 is the inclusion of the Dynamizer as one of its core modules. Previously developed as an Application Domain Extension (ADE) for CityGML 2.0, the Dynamizer allows the integration of dynamic data into an otherwise static CityGML model. It does this by allowing one to specify properties or features to be replaced by dynamic values, either defined in-line or linked externally.

In their analysis of the importance of including time-dependent properties into CityGML, the developers of the Dynamizer, Chaturvedi and Kolbe (2019) have highlighted the role that these kinds of properties play in areas of study such as smart cities and digital twins, urban mobility, urban simulations, urban planning, and history and archeology.

While much work has been put into the integration of various kinds of sensor data within 3D city models over the years, especially through initiatives by the OGC Innovation Program such as the Future City Pilot (Chaturvedi and Kolbe, 2017) and the 3D IoT Platform for Smart Cities Pilot (Coors, 2020), not as much has been done with movement data. Even though urban mobility has been pointed out as an important area of study which can benefit from such integration.

Movement data science is the study of spatial-temporal information, to understand what factors affect how and why dynamic entities move. These studies play an important role in urban planning, crisis mitigation and public health as well as understanding ecology. In recent years, the availability and quality of spatial-temporal information and connected contextual information has increased greatly. This trend is due to advancements in GPS and other tracking technology, and the increase in such data collected through widespread use of mobile phones and social media, resulting in more possibilities and development in movement analysis methods (Dodge et al., 2016).

One of the most important aspects of movement analysis is the study of the relationship between the moving entity and its surroundings (Purves et al., 2014; Siła-Nowicka et al., 2016). Such research requires both the representation of the movement data and the surroundings, preferably in an integrated fashion.

In this study we look at how moving entities may be represented in CityGML. In Section 2, we elaborate on the Dynamizer as a means to integrate dynamic data in CityGML, whereas in Section 3, we compare the use of the Dynamizer method with other methods and techniques described in literature to integrate dynamic data information into CityGML. Subsequently, we distinguish two types of movement data (Section 4) and investigate two different methods of integrating dynamic data with CityGML based on six case studies and data sets (Section 5). Finally, we compare and evaluate the two methods in Section 6 and offer a discussion and conclusion in Section 7.

2. Dynamic Data in CityGML

2.1 CityGML and Dynamizer

CityGML was originally developed to be an open standard for the storage and sharing of 3D city model data. It focused on the documentation of both geometric as well as semantic information of objects within a city (Gröger and Plümer, 2012). It is able to support features and information at different levels of detail and at different connected levels of hierarchy with its semantic structure. For example, a building could be linked to subdivisions or parts like connected blocks or a garage, and these could be further linked to rooms within them. CityGML also had different optional thematic modules such as buildings, transportation and vegetation for describing elements across different domains. In addition to these thematic modules that are built into CityGML, Application Domain Extensions (ADE) allow users to add new object types and attributes to the existing CityGML schema (Biljecki et al., 2018; Biljecki et al., 2021).

During the process of development of CityGML 3.0, one of the main changes planned was the support for changing data (Chaturvedi and Kolbe, 2016; Kutzner and Kolbe, 2018). This was classified into two different types of time-sensitive information. The first was long term and slow changes, such as changes to the city through the construction or demolition of buildings. These were to be updated in the form of different versions of the 3D City Model. A comprehensive versioning system for CityGML was conceptualised by Chaturvedi et al. (2017a). The second type of time-sensitive information was short-term and fast-paced changes, such as data collected from sensors. These were conceptualised to be able to update real-time, without the need to update the other static portions of the model, leading to the inclusion of the Dynamizer module. The Dynamizer module was first introduced in the form of the Dynamizer ADE for CityGML 2.0, developed by Chaturvedi and Kolbe (2016) for integrating dynamic data into CityGML.

Through the years, the Dynamizer ADE and Dynamizer module have been demonstrated to be used with different types of data and in different use cases. One example was HVAC and building energy analysis (Chaturvedi et al., 2019).

2.2 Dynamizer Features and Examples

The Dynamizer ADE for CityGML 2.0 and the Dynamizer module in CityGML 3.0 allow one to add dynamic data to a particular attribute in the model by specifying timeseries data to override its static value. This integration can be done in a number of ways, using different features (Kolbe et al., 2021). A Dynamizer object consists of two main components.

The first is the Dynamizer object itself. This portion describes the details of the attribute or element that is being made dynamic by replacing its value. It contains an attributeRef, which is the reference to the attribute or element. It is defined using XML Path Language (XPath), which is a path notation used to navigate through the hierarchy of an XML file. Other attributes that are defined include the startTime and endTime which specify the period of time in which the attribute or element is dynamic.

The second component is the SensorConnection or TimeSeries object. These define the data that overrides the static value specified in attributeRef. The SensorConnection object is used to link an external sensor or IoT service such as OGC SensorThings API or Sensor Observation Service. The various formats of Timeseries objects are used for specifying existing timeseries data.

2.2.1 SensorConnection: SensorConnection describes details used to retrieve and process dynamic data from an external sensor. This includes links to different sources such as the baseURL and linkToObservation, as well as attributes such as the observationProperty, uom (units of measurement) and datastreamID.

