Exploring the Potential of NoSQL Databases for a BIM and 3D GIS enabled Location Framework for Construction Digital Twins and the Golden Thread

Claire Ellul[a,*], Richard Loo[b], George Floros[c]

[a] University College London, UK - c.ellul@ucl.ac.uk
[b] Singapore Lands Authority, Singapore - richard$_{loo}$@$sla.gov.sg$
[c]Skanksa, London, UK - Georgios.Floros@skanska.co.uk

Keywords: NoSQL, Digital Twins, Construction, Data Engineering, Benchmark

Abstract

As digital transformation accelerates in the construction industry, combining data from Building Information Modelling (BIM) and 3D Geographic Information Systems (3DGIS) has become critical for enhanced decision-making across sectors including construction planning, supply chain management, health and safety and sustainable waste disposal. In combination, BIM and GIS underpin a location framework for Digital Twins across these applications. While traditionally relational databases have been used to integrate the data into one system, they face increasing limitations in handling the massive, complex BIM and 3DGIS data. This research investigates the potential of NoSQL databases as an alternative solution. Focusing on MongoDB, the study conducts a systematic performance comparison with the mature relational database, PostgreSQL. BIM and 3DGIS datasets from a live infrastructure construction project are utilised in three experiments assessing direct query, API-based retrieval, and 3D visualisation. The results provide valuable insights into MongoDB's strengths in managing large spatial data via flexible schemas. However, limitations surface as data complexity and volume increases. Overall, this study concludes that NoSQL databases do offer some advantages for optimising BIM-3DGIS integration for seamless, combined, use, but more work is needed to fully harness their capabilities.

1. Introduction

The UK's construction industry is undergoing a significant digital transformation, driven largely by technologies such as Building Information Modelling (BIM), Internet of Things (IoT), and advanced data analytics (HM Government, 2015). Since 2016, BIM has been mandated for all public sector projects, aiming to revolutionise 3D modelling and data management for infrastructure (HM Government, 2015). This digital transformation is expected to yield significant cost savings and operational efficiencies (ibid.).

Key to this transformation is the concept of **Digital Twins** (DT), a dynamic digital representation of an asset or infrastructure system (National Infrastructure Commission, 2017). DT can facilitate an immersive understanding of infrastructure functionality and its relationship with the surrounding environment, empowering both government and industry to make better-informed decisions for future developments (ibid). Broadly, DT may involve real-time two way communication between the digital and physical environment, digital shadows (one way communication) or human-in-the-loop (two way with manual decision making) situations (Siddorn et al., 2022).

In parallel with DT, the UK is exploring regulations relating to the '**golden thread**' of information relating to the built environment, which aims to: 'have the right information in order to understand the steps needed to keep both the building and the people living in it safe'(Storing your building's information – the golden thread, n.d.). The requirements include: information must be kept digitally, and securely, and be a building's 'single source of truth', as well as being presented in a way that a person can use (Storing your building's information – the golden thread, n.d.).

* Corresponding author

The rapid digitalisation of the construction sector has also led to a proliferation of geospatial big data. These data often surpass current computing capacity (Lee and Kang, 2015), and are typically characterised by the five 'Vs': Volume is the sheer amount of generated data; Velocity is its rapid generation and distribution; Variety refers to the different data formats; Veracity concerns reliability; and Value is about the actionable insights derived from the data (Anuradha et al., 2015).

Amongst the varied data being generated is location data (geospatial data, that can be related in some way to a place on, above or below the surface of the earth). Construction DT are intrinsically location DT: they can underpin construction challenges such as construction site planning, real-time monitoring for health and safety or progress tracking, stakeholder engagement and logistics (see Section 2.1.1). Modelling these examples requires data that includes both engineering detail and a wider context/surrounding the generated data spans BIM and geospatial disciplines.

The combination of BIM and 3D Geographic Information System (3D GIS) underpins both construction DT and the golden thread: "since location is a common attribute among different datasets it can be used to combine and integrate them" (Geospatial Commission, 2023). The resulting framework enables a multi-scale data anchor, where additional data can be linked (via a join) with a real world location in the integrated environment, with the granularity (from detailed engineering data to generalised 2D mapping) selected as appropriate.

To date, relational databases (Section 2.3.1) have primarily been used to create an integrated location framework. However, NoSQL databases - designed to store large quantities of data in a distributed environment - may also show promise (Section 2.3.2). There has been little comparative research as to how these new database types perform with big geospatial vector datasets. This research addresses the question:

How does a NoSQL database perform in storing and querying BIM and 3DGIS data, when compared to a relational database?

2. Background and Literature Review

2.1 Context

To examine this subject within a real-world context, this research was conducted in collaboration with a major UK construction firm., one of the leading global corporations specialising in construction and infrastructure development. They currently rely on relational databases for storing and managing 3D data and proprietary formats for BIM, and face challenges in efficiently querying and retrieval of large 3D models, which are used to facilitate visualisation on web platforms, to better inform decision-making processes and to enable **data democratisation** - i.e. remove the data from silos where expert users are needed to access and explore it, and enable it to be combined with other data across the organisation.

2.1.1 Digital Twin Applications for Constructions Ammar et al. (2022) identify several applications of DTs including facilitating information flow and data transparency between stakeholders by providing accurate and up to date documentation (Ammar et al., 2022) and providing a collaborative environment with a single source of truth (Ammar et al., 2022). DTs could enable real-time monitoring and real-time tracking of construction sites allowing more informed decisions to be made by utilising all the available data (Thomas and Bowman, 2021). They are relevant for construction site logistics and visual presentation of data is a key component of its successful implementation (Greif et al., 2020).

