Department of Information & Communication Engineering
Graduate School of Information Science and Technology
THE UNIVERSITY OF TOKYO

Master's Thesis

A Scalable Spatial ID-Driven Infrastructure for Unified Urban Digital Twins and Spatial Computing

都市デジタルツインと空間コンピューティングの統合に向けた、
スケーラブルな空間 ID 駆動型基盤

Chen Yanru

Supervisor: Professor Hiroshi Esaki

January 2026

Abstract

The transition from static 3D city models to operational Urban Digital Twins is a primary objective of Japan's Society 5.0 strategy. A synchronization barrier currently prevents the real-time integration of archival geospatial layers with high-frequency IoT data. This research proposes a scalable infrastructure driven by Spatial IDs to bridge this gap. The system utilizes a natively 4D voxel-based framework as a universal join key for urban spatial computing.

This infrastructure addresses fundamental technical hurdles, such as inconsistent spatial referencing and the lack of real-time operational awareness and response. Moving intensive voxelization tasks to the network edge establishes a decentralized computing model. This architecture delivers sub-second response times for situational awareness. It significantly reduces the network bandwidth required for city-scale deployments.

A cloud-native distribution framework manages this intelligence through a geospatial database. This database delivers static urban structures via traditional api endpoints and optimized vector tiles. It broadcasts dynamic states through a real-time stream broker. Performance evaluations in realistic urban environments demonstrate that the system maintains interactive responsiveness. It also ensures high-accuracy semantic extraction. These features allow the digital twin to function as a live mirror of the physical environment.

A Mixed Reality platform validates the practical utility of this infrastructure. The platform enables geospatial anchoring. It also supports real-time visualization. Users interact with a standardized 3D grid to facilitate collaborative decision-making. This research establishes a scalable foundation for interoperable urban digital twins. It unifies static models and dynamic streams within a single machine-readable frame.

Ultimately, by establishing the Spatial ID as a universal join key, this research transforms the urban digital twin from a static, retrospective archive into an active, operational layer of the city's nervous system. The seamless integration of edge-cloud processing and immersive visualization provides a trustworthy backbone for future smart city services, where physical movements and digital insights coexist within a single, high-fidelity frame. This establishes continuity not only to resolve the immediate synchronization barrier but also to lay the groundwork for a standardized, cross-platform ecosystem where spatial intelligence is accessible, persistent, and actionable at a national scale.

概要

日本の Society 5.0 戦略において、静的な 3D 都市モデルから「運用型都市デジタルツイン」への移行は極めて重要な目標である。しかし、現状では「同期の壁」が存在し、アーカイブされた地理空間レイヤーと高頻度な IoT データのリアルタイムな統合が困難となっている。本研究では、この課題を解決するため、空間 ID を基盤とした拡張可能なインフラストラクチャを提案する。本システムは、都市空間コンピューティングのためのユニバーサルな結合キーとして、ネイティブな 4D ボクセルベースのフレームワークを活用する。

本インフラストラクチャは、空間参照の不整合や、リアルタイムな運用的状況把握・応答の欠如といった根本的な技術的障壁に対処する。計算負荷の高いボクセル化タスクをネットワークのエッジ側に移行することで、分散型計算モデルを構築した。このアーキテクチャにより、状況把握におけるサブ秒単位の応答速度を実現するとともに、都市規模での展開において課題となるネットワーク帯域消費を大幅に削減する。

クラウドネイティブな配信フレームワークは、地理空間データベースを介してこれらの情報を管理する。このデータベースは、従来の API エンドポイントや最適化されたベクトルタイルを通じて静的な都市構造を配信する一方で、リアルタイム・ストリームブローカーを通じて動的な状態を配信する。現実的な都市環境を用いた性能評価の結果、本システムはインタラクティブな応答性を維持し、高精度なセマンティック抽出を保証することが実証された。これにより、デジタルツインが物理環境の「ライブミラー」として機能することが可能となる。

さらに、複合現実（MR）プラットフォームを用いて、本インフラストラクチャの実用的な有用性を検証した。同プラットフォームは、地理空間アンカリングとリアルタイムな可視化を可能にし、ユーザーは標準化された 3D グリッドを介して対話することで、協調的な意思決定を促進できる。本研究は、相互運用可能な都市デジタルツインのための拡張可能な基盤を確立し、静的なモデルと動的なストリームを単一のマシン読み取り可能なフレームワーク内へと統合した。

結論として、空間 ID をユニバーサルな結合キーとして位置づける本研究は、都市デジタルツインを「静的な過去のアーカイブ」から「都市の神経系における能動的な運用レイヤー」へと変革するものである。エッジ・クラウド処理と没入型可視化のシームレスな統合は、物理的な動きとデジタルな洞察が単一の高忠実度フレーム内で共存する、次世代スマートシティサービスのための信頼すべきバックボーンを提供する。これは、目下の同期課題を解決するだけでなく、空間知能が全国規模でアクセス可能かつ持続的で、実行可能なものとなる、標準化されたクロスプラットフォーム・エコシステムの礎を築くものである。

Contents

Chapter 1 Introduction 1

Chapter 2 Requirements and Definition 3
2.1 Strategic Pillars for Urban Digital Transformation (DX) 3
2.1.1 High-Fidelity Spatial Abstraction 3
2.1.2 Real-Time Operational Awareness and Response 3
2.1.3 Computational Scalability for City-Scale Data 3
2.1.4 Cross-Domain Interoperability and Data Fusion 4
2.1.5 Automated Model Synchronization 4
2.2 System-Level Requirements 4

Chapter 3 Realted Work 5
3.1 Spatial Indexing and Discrete Global Grid Systems 5
3.1.1 Hierarchical 2D Indexing: Google S2 and Uber H3 6
3.1.2 The Volumetric Shift: VDB Family 6
3.2 Existing Research on 3D City Models 7
3.2.1 Semantic Modeling and CityGML Standards 7
3.2.2 CityGML 3.0 and the Dynamizer Module 8
3.3 Frameworks for Real-Time IoT Integration 8
3.4 Edge Computing-Driven Internet of Things(ECDriven-IoT) 10
3.4.1 Transition from Cloud-Centric to Edge-Centric Paradigms . . 10
3.4.2 Technical Advantages and Performance Metrics 11
3.4.3 Multi-Layer Taxonomy of ECDriven-IoT 12
3.4.4 Systemic Challenges and the Research Gap 12

Chapter 4 The Definition of Spatial ID 14
4.1 Concept of Spatial Voxels 14
4.2 Hierarchical Structure and Resolutions 15
4.3 Spatial Partitioning and Reference Planes 15
4.3.1 Horizontal Partitioning 15
4.3.2 Vertical Partitioning 15
4.4 Mathematical Formulation and Computation 16
4.4.1 Horizontal Indices (x, y) 16
4.4.2 Vertical Index (f) 16
4.5 Data Association and Attribute Management 17
4.6 Handling of Temporal Information 17

Chapter 5 System Architecture 19
5.1 Architectural Design Goals 19
5.2 Overview of the Five-Layer Infrastructure 20
5.3 Distributed Perception and Voxelization 21
5.3.1 Distributed Multi-Modal Sensing 21

5.3.2 Edge-Based Voxelization and Near-Field Computation 21
5.4 Data Fusion and Management 22
5.5 Spatial Computing and Visualization 22

Chapter 6 System Implementation and Evaluation 24
6.1 Prototype definition and Scope 24
6.2 City-Scale Spatial ID Generation 25
6.2.1 Data Ingestion and Extraction 25
6.2.2 Surface Voxelization Algorithm 26
6.3 IoT Stream Spatial ID Encoding at the Edge 27
6.3.1 Perception and Semantic Extraction 27
6.3.2 Coordinate Transformation 28
6.3.3 Spatial ID Encoding 29
6.4 Data Management & Distribution Framework 30
6.4.1 Geospatial Database 30
6.4.2 Static Asset Distribution: Vector Tiles and FastAPI 31
6.4.3 Dynamic Stream Brokerage: Node-RED and WebSockets . . 32
6.5 Performance Evaluation 33
6.5.1 Geospatial Database Query Performance 33
6.5.2 Vector Tile Distribution Efficiency 35
6.5.3 Real-Time IoT Stream Synchronization 35

Chapter 7 Use Case: Mixed Reality on Meta Quest 3 37
7.1 Infrastructure-Driven Geospatial Anchoring 37
7.2 Real-Time Visualization of Dynamic Urban Streams 38
7.3 Spatial Grid Interaction and the Voxel Paradigm 38
7.4 Summary of Practical Integration 39

Chapter 8 Conclusion and Future Work 40
8.1 Summary 40
8.2 Future Work 40

Publications 42

References 43

List of Figures

1.1 Illustration of Urban Digital Twin 1

3.1 Converting a polygon feature into a compressed grid structure via H3 hexagon-based (left) and (b) S2 square-based (right) hierarchies[1] 6
3.2 CityGML representation of the same building in the Levels of Detail 0-3. 8
3.3 Conceptual representation of Dynamizers, allowing (i) the representation of time-variant values from sensors, simulation-specific databases, and external files, (ii) enhancing the properties of city objects by overriding their static values [2] 9
3.4 Snap4City Architecture Overview [3] 10
3.5 Cloud, edge nodes, and edge devices. [4] 11
3.6 Taxonomy of edge-computing-driven IoT. [4] 12

4.1 Image of Spatial Voxels and Spatial ID at Low Zoom Level 14
4.2 Basic elements of the spatial partitioning scheme 15
4.3 Examples of using temporal information 17

5.1 Proposed Five-layer Architecture 20

6.1 Abstract the Physical Urban Environment into Two Primary Layers . . 24
6.2 CityGML to Spatial ID Pipeline 25
6.3 Generated Spatial ID overlays on a building model 27
6.4 Generated Spatial ID overlays on a building model 27
6.5 Perception and Visualization of Encoded Spatial ID 28
6.6 Camera Space to Tag Space 29
6.7 JSON Package 30
6.8 Data Management & Distribution Framework 31
6.9 Provided API Endpoints 32
6.10 Node-RED Setup Example for One IoT Stream 32
6.11 Spatial IDs of the Tokyo 23 Wards, Overlaying on Google 3D Tiles . . . 34
6.12 Query latency with varying areas. 34
6.13 Side-by-Side Comparison of the Physical Environment and the Digital Twin 36

7.1 High-level design overview of the system, illustrating how anchoring and visualization enables persistent and collaborative MR authoring of Spatial IDs. 37
7.2 Mixed Reality view showing real-time urban objects represented as color-coded Spatial ID voxels synced with physical counterparts. 38
7.3 A user performing MR authoring by selecting a specific voxel in the 3D Spatial ID grid using the stick-and-cursor paradigm. 39

List of Tables

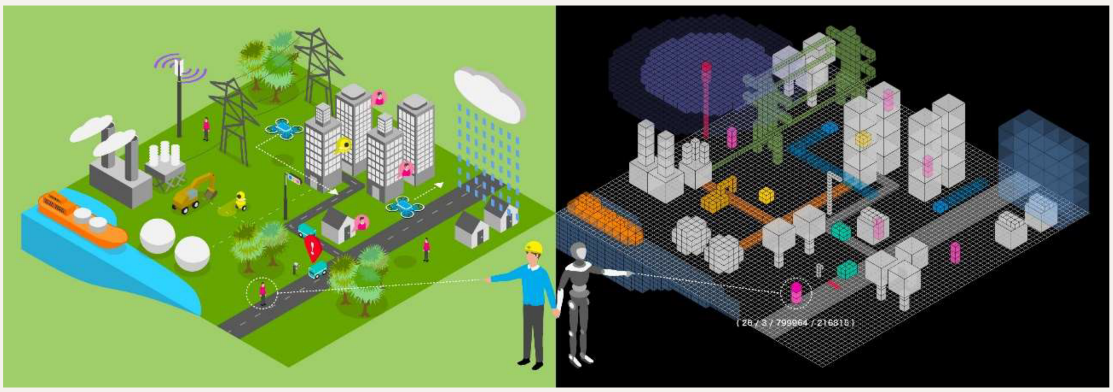
2.1 System Requirements and Pillar Alignment for Urban Digital Twin Infrastructure 4

6.1 Configurations 28
6.2 Configurations 33
6.3 Tile latency and payload size across zoom levels. 35

Chapter 1

Introduction

Urban digital twins(UDT) have transitioned from theoretical concepts to primary frameworks for global infrastructure management. These virtual representations integrate data from multiple sources in real time to support engineering analytics. At the World Economic Forum[5], various countries presented use cases involving digital twins. International standards related to digital twins are also being developed, including ISO/IEC 30173[6]; ISO 23247-1[7]; ISO/IEC/IEEE 24641[8]. Also, demand for digital twins, particularly those that use geospatial information, is rising to promote industrial digital transformation [9]. International initiatives, such as the United States' BuildZero[10], the Hong Kong Smart City Blueprint[11], and Singapore' s Smart Nation program[12], have demonstrated the large-scale utility of these twins for transportation and urban integration. In Japan, this global trend is mirrored by governmental platforms like PLATEAU MLIT Data Platform[13], which promote the open distribution of 3D city models and satellite imagery to foster public-private innovation. This transformation is a primary pillar of Japan' s Society 5.0 strategy, which envisions a highly integrated cyber-physical society capable of resolving complex urban challenges through data-driven simulation, illustrated in Fig 1.1. While technologies like these are gaining global attention, a persistent "Synchronization Barrier" currently prevents digital twins from evolving from static archival snapshots into real-time operational mirrors.