2.2.2 Timeseries: The Dynamizer module has three types of Timeseries objects for defining existing timeseries data from different types of sources, as well as a CompositeTimeseries object for specifying repeating patterns. The first type of Timeseries object is GenericTimeseries. This represents dynamic data as in-line time-value-pairs, directly in the model itself. The second is using TabulatedTimeseries. This inserts the dynamic data by linking to external tabulated files like CSV or Excel files. The respective columns for timestamps and values are specified in the TabulatedTimeseries object, and each row in the file represents one time-value-pair. The third is using StandardTimeseries, which inserts the dynamic data through external files following the OGC TimeseriesML standard or the OGC Observations and Measurements standards. Finally, CompositeTimeseries allows one to specify repeating patterns, such as hourly, weekly or monthly patterns, using a combination of the other features.

3. Integration of Dynamic Data Information with CityGML

Including studies involving CityGML and the Dynamizer, there have been many papers describing different ways that the implementation of integrating CityGML with external data can be handled. We summarise them into three main categories.

3.1 Using CityGML and Dynamizer – Integrate Data into CityGML

Throughout their research and development of the Dynamizer as part of the OGC Future City Pilot (Chaturvedi and Kolbe, 2017), Chaturvedi, Kolbe and their colleagues have demonstrated its use in a number of use cases. This includes the integration of real-time sensor observations in a 3D city model of Greenwich, London (Chaturvedi and Kolbe, 2016), and time-dependent solar irradiation analysis results in a 3D city model of Rennes, France (Chaturvedi et al., 2017b) and HVAC and building energy analysis (Chaturvedi et al., 2019).

To support the use of the Dynamizer, they have also investigated the technologies that are required to store and retrieve such data. They developed an extension for the commonly used CityGML database, 3DCityDB, and a method to import, retrieve and visualise Dynamizer ADE data in CityGML files (Chaturvedi and Kolbe, 2016). Their work involved creating additional tables and structures to manage the time-series data separately. In addition, they also worked on the Mini Sensor Observation Service to support the reading, querying and visualization of Dynamizer timeseries data without requiring a database, using relational adapters to handle the retrieval of data from external files, cloud-based servers and external databases (Chaturvedi et al., 2017b).

Other researchers, such as Chatzinikolaou et al. (2020) also make use of the Dynamizer and 3DCityDB. They also configured 3DCityDB to support the Dynamizer ADE, but instead of using it to import CityGML containing the dynamic data, they made use of a python script to populate the 3DCityDB tables with their time-series data.

3.2 Using CityGML and an External API or Service – Integrate CityGML into Data

There are also studies that do not make use of the Dynamizer ADE. Most make use of one database or application to manage the 3D city models and another separate one to manage the dynamic data.

For example, Santhanavanich and Coors (2021) proposed CityThings. They find that it is difficult to have to represent sensor data in the form of a Dynamizer object and to embed it directly in the CityGML, especially when the people managing and collecting each set of data are different. They suggest instead to link the data externally, by using the CityGML objects' gml-id to identify the respective objects and adding that as an attribute to the sensor data's representation in OGC SensorThings API. Through the SensorThings API, the requesting of the relevant information can be done through a query using the gml-id.

In the OGC 3D IoT Platform for Smart Cities Pilot, that the development of CityThings was part of, multiple implementations to link CityGML with air quality measurements and indoor occupancy readings were carried out by multiple parties (Coors, 2020). These include the STT GeoPortal, Taiwan Civil IoT Taiwan Data Service Platform, and Cyient 3D GeoPortal, among others. These all make use of various implementations of the SensorThings API to manage the sensor data, along with the 3D city model in glTF or 3D tiles format. In this pilot, they also made use of IndoorGML models and created an augmented reality visualisation platform using Unity3D, the Skymantics 3D GeoPortal.

3.3 Using a Separate Database or Model – Integrate Both CityGML and Data into One Model

There are also studies that aim to integrate the two (and other formats) regardless of standards used. These focus on not having to worry about information not matching up to a specified format, standard or organisational structure. This includes the use of a managed object method (implemented in Java) to represent features in CityGML and other data as different objects according to different sets of rules and standards (Wen et al., 2010) and the use of a NoSQL database like MongoDB (document database) which can store CityGML models and other data as separate objects (Mao et al., 2014).

3.4 Movement Data in CityGML

In many studies, the Dynamizer was mainly used to represent dynamic data that is numerical, or in the form of an attribute. In this case when visualising the model, the dynamic data can be retrieved and viewed as an updating property, or in the form of tables or graphs. Although the Dynamizer is also able to represent geometric information, with the option to specify the type of value being replaced as 'geometry', there have not been many examples of such use. According to the requirement analysis by Chaturvedi and Kolbe (2019), such geometric information covers two aspects, changing objects as well as moving objects, the latter which is of interest here.

4. Movement Data

4.1 Trajectories

Movement data is primarily collected in the form of trajectories. These are most commonly represented as a collection of points at which a moving entity was located during their movement, and modelled using Spatiotemporal Data Models. This borrows existing standards from GIS for 2D spatial data and temporal data. For example, with animal tracking, the position of the tracked entity is captured at a certain time interval. This results in a data set with a set of captured values (its position and other properties like temperature or speed) linked to timestamp values. While these time intervals tend to be regular, it is possible that gaps in readings or slight variances may occur due to technical errors or inaccuracies.

4.2 Origin-Destination

When looking at movement data in an urban context, it is also common to find some data collection methods such as public transport ticketing that only capture data about the entry and exit of a passenger. These result in a form of data called Origin-Destination (OD). OD studies aggregate movement data to create datasets of properties (such as the number of trips, or average speed) about movement from one specific point or area to another. The aggregation could group data from a certain time period, like a month, and be categorized by different patterns or properties, such as day of the week or time bands across the day. As such, not all OD datasets may be able to be associated with timestamps in the same way one would expect of data from trajectory datasets.