2.2 Defining Relational and NoSQL Databases

2.3 Databases

A database is a computerized record keeping system (Date, 2004) whilst a database management system (DBMS) is the software that handled access to the data in the database (Date, 2004). Databases are large (making use of both primary and secondary storage), shared (multiple users have simultaneous access to the data), persistent (data is not destroyed on power outage), ensure data privacy (users may be limited as to what data they can access) and should be efficient in performance (Atzeni et al., 1999). Thus, DBMS have key functions that encompass storing data and facilitating concurrent sharing to a large user base (Atzeni et al., 1999). They also offer the key component of **ad-hoc** querying due to **data independence** (Date, 2004) - the ability to slice and dice the stored data in many different ways, and create different views of the data depending on analytical/application requirements, without needing to write additional code.

2.3.1 Relational Databases A relational database employs tables composed of rows and columns to structure data, often spanning multiple tables linked through primary and foreign keys to depict relationships (Atzeni et al., 1999). It stores interconnected data following the relational model, where each table's rows are records with unique identifiers, and columns hold attributes, facilitating establishment of relationship among data (ibid). Structured Query Language (SQL) is the standard language used for relational databases.

Traditionally, relational databases have been the primary choice for managing geospatial data (Guo and Onstein, 2020; Gonçalves et al., 2021). 3D data is commonly stored in relational databases, such as Oracle and PostGIS (Mao et al., 2014; Višnjevac et al., 2017). However, according to (Baralis et al., 2017) and (Višnjevac et al., 2017), relational databases have various drawbacks when it comes to geospatial big data storage, in particular for BIM related information (Munawar et al., 2022) and high concurrency queries or large-volume data access for geospatial applications.

2.3.2 NoSQL Databases When facing rapid transaction challenges in sectors related to online retail and website search, and benefiting from the emergence of commodity hardware a number of organisations developed NoSQL databases to deliver improved performance for complex and massive datasets (NoSQL, 2018) (e.g. Amazon's DynamoDB (DeCandia et al., 2007), Google's BigTable (Chang et al., 2008), Apache's Cassandra (Apache, 2014)). NoSQL - Not Only SQL - can refer to schema-less, schema-free, or flexible schema guidelines to data modelling (Asaad et al., 2020). NoSQL databases have gained prominence due to their inherent capability to manage large volumes of data, particularly in the context of complicated and heterogeneous objects, a capacity that surpasses traditional relational models (Nassif et al., 2020).

In general, there are four types of structures that fall within the category of NoSQL (Krstić and Krstić, 2018; Dancuk, 2020; IBM, n.d.)):

- Key-value: Store data as a collection of key and value pairs
- Document: Store data in documents and typically in JSON, XML, or BSON formats
- Graph: Store data in nodes (entities) and edges (relationships) where edges define relationships between nodes
- Column-oriented: Store data in columns instead of rows.

In a location/geospatial data context they have been used to store large point clouds (Boehm and Liu, 2015) and IoT sensor data (Hong et al., 2023).

Table 1 describes differences between Relational and NoSQL databases(Krstić and Krstić, 2018; Dancuk, 2020; IBM, n.d.))

2.4 BIM and 3D GIS Integration and Interoperability

Interoperability is the ability of a system, or components of a system, to provide information portability and inter-application cooperative process control (Bishr, 1998). Extensive research into resolving "semantic heterogeneity" (Bishr, 1998) of BIM and GIS is underway - where a single feature such as a building is represented using a different data model in different systems (Roxin and Hbeich, 2019; Guyo et al., 2021). Challenges relating to syntactic heterogeneity, where geometry is represented in different ways (Bishr, 1998), are also being addressed (Liu and Ellul, 2022; Borrmann et al., 2015). As an intermediate approach, a DT location framework does not require full data interoperability between BIM and 3D GIS, but rather achieves integration - brining the data into one system for combined analysis - via minimal interoperability concepts which seek to identify any basic commonality as an initial step prior to full semantic mapping (Mulquin, 2023) - in this case, the location / coordinate information is common to both datasets.

	Relational	NoSQL
Model	Tables with rows and columns	Key-value, document, graph, column-oriented
Data	Strict data structures	Can be structured, semi-structured or unstructured data
Schema	Fixed	Flexible
Scalability	Mainly vertical by upgrading hardware	Sharding/horizontal with more servers to distribute load
Transaction Support	ACID (Atomicity, Consistency, Isolation and Durability) properties: • Atomicity: All transactions must either succeed entirely or fail completely. • Consistency: rules must ensure validation and prevent corruption. • Isolation: Concurrent transactions are not allowed to interfere with each other. • Durability: The outcomes of executing a transaction are final, even when failures occur.	Mostly follow BASE (Basically Available, Soft State, Eventual Consistency) properties: • Basically Available: Every user can execute queries, and the database distributes data across multiple systems to avoid a complete outage. • Soft State: The state of the database can evolve over time. • Eventual Consistency: Given a functioning system and sufficient time, the database will attain consistency.

Table 1. Relational versus NoSQL Databases

2.5 Database Performance Comparisons

Baralis et al. (2017) provided an in-depth analysis of four databases for cloud-based applications. Their research illustrated that while Azure Document DB had quick response times, Azure SQL Database showed stronger scalability. On the other hand, Sharma et al. (2018) focused on identifying the best database for GIS applications. They reported MongoDB has consistently outperformed both PostgreSQL and Neo4j, especially in the context of geotagged data queries. Pietroń (2019) measured the performance of selected databases in processing large geospatial datasets, establishing that MongoDB often surpassed PostGIS, especially when enlarging the search area. Meanwhile, Treviño Villalobos et al. (2020) aimed to determine the best fit database for supporting a Web Map Service. Their research evidenced that PostGIS is more apt for managing complex geospatial data types, while emphasizing the importance of multiple factors in selecting a DBMS. Lastly, (Makris et al., 2021) compared MongoDB and PostGIS for managing detailed spatio-temporal queries. They found that PostGIS often performed better, especially with indexes.