Physical Space Urban Digital Twin

Fig. 1.1. Illustration of Urban Digital Twin

This barrier arises from two fundamental challenges in the practical deployment of dynamic digital twins. Widespread inconsistencies in spatial referencing often force datasets to rely on incompatible local benchmarks. These technical gaps grow larger due to differences in positional accuracy across various data sources. Such factors cause spatial

misalignments, making it difficult to reliably compare objects within a unified digital environment.

Industry practitioners emphasize that bridging these gaps is critical for supporting remote accessibility. Rapid situational awareness is necessary during disaster response or infrastructure inspection. Existing asset management systems frequently depend on part numbers or textual references. These methods fail to provide the 3D spatial context needed to efficiently locate objects. Operators need an intuitive understanding of spatial relationships but typical office environments lack the computational resources to process large geospatial datasets at the necessary speeds.

Resolving these operational bottlenecks requires a departure from traditional 2D indexing. This research proposes a natively 3D volumetric backbone provided by Spatial IDs. This voxel-based referencing method partitions 3D space into uniform hierarchical units. The proposed infrastructure can integrate various referencing systems and positional inaccuracies into a single machine-readable index.

The infrastructure achieves sub-second response times for situational awareness by migrating computational tasks to localized edge nodes. These tasks include voxelization and object perception. This decentralized architecture performs near-field computation to reduce network bandwidth requirements. It effectively alleviates the data bottlenecks common in city-wide IoT deployments. The final realization of this infrastructure is demonstrated through an integrated Mixed Reality(MR) platform on the Meta Quest 3. This platform serves as a high-fidelity bridge between city-scale digital twins and human interaction.

The Spatial ID backbone enables spatial computing by supporting persistent authoring and real-time visualization. This functionality remains effective on hardware that lacks native positioning capabilities. The platform aligns static city models with dynamic IoT streams and user annotations within a single interoperable frame. This continuity from physical sensing to immersive interaction provides a scalable foundation for future smart city operations.

Chapter 2

Requirements and Definition

The advancement of UDT is a cornerstone of Japan's Society 5.0 strategy [14], which aims to integrate cyber and physical spaces to address complex social challenges. Central to this transformation is the i-Construction initiative, launched by the Ministry of Land, Infrastructure, Transport and Tourism (MLIT)[15]. This framework promotes the digitalization of infrastructure workflows by synchronizing field conditions with virtual models. It enhances maintenance efficiency and operational safety. However, a significant "Synchronization Barrier" currently prevents the seamless fusion of static geospatial layers with dynamic, high-frequency city operations.

2.1 Strategic Pillars for Urban Digital Transformation (DX)

To overcome this barrier and satisfy the objectives of national DX initiatives, this study defines five strategic pillars. These pillars represent the functional bridge between infrastructure management needs and technical implementation.

2.1.1 High-Fidelity Spatial Abstraction

Current asset management often relies on non-spatial textual databases or 2D parcel maps. Under the i-Construction framework, there is a requirement for Geospatial Object Referencing. This necessitates providing a 3D volumetric context for every urban element, allowing for precise spatial analysis rather than simple identification.

2.1.2 Real-Time Operational Awareness and Response

Static representations of urban environments are insufficient during emergencies or high-stakes operations, such as autonomous navigation. Digital twins must provide Rapid Situational Awareness by delivering real-time spatial overviews. This requires mirroring dynamic objects at high frequencies to support effective response and interference detection.

2.1.3 Computational Scalability for City-Scale Data

Processing city-scale datasets, such as 3D building models, imposes significant performance constraints on typical computing environments. A scalable architecture needs to leverage optimized indexing and multi-access edge computing(MEC) to ensure sub-second retrieval times and low latency, even when handling millions of data entries

2.1.4 Cross-Domain Interoperability and Data Fusion

The "Siloed Standards" problem persists because different domains use an inconsistent spatial reference system (SRS). A unified infrastructure must serve as a Universal Indexing Layer, integrating heterogeneous datasets—such as CityGML building models and live IoT camera streams—into a single, machine-readable framework.

2.1.5 Automated Model Synchronization

Manual updates to a digital twin are cost-prohibitive and prone to human error. There is a critical need for Automated Model Updating, in which the system uses continuous environmental data to self-correct and update the digital twin throughout its lifecycle.

2.2 System-Level Requirements

Based on the strategic pillars identified above, a set of specific system-level requirements has been established. These requirements serve as the evaluation criteria for the proposed architecture described in the subsequent chapters. The following table summarizes the technical requirements for achieving a scalable, unified urban digital twin.

Table 2.1. System Requirements and Pillar Alignment for Urban Digital Twin Infrastructure

ID	System Requirement	Description	Pillar Alignment
i	3D Volumetric Representation	Ability to represent the precise 3D shape, orientation, and position of every urban object.	Spatial Abstraction
ii	Multi-SRS Management	Ability to integrate data recorded across different SRS into a common frame.	Interoperability
iii	Positional Accuracy Fusion	Management of datasets with varying levels of precision and resolution.	Data Fusion
iv	High-Speed Querying	Provision of fast retrieval mechanisms for large-scale spatial and temporal data.	Scalability
v	Real-Time Stream Ingestion	Support for high-frequency data streams from IoT sensors and mobile agents.	Operational Awareness
vi	Modular Scalability	Capability to deploy and scale the infrastructure across an entire city-wide domain.	Scalability

The architecture proposed in this thesis is specifically engineered to satisfy these six requirements, ensuring it can support both the administrative needs of smart infrastructure and the low-latency demands of real-time geospatial applications.

Chapter 3

Realted Work

The transition from static, layer-based Geographic Information Systems (GIS) toward dynamic UDT represents a fundamental shift in how urban environments are monitored and managed [4]. Unlike traditional 3D models, which serve as archival snapshots, a UDT is defined by its ability to mirror the city's physical state in real time, facilitating predictive simulation and autonomous intervention. However, realizing such a "live mirror" at a city scale requires the convergence of multiple currently fragmented technical domains.

This chapter provides a comprehensive review of the existing literature and technological standards that form the landscape of UDT. Following the methodology established for infrastructure-level assessments, existing research is evaluated based on its capability to meet the six system-level requirements defined in Chapter 2, specifically focusing on 3D volumetric representation (Req i), multi-source reference integration (Req ii), and real-time synchronization (Req v).

The review is organized into four technical pillars:

1. **Spatial Indexing and Discrete Global Grid Systems (DGGS):** An analysis of the mathematical methods used to discretize the Earth's surface and the emerging shift toward volumetric structures for spatial AI [1].
2. **3D City Modeling:** An evaluation of semantic standards, primarily CityGML, and the challenges of moving from static archival models to dynamic, temporally aware systems [16].
3. **Real-Time IoT Integration Frameworks:** A review of current standards for fusing high-frequency sensor streams with 3D geometry and the identification of the persistent "Synchronization Barrier."
4. **Edge Computing-Driven IoT (ECDriven-IoT):** An exploration of decentralized computing as a solution for the bandwidth and latency bottlenecks inherent in massive-scale urban sensing [4].

3.1 Spatial Indexing and Discrete Global Grid Systems

The efficacy of any geospatial system is predicated on its indexing strategy—the mathematical method by which continuous space is discretized into addressable units. In the context of UDT, the index must support not only the retrieval of static assets but also the dynamic querying of moving agents and volumetric phenomena. Current literature reveals a stark dichotomy between industry-standard Discrete Global Grid Systems[17], which are optimized for surface analysis, and emerging sparse volumetric data structures required for spatial AI.

3.1.1 Hierarchical 2D Indexing: Google S2 and Uber H3

DGGS are hierarchical data structures that discretize the Earth's surface by tessellating regular polygons. These systems offer a discrete method for spatial referencing, using fixed-cell units rather than coordinates derived from an underlying continuous space [1]. Two of the most prevalent implementations are Google' s S2[18] and Uber' s H3 [19].

Google S2: This system projects the Earth' s sphere onto a circumscribed cube and subdivides it using a hierarchical partition of squares with an aperture of four. It utilizes a Hilbert space-filling curve to assign unique indices to each cell. While square-based grids like S2 are congruent—meaning they can be tessellated into self-similar shapes that fit perfectly within parent boundaries—they possess non-uniform adjacency. This requires complex algorithms for neighbor-finding and directional bias compensation in dynamic urban simulations. [20].

Uber H3: Utilizing an icosahedral projection, H3 partitions the sphere into hexagonal cells. Hexagons are favored for mobility analysis because they possess uniform adjacency; all six neighboring cells are equidistant from the central cell. H3 specifically integrates an aperture-7 grid partitioning, which facilitates easier navigation through the hierarchy. However, hexagons are non-congruent and cannot be refined into self-similar shapes without overlap, which can reduce the precision of the area a cell covers. [21].



Fig. 3.1. Converting a polygon feature into a compressed grid structure via H3 hexagon-based (left) and (b) S2 square-based (right) hierarchies[1]

The fundamental deficiency of both S2 and H3 is their inherent 2D surface-index design, as shown in Fig 3.1. To handle the vertical dimension, developers often resort to "2.5D" approaches, such as extruding 2D cells into prismatoids. These methods fail to represent complex 3D topologies, such as tunnels or overhanging buildings, and suffer from volumetric variability, where a "voxel" near the equator is significantly larger than one at the poles.

The primary advantage of these systems lies in their ability to aggregate or distribute data from multiple sources with arbitrary positional precision, directly supporting Requirement (iii) for managing datasets with different positional accuracies. Furthermore, they introduce methods for efficient storage and rapid access, fulfilling the need for fast retrieval Requirement (iv).

3.1.2 The Volumetric Shift: VDB Family

Despite their global scalability, both S2 and H3 suffer from a fundamental deficiency: their design as surface indices that primarily cover the Earth's 2D manifold. To address the

verticality gap, research has shifted toward sparse volumetric data structures, specifically the Volumetric Data Blocks (VDB) family. The VDB family has emerged as the standard for representing sparse 3D data, facilitating a transition toward true spatial intelligence. This shift is essential for satisfying Requirement (i): the ability to represent the precise 3D shape and position of urban objects beyond simple 2.5D extrusions.

- **OpenVDB: Infinite Sparse Domains:** OpenVDB utilizes a hierarchical B+ tree structure to discretize 3D space into voxels [22]. The primary innovation of this framework is its sparsity: it allocates memory only to voxels that contain matter, leaving the vast majority of "empty" urban space unrepresented. This property allows OpenVDB to manage virtually infinite domains with high geometric fidelity, making it suitable for storing city-scale LiDAR point clouds or signed distance fields (SDFs). However, its reliance on CPU-based pointer chasing introduces latency that can be a bottleneck for real-time digital twin synchronization.
- **NanoVDB: GPU-Accelerated Volumetrics：** NanoVDB addresses the latency issues of the original OpenVDB by linearizing the pointer-based tree structure into a pointer-less, buffer-compatible format [23]. This architecture is specifically optimized for GPU processing, enabling the execution of ray-tracing, collision detection, and volumetric fluid simulation at interactive frame rates. NanoVDB serves as a critical transport layer, moving massive datasets—such as real-time 3D scans of construction sites—directly to the GPU for immediate processing within a digital twin environment.
- **fVDB: Native Neural Spatial Intelligence：** The most recent advancement in this domain is fVDB (Feature VDB) [24], a deep learning framework built on the NanoVDB substrate. fVDB introduces differentiable operators directly into the sparse voxel data structure, allowing neural networks to operate natively on the grid. Unlike traditional pipelines that project 3D data into 2D images or dense 3D tensors, fVDB supports sparse convolution, pooling, and attention mechanisms natively within the voxel grid. This framework is a primary enabler for "Reality Scale" Digital Twins, enabling the continuous ingestion of sparse sensor data to update neural representations, such as Neural Radiance Fields (NeRF), in real time.

By leveraging these sparse volumetric structures, the digital twin can represent complex topologies, including tunnels, overhanging structures, and subterranean infrastructure, mathematically inaccessible to 2D-centric systems. This ensures the high-fidelity spatial abstraction required for automated urban monitoring and autonomous navigation.

3.2 Existing Research on 3D City Models

While spatial indexing addresses the discretization of space, semantic city models define the structural and functional meaning of urban objects. The evolution of 3D city modeling is central to transforming static digital representations into dynamic UDT that can mirror physical reality. Existing research on 3D city models has proposed a digital twin framework for civil infrastructure management [25], as well as guidelines and processes for building digital twins using public data based on CityGML [16] [26].

3.2.1 Semantic Modeling and CityGML Standards

The CityGML standard, managed by the Open Geospatial Consortium (OGC), has emerged as the primary international standard for the representation and exchange of

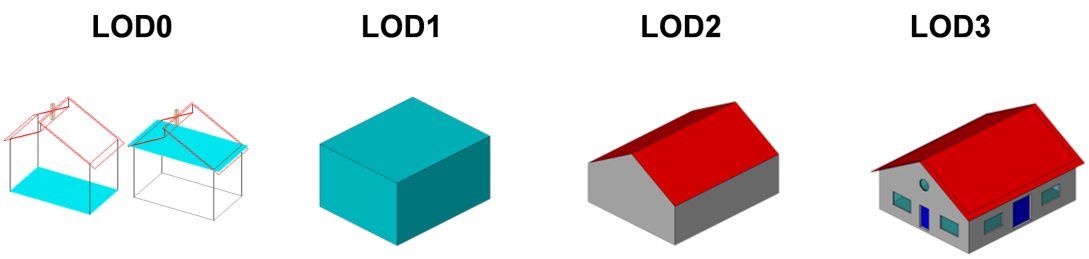


Fig. 3.2. CityGML representation of the same building in the Levels of Detail 0-3.

semantic 3D city models [27]. Unlike purely geometric formats, CityGML combines 3D geometry with a rich semantic hierarchy, in which urban elements are classified into categories such as buildings, bridges, and tunnels.

