

Augmented Reality visualisation on site: BIM semantics and communication

Linked Data

Master's dissertation submitted in order to obtain the academic degree of

Master of Science in de ingenieurswetenschappen: architectuur

Supervisor: Prof. ir.-arch. Paulus Present

Counselors: Ir.-arch. Jeroen Werbrouck
Prof. dr. ir. arch. Ruben Verstraeten

Philippe Soubrier 01702837 philippe.soubrier@ugent.be
Academic year: 2022–2023

Contents

1	Introduction	6
1.1	Context	6
1.1.1	3D viewers	6
1.1.2	BIM geometry	6
1.1.3	LDBIM	6
1.1.4	Computing power dilemma	6
1.2	Research questions	7
1.2.1	Can LDBIM be culled?	7
1.2.2	Can existing semantic be used?	7
1.3	Research objectives	7
1.3.1	Advantages of LDBIM	7
1.3.2	Showcase the fisability	7
2	State of the art	26
2.1	AR on-site approaches	27
2.1.1	Autodesk xyz - Atom	27
2.1.2	Openspace AI	27
2.2	Linked geometric data	27
2.2.1	BOT	27
2.2.2	ifcOWL	27
2.2.3	Other ontologies	27
2.3	LDBIM viewers	27
2.3.1	ld-bim.web.app	27
2.4	Rendering optimizations	27
2.4.1	Culling algorithms	27
2.4.2	Qonic	27
3	Methodology	28
3.1	RDF data	28
3.1.1	Existing data	28
3.1.2	Adding data	28
3.2	LOD streaming model	28

3.2.1	Culling	28
3.2.2	Redundancy avoidance with BCF history	28
3.3	Querying	28
3.3.1	Front-end	28
3.3.2	Back-end	28
3.4	Rendering	28
3.4.1	Three.js	28
3.4.2	Shaders	28
4	Website	29
4.1	Setup	29
4.1.1	Software	29
4.1.2	Hardware	29
4.2	User interface	29
5	Results	30

List of Figures

Short abstract

This is my short abstract.

Abstract

This is my abstract

Chapter 1

Introduction

1.1 Context

1.1.1 3D viewers

- ▶ Applications?
- ▶ Who uses them?
- ▶ What for?

1.1.2 BIM geometry

- ▶ What is Building Information Modeling (BIM)? (short)
- ▶ Extend of BIM geometry?
- ▶ Complexity of BIM geometry?

1.1.3 LDBIM

- ▲ Focus on geometry!
- ▶ What is Linked Data BIM (LDBIM)?
- ▶ Why the need / What are the advantages of LDBIM?
- ▶ Context of enrichment and complexity

1.1.4 Computing power dilemma

- ▶ What is the hardware problem?

► Why is it that important for the Architecture, Engineering and Construction ([AEC](#)) industry?

►

1.2 Research questions

► Why the need for this thesis? (why a [LDBIM](#) viewer?)

► What is the possible solution? (Culling algorithms)

► Why the need for research questions? (culling algorithms are not new, always progress, see later)

1.2.1 To which extent can [LDBIM](#) geometry be culled to be streamed to lightweight viewers?

► What can be culled exactly?

► What needs to be streamed?

► What is the impact of culling on the viewing experience?

1.2.2 Can existing semantic and ontologies be used to feed possible culling algorithms?

► What are ontologies?

► Can GIS ontologies be used too?

► What are the advantages of using ontologies?

1.3 Research objectives

1.3.1 Bring forward the advantages of [LDBIM](#) for vizualization of big 3D models

► Showcase that existing models are already mature enough for these usecases.

►

1.3.2 Showcase the fisability of [LDBIM](#) for vizualization of big 3D models

►

GPT chat

BIM and Augmented Reality (AR)

Me: What is linked data?

GPT chat: Linked data is a way of representing and sharing data on the internet so that it can be easily linked to related data and more easily used by machines. This is achieved by using standardized formats and protocols, such as RDF (Resource Description Framework), which allows data from different sources to be connected and used together. The goal of linked data is to create a web of interconnected data that can be easily accessed and used by anyone, with the ultimate aim of enabling people and machines to better understand the world around them.

Me: What are the opportunities for the AEC industry?