3. Data

Two types of data are used in this research - BIM for a road construction project and 3D GIS topographic mapping of the surrounding area (at Level of Detail 1 (Kutzner et al., 2023)). BIM is delivered as 15 files in Industry Foundation Classes (IFC) format (International Standards Organisation, 2018), version 2x3, with a total size of around 12.9 gigabytes, using a local grid referencing system. A screenshot of the IFC data is shown in Figure 1. IFC is an open, international standard (ISO 16739-1:2018) and is vendor-neutral (International Standards Organisation, 2018). It is not used for storing BIM data, but is used to exchange information from one party to another for a specific business transaction (Industry Foundation Classes - An Introduction, n.d.).

GIS data was sourced from Digimap's (Edina DigiMap Service, n.d.) Ordnance Survey (OS, the National Mapping Agency of Great Britain) Collection where building footprint polygons for the intended area of study were downloaded, part of the OS MasterMap (Mastermap Topography Layer, n.d.) topographic mapping dataset. Building heights are also included in the dataset. The downloaded data was in Esri's file geodatabase format with a total size of about 200 megabytes. Extrusion is performed to create 3D geometry.

4. Method

4.1 Selecting a NoSQL Database

4.1.1 Developing Selection Criteria Table 2.3.2 shows support for geospatial data in various NoSQL databases. Expanding upon that foundation and with the understanding of the dataset and communication with construction partner's team, a set of criteria for qualitative comparison among different NoSQL databases was developed. Each criterion is assigned a weight based on an evaluation of its significance in addressing the research topic. Table 2 shows the selection criteria

Within the above criteria, ratings from 5 to 1 were given depending on how the selected NoSQL databases met requirement: fully supported = 5; supported = 4;partially supported = 3;limited support = 2; not supported = 1.

4.2 Test 1 - Inserting Data

The hardware for performance testing was a Lenovo device with an AMD Ryzen 7 4800H with Radeon Graphics @ 2.90 GHz, 16GB RAM and a solid state drive. Safe Soft's FME (FME By Safe Software, n.d.) was used to import the data (both the IFC and the 3D building models), transforming/georeferencing the IFC from local coordinates to British National Grid for PostgreSQL (PostGIS, n.d.) with the PostGIS extension (PostGIS, n.d.). A transformation of the data from British National Grid to WGS84 is required for MongoDB (as the latter does not support British National Grid), and MongoDB's (MongoDB, n.d.) JSON importer was used to import the resulting GeoJSON data. Five different-sized datasets were created for the IFC data, and five additional datasets created for the 3D GIS data, to enable queries to be run to understand how the databases perform under varying loads, with the data selected to test NoSQL suitability for the static 3D city models and BIM that are the focus of this paper.

4.3 Test 2 - Direct Query

Select queries were conducted for the above datasets, using PGAdmin (PGAdmin, n.d.) for PostgreSQL (PostgreSQL, n.d.) with the PostGIS extension (PostGIS, n.d.), with SQL select statements (*select * from*) issued. The MongoDB (MongoDB, n.d.) shell was used to query the MongoDB system, by running find queries (*db.collection.find(query))*. Both clients were installed on the same machine as the server in order to avoid network latency.

4.4 Test 3 and 4 - API Query and Retrieving Data for Visualisation

NodeJS (NodeJS, n.d.) APIs were created to connect to the database servers. Two tests were carried out - a URL-based API

#	Criterion	Descriptions	Wt
1	Native storage of spatial data	Handles spatial data without the need for third-party extensions	0.15
2	Supports geometry types	Stores points, lines, polygons, multi-points, multi-polygons, and geometry collections?	0.25
3	Supports spatial operations	Supports spatial functions and operations such as spatial join, buffer, intersect, contains, and distance calculations?	0.15
4	Supports spatial indexes	Offers spatial indexing to optimise query performance over large spatial datasets?	0.15
5	Supports GeoJSON	Stores data in GeoJSON and JSON formats - i.e. web-based open standard formats	0.10
6	Supports coordinate reference system (CRS)s	Stores data in different coordinate systems?	0.10
7	API support	Can be extended to retrieve and interact with the data through Application Programming Interfaces (APIs)	0.05
8	Documentation	Availability of comprehensive documentation specifically focusing on geospatial features of the database, ensuring easier implementation and troubleshooting	0.05

Table 2. Criteria for evaluating geospatial suitability of different NoSQL databases

GeoJSON retrieval, and retrieving the test datasets for display inside a CesiumJS (CesiumJS, n.d.) globe environment.

5. Results

5.1 Selecting a NoSQL Database

Data Model	Database Name	Support for spatial (native)
Key-value	Redis	Yes
Document	MongoDB	Yes
Document	CouchDB	Yes
Graph	Neo4j	Yes
Column	Cassandra	No
Column	HBase	No

Table 3. Popular NoSQL Databases

A list of popular databases (Table 3 was generated combining information from (Guo and Onstein, 2020) and (Gonçalves et al., 2021). The selection criteria were then applied through a desk-based review, with results shown in Table 4.