The CityGML 2.0 version was primarily designed for the exchange of static 3D city objects. It has been widely adopted in national initiatives, most notably Japan's PLATEAU project, which has developed semantically rich models for numerous cities to support urban planning and disaster management. CityGML 2.0 is fundamentally limited by its focus on "archival" snapshots of the built environment. To represent time-variant data—such as energy consumption or real-time traffic flow—developers were historically forced to use non-standard Application Domain Extensions (ADEs) or external file links, leading to fragmented implementations that lack interoperability.

3.2.2 CityGML 3.0 and the Dynamizer Module

The transition to CityGML 3.0 represents a significant architectural shift from static geometric snapshots toward dynamic, temporally aware urban systems [28]. The most critical advancement in this version is the introduction of the Dynamizer module, which provides a standardized mechanism to inject time-series data into static city objects. Instead of the computationally prohibitive process of updating geometry files whenever a sensor value changes, a Dynamizer object serves as a link between a specific attribute of a static feature and a dynamic data source, such as a time-series database or an OGC SensorThings API stream. This allows the underlying geometry to remain stable while the semantic attributes reflect real-time conditions.

Beyond real-time data ingestion, CityGML 3.0 introduces bitemporal versioning, allowing the digital twin to track the historical lifecycle of urban objects from construction to demolition. This is complemented by a revised "Space" concept that permits a more abstract definition of volumes, such as distinguishing legal boundaries in a 3D cadastre from physical architectural boundaries. These enhancements collectively move the standard closer to the requirements of a multi-dimensional urban digital twin, though they also introduce significant complexity into the underlying data management layers.

3.3 Frameworks for Real-Time IoT Integration

The successful realization of an urban digital twin depends on the seamless integration of the physical world, captured through IoT sensor networks, with the virtual world, represented by semantic 3D geometry [29]. To facilitate this integration, several standardized frameworks have been developed to manage the high-frequency data streams generated by city-scale deployments. The OGC SensorThings API (STA) serves as a critical semantic bridge in this domain [30] [31], providing an open, geospatial-enabled standard for inter-

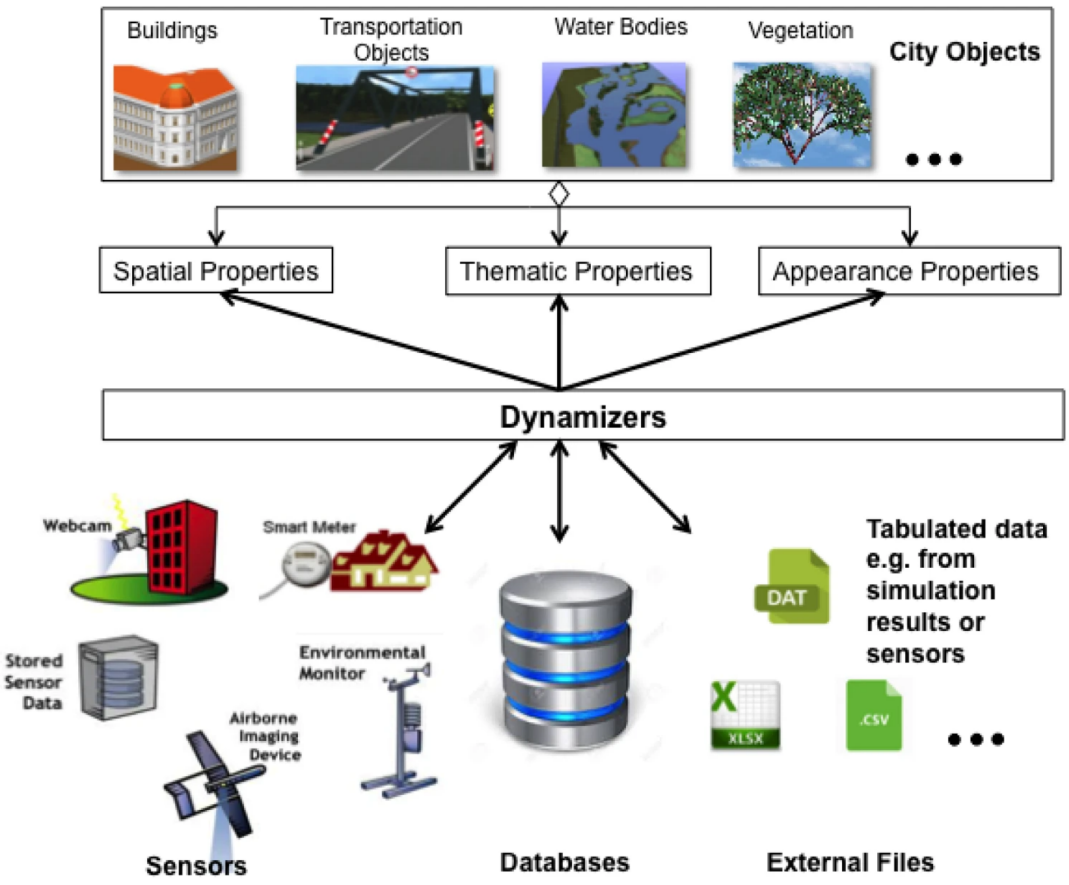


Fig. 3.3. Conceptual representation of Dynamizers, allowing (i) the representation of time-variant values from sensors, simulation-specific databases, and external files, (ii) enhancing the properties of city objects by overriding their static values [2]

connecting IoT devices and applications over the web. This API utilizes a data model based on the ISO/OGC Observations and Measurements standard, which organizes the IoT ecosystem into discrete entities such as Things, Datastreams, and Observations. In contemporary UDT architectures, researchers propose mapping CityGML CityObjects to SensorThings, effectively creating a linked-data bridge in which the CityGML Dynamizer module references the URIs of real-time SensorThings data streams [32] [33].

Beyond localized sensor management, holistic aggregation platforms such as Snap4City(Fig 3.4) [3] have been highlighted in recent literature as robust solutions for managing heterogeneous urban data sources. These platforms typically employ a knowledge base driven by RDF or SPARQL to map sensor data to specific city entities, facilitating complex inferential queries that can identify relationships between environmental conditions and urban infrastructure. By indexing both static 3D building models and real-time IoT entities, these frameworks provide the necessary substrate for "what-if" analysis and executive decision support. However, despite these advancements, a significant "Synchronization Barrier" remains a primary obstacle to the deployment of truly responsive digital twins.

This barrier arises from a deep-seated mismatch between the continuous, streaming nature of IoT data and the discrete, object-oriented nature of 3D modeling standards.

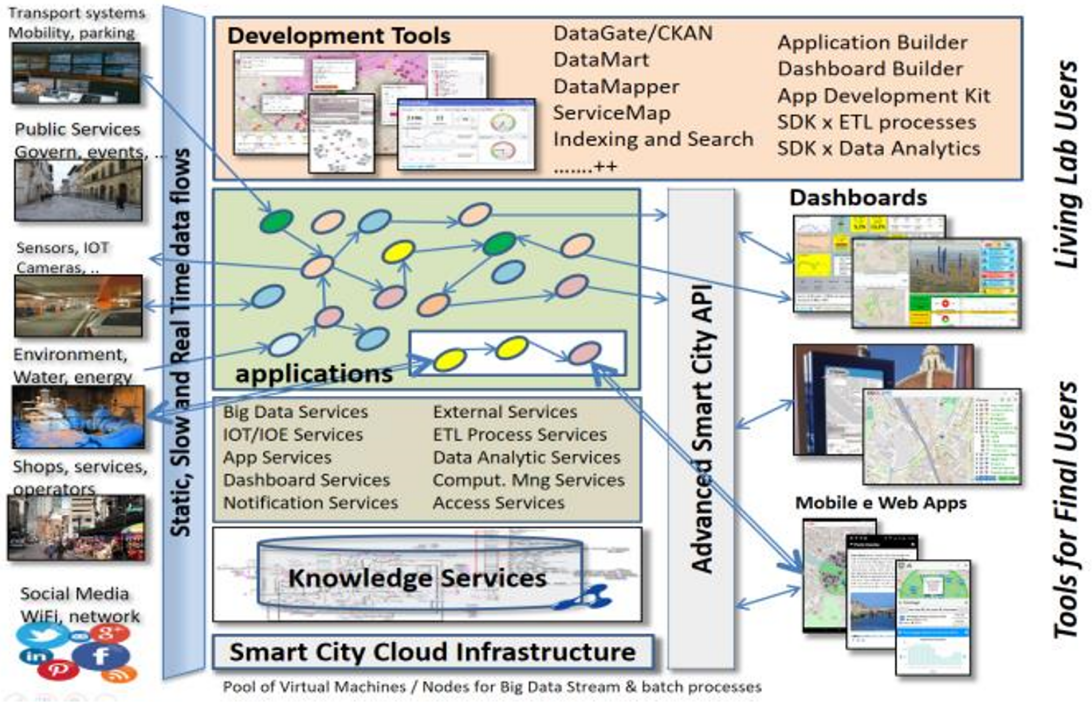


Fig. 3.4. Snap4City Architecture Overview [3]

IoT data arrives effectively as a continuous signal, whereas CityGML objects represent discrete states within a database. When high-frequency streams are integrated with a 3D model that is updated at a significantly lower frequency, a temporal misalignment occurs. This creates an "aliasing" or "stutter" effect where the digital twin mirrors a retrospective snapshot rather than a live state, lagging behind the physical world by a variable delta. Furthermore, semantic ambiguity persists in defining what constitutes a Thing across different standards. Reconciling these ontologies at scale, such as the automatic discovery and semantic placement of millions of sensors within a 3D CityGML hierarchy, remains an unresolved research challenge that often necessitates labor-intensive manual mapping.

3.4 Edge Computing-Driven Internet of Things(ECDriven-IoT)

The architectural paradigm of the Smart City is transitioning from centralized cloud-based systems to decentralized edge-based infrastructures to accommodate the explosive growth of the IoT, which reached 18.5 billion in 2024 [34]. While the traditional cloud-based computing model provides significant centralized resources, it faces non-trivial challenges regarding network bandwidth and unacceptable latency when managing massive-scale urban data. Edge computing addresses these limitations by migrating computational data, applications, and services from remote cloud servers to the network edge, enabling processing in geographic proximity to the data sources [4].

3.4.1 Transition from Cloud-Centric to Edge-Centric Paradigms

In a cloud-based architecture, the transfer of large amounts of IoT data for management and analysis places a heavy workload on central servers and often results in bottlenecks in data transmission rates. Edge computing is characterized by high bandwidth, ultra-low

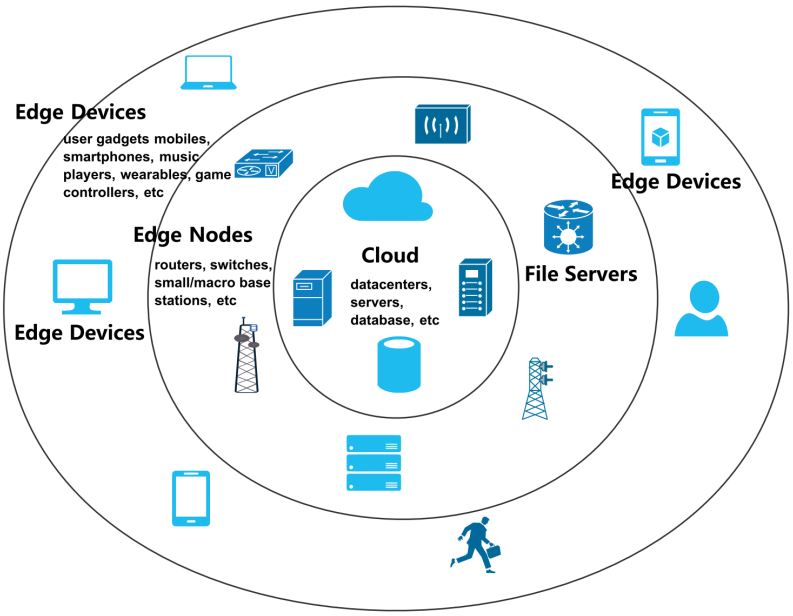


Fig. 3.5. Cloud, edge nodes, and edge devices. [4]

latency, and real-time access to network information, making it inherently better suited to the requirements of large-scale IoT applications such as smart cities and autonomous driving.

Previous surveys [35][36][37] on the integration of IoT and edge computing have discussed specialized areas, including distributed architectures, real-time resource management, and various security and privacy vulnerabilities. However, as noted in recent literature, these surveys often cover limited research areas and frequently fail to explore the new challenges arising specifically from the combination of these two technologies.