GPT chat: The AEC (architecture, engineering, and construction) industry has many opportunities to benefit from linked data. For example, linked data can be used to more easily share and integrate data from different sources, such as building information models (BIM) and GIS (geographic information systems), which can help improve collaboration and decision-making among different stakeholders in a construction project. Linked data can also be used to create more accurate and detailed models of buildings and infrastructure, which can help improve the design and planning of projects. Additionally, linked data can be used to create more comprehensive and accessible datasets that can be used for research and analysis, helping to drive innovation and progress in the AEC industry.

Me: Will the ifc format still have a future?

GPT chat: Yes, linked data is interoperable with the IFC (Industry Foundation Classes) standards. IFC is a standardized data model for building and construction, and it can be used to represent and share data in a consistent and standardized way. This makes it possible to easily integrate linked data with IFC data, allowing different sources of data to be more easily combined and used together. Additionally, RDF, the standard format for linked data, is

designed to be interoperable with other data formats, including IFC, which further enhances its ability to integrate with other data sources and systems.

Me: How can IFC 3d data be translated into the RDF graph?

GPT chat: One way to represent the 3D geometry of an IFC file in an RDF graph would be to use a geometry-specific vocabulary, such as the GeoSPARQL vocabulary, to describe the geometry. This vocabulary defines a set of terms and properties that can be used to represent geometric objects and their spatial relationships, such as points, lines, and polygons. Using this vocabulary, the 3D geometry from an IFC file could be represented as a series of RDF triples, with each triple consisting of a subject, predicate, and object. The subject would be the IFC element or object that the geometry represents, the predicate would be a term from the GeoSPARQL vocabulary that describes the geometry, and the object would be a representation of the geometry in a standardized format, such as WKT (well-known text) or GML (geography markup language). This would allow the 3D geometry to be integrated into the RDF graph and linked to other data in the graph using the standardized vocabulary and formats.

Me: Wouldn't it be better to store 3d data in GLTF files outside the graph and link to them?

GPT chat: Storing 3D data in GLTF (GL Transmission Format) files and linking to them from an RDF graph can be a good approach in some cases, but it also has some limitations. One advantage of using GLTF files is that they are a widely-used and well-supported format for storing 3D geometry, and many tools and applications can read and use GLTF files. Additionally, GLTF files are designed to be small and efficient, which can be important for large or complex 3D models. However, using GLTF files also has some disadvantages. For example, it can be difficult to link or relate the data in a GLTF file to other data in an RDF graph, because the data in a GLTF file is not structured in the same way as RDF data. Additionally, using GLTF files can make it more difficult to query or analyze the 3D data using the full range of capabilities offered by RDF and SPARQL (the standard query language for RDF), because the data is not directly integrated into the RDF graph. Overall, whether it is better to store 3D data in GLTF files or to represent it directly in an RDF graph will depend on the specific requirements and goals of the project.

Me: What if that project is about visualizing BIM data via augmented reality?

GPT chat: If the project is about visualizing BIM (building information modeling) data via augmented reality, then representing the 3D geometry directly in the

RDF graph can be a more effective approach. This is because AR (augmented reality) applications typically require real-time access to the 3D geometry, and accessing this data from a separate GLTF file can be slow and cumbersome. By representing the 3D geometry directly in the RDF graph, it can be accessed more quickly and easily, which can improve the performance and user experience of the AR application. Additionally, representing the 3D geometry in the RDF graph can also make it easier to link the geometry to other data in the graph, such as metadata about the elements in the BIM model, which can provide more context and information to the user of the AR application.

Me: But would it not overpopulate the graph with geometric data, especially in the AEC industry?

GPT chat: Representing the 3D geometry of a BIM model directly in an RDF graph can potentially increase the size of the graph, but this can be managed in various ways to avoid overpopulating the graph. For example, the 3D geometry can be represented using a compact and efficient format, such as WKT or GML, which can help reduce the size of the data in the graph. Additionally, the RDF graph can be partitioned or divided into smaller sub-graphs, with each sub-graph containing the data for a specific aspect or element of the BIM model. This can help to distribute the data and avoid overloading any one part of the graph. Furthermore, the RDF graph can be stored and managed using specialized tools and technologies, such as triple stores or graph databases, which are designed to efficiently manage and query large amounts of RDF data. Overall, there are various approaches that can be used to effectively manage the size and complexity of an RDF graph containing 3D geometry data from a BIM model, and these approaches can help to support the use of linked data in the AEC industry.