To assess performance comparison between a NoSQL and a relational database in the context of BIM and GIS data management, it is important to ensure that the NoSQL can handle a wide range of geometries and support a variety of spatial operations. MongoDB stands out as the most suitable NoSQL candidate for this purpose: Couchbase aligns with MongoDB regarding geometry types, however its spatial operators are limited to a bounding box operation. Neo4j supports point geometry only and it has a restricted spatial operation where it covers only distance calculation among various point types. MongoDB supports a wider range of geometry types and offers diverse spatial query operators. Therefore, the MongoDB database was selected for the next phase of research.

5.2 Test 1 - Inserting Data

As shown in Figure 1, one individual 3D object could consist of tens of thousands of vertices for geometry. As such, both BIM and 3DGIS data were classified into five categories of varying sizes using a Data Sampler tool in FME (which creates subsets of the data), each category representing different complexities and number of vertices (Table 5).

5.3 Test 2 - Direct Query

Figure 2 and Figure 3 illustrate the results of a standard command line or PGAdmin query for the different dataset sizes. As can be expected, response times for the smaller datasets (ifc5

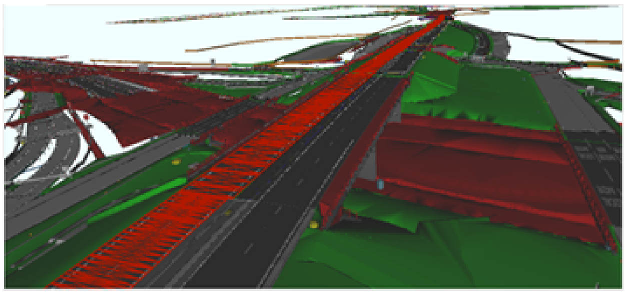

Figure 1. Vertices in a Single Feature

and gis5) are lower than those for larger data sizes. There is an inverse trend in terms of the comparatives - MongoDB out performs PostgreSQL/PostGIS for larger datasets, but slower retrieval times are shown for the very smallest datasets.

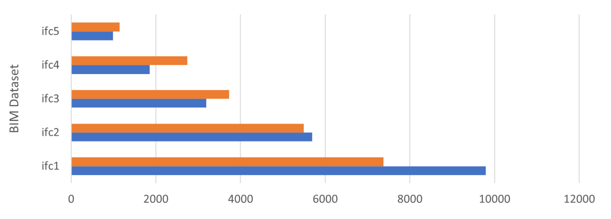

Figure 2. Direct Retrieval Times in ms, BIM Data (Orange=MongoDB, Blue=PostGIS)

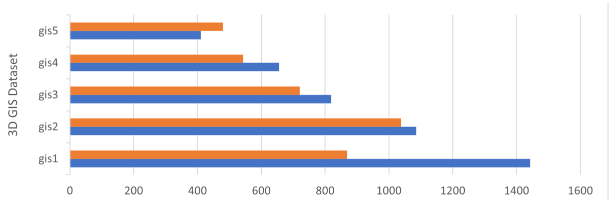

Figure 3. Direct Retrieval Times in ms, GIS Data (Orange=MongoDB, Blue=PostGIS)

5.4 Test 3 - API Query

10 API calls were to each database for each dataset (200 in total). Figure 4 shows the average response time for querying BIM via these APIs. A consistent pattern was observed – response times increased for both databases as the size of the queried data escalated from ifc5 to ifc2. Both PostGIS and MongoDB failed to return results for the largest dataset (ifc1), possibly indicating limitations in handling extensive data

S/N	Criteria	Weights	Redis	MongoDB	Couchbase	Neo4j	Cassandra	HBase
1	Native storage of spatial data	0.15	0.75	0.75	0.75	0.75	0.15	0.15
2	Supports geometry types	0.25	0.50	1.25	1.25	1.00	0.75	0.25
3	Supports spatial operations	0.15	0.60	0.60	0.45	0.60	0.45	0.15
4	Supports spatial indexes	0.15	0.45	0.60	0.30	0.60	0.30	0.15
5	Supports GeoJSON	0.10	0.50	0.50	0.50	0.10	0.10	0.10
6	Supports coordinate reference system (CRS)	0.10	0.40	0.40	0.30	0.30	0.40	0.10
7	API support	0.05	0.15	0.25	0.25	0.25	0.15	0.25
8	Documentation	0.05	0.20	0.25	0.20	0.20	0.15	0.15
	Total Weighted Score		3.55	4.60	4.00	3.80	2.45	1.30

Table 4. Comparison of criteria matrix and weighted score for selected NoSQL databases

Datasets		3D Object Count
BIM	ifc1	28848
	ifc2	14424
	ifc3	9616
	ifc4	7212
	ifc5	5769
3DGIS	gis1	6564432
	gis2	23020
	gis3	15346
	gis4	11510
	gis5	9208

Table 5. Dataset Insertion Results

volumes. As for the remaining data sizes, MongoDB consistently outperformed PostGIS, showing quicker response times across the board.

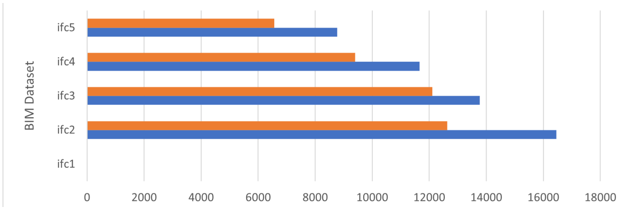

Figure 4. API Retrieval Time in ms, BIM Data (Orange=MongoDB, Blue=PostGIS)

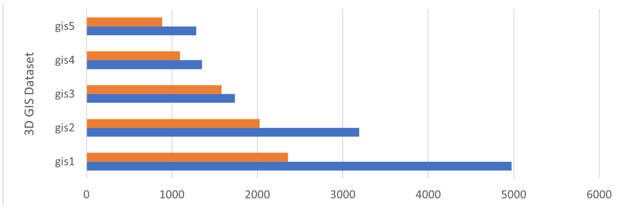

Figure 5. API Retrieval Times in ms, GIS Data (Orange=MongoDB, Blue=PostGIS)

For 3DGIS queries (Figure 5), both databases exhibited an increase in response time as the size of the data increases from gis5 to gis1. It was observed that MongoDB consistently outpaced PostGIS in all data sizes. The most noticeable performance difference was with the largest dataset (gis1), where MongoDB's response time was less than half of that for PostGIS.