3.4.2 Technical Advantages and Performance Metrics

The shift toward an ECDriven-IoT model brings several critical advantages to urban infrastructure management:

- **Real-time Response and Quality of Service:** By placing edge servers closer to IoT devices, network latency is significantly reduced compared to cloud computing. This superiority supports high-demand real-time applications and reduces the amount of data offloaded to the cloud.
- **Low Energy Consumption:** Most IoT nodes are power-limited devices. Synchronizing large amounts of data to a remote cloud wastes power, whereas sending data to local edge servers decreases the energy consumption of these nodes, thereby extending their operational lifetime.
- **High Scalability:** While cloud servers can become system bottlenecks due to massive concurrent connections, edge servers provide moderate computing resources in a distributed manner. This distributed infrastructure supports the large-scale access requirements of dense urban environments.

3.4.3 Multi-Layer Taxonomy of ECDriven-IoT

Research into ECDriven-IoT systems typically categorizes the field into six distinct layers that work together to form a complete system, illustrated in Fig 3.6:

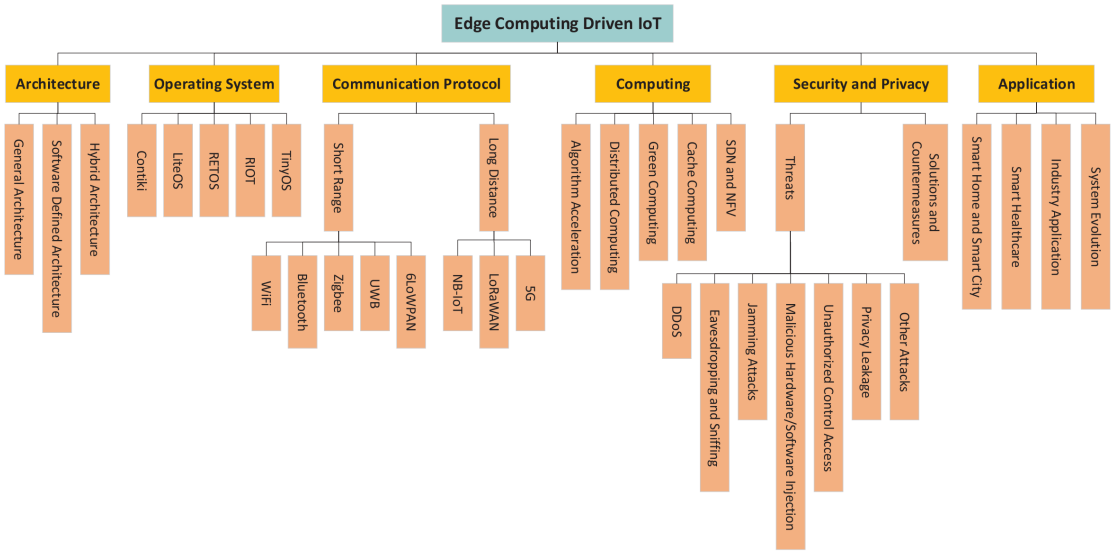


Fig. 3.6. Taxonomy of edge-computing-driven IoT. [4]

1. **Hardware Architecture:** This lowest layer includes terminal things and edge network devices, ranging from general and mobile architectures to scalable hierarchical structures suitable for different deployment scenarios.
2. **Operating Systems:** IoT operating systems, such as LiteOS, RIOT, and TinyOS, serve as bridges across diverse hardware platforms, providing unified programming interfaces while meeting real-time, high-reliability, and energy-saving requirements.
3. **Communication Protocols:** This layer manages both short-range protocols and long-distance protocols. 5G technology, in particular, is capable of supporting synchronization down to the nanosecond level with high reliability.
4. **Computing Paradigms:** This includes computation offloading, distributed computing, and caching strategies to alleviate backhaul traffic and reduce system delay.
5. **Security and Privacy:** This layer addresses threats such as DDoS attacks, eavesdropping, and privacy breaches through countermeasures such as cryptographic primitives, blockchain technologies, and trusted execution environments.
6. **Applications:** The top layer implements these technologies in practical scenarios, including smart healthcare, industrial automation, and smart home systems.

3.4.4 Systemic Challenges and the Research Gap

Despite the benefits, several new challenges emerge in ECDriven-IoT systems. First, the heterogeneity of hardware and communication protocols across different scenarios makes unifying the diversity of IoT and edge computing difficult. Second, there is a critical need for coordination between communication and computation; since edge servers have limited power and capacity, the system must efficiently allocate workloads to balance these

costs. Finally, security and privacy issues become more complex due to the architecture's distributed nature and the limited computing capabilities of edge nodes.

The prevailing research highlights a significant research gap: while individual technologies have matured, the "Synchronization Barrier" remains. This barrier stems from the temporal and semantic mismatch between the continuous, streaming nature of IoT data and the discrete, object-oriented nature of 3D modeling standards. Existing infrastructures often fail to provide a unified, reality-scale digital mirror that can handle 4D (spatio-temporal) complexity in real-time. This necessitates the development of a unified 3D-native indexing backbone, such as the Spatial ID, to bridge the gap between edge-processed sensor data and static semantic urban models.

Chapter 4

The Definition of Spatial ID

To achieve the "Unified Referencing" and "Cross-Domain Interoperability" required for a city-scale digital twin, this research adopts the Spatial ID standard. This chapter provides the formal definition of Spatial IDs [38][39], the underlying concept of spatial voxels, and the mathematical framework for their computation.

4.1 Concept of Spatial Voxels

The foundation of the Spatial ID system is the "spatial voxel," a discrete unit of three-dimensional space. A spatial voxel is a rectangular parallelepiped (rectangular grid) that partitions the environment into uniquely identifiable volumes. This system is based on the concept of Discrete Global Grid Systems (DGGS), which provides a common framework for distributing spatial attribute information regardless of the original data format.

Unlike traditional point-level coordinate systems, spatial voxels define a volumetric region. This allows the infrastructure to abstract complex geographic features into a unified standard that is easily searchable and integrable. Because spatial voxels do not overlap at the same zoom level, they provide an unambiguous indexing mechanism for the entire Earth, including the air, ground, and underground.

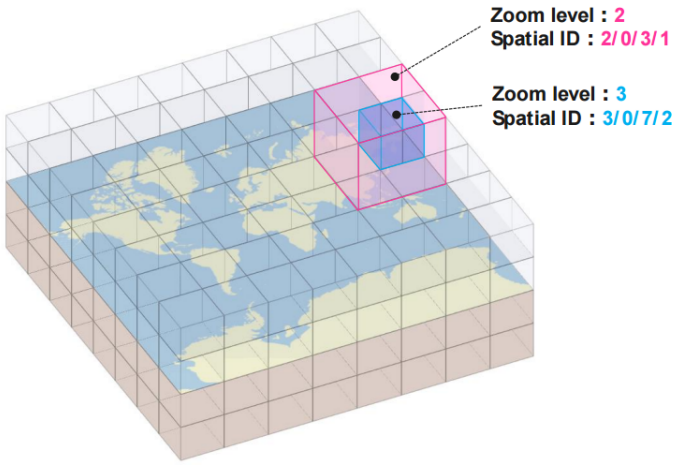


Fig. 4.1. Image of Spatial Voxels and Spatial ID at Low Zoom Level

4.2 Hierarchical Structure and Resolutions

The Spatial ID system is inherently hierarchical. The division process begins at zoom level 0, which covers the entire Earth's target area, as shown in Fig 4.1. Each time the zoom level increases by 1, the spatial voxel is divided into 8 segments (2 divisions along each of the X, Y, and Z axes). This recursive partitioning creates a parent-child relationship between levels, enabling flexible resolution management.

The system's granularity can range from a global overview to highly precise urban details. For instance, at zoom level 25, a voxel in Tokyo represents a volume of approximately 1 m^3 (0.97 m horizontally and 1.00 m vertically). This multi-resolution capability allows the system to support diverse use cases, ranging from city-wide demographics at low zoom levels to millimeter-level infrastructure inspection at high zoom levels.

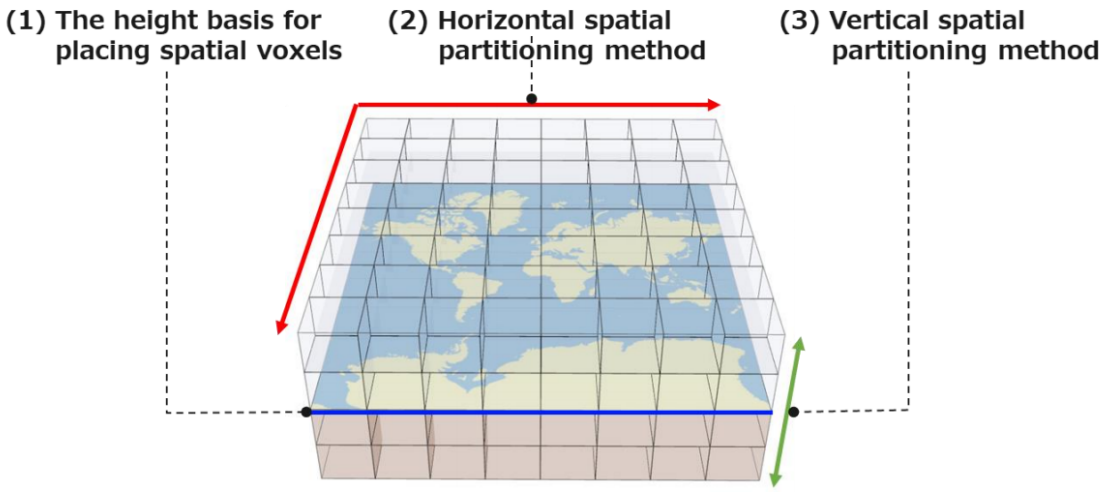


Fig. 4.2. Basic elements of the spatial partitioning scheme

4.3 Spatial Partitioning and Reference Planes

The Spatial ID partitions space using established geodetic and projection standards to ensure global consistency.

4.3.1 Horizontal Partitioning

The horizontal grid is based on the Web Mercator projection, partitioning the Earth between 180 degrees west and east longitude, and approximately 85.05 degrees north and south latitude. This partitioning method is identical to the XYZ tiles used by most web map services, ensuring high system compatibility. The standard geodetic systems for these calculations are WGS84.

4.3.2 Vertical Partitioning

The vertical partitioning uses orthometric elevation as its basis. The reference plane is the geoid, which represents the mean sea level. In Japan, the "Japan Geoid 2011"[40] is

used as the standard height reference. The system partitions both positive and negative height ranges, covering elevations from -33,554,432 m to +33,554,432 m.

4.4 Mathematical Formulation and Computation

A Spatial ID is computed deterministically from geographic coordinates (longitude, latitude, elevation) and a target zoom level (z). The ID is typically represented as a structured string in the format $\{z\}/\{f\}/\{x\}/\{y\}$.

4.4.1 Horizontal Indices (x, y)

The horizontal indices represent the grid position in the east-west (x) and north-south (y) directions. Given longitude (lng) and latitude in radians (lat_{rad}), the indices at zoom level z are calculated as follows:

lng	Longitude in decimal degrees
lat_{rad}	Latitude in radians
h	Elevation in meters
z	Current zoom level
Z	Reference zoom level (typically $Z = 25$, where voxel height $\approx 1\,\text{m}$)

1. Define the partition factor n:
$$n = 2^z \tag{4.1}$$
2. Calculate the x-index (longitude):
$$x = \text{floor}\left(n \times \frac{lng + 180}{360}\right) \tag{4.2}$$
3. Calculate the y-index (latitude):
$$y = \text{floor}\left(\frac{n}{2} \times \left(1 - \frac{\log\left(\tan(lat_{rad}) + \frac{1}{\cos(lat_{rad})}\right)}{\pi}\right)\right) \tag{4.3}$$

4.4.2 Vertical Index (f)

The vertical index (f) represents the elevation or floor number. To ensure consistency across resolutions, the vertical partitioning uses a maximum height factor H based on the reference zoom level where voxel height equals 1 meter ($Z = 25$):

1. Define the reference vertical range H:
$$H = 2^{25} \quad \text{(Reference vertical range in meters)} \tag{4.4}$$
2. Calculate the f-index:
$$f = \text{floor}\left(n \times \frac{h}{H}\right) \tag{4.5}$$

Positive elevation values result in $f \geq 0$, while negative elevations (underground or undersea) result in $f < 0$. For data without height information, a 2D format $\{z\}/\{x\}/\{y\}$ is applied, maintaining compatibility with standard web map tile specifications.

4.5 Data Association and Attribute Management

Spatial ID serves as the "common key" for data integration by acting as a universal index that relates geographic features to the specific volumetric regions they occupy. This framework defines a "spatial voxel" as the fundamental unit for distributing "Spatial Attribute Information"—data that represents real-world properties for a specific purpose—independent of its original data format. By abstracting information into these standardized units, the infrastructure enables rapid retrieval, cross-domain fusion, and a comprehensive understanding of the general situation within a given space.