Me: What are the options when it comes to query that 3d data if it would be for example in the view of an augmented reality interface?

GPT chat: If the 3D data is represented in an RDF graph, it can be queried using SPARQL, the standard query language for RDF. SPARQL allows you to specify the data that you want to retrieve from the graph in the form of a query, and the query engine will return the matching data from the graph. This can be useful for an augmented reality (AR) interface, because it allows you to specify exactly what data you want to retrieve from the graph and display in the AR view. For example, you could use SPARQL to query the graph for the 3D geometry of specific elements in the BIM model, or to query for metadata about those elements. You could also use SPARQL to query for information about the spatial relationships between different elements in the BIM model, such as the distance between two elements or the orienta-

tion of an element relative to another element. Overall, using SPARQL to query an RDF graph containing 3D data can provide a powerful and flexible way to access and use that data in an AR interface. can you give me an example for a sparql query for the visible objects in the ar view? Here is an example of a SPARQL query that can be used to retrieve the visible objects in an augmented reality (AR) view:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ar: <http://example.org/ar#>
PREFIX bim: <http://example.org/bim#>

SELECT ?object ?geometry
WHERE {
    ?object rdf:type bim:Object .
    ?object ar:isVisible true .
    ?object bim:hasGeometry ?geometry .
}
```

This query uses the `ar:isVisible` property to select only those objects that are marked as being visible in the AR view. It then uses the `bim:hasGeometry` property to retrieve the 3D geometry for each of those objects. The `?object` and `?geometry` variables represent the objects and their corresponding geometry that are returned by the query. This query can be modified and extended as needed to meet the specific requirements of the AR application.

Me: yes, but how is that 'ar:isVisible' property computed in the first place? Do you have a code for that?

GPT chat: The `ar:isVisible` property in the SPARQL query above is a hypothetical property that is used to indicate whether an object is visible in the AR view. In order to compute this property, you would need to write code that determines which objects are currently visible in the AR view, based on the user's current location and orientation, the position and orientation of the objects in the 3D model, and any other relevant factors. The code for computing this property would depend on the specific details of the AR application, such as the types of objects that are being modeled, the algorithms and data structures that are used to represent the 3D geometry, and the capabilities and limitations of the AR platform. Therefore, it is not possible to provide a specific code example for computing the `ar:isVisible` property without more information about the AR application.

Me: Let us pretend we are writing a web-based AR app, and that the 3d geometry is in its simplest form 3d points defined in the graph, which points are in the view if you have the 3d position and orientation of the camera?

GPT chat: If you have the 3D position and orientation of the camera in a web-based AR app, and the 3D geometry of the objects in the scene is represented as

a series of 3D points in an RDF graph, then you can use a SPARQL query to retrieve the points that are visible in the current view. Here is an example of a SPARQL query that can be used to do this:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ar: <http://example.org/ar#>
PREFIX bim: <http://example.org/bim#>

SELECT ?point
WHERE {
    ?point rdf:type bim:Point .
    ?point bim:hasCoordinates ?coordinates .
    ?coordinates bim:x ?x .
    ?coordinates bim:y ?y .
    ?coordinates bim:z ?z .
    FILTER (
        ?x > ?minX &&
        ?x < ?maxX &&
        ?y > ?minY &&
        ?y < ?maxY &&
        ?z > ?minZ &&
        ?z < ?maxZ
    )
}
```

This query uses the `bim:hasCoordinates` and `bim:x`, `bim:y`, and `bim:z` properties to retrieve the coordinates of each point in the 3D model. It then uses a `FILTER` clause to select only those points that have coordinates within the specified range for each dimension (`?minX`, `?maxX`, `?minY`, `?maxY`, `?minZ`, and `?maxZ`). These ranges would need to be set to the minimum and maximum values for each dimension that are within the current view of the camera. This query can be modified and extended as needed to meet the specific requirements of the AR application.

Me: and if you add a parameter to filter points at a certain distance of the camera?