5.5 Retrieving Data for Visualisation

Table 6 shows the results of data visualisation using CesiumJS, with Figures 6 and 7 showing partial visualisations - the ifc5 and gis5 datasets respectively.

In terms of performance evaluation, another set of 200 queries was executed via Cesium, with each database responsible for

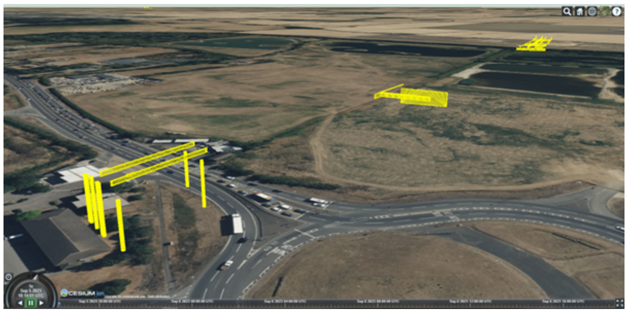

Figure 6. Partial BIM Dataset (ifc5) Displayed in Cesium

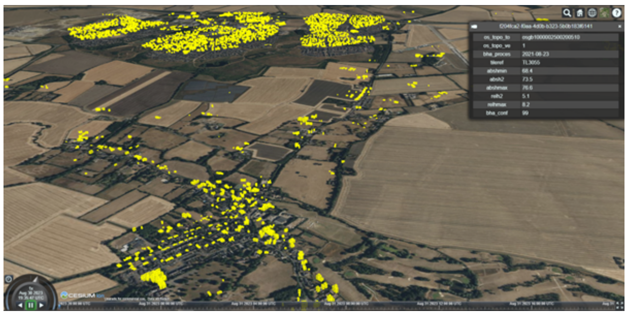

Figure 7. Partial GIS Dataset (gis5) Displayed in Cesium

Database	ifc1	ifc2	ifc3	ifc4	ifc5
MongoDB	N/A	N/A	17304.3	12335.8	11472.3
PostGIS	N/A	N/A	N/A	19987.5	17954.3
Database	gis	gis2	gis3	gis4	gis5
MongoDB	10600.1	9843.4	7649.9	7146.8	6269.1
PostGIS	14316.1A	13482.7	10372.7	10240.1	9090.3

Table 6. BIM and GIS Queries via Cesium (N/A means test was not completed

100 queries across the five varying sizes of BIM and 3DGIS data. Both PostGIS and MongoDB failed to visualise for the larger BIM datasets (ifc1 and ifc2). MongoDB was able to successfully handle the medium-sized dataset (ifc3), outperforming PostGIS which failed in this case as well. For the smaller datasets (ifc4 and ifc5), PostGIS and MongoDB managed to return results, albeit with considerably longer response times compared to previous experiments. MongoDB notably outperformed PostGIS in these instances, delivering quicker response times by some margin. For example, with the ifc4 dataset, MongoDB was approximately 7600 ms faster, and for the smallest dataset, ifc5, MongoDB was about 6500 ms quicker.

On the other hand for 3DGIS retrieval via Cesium, both PostGIS and MongoDB managed to process all five sizes of 3DGIS data, as presented in Table 6. Here, MongoDB consistently outperformed PostGIS across all dataset sizes. For the largest dataset (gis1).

6. Discussion

The aim of this research was to compare performance of NoSQL databases and relational databases for facilitating the incorporation of BIM and 3DGIS into a common environment. The study was approached in the context of a large construction firm's real-world data management issues, particularly the prolonged retrieval times for large 3D models that affect user interaction and decision-making.

MongoDB was selected due to its ability to handle diverse geospatial geometry types and greater spatial capabilities.

6.1 NoSQL Support for Spatial Data Storage

The selection of MongoDB mirrors many of the previous research studies have opted for MongoDB when comes to dealing with geospatial data tasks(Pietroń, 2019; Treviño Villalobos et al., 2020; Makris et al., 2021).

More broadly, the qualitative review revealed that most NoSQL databases offer limited support for geospatial data when compared to relational databases such as PostgreSQL/PostGIS. Popular NoSQL databases currently provide only basic features: basic spatial indexing with geohash, and a limited set of spatial operations. In contrast, mature spatial databases like PostGIS offer comprehensive geometry support, various coordinate systems, spatial analytics functions, and compliance with international standards. Additionally, the lack of support for a broader range of coordinate systems beyond WGS84 restricts the practicality of NoSQL, especially in areas where accuracy matters, such as construction. The limited spatial capabilities in NoSQL databases suggest that while developers recognise the need for spatial data management, functionality and the ability to support a wide range of geospatial data is not yet a top priority(Guo and Onstein, 2020).