5. Study

In light of the possibilities of existing methods and the lack of examples of such use, we investigate how different methods work for combining CityGML models and movement data of different types. We have focused on the first two methods, using CityGML and Dynamizer, and using CityGML and an external API or service, so that we may take advantage of existing implementations and standards. For each method, we investigate the differences and issues faced at two points of the process: first, the conversion and processing phase of turning the available datasets into their target formats and, second, the usage and visualization phase of accessing the data in the resulting formats.

5.1 Datasets

We have selected a range of city models, in the CityGML and CityJSON formats, as well as movement data of different types, including animal tracking trajectories, human movement through spaces, and traffic data. All movement data considered constitutes historical data, no live tracking was considered for this study. Seven city models from different sources and six datasets for movement data were used. To see how different combinations of datasets work with the two implementations, the city models and movement datasets have been combined into six case studies, as shown in table 1.

	CityGML 1.0	CityGML 2.0	CityGML 3.0	CityJSON
Trajectory (Animal)	Brussels & Gulls	Berlin and Potsdam & White Storks		New York and Philadelphia & Osprey
Trajectory (Human)			University Supermarket & Customers	
Origin-Destination		University Campus & Bus	London & London Tube	

Table 1. Combinations of city models and datasets used as case studies in this study.

5.1.1 CityGML Models: The CityGML models we used include:

1. CityGML 1.0 model of Brussels (Brussels Regional Informatics Centre, 2016).
2. CityGML 3.0 model of Student Supermarket in Tsinghua University, Beijing, China. Model created from floorplans used in study of customer behaviour (Yang et al., 2019).
3. CityGML 2.0 model of Berlin (Berlin Business Location Center, 2015).
4. CityGML 1.0 model of Potsdam (Bereich Vermessung, Geodateninfrastruktur, 2012).
5. CityGML 2.0 model of a university campus in Singapore.
6. CityJSON model of part of London, generated using 3dfier (Ledoux et al., 2021) and GeoJSON data tiles of building footprints from OSM Buildings.
7. CityJSON models of New York and Philadelphia (BuildZero.Org, 2019).

A summary of the city models used in the study, including their location, city model type and level of detail is shown in Table 2. Of these models, the six LOD1 and LOD2 models here focus on the exterior of the buildings, while the LOD3 model of the University Supermarket is a model of the interior of the building.

Country	Area	City model type	LOD	Total size (kB)
Belgium	Brussels	CityGML 1.0	2	3975763
China	University Supermarket	CityGML 3.0	3	5238
Germany	Berlin	CityGML 2.0	2	39767944
Germany	Potsdam	CityGML 1.0	2	1454187
Singapore	University Campus	CityGML 2.0	1	2151
UK	London	CityJSON	1	58187
USA	New York & Philadelphia	CityJSON	1	553275

Table 2. Details of the city models used in this study.

5.1.2 Trajectories: The datasets of the trajectories type we used include:

1. Lesser black-backed gulls tagged along the southern North Sea Coast (data collected by the LifeWatch GPS tracking network for large birds and published by the Research Institute for Nature and Forest (INBO)) (Stienen et al., 2021).
2. White Storks from 3 populations in Beuster, Dromling and Loburg (Carlson et al., 2021b), retrieved from Movebank (Carlson et al., 2021a).
3. Fall migration routes of Osprey in USA, from the areas of the lower Columbian river, north-central Minnesota, Shelter Island, New York and southern New Jersey (Martell et al., 2001), retrieved from Movebank (Martell and Douglas, 2019).
4. Customer trajectories collected in a student supermarket in Tsinghua University (Yang et al., 2019).

5.1.3 Origin-Destination: The datasets of the OD type we used include:

1. Origin-Destination data between bus stops in Singapore (Land Transport Authority, 2021).
2. Origin-Destination data between London Underground Stations (Transport for London, 2021; as part of the NUMBAT dataset).

A summary of the movement datasets used in the study, including their location and data type is shown in Table 3. Of these models, three are animal tracking trajectory datasets, one tracks the trajectories of human movement indoors and two contain Origin-Destination data from public transportation.

5.1.4 Differences Between Scale and Frequency: The movement data we used range in scale and time frequency. The data collected by Tsinghua University covers only a small area of less than 50 m by 50 m, and across a period of 3 days, for less than an hour each. But the positions are collected at a high frequency of once every 0.02 seconds. Meanwhile, the animal tracking datasets cover a much larger distance, in this case, across continents, between North and South America, and between Europe and Africa. However, the positions are collected much less frequently, such as once every half hour, or sporadically, for much longer periods of time of up to 2-5 years.

5.2 Method 1

The first implementation makes use of embedding the dynamic data into CityGML using Dynamizer.

Country	Dataset	Data type	Total size (KB)	Sample trajectory or OD size (KB)	Sample size as CityGML + CSV (KB)
Belgium	GPS tracking of Lesser Black-backed Gulls and Herring Gulls breeding at the southern North Sea coast	Trajectory	613441	557 - 49594	108 - 9603
China	Customer movement in University Supermarket	Trajectory	299325	55 - 2766	404 - 2252
Germany	HUJ MPIAB White Stork E-Obs	Trajectory	550054	4234 - 20592	1501 – 7302
Singapore	Passenger volume by origin-destination bus stops	Origin-Destination	195688	-	14 - 786
UK	TfL Rolling Origin and Destination Survey	Origin-Destination	19522	-	2729 - 3559
USA	Osprey in North and South America 1995-2002 (Martell)	Trajectory	21989	19 - 692	8 - 9

Table 3. Details of the movement datasets used in this study.