The system manages the relationship between spatial voxels and features through three primary cardinality models to ensure flexible data association:

1. One-to-One (1:1) Relationship: This occurs when a single spatial voxel contains exactly one feature. In this scenario, the feature's attribute information is directly related to the spatial voxel, and the corresponding Spatial ID is assigned as a property of that feature' s attributes.
2. One-to-Many (1:N) Relationship: This model is applied when a single spatial voxel encompasses multiple distinct features. Each individual feature's attribute information is linked to the voxel, and a unique Spatial ID is assigned to every feature within that volume. Furthermore, this relationship allows the system to generate statistical metadata—such as counts, averages, and maximum/minimum values—for all features within a voxel, providing a high-level overview of the space.
3. Many-to-One (N:1) Relationship: When a large geographic feature overlaps multiple spatial voxels, the same attribute information is associated with every voxel it occupies. Simultaneously, the feature's attributes are assigned a list or array of all Spatial IDs it intersects. This ensures that a query for any individual voxel within the object' s footprint will successfully retrieve the full semantic context of the entire feature.

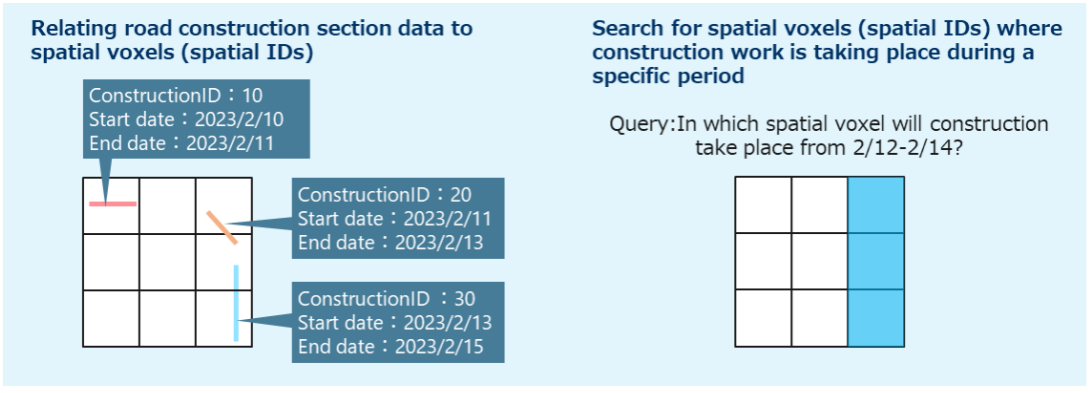


Fig. 4.3. Examples of using temporal information

4.6 Handling of Temporal Information

To realize a complete "4D" spatiotemporal ID, the system incorporates temporal identifiers along the time axis. By relating temporal information (year, month, day, and time)

to the three-dimensional Spatial ID, the infrastructure can manage and search city-scale data across both spatial and temporal dimensions simultaneously. Temporal attributes are categorized into two primary types:

1. Instant Information: This captures the state or predicted situation of a feature at a specific point in time. Typical examples include real-time meteorological conditions or sensor readings from IoT devices.
2. Period Information: This defines the validity or existence of a feature between a specific start and end time. This type of information is critical for managing transient urban events, such as road construction schedules or the temporary deployment of equipment in construction sites.

Using these mechanisms, users can perform complex queries, such as identifying all spatial voxels where construction work is taking place during a specific timeframe, thereby facilitating proactive urban management and autonomous navigation

Chapter 5

System Architecture

The proposed infrastructure is engineered to resolve the "dimensional schism" and "synchronization barrier" that currently limit the utility of UDT. This chapter details a unified, 4D voxel-based architectural framework that utilizes Spatial IDs as a universal indexing key within an edge-cloud continuum.

5.1 Architectural Design Goals

The central objective is to transition from static, layer-based geospatial systems to a "live" digital mirror capable of high-frequency synchronization. This transition is a prerequisite for the realization of Japan's Society 5.0 vision. To address the six core requirements defined in Chapter 2, the architecture shifts from a traditional cloud-centric processing model to an edge-cloud hybrid model, specifically designed to eliminate latency and bandwidth bottlenecks.

The design is governed by the following fundamental principles:

- **Volumetric Consistency (Requirement i):** Moving beyond 2D surface representations, the architecture employs a true 3D volumetric index. This ensures high-fidelity abstraction for complex urban topologies, such as multi-level interchanges, tunnels, and overhanging structures, that are often misrepresented in 2D-centric mapping.
- **Unified Reference Management (Requirement ii, iii):** The system utilizes the Spatial ID as a Universal Indexing Layer. This allows for the seamless fusion of heterogeneous datasets—including static CityGML building models and dynamic IoT sensor streams—into a single machine-readable framework, regardless of their native spatial referencing systems or varying positional accuracies.
- **Low-Latency Synchronization (Requirement v):** To ensure the digital twin reflects physical reality with sub-second latency, the architecture migrates critical computing tasks from remote cloud servers to local edge nodes. This proximity provides the rapid response times necessary for real-time operational awareness.
- **High Scalability and Flexibility (Requirement vi):** By distributing computing resources across edge nodes, the infrastructure overcomes the "cloud bottleneck" typical of city-scale deployments. This Edge-Cloud Driven IoT approach reduces long-haul bandwidth consumption by performing data preprocessing and spatial filtering near the data source.

5.2 Overview of the Five-Layer Infrastructure

To achieve automated synchronization and real-time operational awareness, the architecture is structured into a five-layer stack. This hierarchy facilitates the transformation of raw physical signals into a synchronized 4D digital environment, using the Spatial ID as the "Universal Join-Key" at every stage.As illustrated in Fig 5.1, this layered organization ensures that high-frequency updates from the physical world are processed efficiently while maintaining the integrity of the semantic city model. The following sections will detail the specific functions and components of each layer within this stack.

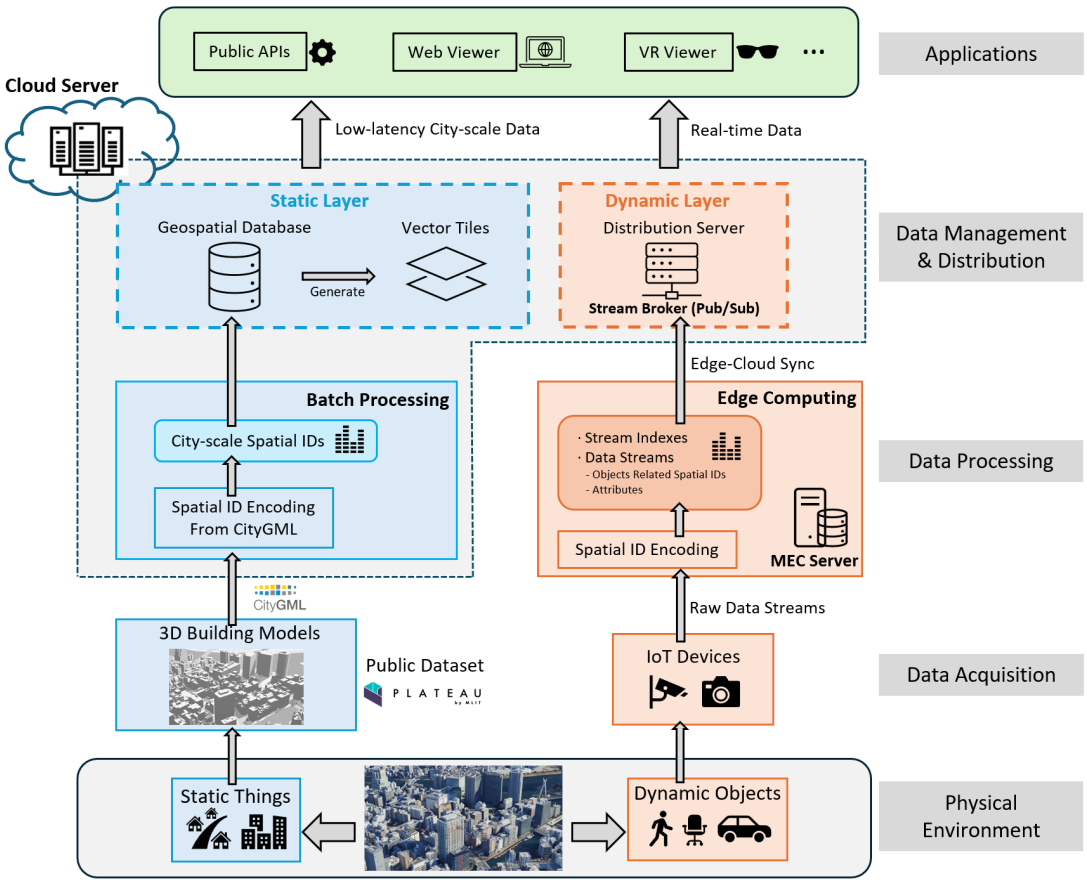


Fig. 5.1. Proposed Five-layer Architecture

The five layers are defined as follows:

1. **Physical Environment** The foundation of the system is the physical city itself, providing the ground truth for all digital representations. This layer is categorized into static entities, such as buildings and permanent infrastructure, and dynamic objects, including pedestrians, vehicles, and temporary equipment. These elements represent the primary sources of environmental data.
2. **Data Acquisition** This layer serves as the interface for ingesting heterogeneous data types. For the static environment, it leverages semantically rich 3D datasets, such as Japan's PLATEAU (MLIT), to establish a CityGML-based skeleton. For

dynamic states, a distributed network of IoT devices—including cameras and environmental sensors—captures high-frequency raw data streams, prioritizing endpoint sensing to feed the subsequent computational pipelines.

3. **Data Processing** To handle data across different scales, this layer is bifurcated into batch and real-time streams. Batch Processing is utilized for city-scale static models, where vector geometries are encoded into Spatial IDs. Simultaneously, MEC servers perform real-time voxelization of dynamic streams. This decentralized approach allows for data preprocessing at the source, significantly reducing long-haul bandwidth requirements.
4. **Data Management & Distribution** Situated in a cloud environment, this layer serves as the relational and temporal backbone. It manages a Static Layer for geospatial databases and a Dynamic Layer that utilizes a stream broker for publish/subscribe distribution. The Spatial ID acts as the universal join-key, allowing the system to aggregate data from multiple sources while maintaining a unified, queryable city-wide state.
5. **Applications** The top layer provides the end-user interface for interaction and visualization. It supports high-level applications including Public APIs for third-party development, Web Viewers for administrative monitoring, and VR Viewers for spatial computing on devices such as the Meta Quest 3. These applications consume synchronized 4D voxel grids to provide real-time situational awareness and "what-if" simulations.

5.3 Distributed Perception and Voxelization

The first stage of the architecture is designed to bridge the "synchronization barrier" by converting raw physical events into structured spatial intelligence. Rather than treating data ingestion as a passive collection process, this layer employs high-frequency ingestion coupled with immediate, localized processing at the network edge. This ensures that the digital twin is not merely a historical record but a live reflection of the city.

5.3.1 Distributed Multi-Modal Sensing

The system's foundation relies on a heterogeneous array of urban sensors that capture the city's state across multiple modalities. In alignment with Japan' s i-Construction and Society 5.0 frameworks, the architecture ingests semantically rich 3D building models—such as those from Project PLATEAU—to establish a high-fidelity "spatial skeleton."

This static framework is then augmented by a distributed network of IoT devices, including vision-based sensors and environmental monitors. These devices provide the high-frequency raw data streams required for real-time mirroring. By integrating these diverse sources, the system satisfies the operational awareness requirements of Requirement (v), creating a hybrid environment where static infrastructure and dynamic agents coexist.

5.3.2 Edge-Based Voxelization and Near-Field Computation

To solve the latency and bandwidth issues that typically plague cloud-only models, the architecture utilizes Multi-access Edge Computing (MEC) for "near-field computation." In a traditional setup, transmitting raw, high-bandwidth video feeds to a central server would lead to significant network congestion and delayed response times. Instead, the proposed system executes perception and voxelization algorithms directly on edge nodes

located near the data source.

The core of this edge process is Real-time Spatial ID Encoding. This process takes the raw 3D coordinates of detected objects and maps them to discrete, voxel-based identifiers. By adopting this fixed-cell reference system, data from multiple, disparate sources can be aggregated with arbitrary positional precision, fulfilling Requirement (iii).

The primary advantage of performing this voxelization at the edge is the drastic reduction in data volume. The system extracts lightweight semantic metadata—effectively converting megabytes of raw sensor data into bytes of structured text strings (Spatial IDs). This transformation enables the ultra-low-latency synchronization necessary for a truly responsive urban digital twin.

5.4 Data Fusion and Management

Once the urban state has been discretized into voxels, the infrastructure must manage the fusion of these high-frequency dynamic updates with the stable, archival city model. This layer serves as the relational backbone of the digital twin and runs in a cloud-native environment to ensure global accessibility and persistence.

The data management process follows a dual-path architecture to accommodate different temporal scales. The static branch handles the geospatial database and the long-term storage of city-scale models. By using batch processing to pre-encode CityGML features into a unified set of Spatial IDs, the system establishes a machine-readable foundation for the built environment. This pre-indexing is crucial for meeting the fast retrieval and search capabilities specified in Requirement (iv).

In contrast, the dynamic branch manages high-velocity state changes through a specialized stream broker. This component facilitates the distribution of real-time voxel occupancy data from edge nodes to application subscribers, maintaining the "Edge-Cloud Sync" required for live mirroring.

The Spatial ID serves as the universal join-key that bridges these two specialized paths. Because the cells within a Discrete Global Grid System are geographically fixed, the system uses the ID as a common reference point to fuse real-time IoT streams with static semantic attributes. This hierarchy-based indexing allows the architecture to manage data recorded with different spatial reference systems in an integrated manner, effectively fulfilling Requirement (ii). The resulting 4D infrastructure tracks the movement and state of urban objects through both space and time, providing a consistent "Single Source of Truth."