6.2 Data Interoperability and Coordinate Transformation

The research used IFC files as provided by the construction firm. A significant challenge involved converting these 3D BIM models into intermediate GeoJSON/JSON formats before inserting them into the MongoDB database. Multiple approaches were employed to import the BIM data into MongoDB. During this process, it was observed that the geometry type converted from IFC to GeoJSON was in “MultiLineString” despite the original IFC models being in B-rep format. This resulted in a higher number of vertices for data stored in MongoDB compared to PostGIS (Table 5). These findings of data incompatibilities aligned with interoperability issues when converting BIM data, as reported by (Fosu et al., 2015; Noardo et al., 2020).

6.3 Performance Comparison

The three experiments conducted provide some empirical insights into the geospatial big data handling capabilities of MongoDB in comparison to the relational database, PostgreSQL. When it comes to direct queries from databases, MongoDB displayed promising results, particularly in retrieving large and complex BIM and 3DGIS data more efficiently than PostgreSQL. This highlights MongoDB's strength in using a flexible document schema to store 3D data in various formats, including GeoJSON (Višnjevac et al., 2017). However, MongoDB was comparatively slower when dealing with smaller data sizes, highlighting PostgreSQL's maturity in these scenarios. Thus, MongoDB's performance gains appear to be closely tied to the complexity and volume of the big data, which aligns with findings from (Pietroń, 2019).

In the experiment involving the web API endpoint, which aimed to replicate real-world spatial data retrieval, MongoDB delivered strong performance across the board for both BIM and 3DGIS datasets. However, both databases faced challenges when dealing with the largest BIM data (ifc1), revealing limitations as the complexity and sizes scaled, although it is yet to be determined if this was due to the hardware used for testing. Finally, on visualisation tests using Cesium, MongoDB once again demonstrated superiority over PostgreSQL for handling of both BIM and 3DGIS datasets across all data sizes. However, large data volumes remained a challenge for both databases.

Overall, while MongoDB holds potential for effectively managing extensive spatial data applications related to BIM and 3DGIS, it is worth noting that these advantages tend to diminish as complexity and data volume grows exponentially. It can also be noted that the while the hardware used for comparison is well above the minimum specification for both servers (MongoDB requires 4GB RAM, PostgreSQL requires 2GB) in practice this may also impact results.

6.4 NoSQL for Spatial Data Query

Due to their restricted geospatial analytical options, this study focussed on the storage and retrieval potential of NoSQL database, mirroring real-world situations through the use of both command line and visualisation queries. Our results mirror the studies in Section 2.5 highlight a consistent theme emerges – while NoSQL databases, especially MongoDB, exhibit merits in specific scenarios, relational databases like PostGIS generally fare better in complex geospatial tasks (which are not available directly from MongoDB without additional software development and/or use of external libraries).

6.5 Data Engineering Challenges for Construction Digital Twins and the Golden Thread

As noted in Section 2.1 one of the key challenges facing the built environment in the UK today is the concept of a 'golden thread' that is the record of the structure from cradle to grave. Conceptually, this requires all relevant data to be stored and shared in an open format, that can easily be migrated through the project's life cycle, and can be accessed by any software that wishes to make use of this data. To enable this reality, changes in the data should immediately be shared to all users in real time. This is fundamental for both construction monitoring and health and safety applications

To date, BIM data is stored and managed in proprietary formats, with IFC used as an exchange format. As noted, interoperability between IFC and GeoJSON is challenging, in particular due to geometry modelling approaches - boundary-representation versus constructive solid geometry (Donkers et al., 2016). Additionally, IFC only provides a static snapshot of the BIM model. This contrasts with the multi-user, real-time approach enabled by DBMS (Section 2.3).

A key assumption of the work also involved the use of GeoJSON, with a primary focus on geometry. This was driven by the fact that GeoJSON is the data format predominantly used in web mapping, and can be directly visualised in CesiumJS (GeoJSON Data Source, n.d.). It is also a *document* format and thus handled natively by MongoDB. As a document format, it has the advantage of providing directly embedded options for a single geometry to be linked to multiple data points - e.g. a sensor to many readings - a relationship which in a relational database has to be constructed via an expensive JOIN query. This may explain the performance issues encountered with PostGIS, as an additional JOIN and conversion task (using *ST_AsGeoJSON*) was required to generate GeoJSON. While GeoJSON - and JSON formats in general - can be stored in PostgreSQL (JSON Types in PostgreSQL, n.d.) this would degrade performance for spatial query functionality (as a conversion to geometry would be required).

It can also be noted that CesiumJS offers alternative options for visualisation of large, complex datasets - in particular 3D Tiles (Cesium 3D Tiles, n.d.). These offer a very performant option for visualisation, but have the disadvantage of being file based and thus not able to easily support the multi-user ad-hoc query (including real-time update of individual features as construction progresses) offered by a DBMS.

6.6 Recommendations for Future Work

The work presented here focussed on comparing the performance of a relational and NoSQL database when handling construction data, in order to enable Digital Twins and the construction golden thread. Future work should address the following:

- The experiments were conducted on a standalone computer system. Assessing distributed database architectures - and in particular the low cost, distributed, commodity hardware that NoSQL is designed to use (Section 2.3), or the relational database equivalent of partitioning (or even replication) - could provide insights into scalability.
- The data types selected reflect those required to provide the underpinning location framework - the static anchor data to which other data can be referenced. A more realistic test would involve combining both static and dynamic data - e.g. sensor data. Increasing dataset sizes, considering alternative storage formats other than GeoJSON and adding data such as LiDAR/laser scanning data (e.g. to support applications such as construction monitoring) would also provide useful evidence. Extending this further, NoSQL approaches to underpinning full digital twins (with two-way communication between physical and digital environments) would provide additional insights.
- The queries used for benchmarking related to basic retrieval and visualisation operations. A more comprehensive assessment that includes complex spatial functions, such as topological processing, could provide a more thorough understanding of the NoSQL databases' capabilities. Aligned to real world case studies, these would allow the construction partner to better understand the benefits and issues of using a NoSQL approach to provide a primary database for all location data storage. In particular, a list of spatial queries used in various construction applications - with a focus on the need for ad-hoc querying of data - would assist with this task.