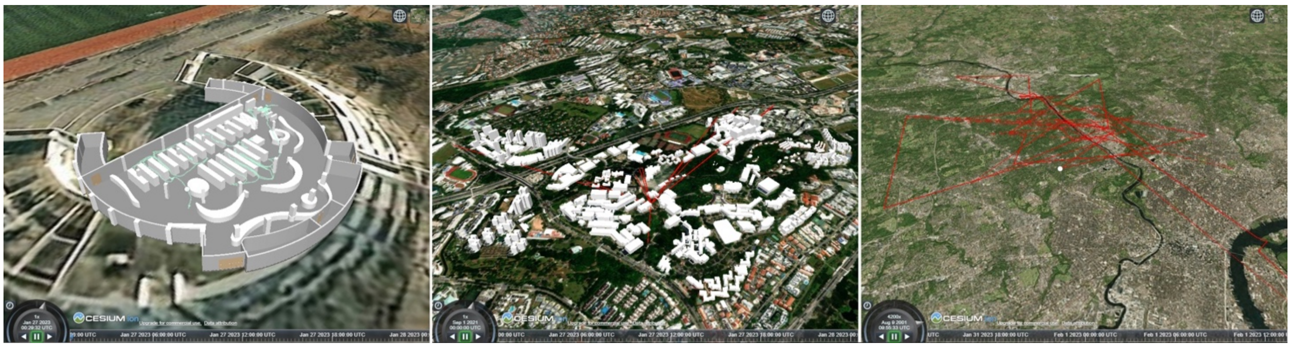

Figure 1. University supermarket and customer trajectory (left), University Campus and Bus OD (center) and New York and Philadelphia and Osprey trajectory (right) viewed in Angular Cesium.

5.2.1 Conversion and Processing: Citygml4j, an open source Java API for reading, processing and writing CityGML, was used to convert the movement data into CityGML models, with the dynamic data embedded using the Dynamizer module (for CityGML 3.0) or Dynamizer ADE (for CityGML 2.0) and CSV files. Here, we have created new CityGML files for each dataset but the data can also be added into existing files. With the range of datasets used, we were able to test out the usage of different Dynamizer features, such as TabularTimeseries for trajectories and CompositeTimeseries with GenericTimeseries for OD data. The evaluation follows in Section 6. Note that for trajectory data, the data was not included directly into the CityGML, but was linked as a CSV file. For O-D data, the data was explicitly integrated into the CityGML file (in order to make use of the repeating feature of CompositeTimeseries).

5.2.2 Usage and Visualization: A viewer to view the resulting CityGML files was implemented using Angular Cesium, an extension of the CesiumJS API (Figure 1). CesiumJS is an open source JavaScript library commonly used to create web platforms for the visualisation of city models, and Angular Cesium builds upon it with RxJS, which makes it easier to update groups of entities. The CityGML files are read and city objects are created as Cesium entities for visualization. To view objects in motion, Cesium's built-in clock was used. If Dynamizer objects are present, the timestamps are compared against the Cesium clock to determine what geometry or attributes to display, updating the Cesium entities accordingly based on their gml id.

5.3 Method 2

The second implementation stores the data separately from CityGML, in the SensorThings API.

5.3.1 Conversion and Processing: For this method, the movement data was first processed and imported into an implementation of SensorThings API. FROST-Server was selected as it is a complete and well-maintained open-source implementation.

5.3.2 Usage and Visualization: The city models were imported into 3DCityDB, an open source CityGML database, and exported as tiled KML files for viewing. The data is viewed on the Web Map Client that comes with 3DCityDB. It was extended to be able to retrieve and process the JSON data from the SensorThings API.

5.4 Case Studies and Compatibility with Methods

Due to a lack of support, some datasets are not compatible with some methods. A summary of the compatibility and the reasons for incompatibility are shown in Table 4.

	Implementation 1	Implementation 2
Brussels & Gulls	No Dynamizer in CityGML 1.0	Yes
University Supermarket & Customers	Yes	No CityGML 3.0 in 3DCityDB
Berlin and Potsdam & White Storks	Yes	Yes
University Campus & Bus	Yes (OD in CityGML 3.0)	Yes
London & London Tube	Yes (OD in CityGML 3.0)	Yes
New York and Philadelphia & Osprey	No Dynamizer in CityJSON	Yes

Table 4. Combinations of case study datasets and their compatibility with each implementation.

6. Evaluation of the Two Methods

6.1 Compatibility with Movement Data

Although possible, neither method seems to be designed to store information for moving entities. The process is not as intuitive as it would be for numerical or attribute data.

6.1.1 Movement in FROST-Server: While it is possible to simply store locations as a string in an Observation, semantically, SensorThings requires the location of Things to be defined as a fixed Location property at the time of creation. This Location property can be updated, and previous locations can be stored as HistorialLocation properties. However, by storing it in this way, the location of the object is not treated as an Observation, and is not as intuitive to retrieve, nor is it possible to assign any additional properties. One workaround method that the developers recommend, is to instead link each Observation to a FeatureOfInterest entity which can be assigned its own location (OGC SensorThings API, 2017). In this way, the position of the sensor can be retrieved from each Observation alongside any other properties it may have. But this results in a large amount of additional entities and an extra step in the data retrieval process.