GPT chat: If you want to filter the points in the 3D model based on their distance from the camera in a web-based AR app, then you can add a `FILTER` clause to the SPARQL query to do this. Here is an example of how this could be done:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ar: <http://example.org/ar#>
PREFIX bim: <http://example.org/bim#>

SELECT ?point
WHERE {
    ?point rdf:type bim:Point .
    ?point bim:hasCoordinates ?coordinates .
    ?coordinates bim:x ?x .
```

```

?coordinates bim:y ?y .
?coordinates bim:z ?z .
FILTER (
  ?x > ?minX &&
  ?x < ?maxX &&
  ?y > ?minY &&
  ?y < ?maxY &&
  ?z > ?minZ &&
  ?z < ?maxZ &&
  (
    (?x - ?cameraX) * (?x - ?cameraX) +
    (?y - ?cameraY) * (?y - ?cameraY) +
    (?z - ?cameraZ) * (?z - ?cameraZ)
  ) < ?distance * ?distance
)
}

```

This query is similar to the previous example, but it includes an additional FILTER clause that uses the distance formula to calculate the distance between each point and the camera. The ?distance variable specifies the maximum distance that a point can be from the camera in order to be included in the query results. This query can be modified and extended as needed to meet the specific requirements of the AR application.

Me: what if you now don't have points but links to gltf files for each object, and that you would first need to compute there center points?

...

URIs

Me: What are URI

GPT chat: A URI, or Uniform Resource Identifier, is a string of characters that identify a name or a resource on the Internet. URIs can be broken down into two types: URLs and URNs. A URL, or Uniform Resource Locator, is the web address of a specific page or file on the internet, while a URN, or Uniform Resource Name, is a unique identifier for a resource that may or may not be available on the internet. For example, "http://www.example.com" is a URL, while "urn:isbn:978-1-60309-024-3" is a URN

Me: does everything have a uri in an rdf graph?

GPT chat: In an RDF graph, every resource is identified by a URI. The URI serves as a unique identifier for the resource, and it can be used to refer to the resource in RDF statements. The URI can be thought of as the name of the resource, and it is used to distinguish the resource from other resources in the graph.

Me: but does every triple have a uri?

GPT chat: In an RDF graph, every resource is identified by a URI, but not every triple necessarily has a URI. A triple in an RDF graph consists of a subject, a predicate, and an object, and each of these components can be either a URI or a literal value. In some cases, all three components of a triple may be URIs, while in other cases, one or more of the components may be a literal value. For example, the following triple has a URI for the subject and predicate, but the object is a literal string:

```
`<http://www.example.com/books/book1> <http://www.example.com/ns/
title> "The Great Gatsby"`
```

In this triple, the subject is the URI `http://www.example.com/books/book1`, the predicate is the URI `http://www.example.com/ns/title`, and the object is the string "The Great Gatsby".

Me: why does querying resources gives back uri's?

GPT chat: When you query a resource in an RDF graph, the response typically includes the URI of the resource. The URI serves as a unique identifier for the resource, and it can be used to refer to the resource in other queries or statements. Additionally, the URI may be used to dereference the resource and retrieve additional information about it.

Me: Is that resource URI automatically assigned by the sparql endpoint you are querying?

GPT chat: The URI for a resource in an RDF graph is not automatically assigned by a SPARQL endpoint. In most cases, the URI for a resource is determined by the creator of the RDF data, and it is typically chosen to be a unique and meaningful identifier for the resource. For example, a URI for a book might include the title of the book and the author's name, while a URI for a person might include their full name and date of birth. The URI is then used to identify the resource in the RDF data, and it can be used to query the resource using a SPARQL endpoint.

Me: can you give me an example of triple data with an ressource uri?

GPT chat: Sure, here is an example of a triple with a resource URI:

```
`<http://www.example.com/books/book1> <http://www.example.com/ns/  
title> <http://www.example.com/ns/TheGreatGatsby>`
```

In this triple, the subject is the URI `http://www.example.com/books/book1`, the predicate is the URI `http://www.example.com/ns/title`, and the object is the URI `http://www.example.com/ns/TheGreatGatsby`. The subject of this triple identifies a specific book, the predicate specifies that the object of the triple is the title of the book, and the object is the URI of the resource that represents the title of the book. This URI could be used in other triples to provide additional information about the title of the book, such as the language in which it was written or the author who wrote it.

Me: Does that uri then directly refers to data? I mean, can I look it up in my browser, or is it just semantics for the sparql endpoint?

GPT chat: A URI in an RDF graph typically does not directly refer to data that can be looked up in a web browser. Instead, the URI serves as a unique identifier for a resource in the RDF graph, and it is used to distinguish that resource from other resources in the graph. The URI itself does not contain any information about the resource, but it can be used to retrieve additional information about the resource through a process called dereferencing. To dereference a URI, you can send a request to the URI using a web client, such as a web browser or a SPARQL endpoint. If the URI is associated with a resource that is available on the internet, the request will return information about the resource. For example, if the URI identifies a web page, the request will return the contents of the web page. However, not all URIs are associated with resources that are available on the internet. In some cases, the URI may simply serve as a unique identifier for a concept or an idea, without referring to any specific data. In these cases, dereferencing the URI will not return any information about the resource.

...

RDF reasoners

Me: What is a rdf reasoner?

GPT chat: An RDF reasoner is a type of software that can automatically infer new information based on existing data in an RDF (Resource Description Framework) graph. An RDF graph is a way of representing information using the RDF data model, which is a set of interconnected statements about resources. For example, if an RDF graph contains statements that indicate that a particular person is the child of a certain parent, an RDF reasoner could infer that the person is also the sibling of the parent's other children. In this way, an RDF reasoner can help to automatically organize and make sense of complex data in an RDF graph.

Me: Does it add data to the graph or is only used for the query?

GPT chat: An RDF reasoner does not add data to the RDF graph directly. Instead, it analyzes the existing data in the graph and uses rules of inference to automatically generate new statements that can be added to the graph. These new statements are typically derived from the existing data in the graph and reflect relationships or connections that can be logically inferred from that data. For example, if an RDF graph contains statements indicating that person A is the child of person B, and person B is the child of person C, an RDF reasoner could infer that person A is the grandchild of person C and generate a new statement to that effect. In this way, an RDF reasoner can help to enrich the data in an RDF graph and make it more useful for querying and analysis.

Me: Does it then run continuously, searching for new correlations?

GPT chat: It depends on the specific implementation of the RDF reasoner. Some RDF reasoners may be designed to run continuously, continuously analyzing the data in an RDF graph and generating new statements as needed. Other RDF reasoners may be designed to run only when explicitly invoked by a user or application, in which case they would not be continuously searching for new correlations. In general, whether an RDF reasoner runs continuously or not will depend on the specific requirements and goals of the system in which it is being used.

Me: Is it part of the triplestore or the sparql endpoint?

GPT chat: An RDF reasoner is not necessarily part of a triplestore or a SPARQL endpoint. A triplestore is a type of database specifically designed for storing and managing RDF data, while a SPARQL endpoint is a web service that allows users to query an RDF graph using the SPARQL query language. An

RDF reasoner, on the other hand, is a type of software that can automatically infer new information from an RDF graph, but it is not necessarily part of a triplestore or a SPARQL endpoint. In some cases, an RDF reasoner may be used in conjunction with a triplestore or a SPARQL endpoint as part of a larger system for managing and querying RDF data. In other cases, an RDF reasoner may be used independently of these technologies.

Me: Wait a minute. Does all triplestores have sparql endpoints?

GPT chat: Not all triplestores have SPARQL endpoints. A triplestore is a type of database specifically designed for storing and managing RDF data, while a SPARQL endpoint is a web service that allows users to query an RDF graph using the SPARQL query language. A triplestore and a SPARQL endpoint are separate, but related, technologies. A triplestore provides a backend storage mechanism for RDF data, while a SPARQL endpoint provides a way for users to access and query that data. In some cases, a triplestore may include a SPARQL endpoint as part of its functionality, allowing users to query the data stored in the triplestore using SPARQL. In other cases, a triplestore may not include a SPARQL endpoint, and a separate SPARQL endpoint may need to be used in order to query the data stored in the triplestore.