7. Conclusion

This research explored options for integrated location data storage for construction applications, to underpin the development of a location framework for construction Digital Twins and more broadly provide an anchor point for a golden thread and single source of truth for the built environment. The potential of storing location data in a NoSQL database was demonstrated and some clear performance benefits were highlighted for data retrieval for visualisation. However, the challenges encountered highlighted that there is still some way to go before these end goals are reached. Proprietary data formats in BIM require extract/transform/load procedures to convert the data to a format where the ad-hoc querying required to enable multiple Digital Twins is possible. Coordinate system transformation issues were challenging and the lack of support for spatial analysis/queries in NoSQL databases means that the real-time, ad-hoc query environment and advanced spatial functionality of a relational database may still be preferable.

References

Ammar, A., Nassereddine, H., AbdulBaky, N., AbouKansour, A., Tannoury, J., Urban, H., Schranz, C., 2022. Digital twins in the construction industry: A perspective of practitioners and building authority. *Frontiers in Built Environment*, 8.

Anuradha, J. et al., 2015. A brief introduction on Big Data 5Vs characteristics and Hadoop technology. *Procedia computer science*, 48, 319–324.

Apache, 2014. Apache Cassandra. *Website. Available online at http://planetcassandra. org/what-is-apache-cassandra*, 13.

Asaad, C., Baïna, K., Ghogho, M., 2020. NoSQL databases: yearning for disambiguation. *arXiv preprint arXiv:2003.04074*.

Atzeni, P., Ceri, S., Paraboschi, S., Torlone, R., 1999. *Database Systems: Concepts, Languages & Architectures*. (McGraw Hill).

Baralis, E., Dalla Valle, A., Garza, P., Rossi, C., Scullino, F., 2017. SQL versus NoSQL databases for geospatial applications. *2017 IEEE International Conference on Big Data (Big Data)*, IEEE.

Bishr, Y., 1998. Overcoming the semantic and other barriers to GIS interoperability. *International journal of geographical information science*, 12(4), 299–314.

Boehm, J., Liu, K., 2015. NoSQL for storage and retrieval of large LiDAR data collections. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 40, 577–582.

Borrmann, A., Kolbe, T. H., Donaubauer, A., Steuer, H., Jubierre, J. R., Flurl, M., 2015. Multi-scale geometric-semantic modeling of shield tunnels for GIS and BIM applications. *Computer-Aided Civil and Infrastructure Engineering*, 30(4), 263–281.

Cesium 3D Tiles, n.d. https://cesium.com/why-cesium/3d-tiles/. Accessed: 28/01/2024.

CesiumJS, n.d. https://cesium.com/platform/cesiumjs/. Accessed: 28/01/2024.

Chang, F., Dean, J., Ghemawat, S., Hsieh, W. C., Wallach, D. A., Burrows, M., Chandra, T., Fikes, A., Gruber, R. E., 2008. Bigtable: A distributed storage system for structured data. *ACM Transactions on Computer Systems (TOCS)*, 26(2).

Dancuk, M., 2020. SQL vs NoSQL: What's the Difference? Accessed: 4/08/2023.

Date, C., 2004. An Introduction to Database Systems (Eighth Edition). *USA, Addison*.

DeCandia, G., Hastorun, D., Jampani, M., Kakulapati, G., Lakshman, A., Pilchin, A., Sivasubramanian, S., Vosshall, P., Vogels, W., 2007. Dynamo: Amazon's highly available key-value store. *ACM SIGOPS operating systems review*, 41(6).

Donkers, S., Ledoux, H., Zhao, J., Stoter, J., 2016. Automatic conversion of IFC datasets to geometrically and semantically correct CityGML LOD3 buildings. *Transactions in GIS*, 20(4), 547–569.

Edina DigiMap Service, n.d. https://digimap.edina.ac.uk/. Accessed: 28/01/2024.

FME By Safe Software, n.d. https://fme.safe.com/. Accessed: 28/01/2024.

Fosu, R., Suprabhas, K., Rathore, Z., Cory, C., 2015. Integration of building information modeling (bim) and geographic information systems (gis)–a literature review and future needs. *Proceedings of the 32nd CIB W78 Conference, Eindhoven, The Netherlands*, 27–29.

GeoJSON Data Source, n.d. https://cesium.com/learn/cesiumjs/ref-doc/GeoJsonDataSource.html. Accessed: 28/01/2024.

Geospatial Commission, 2023. UK Geospatial Strategy 2030 – Unlocking the Power of Location.

Gonçalves, H. C., Carniel, A. C., Vizinhos-PR-Brazil, D., Carlos-SP-Brazil, S., 2021. Spatial data handling in nosql databases: a user-centric view. *GeoInfo*, 167–178.

Greif, T., Stein, N., Flath, C. M., 2020. Peeking into the void: Digital twins for construction site logistics. *Computers in Industry*, 121.

Guo, D., Onstein, E., 2020. State-of-the-art geospatial information processing in NoSQL databases. *ISPRS International Journal of Geo-Information*, 9(5).

Guyo, E., Hartmann, T., Ungureanu, L., 2021. Interoperability between BIM and GIS through open data standards: An overview of current literature. *sign*, 3, 5–9.

HM Government, 2015. Digital Built Britain – Level 3 Building Information Modelling - Strategic Plan. Technical report, Department for Business, Energy Industrial Strategy.