6.1.2 Movement in CityGML Dynamizer: While the Dynamizer does have geometry as an option for the value it is replacing, it can only replace one value from one input source. This may work fine for updating a property value such as a temperature reading, but if it is used to update the location of an object, the entire poslist of the object has to be replaced. This is additional processing work as location is not collected and stored in such a manner. Even for a point, if the location is stored as seperate latitude and longtitude or x, y and z values, they will need to be combined first before they can be linked via the Dynamizer. If the entity is more complex, the data that needs to be replaced becomes significantly larger and more work to process as well.

6.2 Retrieval Flexibility

FROST-Server supports the use of request parameters that, when added to the url used to request data from the server, can provide additional processing of the return data. These include sorting, filtering and specifying how many entities to return. If requesting data from FROST-Server separately, this allows for a lot more flexibility in accessing the data.

With CityGML Dynamizer, the data is more or less fixed within the CityGML file. Even for external links, the url is specified directly in the CityGML file. Therefore, although a url to FROST-Server can be used in the description of a Dynamizer object, it is fixed. One could alter this url before retrieving any data but it would not be intuitive as this data is designed to be retrieved in the background. Any changes would have to be reflected in the CityGML file itself or upon the actual data being linked. Otherwise, any sort of processing to the data would have to be implemented and performed separately upon the retrieved data afterwards .

6.3 Input Flexibility

SensorThings and the FROST-Server implementation were designed for streamed real- time sensor data. In order to POST a new observation to the server, a timestamp is required, otherwise it is automatically set to the time at the time of sending the request. In this case, the movement data in the form of timestamped positions would be easy to input. Meanwhile, aggregated data such as Origin-Destination data would not be as intuitive to input. A user-defined timestamp would have to be assigned based on the time period the data is from.

On the other hand, CityGML Dynamizer supports the input of data in a number of different ways. One such way is CompositeTimeseries. It allows one to define recurring patterns of data, such as repeating daily or weekly data patterns. This allows for more flexibility in storing aggregated data. The downside of this is that the data is more difficult to use due to a lack of support. For example, it would be necessary to reconstruct timestamps from the patterns found in CompositeTimeseries in order to use them for visualization or analysis. A summary of the evaluation of the two methods is shown in Table 5.

	CityGML Dynamizer	**FROST-Server**
Compatibility with movement data	Requires additional processing into right format for geometry	Requires additional entities and additional retrieval step
Retrieval flexibility	Data will be retrieved as defined in CityGML file / data source	Supports processing of data before retrieval by adding parameters to url
Input flexibility	Supports recurring data patterns, allowing some storage of aggregated data	Requires timestamp for each observation, not suitable for aggregated data

Table 5. Comparisons between the use of CityGML Dynamizer and FROST-Server.

7. Discussion and Conclusion

7.1 Conclusion

Dynamic data is becoming more relevant in urban digital twins, and its support has been growing thanks to the development of the infrastructure such as the Dynamizer module in CityGML. However, the integration of movement data in 3D city models has not been been subject of research so far. This paper explored the process of integrating movement data with CityGML in two different ways. One using the CityGML Dynamizer module to integrate the movement data directly into CityGML and the other using FROST-Server to provide the movement data alongside the CityGML model.

Overall, the Dynamizer provides a more flexible way of representing the movement data, allowing the definition of repeating patterns and aggregated data. However, it is more static and restrictive when it comes to accessing and utilising the data. It also has less software support.

FROST-Server, on the other hand, has a more restrictive structure for representing data that does not support moving locations and aggregated data well. However, accessing the data is more flexible, with the ability to request for data with filtering or sorting. In this case, there is support for the data and models separately, but support for integration is not as accessible.

7.2 Support and Integration

As noted in the report for the 3D IoT Platform for Smart Cities Pilot (Coors, 2020), existing data models and APIs to support 3D city models and dynamic data do not integrate well. This is especially so for movement data. More commonly used software and methods of handling city models do not support dynamic values. While Dynamizer allows the inclusion of dynamic data, support for it is limited.

On the other hand, for sensor data, commonly used APIs like SensorThings API that handle streams of dynamic data do not support any form of feature representation or geometry. While it has ways to specify geometric locations, these do not work well with constantly changing locations.

7.3 Visualization and Utilization

With almost all the methods studied in literature, CityGML is visualised after conversion into other formats rather than using its geometric data directly.

When it comes to large datasets, conversion into a tiled format like 3D Tiles or a tiled collection of KML, COLLADA or glTF files (through 3DCityDB) allows for more efficient loading of the model in smaller batches as needed. When converting into such formats, most ways combine all the geometry at a CityObject level, such as the Building level, with all the separation below that level removed.

This is the same for properties and attributes found in CityGML. Most implementations have all the information extracted into a separate tabular dataset or retrieve the information to be viewed from a completely external datasource. In such implementations, the information is also often exported only from the Building level, or completely absent from the city model to begin with.

7.4 Future Work

One way that CityGML is being developed to make it more user-friendly is by expanding on its encoding formats, such as CityJSON for more compact and developer-friendly files. They have also been exploring other encoding formats such as CitySQL for database storage and management and glTF for ease of visualisation with existing software (Open Geospatial Consortium CityGML Standards Working Group, 2022). At the same time, SensorThings API is also working towards their 2.0 version, including updates that could help with the retrieval and visualisation of moving objects as a trajectory or collection of points. With these new possibilities, utilising movement data and CityGML in conjunction may be more accessible. It would be interesting to see how they integrate moving forward.