5.5 Spatial Computing and Visualization

The top layer of the architecture transforms the synchronized 4D voxel grid into actionable intelligence for both human and machine operators. This stage focuses on high-level interaction and provides the necessary interfaces for a wide range of urban applications.

To ensure the infrastructure is scalable for city-wide deployment as dictated by Requirement (vi), the architecture provides a set of Public APIs. These interfaces allow third-party developers to access the synchronized 4D data for specialized use cases like autonomous driving, disaster response, and utility management. The system is designed to support multiple visualization modalities, ranging from standard web-based monitoring dashboards to immersive Virtual Reality and MR environments.

The hierarchical nature of the Spatial ID is particularly beneficial for these applications because it allows for multi-resolution visualization. Users can subscribe to coarse-grained voxels when they need a broad city overview or switch to fine-grained voxels for pre-

cise object-level interactions. This flexibility ensures computational efficiency and rapid response times even in the dense geometric environments of complex urban canyons. Ultimately, this layer realizes the digital twin as an interactive, live mirror of the physical city, capable of supporting sophisticated spatial computing tasks with minimal latency.

Chapter 6

System Implementation and Evaluation

This chapter details the prototype implementation of the proposed five-layer architecture. This transition from a conceptual framework to a functional system serves to validate the Edge-Cloud hybrid model and demonstrate how real-time synchronization can be maintained under practical urban conditions.

6.1 Prototype definition and Scope

To test the feasibility of a Spatial ID-driven infrastructure, we abstracted the physical urban environment into two primary layers: static infrastructure and dynamic agents, as shown in Fig 6.1. The scope of the prototype focuses on integrating high-fidelity city models with high-frequency sensor data, effectively bridging the "synchronization barrier" discussed in earlier chapters.

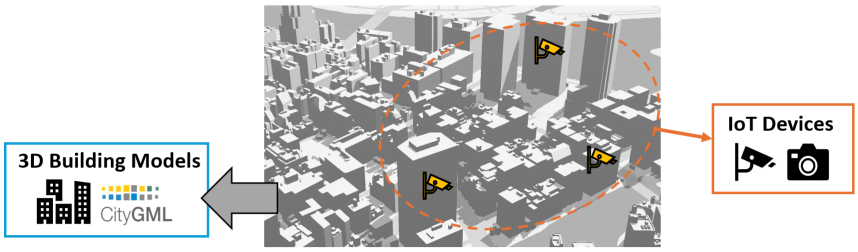


Fig. 6.1. Abstract the Physical Urban Environment into Two Primary Layers

Static Environment: PLATEAU Tokyo 23 Wards The built environment for this prototype is represented using the PLATEAU dataset, an open-source 3D city model project managed by Japan's Ministry of Land, Infrastructure, Transport and Tourism (MLIT).

For this implementation, the scope is focused on the Tokyo 23 Wards area. The data is utilized in CityGML format, which provides the semantically rich 3D geometries that form the structural backbone of the digital twin. By leveraging PLATEAU, the system establishes a pre-indexed spatial reference at a city-wide scale. These models provide the "spatial skeleton" required for the Universal Indexing Layer, allowing the system to perform high-speed geospatial queries without the computational burden of processing raw geometry in real time.

Dynamic Environment: RGB-D Sensing Unit Real-time changes within the envi-

ronment are captured using a distributed IoT sensing unit. In this prototype, we used an RGB-D camera as the primary sensing device. Standard RGB cameras only provide 2D projections, which are insufficient for precise spatial mapping. In contrast, the depth data from an RGB-D sensor is critical for satisfying Requirement (i), as it allows the system to calculate the exact 3D volumetric position of detected objects relative to the camera's pose. By fusing color and depth streams, the system extracts 3D coordinates (x, y, z) for dynamic agents, which are then converted into Spatial IDs at the network edge.

The prototype focuses on two classes of dynamic entities to test the system's versatility:

- Pedestrians: These represent high-frequency, non-deterministic movements. They are used to evaluate whether the system can maintain sub-second synchronization and provide the rapid situational awareness necessary for safety-critical applications.
- Furniture and Temporary Equipment: These represent semi-static objects, such as street barriers or temporary signage. These entities are used to test the "Automated Model Updating" capability—the system' s ability to self-correct the digital twin environment when physical layouts change, reducing the need for manual oversight.

6.2 City-Scale Spatial ID Generation

The implementation of the static layer centers on the transformation of Project PLATEAU vector datasets into a discrete hierarchical grid. By converting the continuous coordinate space of Tokyo' s built environment into a structured voxel format, the system enables significantly faster spatial querying and more efficient cross-domain data integration.

6.2.1 Data Ingestion and Extraction

The pipeline begins with the ingestion of CityGML files covering the Tokyo 23 Wards. As illustrated in Fig 6.2, the system decomposes the raw XML-based data into two primary components: geometric meshes and semantic properties. This ensures that metadata—such as building age, usage, or structural material—remains associated with the spatial volume after discretization. This approach allows the static infrastructure to serve as more than just a visual backdrop; it becomes a semantically rich "spatial skeleton" that provides context to the dynamic data streams processed in real time.

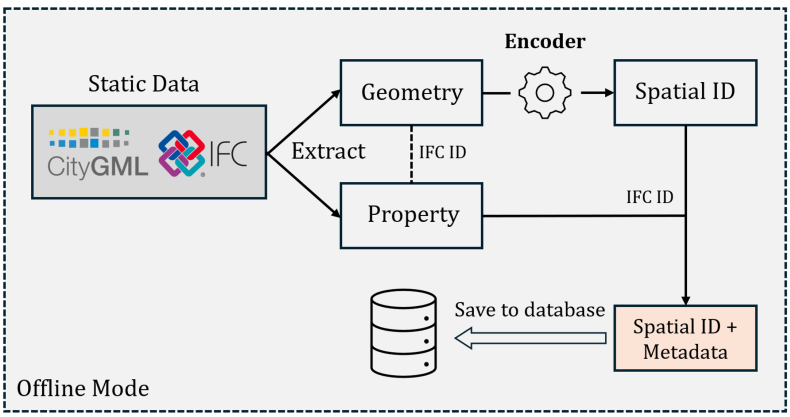


Fig. 6.2. CityGML to Spatial ID Pipeline

6.2.2 Surface Voxelization Algorithm

At the heart of the static generation process is a surface voxelization algorithm designed to map complex 3D meshes onto a set of unique volumetric identifiers. This algorithm avoids the computational overhead of traditional volumetric filling by focusing on the exterior boundary of urban objects, which is sufficient for most spatial interaction and interference detection tasks. The conversion process follows a rigorous multi-stage logic:

1. **Feature Parsing:** The system parses the CityGML file based on the target Level of Detail (LOD) and Zoom Level.
2. **Vertex Extraction:** Vertices are extracted from the geometric meshes of each feature.
3. **Coordinate Transformation:** These continuous points are converted into discrete grid indices (x, y, z) based on a specified Zoom Level (Z). For this prototype, a zoom level of 25 is used to provide approximately 1-meter resolution.
4. **ZFXY Encoding:** The indices are standardized into the ZFXY format $\{z\}/\{f\}/\{x\}/\{y\}$, where f represents the altitude index.

Algorithm 1: Spatial ID Generation from Algorithm CityGML

Input : CityGML file (C), Target LOD (L), Zoom Level (Z), Fill Flag (F)
Output: Mapping $M : ID_{gml} \rightarrow \{ID_{spatial}\}$

$M \leftarrow \emptyset$
$Features \leftarrow$ ParseCityGML(C, L) `// Extract features based on LOD`
foreach $feature \in Features$ **do**
  $id_{gml} \leftarrow$ GetAttribute($feature$,"gml:id")
  $geom \leftarrow$ GetGeometry($feature$)
  if *geom is null* **then**
    continue
  `// Surface Voxelization`
  $Points \leftarrow$ ExtractVertices($geom$)
  $Voxels \leftarrow \emptyset$
  foreach $p \in Points$ **do**
    $v \leftarrow$ CalculateGridIndices(p, Z) `// Convert coords to Z, X, Y`
    $Voxels \leftarrow Voxels \cup \{v\}$
  `// Standardize to ZFXY Format`
  $SIDs \leftarrow \emptyset$
  foreach $v \in Voxels$ **do**
    $sid \leftarrow$ EncodeZFXY(v, Z)
    $SIDs \leftarrow SIDs \cup \{sid\}$
  $M(id_{gml}) \leftarrow SIDs$
return M

The outcome of this surface voxelization is a structured digital mirror of the physical environment. In the prototype results, showing in Fig 6.3, the original vector geometry is effectively discretized into addressable volumetric cells. For visual clarity in technical documentation, these voxels are often rendered at a lower zoom level ($Z = 18$), where the

overlay of the standardized grid on the original shaded mesh demonstrates the mapping's precision.

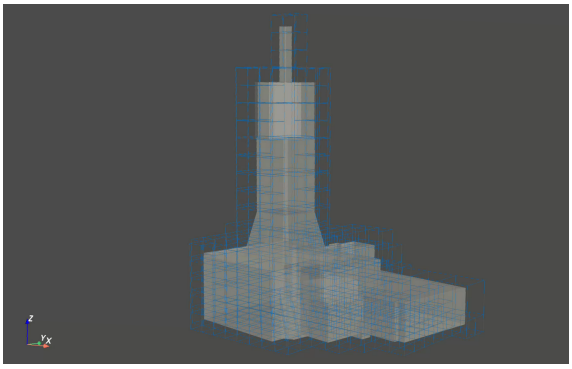


Fig. 6.3. Generated Spatial ID overlays on a building model

6.3 IoT Stream Spatial ID Encoding at the Edge

The real-time synchronization of dynamic agents is handled through a decentralized pipeline running on MEC servers Fig 6.4. This implementation is the primary mechanism for overcoming the "Synchronization Barrier," as it shifts the heavy lifting of voxelization to the network edge. By processing data near the source, the system ensures ultra-low latency and drastically reduces the volume of traffic sent over the backhaul network, satisfying the performance constraints of a live city-scale infrastructure.

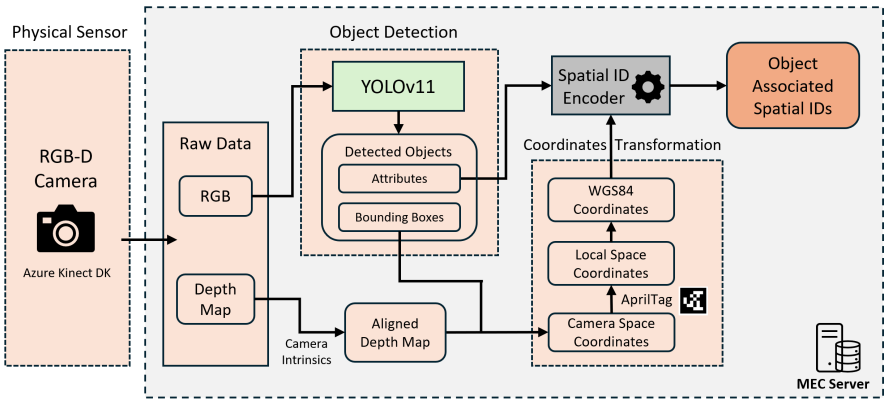


Fig. 6.4. Generated Spatial ID overlays on a building model

For this prototype, the edge node is interfaced with an Azure Kinect DK RGB-D camera, configured according to the parameters in Table 6.1.

6.3.1 Perception and Semantic Extraction

The perception stage begins by ingesting two synchronized data streams: a high-resolution 1536p RGB frame and a corresponding depth map. For real-time object detection, the system employs the YOLOv11m model. This specific model was selected for its high

Table 6.1. Configurations

Parameter	Specification
Color Resolution	RES_1536P
Depth Mode	NFOV_2X2BINNED
Frame Rate	30 FPS
Detection Model	YOLOv11m
Training Dataset	COCO Dataset

throughput and precision, making it suitable for the computational limits of edge hardware.

Once an object is detected in the 2D frame, its depth information must be precisely mapped to its visual representation. This is achieved by aligning the depth map with the RGB frame using the camera' s factory-calibrated intrinsic parameters. To maintain the accuracy of the 3D coordinates and mitigate inherent sensor noise or flickering, the pipeline applies a spatial averaging technique. Rather than calculating coordinates from a single central pixel—which can be unstable—the system utilizes a 5×5 window centered on the object's centroid to derive representative XYZ coordinates in camera space.

This extraction process effectively converts raw, high-bandwidth video feeds into lightweight, structured metadata. This metadata includes not only the precise spatial location of each agent but also critical semantic attributes such as its category classification (e.g., "pedestrian" or "vehicle") and detection confidence.