Hong, S.-S., Lee, J., Chung, S., Kim, B., 2023. Fast Real-Time Data Process Analysis Based on NoSQL for IoT Pavement Quality Management Platform. *Applied Sciences*, 13(1).

IBM, n.d. SQL vs. NoSQL Databases: What's the Difference? . https://www.ibm.com/blog/sql-vs-nosql. Accessed: 5/08/2023.

Industry Foundation Classes - An Introduction, n.d. https://technical.buildingsmart.org/standards/ifc/. Accessed: 28/01/2024.

International Standards Organisation, 2018. Industry Foundation Classes - ISO 16739-1:2018. Technical report, Building Smart International.

JSON Types in PostgreSQL, n.d. https://www.postgresql.org/docs/current/datatype-json.html. Accessed: 28/01/2024.

Krstić, L. J., Krstić, M. S., 2018. Testing the performance of NoSQL databases via the database benchmark tool. *Vojnotehnički glasnik*, 66(3).

Kutzner, T., Smyth, C. S., Nagel, C., Coors, V., Vinasco-Alvarez, D., Ishimaru, N., Yao, Z., Heazel, C., Kolbe, T. H., 2023. OGC City Geography Markup Language (CityGML) Part 2: GML Encoding Standard. Technical report, Open Geospatial Consortium.

Lee, J.-G., Kang, M., 2015. Geospatial big data: challenges and opportunities. *Big Data Research*, 2(2), 74–81.

Liu, A., Ellul, C., 2022. Quantifying Geometric Changes in BIM-GIS Conversion. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 10, 185–192.

Makris, A., Tserpes, K., Spiliopoulos, G., Zissis, D., Anagnostopoulos, D., 2021. MongoDB Vs PostgreSQL: A comparative study on performance aspects. *GeoInformatica*, 25, 243–268.

Mao, B., Harrie, L., Cao, J., Wu, Z., Shenc, J., 2014. Nosql based 3d city model management system. *ISPRS Symposium on Geospatial databases and location based services, 2014*, The International Society for Photogrammetry and Remote Sensing.

Mastermap Topography Layer, n.d. https://www.ordnancesurvey.co.uk/products/os-mastermap-topography-layer. Accessed: 28/01/2024.

MongoDB, n.d. https://www.mongodb.com/. Accessed: 28/01/2024.

Mulquin, M., 2023. Interoperability and the Minimal Interoperability Mechanisms. *Personal Data-Smart Cities: How cities can Utilise their Citizen's Personal Data to Help them Become Climate Neutral*.

Munawar, H. S., Ullah, F., Qayyum, S., Shahzad, D., 2022. Big data in construction: current applications and future opportunities. *Big Data and Cognitive Computing*, 6(1), 18.

Nassif, E. H., Hicham, H., Yaagoubi, R., Badir, H., 2020. Assessing nosql approaches for spatial big data management. *Advanced Intelligent Systems for Sustainable Development (AI2SD'2019) Volume 6-Advanced Intelligent Systems for Networks and Systems*, Springer, 49–58.

National Infrastructure Commission, 2017. Data for the Public Good. https://www.nic.org.uk/app/uploads/Data-for-the-Public-Good-NIC-Report.pdf. Accessed: 23/08/2023.

Noardo, F., Ellul, C., Harrie, L., Overland, I., Shariat, M., Arroyo Ohori, K., Stoter, J., 2020. Opportunities and challenges for GeoBIM in Europe: developing a building permits use-case to raise awareness and examine technical interoperability challenges. *Journal of Spatial Science*, 65(2), 209–233.

NodeJS, n.d. https://nodejs.org/en/about. Accessed: 28/01/2024.

NoSQL, W., 2018. What Is NoSQL? *http://nosql-database. org*.

PGAdmin, n.d. https://www.pgadmin.org/. Accessed: 28/01/2024.

Pietroń, M., 2019. Analysis of performance of selected geospatial analyses implemented on the basis of relational and NoSQL databases. *Polish Cartographical Review*, 51(4).

PostGIS, n.d. https://postgis.net/. Accessed: 28/01/2024.

PostgreSQL, n.d. https://www.postgresql.org/. Accessed: 28/01/2024.

Roxin, A., Hbeich, E., 2019. Semantic interoperability between BIM and GIS–review of existing standards and depiction of a novel approach. *36th CIB W78–Information Technology for Construction*.

Sharma, M., Sharma, V. D., Bundele, M. M., 2018. Performance analysis of rdbms and no sql databases: Postgresql, mongodb and neo4j. *2018 3rd International Conference and Workshops on Recent Advances and Innovations in Engineering (ICRAIE)*, IEEE, 1–5.

Siddorn, J., Blair, G., Boot, D., Buck, J., Kingdon, A., Kloker, A., Kokkinaki, A., Moncoiffe, G., Blyth, E., Fry, M. et al., 2022. An information management framework for environmental digital twins (imfe). NOC.

Storing your building's information – the golden thread, n.d. https://www.hse.gov.uk/building-safety/golden-thread.htm. Accessed: 28/01/2024.

Thomas, E., Bowman, J., 2021. Harnessing the data advantage in construction.

Treviño Villalobos, M., Víquez Acuña, L., Quirós Oviedo, R., 2020. Comparison of the Response Times of MongoDB and PostgreSQL According to Type of Query in Geographical Databases. *Computación y Sistemas*, 24(4).

Višnjevac, N., Mihajlović, R., Šoškić, M., Cvijetinović, Ž., Bajat, B., 2017. Using NoSQL databases in the 3D cadastre domain. *Geodetski vestnik*, 61(3).