Me: how would a sparql endpoint talk to a separate triplestore? By using lots of http requests?

GPT chat: A SPARQL endpoint can talk to a separate triplestore by using a network protocol such as HTTP to send SPARQL queries to the triplestore and receive the results of those queries. The exact details of how this communication occurs will depend on the specific implementation of the SPARQL endpoint and the triplestore. In some cases, the SPARQL endpoint and the triplestore may be part of the same system and may be designed to communicate with each other using a custom, proprietary protocol. In other cases, the SPARQL endpoint and the triplestore may be part of separate systems and may communicate with each other using a standard network protocol such as HTTP. In either case, the goal is to allow the SPARQL endpoint to send queries to the triplestore and receive the results of those queries in a way that is efficient and transparent to the user.

Me: So a triplestore CAN interpret sparql queries?

GPT chat: A SPARQL endpoint is a web service that allows users to query an RDF graph using the SPARQL query language. It provides a convenient way for users to access and query RDF data without having to directly interact with the underlying triplestore. This can be useful in a number of situations. For example, a SPARQL endpoint can provide a standard, consistent interface for querying RDF data, allowing different applications and users to access

the data in the same way. Additionally, a SPARQL endpoint can provide additional features and functionality beyond what is available in a triplestore, such as support for querying multiple RDF graphs, advanced query optimization, and caching of query results. In this way, a SPARQL endpoint can provide a useful layer of abstraction between the triplestore and the applications and users that need to access the data stored in the triplestore.

Me: So, in a certain way it is a standardisation of the querying?

GPT chat: Yes, you could say that a SPARQL endpoint provides a standardized interface for querying RDF data. The SPARQL query language is a standardized, widely-used language for querying RDF data, and a SPARQL endpoint provides a way for users and applications to access RDF data using this standard language. This can help to ensure that different applications and users are able to access RDF data in a consistent and predictable way, regardless of the specific triplestore or other backend technology being used to store the data. In this way, a SPARQL endpoint can help to promote interoperability and facilitate the use of RDF data in a variety of different contexts.

...

Linked Data and BIM

Me: What are the disadvantages of linked data for the AEC industry? Specifically when it comes to communication on the building site?

- GPT chat:**
- 1.The complexity of linked data can be overwhelming and difficult to understand for those without a strong technical background. This can hinder communication on the building site and lead to misunderstandings or errors.
 - 2.Linked data often requires specialized software and tools to access and interpret, which may not be readily available on the building site. This can make it difficult for teams to share and collaborate on information in real-time.
 - 3.The sheer volume of data that is generated and linked through the use of linked data can be overwhelming, making it difficult to identify and access the specific information that is needed for a particular project or task.
 - 4.The integration of linked data into existing systems and processes can be time-consuming and costly, requiring significant training and resources to implement effectively.
 - 5.The security and privacy of linked data can be a concern, as the interconnected nature of the data means that any breach or unauthorized access could potentially affect multiple systems and parties.

Overall, the disadvantages of linked data for the AEC industry include a potential lack of understanding and specialized tools, difficulty managing and accessing data, and concerns over security and privacy.

Me: Which type of software could help for that matter?

- GPT chat:**
- 1.There are several types of software that could help with the challenges of linked data in the AEC industry, including:
 - 2.Collaboration and project management software, such as Asana or Trello, which can help teams to organize and share information in real-time.
 - 3.Data visualization and analysis tools, such as Tableau or Qlik, which can help to make sense of large volumes of linked data and identify patterns and trends.
 - 4.Data management and governance platforms, such as Talend or Informatica, which can help to ensure the integrity and security of linked

data, as well as facilitate its integration into existing systems and processes.

5. Linked data tools, such as Apache Jena or GraphDB, which can help to create, manage, and query linked data sets.

Overall, the use of specialized software can help to overcome the challenges of linked data in the AEC industry, enabling teams to better communicate, collaborate, and make sense of the data.

Me: Which type of software could take advantage of linked data on the building site itself?