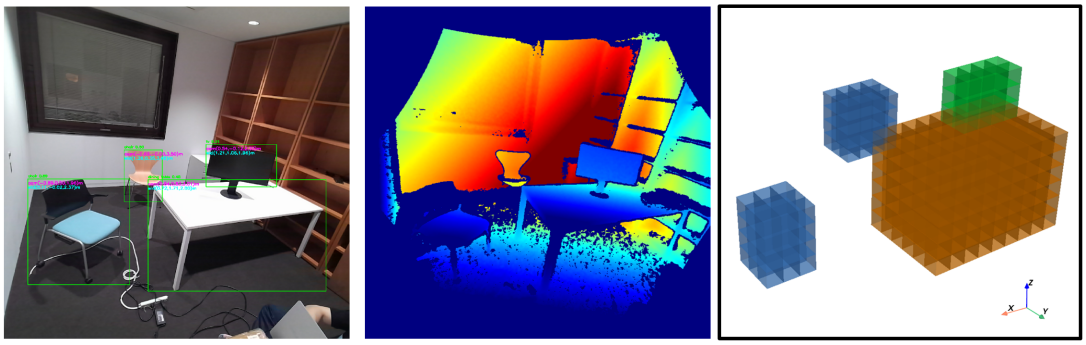


Fig. 6.5. Perception and Visualization of Encoded Spatial ID

6.3.2 Coordinate Transformation

To achieve a globally consistent digital twin, the system must bridge the gap between the camera' s relative optical space and the Earth' s geodetic frame. This transformation is necessary to satisfy Requirement (ii) and enable the integration of heterogeneous sensor data into a unified spatial context. The process is implemented as a three-stage mathematical pipeline that converts local camera-space coordinates (P_{cam}) into world-space Earth-Centered, Earth-Fixed (ECEF) coordinates (P_{ECEF}).

1. **Camera Space to Tag Space** ($T_{cam2tag}$) The transformation begins by establishing the camera' s pose relative to a physical anchor. The system utilizes an AprilTag (family 36h11, size 0.175m) with a predefined global position. By detect-

ing the tag in the RGB frame, the edge server computes the tag-to-camera pose ($T_{tag2cam}$) using the camera' s internal matrix ($Kmat$). To obtain the necessary mapping from the camera' s origin to the tag, the system inverts this matrix:

$$T_{cam2tag} = (T_{tag2cam})^{-1}$$

This ensures that all detected objects are first normalized to the tag' s localized coordinate system Fig 6.6.

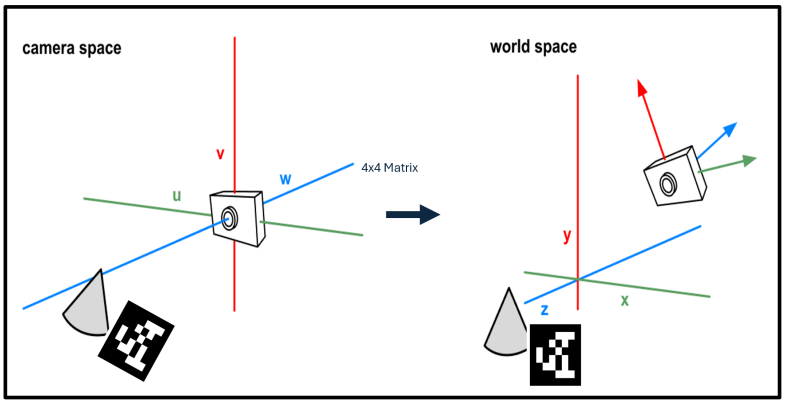


Fig. 6.6. Camera Space to Tag Space

2. **Tag Space to Local ENU Space** ($T_{tag2ENU}$) Because the physical orientation of the tag may not align with cardinal directions, a rotation matrix is applied to align the tag' s axes with a local East-North-Up (ENU) frame. The tag' s orientation is defined by a predefined yaw, measured anticlockwise from the East axis. The transformation matrix $T_{tag2enu}$ maps the tag' s +X (right) to the ground plane, +Y (down) to the negative vertical, and +Z (into the wall) to the corresponding horizontal vector. This stage creates a localized, gravity-aligned environment for stable object positioning.
3. **ENU Space to World ECEF Space** ($T_{ENU2ECEF}$) The final stage projects the localized ENU coordinates into the global WGS-84/ECEF frame (EPSG:4978). Using the pyproj library, the system transforms the tag' s geodetic coordinates into an ECEF origin point (x_0, y_0, z_0). A rotation matrix R is constructed to account for the Earth's curvature at that specific latitude and longitude:

$$R = \begin{bmatrix} -\sin(\lambda) & -\sin(\phi)\cos(\lambda) & \cos(\phi)\cos(\lambda) \\ \cos(\lambda) & -\sin(\phi)\sin(\lambda) & \cos(\phi)\sin(\lambda) \\ 0 & \cos(\phi) & \sin(\phi) \end{bmatrix}$$

Where ϕ is the latitude and λ is the longitude. The final global position is derived by concatenating these matrices:

$$P_{ECEF} = T_{ENU \to ECEF} \cdot T_{Tag \to ENU} \cdot T_{Cam \to Tag} \cdot P_{Cam}$$

6.3.3 Spatial ID Encoding

The final stage is the encoding of these global coordinates into the standardized Spatial ID format. To represent the physical volume of agents without the computational overhead of

high-resolution point cloud transmission, the system utilizes category-specific dimensions (Length, Width, Height) to define a volumetric bounding box.

The encoder calculates only the minimum and maximum corner Spatial IDs for the object's volume. This efficient representation allows the central cloud to reconstruct the full set of occupied voxels across multiple zoom levels without receiving massive data packets. As illustrated in Fig 6.7, the output is a structured JSON package containing the min-max Spatial IDs at zoom level 25, along with confidence scores, central coordinates, and high-precision timestamps.


```
{
  "spatial_ids": [
    {
      "zoom": 25,
      "min_corner": "25/29407096/13327667/14",
      "max_corner": "25/29407096/13327668/15"
    }, ...
  ],
  "name": "person",
  "confidence": "0.91",
  "llh": [135.5039, 34.6757, 13.5098],
  "timestamp": "2025-06-18T23:38:42.776493"
}
```


Fig. 6.7. JSON Package

By performing this encoding at the edge, the infrastructure achieves millisecond-level synchronization required for "live" mirroring, transforming megabytes of raw sensor data into machine-readable spatial intelligence.

6.4 Data Management & Distribution Framework

The Data Management and Distribution Framework functions as the central operational hub of this cloud-native infrastructure. It is responsible for the persistent storage, orchestration, and dissemination of urban spatial intelligence. As illustrated in the detailed architecture Fig 6.8, the framework implements a bifurcated pipeline that treats static and dynamic data as distinct but interconnected streams.

6.4.1 Geospatial Database

The core of the distribution framework is built upon a PostgreSQL database with the PostGIS extension. This setup is chosen specifically because PostGIS provides specialized geometry column types and spatial indexing capabilities that are essential for handling complex 3D urban data at scale. The database functions as the "Single Source of Truth," maintaining two critical tables that coordinate the system's spatial awareness:

Spatial_ids Table: This table stores the pre-indexed urban skeleton. It maintains the relationships between building geometries, the unique Spatial IDs generated from the PLATEAU datasets, and their associated semantic metadata. By providing a persistent 3D volumetric context for urban objects, this structure directly satisfies Requirement (i).

Stream_Indexes Table: This table orchestrates real-time data flows by mapping the geographic locations of active sensors to specific WebSocket endpoints. This indexing allows the system to dynamically route IoT data based on a user' s spatial query, fulfilling Requirement (ii) for the integrated management of heterogeneous data sources.

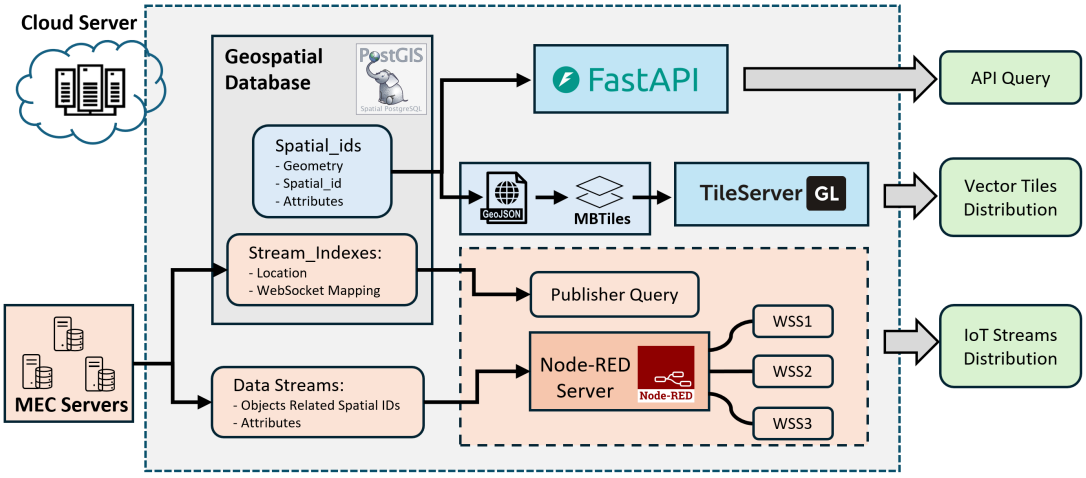


Fig. 6.8. Data Management & Distribution Framework

6.4.2 Static Asset Distribution: Vector Tiles and FastAPI

The distribution of static assets is optimized to ensure that city-scale data remains accessible and searchable without overwhelming the client or the network. To meet Requirement (iv)—fast retrieval and high-performance searching—the framework avoids the heavy computational cost of serving raw, uncompressed geometries. Instead, it employs a combination of standardized tiling and a programmable interface.

Vector Tile Distribution

Directly serving raw Spatial ID geometries is not feasible at high zoom levels due to the massive number of data points. To solve this, the framework implements a vector tile pipeline that compresses and segments the city model into manageable packets.

1. **Extraction:** The process begins by pulling static building geometries and their corresponding Spatial IDs from the PostGIS database and converting them into the GeoJSON format.
2. **Conversion:** This data is then transformed into MBTiles, a compact, SQLite-based container. This format is highly efficient for storing hierarchical geospatial data, as it allows for rapid lookups of specific map areas.
3. **Serving:** A TileServer GL instance acts as the distribution engine, exposing these MBTiles as vector tiles. The retrieval follows a standard RESTful pattern: http://localhost:8080/data/tiles/z/x/y. This design allows the client-side viewer to fetch only the data relevant to the user' s current field of view, enabling stateless, on-demand access that is standard in modern web mapping.

Programmable API Layer

While tiles handle visualization, a FastAPI wrapper provides the programmable logic needed for spatial computing and third-party integration. This layer transforms the database into a queryable engine, supporting several core functions:

- **Area Queries:** Through the /spatial-ids/area endpoint, developers can retrieve all Spatial IDs within a specific geographic bounding box, which is essential for regional analysis.

- **Point-to-ID Conversion:** The /spatial-ids/point endpoint allows for the translation of GPS coordinates into a specific Spatial ID voxel, as well as the inverse operation (finding the center coordinates of a given ID).
- **Semantic Lookup:** The /spatial-ids/spatial_id/attributes endpoint bridges the gap between geometry and information, fetching the metadata (such as building usage or height) tied to a specific voxel.
- **Hierarchical Navigation:** By using the /spatial-ids/spatial_id/neighbors endpoint, the system supports complex spatial reasoning, such as finding adjacent voxels or navigating up and down the grid hierarchy (parents and children).



Fig. 6.9. Provided API Endpoints

6.4.3 Dynamic Stream Brokerage: Node-RED and WebSockets

The dynamic layer addresses the "Synchronization Barrier" by facilitating the live flow of state updates from distributed edge nodes to applications. This pipeline is implemented via a Publish/Subscribe model to maintain "Edge-Cloud Sync" with high reliability.

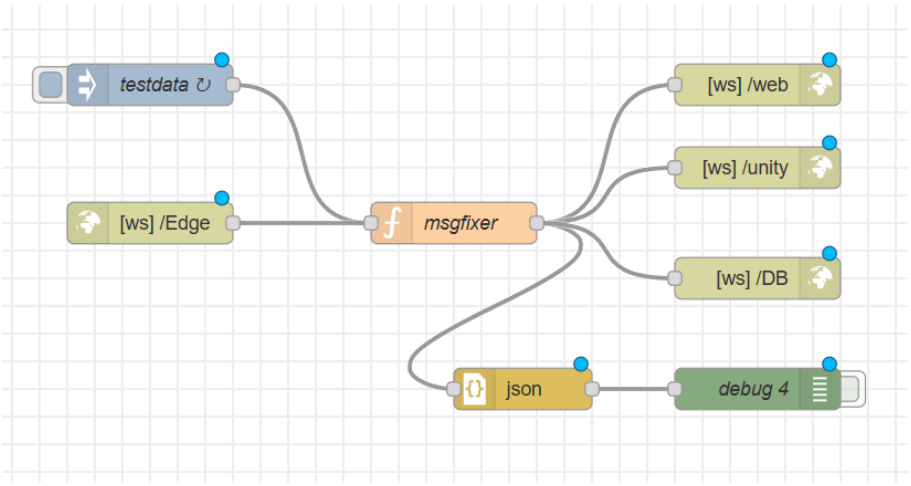


Fig. 6.10. Node-RED Setup Example for One IoT Stream

- Stream Brokerage via Node-RED: The system utilizes a Node-RED Server as the

primary stream broker, Fig 6.10. This server receives encoded Spatial ID data streams and attributes from the MEC servers.