Acknowledgements

This research was supported by the Ng Teng Fong Charitable Foundation research funding (WBS A-0008245-01-00). Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of the Ng Teng Fong Charitable Foundation.

The authors kindly acknowledge Huang Weixin, School of Architecture, Tsinghua University, for sharing the trajectory records from the university supermarket.

References

Agalianos, K., Ponis, S., Aretoulaki, E., Plakas, G., Efthymiou, O. 2020. Discrete event simulation and digital twins: review and challenges for logistics. *Procedia Manufacturing*, 51, 1636-1641. doi.org/10.1016/j.promfg.2020.10.228.

Bereich Vermessung, Geodateninfrastruktur. 2012. 3D-gebäudemodell Potsdam 2012. potsdam.opendatasoft.com/explore/dataset/3d-gebaudemodell-lod2-citygml/information (27 January 2024).

Berlin Business Location Center. 2015. Berlin 3D – download portal. www.businesslocationcenter.de/en/economic-atlas/download-portal (27 January 2024).

Biljecki, F., Kumar, K., Nagel, C. 2018. CityGML application domain extension (ADE): overview of developments. *Open Geospatial Data, Software and Standards*, 3(13). doi.org/10.1186/s40965-018-0055-6.

Biljecki, F., Lim, J., Crawford, J., Moraru, D., Tauscher, H., Konde, A., Adouane, K., Lawrence, S., Janssen, P., Stouffs, R. 2021. Extending CityGML for IFC-sourced 3D city models, *Automation in Construction* 121, 103440. doi.org/10.1016/j.autcon.2020.103440

Bouzguenda, I., Alalouch, C., Fava, N. 2019. Towards smart sustainable cities: a review of the role digital citizen participation could play in advancing social sustainability. *Sustainable Cities and Society*, 50, 101627. doi.org/10.1016/j.scs.2019.101627.

Brussels Regional Informatics Centre. 2016. datastore.brussels - urbis download. datastore.brussels/web/urbis-download (27 January 2024).

BuildZero.Org. 2019. Open city model. github.com/opencitymodel/opencitymodel (27 January 2024).

Carlson, B.S., Rotics, S., Nathan, R., Wikelski, M., Jetz, W. 2021a. Data from: individual environmental niches in mobile organisms. Movebank Data Repository. doi.org/10.5441/001/1.rj21g1p1/2.

Carlson, B.S., Rotics, S., Nathan, R., Wikelski, M., Jetz, W. 2021b. Individual environmental niches in mobile organisms. *Nature Communications* 12(1). doi.org/10.1038/s41467-021-24826-x.

Chaturvedi, K., Kolbe, T.H. 2016. Integrating dynamic data and sensors with semantic 3d city models in the context of smart cities. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-2/W1, 31-38. doi.org/10.5194/isprs-annals-IV-2-W1-31-2016.

Chaturvedi, K., Kolbe, T.H. 2017. Future city pilot 1 engineering report. www.opengis.net/doc/PER/FCP1-ER (27 January 2024).

Chaturvedi, K., Kolbe, T.H. 2019. A requirement analysis on extending semantic 3d city models for supporting time-dependent properties. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* IV-4/W9, 19-26. doi.org/10.5194/isprs-annals-IV-4-W9-19-2019.

Chaturvedi, K., Smyth, C.S., Gesquière. G., Kutzner, T., Kolbe, T.H. 2017a. Managing versions and history within semantic 3D city models for the next generation of CityGML. Abdul-Rahman, A. (ed.) *Lecture Notes in Geoinformation and Cartography*, Advances in 3D Geoinformation, 191-206. Springer, Cham. doi.org/10.1007/978-3-319-25691-7\ 11.

Chaturvedi, K., Willenborg, B., Sindram, M., Kolbe, T.H. 2017b. Solar potential analysis and integration of the time-dependent simulation results for semantic 3D city models using dynamizers. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 44W5, 25–32. doi.org/10.5194/isprs-annals-IV-4-W5-25-2017.

Chaturvedi, K., Yao, Z., Kolbe, T.H. 2019. Integrated management and visualization of static and dynamic properties of semantic 3D city models. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLII-4/W17, 7–14. doi.org/10.5194/isprs-archives-XLII-4-W17-7-2019.

Chatzinikolaou, E., Pispidikis, I., Dimopoulou, E. 2020. A semantically enriched and web-based 3D energy model visualization and retrieval for smart building implementation using CityGML and Dynamizer ADE. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, VI-4/W1-2020, 53–60. doi.org/10.5194/isprs-annals-VI-4-W1-2020-53-2020.

Coors, V. 2020 OGC 3D-IoT platform for smart cities engineering report. www.opengis.net/doc/PER/3D-IoT-Platform (27 January 2024).

Dembski, F., Wössner, U., Letzgus, M., Ruddat, M., Yamu, C. 2020. Urban digital twins for smart cities and citizens: the case study of Herrenberg, Germany. *Sustainability*, 12(6). doi.org/10.3390/su12062307.

Dodge, S., Weibel, R., Ahearn, S.C., Buchin, M., Miller, J.A. 2016. Analysis of movement data. *International Journal of Geographical Information Science*, 30(5), 825-834. doi.org/10.1080/13658816.2015.1132424.

Duan, W., Nasiri, R., Karamizadeh, S. 2020. Smart city concepts and dimensions. *Proceedings of the 2019 7th International Conference on Information Technology: IoT and Smart City*, 488-492. Association for Computing Machinery, New York. doi.org/10.1145/3377170.3377189.