GPT chat: There are several types of software that could take advantage of linked data on the building site itself, including:

1. Augmented reality (AR) and virtual reality (VR) tools, such as HoloLens or Sketchfab, which can enable teams to visualize and interact with linked data in real-time, providing a more immersive and intuitive way of understanding complex information.
2. Sensors and IoT technologies, such as BIM sensors or connected devices, which can generate and link data in real-time from the building site, providing teams with real-time insights and updates on the status and performance of the project.
3. Building information modeling (BIM) software, such as Autodesk Revit or ArchiCAD, which can help teams to create and manage linked data sets that represent the design, construction, and operation of a building, providing a comprehensive and integrated view of the project.

Overall, the use of specialized software can help teams to take advantage of linked data on the building site itself, enabling them to better understand and manage the project in real-time.

Me: What would be the greatest challenge for AR on the construction site, using linked data?

GPT chat: The greatest challenge for AR on the construction site, using linked data, would be ensuring that the data is accurate, up-to-date, and properly linked. As linked data is often generated from multiple sources and can be updated in real-time, it is critical that the data is accurate and consistent across all systems and stakeholders. This can be a challenge, as errors or inconsistencies in the data could lead to misunderstandings or errors on the construction site. Additionally, ensuring that the data is properly linked and interconnected can also be a challenge, as it requires specialized knowledge and tools to create and manage linked data sets. This can be a barrier

for teams that are not familiar with linked data and its potential benefits. Overall, the greatest challenge for AR on the construction site, using linked data, would be ensuring the accuracy, consistency, and connectivity of the data, in order to support effective decision-making and collaboration on the building site.

...

Coherent Hierarchical Culling algorithm (CHC)

Me: Can you explain me the Coherent Hierarchical Culling algorithm?

GPT chat: Coherent Hierarchical Culling (CHC) is an algorithm used in computer graphics to improve the performance of rendering 3D scenes by reducing the number of objects that need to be processed by the graphics pipeline. The goal of CHC is to quickly and efficiently determine which objects in a scene are visible to the viewer and which objects are occluded (hidden) by other objects and therefore do not need to be drawn. This can greatly improve the performance of a graphics application, especially when dealing with complex or large scenes.

CHC works by dividing the scene into a hierarchy of bounding volumes, with each level of the hierarchy representing a different level of detail. The hierarchy is constructed so that each bounding volume encloses one or more objects in the scene, and the bounding volumes at higher levels of the hierarchy enclose groups of objects at lower levels.

The CHC algorithm begins by determining the bounding volume at the top level of the hierarchy that encloses the viewer's current viewpoint. It then uses this bounding volume to recursively traverse the hierarchy and identify the objects that are potentially visible to the viewer. As the algorithm traverses the hierarchy, it eliminates objects and bounding volumes that are completely occluded by other objects or bounding volumes. This process continues until the algorithm reaches the lowest level of the hierarchy, at which point it has a list of the objects that are potentially visible to the viewer.

CHC is an efficient algorithm because it can quickly eliminate large groups of objects and bounding volumes that are not visible to the viewer, reducing the number of objects that need to be processed by the graphics pipeline. It is also a "coherent" algorithm because it takes advantage of spatial coherence in the scene, meaning that objects that are near each other in the scene are likely to be occluded by similar objects. This allows the algorithm to quickly eliminate large groups of objects that are occluded by other objects.

Me: How does it compute if a bounding volume is completely occluded?

GPT chat: In the Coherent Hierarchical Culling (CHC) algorithm, the occlusion of a bounding volume is determined by comparing it to other bounding volumes in the scene. Specifically, the algorithm compares the bounding volume being tested to the bounding volumes of other objects in the scene to see if it is completely contained within another bounding volume. If

the bounding volume being tested is completely contained within another bounding volume, it is considered to be occluded and is eliminated from further processing.

To determine if one bounding volume is completely contained within another bounding volume, the CHC algorithm compares the vertices of the bounding volume being tested to the planes of the bounding volume it is being compared to. If all of the vertices of the bounding volume being tested lie on the same side of all of the planes of the other bounding volume, then the bounding volume being tested is completely contained within the other bounding volume and is considered occluded.