El Saddik, A. 2018. Digital twins: the convergence of multimedia technologies. *IEEE MultiMedia*, 25(2), 87-92. doi.org/10.1109/MMUL.2018.023121167.

Glaessgen, E.H., Stargel, D.S. 2012. The digital twin paradigm for future NASA and U.S. Air Force vehicles. *Structural Dynamics and Materials Conference - Special Session on the Digital Twin*, volume 53. American Institute of Aeronautics and Astronautics.

Goldin, M. 2022. Los Angeles pilots digital twin project to aid building decarbonization. *Smart Cities Dive*. www.proquest.com/trade-journals/los-angeles-pilots-digital-twin-project-aid/docview/2675091961/se-2 (27 January 2024).

Grieves, M. 2014. Digital twin: manufacturing excellence through virtual factory replication. www.researchgate.net/publication/275211047_Digital_Twin_Manufacturing_Excellence_through_Virtual_Factory_Replication (27 January 2024).

Gröger, G., Plümer, L. 2012. CityGML: interoperable semantic 3D city models. *ISPRS Journal of Photogrammetry and Remote Sensing*, 71, 12-33. doi.org/10.1016/j.isprsjprs.2012.04.004.

Guo, J., Lv, Z. 2022. Application of digital twins in multiple fields. *Multimedia Tools and Applications*, 81(19), 26941-26967. doi.org/10.1007/s11042-022-12536-5.

Jwo, J.S., Lee, C.H., Lin, C.S. 2022. Data twin-driven cyber-physical factory for smart manufacturing. *Sensors*, 22(8). doi.org/10.3390/s22082821.

Kaiblinger, A., Woschank, M. 2022. State of the art and future directions of digital twins for production logistics: a systematic literature review. *Applied Sciences*, 12(2). doi.org/10.3390/app12020669.

Karakra, A., Fontanili, F., Lamine, E., Lamothe, J., Taweel, A. 2018. Pervasive computing integrated discrete event simulation for a hospital digital twin. *2018 IEEE/ACS 15th International Conference on Computer Systems and Applications (AICCSA)*, 1-6. doi.org/10.1109/AICCSA.2018.8612796.

Kobie, N. 2022. Mapping an entire country: meet Singapore's digital twin. *PC Pro*, 126-128. www.proquest.com/magazines/mapping-entire-country-meet-singapores-digital/docview/2708429196/se-2 (27 January 2024).

Kolbe, T.H., Kutzner, T., Smyth, C.S., Nagel, C., Roensdorf, C. Heazel, C. 2021. OGC City Geography Markup Language (CityGML) Part 1: Conceptual Model Standard. Open Geospatial Consortium. www.opengis.net/doc/IS/CityGML-1/3.0.

Kutzner, T., Kolbe, T.H. 2018. CityGML 3.0: sneak preview. Kersten, T.P., Gülch, E., Schiewe, J., Kolbe, T.H., Stilla U (eds.) *Photogrammetrie, Fernerkundung, Geoinformatik, Kartographie, 2018 Jahrestagung. Publikationen der Deutschen Gesellschaft für Photogrammetrie, Fernerkundung und Geoinformation (DGPF) e.V.*, volume 27, 835–839.

Land Transport Authority. 2021. LTA - dynamic datasets. datamall.lta.gov.sg/content/datamall/en/dynamic-data.html.

Ledoux, H., Biljecki, F., Dukai, B., Kumar, K., Peters, R., Stoter, J., Commandeur, T. 2021. 3dfier: automatic reconstruction of 3D city models. *Journal of Open Source Software*, 6(57), 2866. doi.org/10.21105/joss.02866.

Lei, B., Janssen, P., Stoter, J., Biljecki, F. 2023. Challenges of urban digital twins: a systematic review and a Delphi expert survey. *Automation in Construction*, 147, 104716. doi.org/10.1016/j.autcon.2022.104716.

Li, W., Zlatanova, S., Diakite, A.A., Aleksandrov, M., Yan, J. 2020. Towards integrating heterogeneous data: a spatial DBMS solution from a CRC-LCL project in Australia. *ISPRS International Journal of Geo-Information*, 9(2). doi.org/10.3390/ijgi9020063.

Lu, Q., Parlikad, A.K., Woodall, P., Ranasinghe, G.D., Xie, X., Liang, Z., Konstantinou, E., Heaton, J., Schooling, J. 2020. Developing a digital twin at building and city levels: case study of West Cambridge campus. *Journal of Management in Engineering*, 36(3), 05020004. doi.org/10.1061/(ASCE)ME.1943-5479.0000763.

Mao, B., Harrie, L., Cao, J., Wu, Z. Shen, J. 2014 NoSQL based 3D city model management system. *ISPRS International*

Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 169-173.

Martell, M., Douglas, D. 2019. Data from: fall migration routes, timing, and wintering sites of North American ospreys as determined by satellite telemetry. doi.org/10.5441/001/1.sv6335t3/1.

Martell, M.S., Henny, C.J., Nye, P.E., Solensky, M.J. 2001. Fall migration routes, timing, and wintering sites of North American ospreys as determined by satellite telemetry. *The Condor*, 103(4), 715-724. doi.org/10.1093/condor/103.4.715.

Milner, G. 2021. South Korean city uses a digital twin to meet challenges. www.esri.com/about/newsroom/arcuser/south-korean-city-uses-a-digital-twin-to-meet-challenges/.