There are several different types of bounding volumes that can be used with the CHC algorithm, including axis-aligned bounding boxes (AABBs), oriented bounding boxes (OBBs), and bounding spheres. The specific method for determining occlusion will depend on the type of bounding volume being used.

Me: Can you explain it all again in simpler terms? I am confused about what bounding volumes actually are.

GPT chat: Certainly! Coherent Hierarchical Culling (CHC) is an algorithm used to improve the performance of rendering 3D graphics by reducing the number of objects that need to be processed. It does this by dividing the scene into a hierarchy of bounding volumes, with each level of the hierarchy representing a different level of detail.

A bounding volume is a geometric shape that encloses an object or group of objects in the scene. The bounding volume is typically chosen to be as small as possible while still enclosing the object or objects it represents. There are several different types of bounding volumes that can be used, including axis-aligned bounding boxes (AABBs), oriented bounding boxes (OBBs), and bounding spheres.

The CHC algorithm begins by determining the bounding volume at the top level of the hierarchy that encloses the viewer's current viewpoint. It then uses this bounding volume to recursively traverse the hierarchy and identify the objects that are potentially visible to the viewer. As the algorithm traverses the hierarchy, it eliminates objects and bounding volumes that are completely occluded by other objects or bounding volumes. This process continues until the algorithm reaches the lowest level of the hierarchy, at which point it has a list of the objects that are potentially visible to the viewer.

To determine if a bounding volume is occluded, the CHC algorithm com-

compares it to the bounding volumes of other objects in the scene. If the bounding volume being tested is completely contained within another bounding volume, it is considered to be occluded and is eliminated from further processing. This allows the CHC algorithm to quickly eliminate large groups of objects that are not visible to the viewer, improving the performance of the graphics application.

Me: And in very simple terms, how does it compute if bounding volumes are occluded?

GPT chat: To determine if a bounding volume is occluded (hidden) by another bounding volume, the Coherent Hierarchical Culling (CHC) algorithm compares the vertices of the bounding volume being tested to the planes of the bounding volume it is being compared to. If all of the vertices of the bounding volume being tested lie on the same side of all of the planes of the other bounding volume, then the bounding volume being tested is considered to be completely contained within the other bounding volume and is therefore occluded.

For example, if the bounding volume being tested is an axis-aligned bounding box (AABB) and the bounding volume it is being compared to is also an AABB, the CHC algorithm would compare the vertices of the AABB being tested to the planes of the other AABB. If all of the vertices of the AABB being tested lie on the same side of all of the planes of the other AABB, then the AABB being tested is considered to be occluded.

This process is repeated for each bounding volume in the scene, allowing the CHC algorithm to quickly eliminate objects and bounding volumes that are not visible to the viewer and improve the performance of the graphics application.

Me: A bounding volume A sitting inside another bounding volume B means all of the geometry in A is not visible outside of B?

GPT chat: Yes, that's correct! If a bounding volume (A) is completely contained within another bounding volume (B), it means that all of the geometry represented by bounding volume A is completely occluded (hidden) by the geometry represented by bounding volume B. This is because the geometry represented by bounding volume A is completely enclosed by the geometry represented by bounding volume B, and therefore none of it is visible to the viewer.

In the context of the Coherent Hierarchical Culling (CHC) algorithm, this means that the objects represented by bounding volume A can be eliminated from further processing, since they are not visible to the viewer. This

can greatly improve the performance of a graphics application, especially when dealing with complex or large scenes.

...

Chapter 2

State of the art

2.1 [AR](#) on-site approaches

2.1.1 Autodesk xyz - Atom

2.1.2 Openspace AI

2.2 Linked geometric data

2.2.1 [BOT](#)

2.2.2 [ifcOWL](#)

2.2.3 Other ontologies

[bcfOWL](#)

2.3 [LDBIM](#) viewers

2.3.1 ld-bim.web.app

2.4 Rendering optimizations

2.4.1 Culling algorithms

Bounding Box Culling

Frustum Culling

Octree Culling

Quadtree Culling

Spatial Hashing

Spatial Partitioning

Visibility Culling

Chapter 3

Methodology

3.1 [RDF](#) data

3.1.1 Existing data

3.1.2 Adding data

3.2 [LOD](#) streaming model

3.2.1 Culling

3.2.2 Redundancy avoidance with [BCF](#) history

3.3 Querying

3.3.1 Front-end

3.3.2 Back-end

3.4 Rendering

3.4.1 Three.js

3.4.2 Shaders

Chapter 4

Website

4.1 Setup

4.1.1 Software

4.1.2 Hardware

4.2 User interface

Chapter 5

Results

List of Acronyms

AR	Augmented Reality	8
AEC	Architecture, Engineering and Construction	7
BCF	BIM Collaboration Format	
bcfOWL	BIM Collaboration Format Ontology	
BIM	Building Information Modeling	6
BOT	Building Topology Ontology	
CHC	Coherent Hierarchical Culling algorithm	22
ifcOWL	Industry Foundation Classes Ontology	
LDBIM	Linked Data BIM	6
LOD	Level of Detail	
RDF	Resource Description Framework	