OGC SensorThings API. 2017. Additional moving object issue. github.com/opengeospatial/sensorthings/issues/33.

Open Geospatial Consortium CityGML Standards Working Group. 2022. CityGML 3.0 encodings. github.com/opengeospatial/CityGML-3.0Encodings.

Parris, C.J., Laflen, J.B., Grabb, M.L., Kalitan, D.M. 2016. The future for industrial services: the digital twin. *Infosys Insights.* www.infosys.com/insights/iot/future-industrial-digital-twin.html.

Plautz, J. 2022. Las Vegas unveils digital twin at CES as part of sustainability push. *Smart Cities Dive*. www.proquest.com/trade-journals/las-vegas-unveils-digital-twin-at-ces-as-part/docview/2620906955/se-2 (24 January 2024).

Purves, R.S., Laube, P., Buchin, M., Speckmann, B. 2014. Moving beyond the point: an agenda for research in movement analysis with real data. *Computers, Environment and Urban Systems*, 47, 1-4. doi.org/10.1016/j.compenvurbsys.2014.06.003.

Qi, Q., Tao, F., Zuo, Y., Zhao, D. 2018. Digital twin service towards smart manufacturing. *Procedia CIRP*, 72, 237-242. doi.org/10.1016/j.procir.2018.03.103.

Ramu, S.P., Boopalan, P., Pham, Q.V., Maddikunta, P.K.R., Huynh-The, T., Alazab, M., Nguyen, T.T., Gadekallu, T.R. 2022. Federated learning enabled digital twins for smart cities: concepts, recent advances, and future directions. *Sustainable Cities and Society*, 79, 103663. doi.org/10.1016/j.scs.2021.103663.

Reifsnider, K., Majumdar, P. 2013. Multiphysics stimulated simulation digital twin methods for fleet management. *54th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*. American Institute of Aeronautics and Astronautics. doi.org/10.2514/6.2013-1578.

Rivera, L.F., Müller, H.A., Villegas, N.M., Tamura, G., Jiménez, M. 2020. On the engineering of iot-intensive digital twin software systems. *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*, 631-638. Association for Computing Machinery. New York. doi.org/10.1145/3387940.3392195.

Ruohomäki, T., Airaksinen, E., Huuska, P., Kesäniemi, O., Martikka, M., Suomisto, J. 2018. Smart city platform enabling digital twin. *2018 International Conference on Intelligent Systems (IS)*, 155-161. doi.org/10.1109/IS.2018.8710517.

Santhanavanich, T., Coors, V. 2021. CityThings: An integration of the dynamic sensor data to the 3D city model. *Environment and Planning B: Urban Analytics and City Science*, 48(3), 417-432. doi.org/10.1177/2399808320983000.

Siła-Nowicka, K., Vandrol, J., Oshan, T., Long, J.A., Demšar, U., Fotheringham, A.S. 2016. Analysis of human mobility patterns from GPS trajectories and contextual information. *International Journal of Geographical Information Science*, 30(5), 881-906. doi.org/10.1080/13658816.2015.1100731.

Stienen, E.W., Desmet, P., Milotic, T., Hernandez, F., Deneudt, K., Bouten, W., Müller, W., Matheve, H., Lens, L. 2021. LBBG ZEEBRUGGE - lesser black-backed gulls (Larus fuscus, Laridae) breeding at the southern North Sea coast (Belgium and the Netherlands). doi.org/10.5281/zenodo.5068540.

Svítek, M., Skobelev, P., Kozhevnikov, S. 2020. Smart city 5.0 as an urban ecosystem of smart services. Borangiu, T., Trentesaux, D., Leitão, P., Giret Boggino, A., Botti, V. (eds.) *Service Oriented, Holonic and Multi-agent Manufacturing Systems for Industry of the Future*, 426-438. Springer, Cham.

Tao, F., Zhang, H., Liu, A., Nee, A.Y.C. 2019. Digital twin in industry: state-of-the-art. *IEEE Transactions on Industrial Informatics*, 15(4), 2405-2415. doi.org/10.1109/TII.2018.2873186.

Transport for London. 2021. Our open data - Transport for London. tfl.gov.uk/info-for/open-data-users/our-open-data (27 January 2024).

Wen, W., Kjems, E., Bodum, L., Kolar, J. 2010. Dynamic features in a 3D city model as an energy system. Kolbe, T.H., König, G., Nagel C. (eds.) *ISPRS Conference: International Conference on 3D Geoinformation*, volume. XXXVIII-4 W15, 73-78. International Society for Photogrammetry and Remote Sensing.

Yang, L., Cheng, B., Deng, N., Zhou, Z., Huang, W. 2019. The influence of supermarket spatial layout on shopping behavior and product sales - an application of the ultra-wideband indoor positioning system. *Intelligent & Informed*, volume 1, 301-310. CAADRIA, Hong Kong. doi.org/10.52842/conf.caadria.2019.1.301.

Yu, G., Wang, Y., Hu, M., Shi, L., Mao, Z., Sugumaran, V. 2021. Rioms: an intelligent system for operation and maintenance of urban roads using spatio-temporal data in smart cities. *Future Generation Computer Systems*, 115, 583-609. doi.org/10.1016/j.future.2020.09. 010.

Yun, Y., Lee, M. 2019 Smart city 4.0 from the perspective of open innovation. *Journal of Open Innovation: Technology, Market, and Complexity*, 5(4). doi.org/10.3390/joitmc5040092.