- Dynamic Mapping and Distribution: For each incoming IoT stream, the broker performs a Publisher Query against the Stream_Indexes table. By checking the sensor's location and its corresponding WebSocket mapping, Node-RED determines how many sensors are active in a given area and routes the data to the appropriate subscribers.
- WebSocket Delivery: The real-time states are broadcast via secure WebSockets (WSS). Dedicated channels (/web, /unity, /DB) allow different viewers—such as web-based monitors or MR headsets—to receive synchronized updates simultaneously.

By decoupling the heavy archival database from the high-frequency dynamic streams, the distribution framework ensures that the digital twin remains a "live mirror" of reality, providing sub-second response times for situational awareness while maintaining the scalability needed for city-wide deployment.

6.5 Performance Evaluation

The prototype' s capacity to resolve the "Synchronization Barrier" was tested by evaluating three core functional areas: geospatial database responsiveness, vector tile distribution efficiency, and the latency inherent in real-time IoT stream processing. This analysis provides an empirical baseline to determine if the proposed architecture can sustain a persistent, high-fidelity digital twin under realistic urban conditions.

Processing the PLATEAU datasets for Tokyo' s 23 special wards through the City-Scale Spatial ID Generation process detailed in Section 6.2 yielded a comprehensive spatial index at zoom level 25. This derived dataset [41] serves as a static structural backbone for subsequent evaluations. The resulting Spatial IDs, when rendered within the custom web viewer, align precisely with Google 3D Tiles[42], as illustrated in Fig 6.11. This visual verification suggests that the voxelization process maintains spatial integrity relative to established geospatial standards, even at a resolution approximating 1 m^3. Such a configuration appears to provide the necessary granularity for addressable spatial computing while remaining compatible with existing cloud-based 3D tile architectures.

Table 6.2 shows the equipment's configuration.

Table 6.2. Configurations

Component	**Specification**
GPU	NVIDIA RTX 4070 GPU
CPU	13th Gen Intel i9-13900H 20-core CPU
RAM	30 FPS
SYSTEM	Ubuntu 24.04.2

6.5.1 Geospatial Database Query Performance

The efficiency of city-scale data retrieval depends largely on the underlying PostGIS database. To simulate realistic demand, we populated the database with 148.5 million rows of Spatial ID–encoded geometries covering Tokyo' s Bunkyo and Chiyoda wards. These central districts represent a high-density urban environment ideal for stress-testing. We selected a zoom level of 25—equivalent to a 1 m^3 voxel resolution—to ensure the dataset



Fig. 6.11. Spatial IDs of the Tokyo 23 Wards, Overlaying on Google 3D Tiles

was sufficiently detailed for practical use.

Performance was measured by executing 90 randomized queries for bounding boxes ranging from 0.1 km² to 1.5 km². To eliminate cold-start discrepancies, we conducted an initial warm-up phase before recording latency from a remote client.

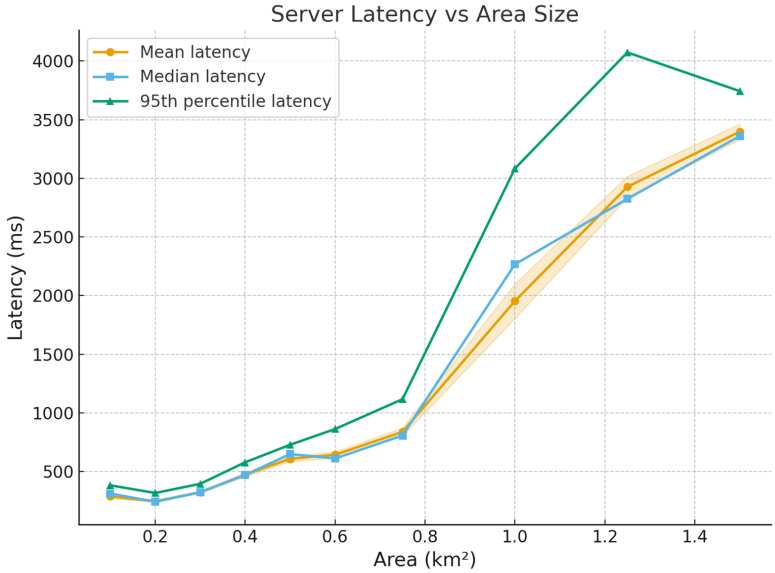


Fig. 6.12. Query latency with varying areas.

Fig. 6.12 presents the relationship between query latency and area size. The results confirm that the system maintains millisecond-to-second responsiveness across the tested range. For small-scale queries (under 0.5 km²), both mean and median latencies stayed below 600 ms. While latency increases non-linearly as the area grows—with the 95th

percentile exceeding 4 seconds at 1.2 km² due to intensive spatial filtering—the median latency at the district level (approximately 1 km²) remains within a few seconds. This confirms that Spatial ID–indexed queries are feasible for real-time UDT operations where "sub-minute" updates for large areas are acceptable.

6.5.2 Vector Tile Distribution Efficiency

To optimize the visualization of these static assets, we generated a vector tile set in MBTiles format for a 2 km² area in Chiyoda Ward. This involved 3,179 tiles across zoom levels 15–20, totaling 209.7 MB. Using TileServer-GL, we measured performance by issuing 300 randomized fetches per zoom level using a standard REST-style pattern (/data/tiles/z/x/y).

Table 6.3. Tile latency and payload size across zoom levels.

Zoom	Avg Latency (ms)	P95 Latency (ms)	Payload Size (KB)
15	98.36	111.52	479.94
16	107.72	121.71	481.97
17	97.21	110.24	395.18
18	43.86	52.93	183.01
19	19.01	25.46	70.63
20	64.89	75.12	257.63

The average latency remained remarkably low, fluctuating between 19 ms and 108 ms. The 95th percentile values were consistently under 125 ms, which is the threshold typically required for a "seamless" user experience in interactive mapping.

Notably, payload sizes dropped significantly at higher zoom levels, reaching a minimum of 70 KB at zoom 19. This reduction in data volume per tile is a direct benefit of the hierarchical indexing, as it limits the number of features the system must transfer for a given view. The slight increase at zoom 20 reflects the overhead of fine-grained geometry fragmentation. Ultimately, these results validate that a tile-based approach provides the sub-100 ms responsiveness needed for interactive digital twin visualization.

6.5.3 Real-Time IoT Stream Synchronization

The final evaluation phase focused on the system' s capacity to mirror dynamic physical events as they occur. To test this, an Azure Kinect RGB-D camera generated synchronized **30 FPS** streams, with the entire perception-to-voxelization pipeline executed locally on the MEC server.

The complete Spatial ID encoding pipeline—encompassing YOLOv11m object detection, depth alignment, and coordinate transformation—averaged a processing time of just **25 ms** per frame. This efficiency is critical. It allows the system to sustain a full 30 FPS throughput without dropping frames. Consequently, the digital twin can reflect physical movements with sub-second latency, providing the "live" mirroring required for safety-critical applications like autonomous navigation or real-time crowd management.

Fig 6.13 provides a side-by-side comparison of the physical environment and its digital counterpart reveals the precision of this synchronization. For visual clarity during testing, building-level Spatial IDs were hidden to better highlight the behavior of dynamic agents. As the sequence in the prototype demonstrates, moving pedestrians and temporary equipment are precisely mapped in cyberspace as discrete, addressable voxels. This alignment

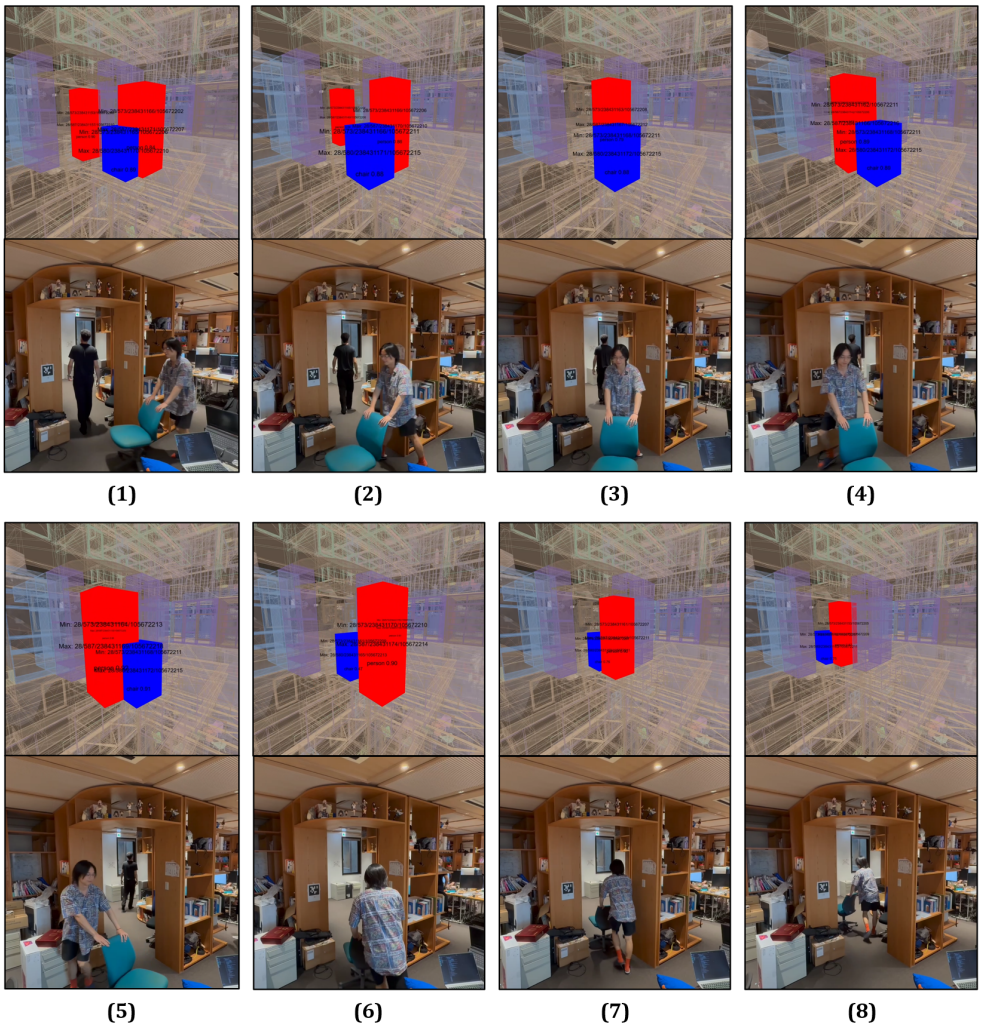


Fig. 6.13. Side-by-Side Comparison of the Physical Environment and the Digital Twin

confirms that the proposed architecture effectively bridges the "Synchronization Barrier," creating a reliable, 4D foundation for high-stakes spatial computing.

Chapter 7

Use Case: Mixed Reality on Meta Quest 3

This final chapter illustrates how the proposed infrastructure moves beyond a backend data system to become a functional substrate for MR. By using the Meta Quest 3, we demonstrate that a high-fidelity bridge between city-scale digital twins and immersive human interaction is not only possible but scalable. This platform isn't just an application; it is a direct extension of the five-layer architecture, providing the anchoring, synchronization, and persistence that standalone headsets cannot achieve alone.

7.1 Infrastructure-Driven Geospatial Anchoring

The primary hurdle for urban-scale spatial computing is geospatial anchoring. Consumer hardware, such as the Meta Quest 3, lacks native GNSS or absolute geolocation capabilities, making it difficult to "pin" digital content to the physical world accurately. Our infrastructure solves this by acting as a global reference backbone, using the hierarchical 4D indexing of Spatial IDs to ensure virtual objects stay fixed to their real-world counterparts.

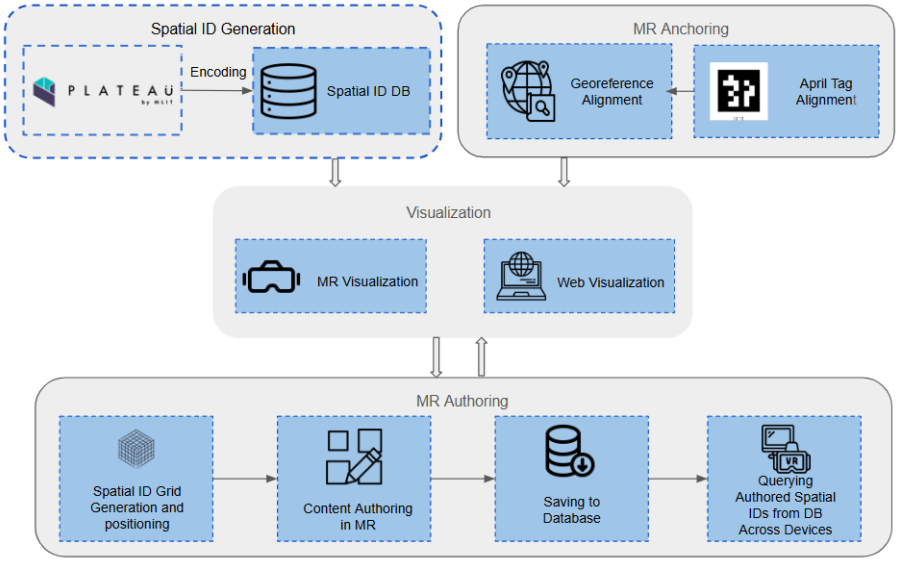


Fig. 7.1. High-level design overview of the system, illustrating how anchoring and visualization enables persistent and collaborative MR authoring of Spatial IDs.

As illustrated in Fig. 7.1, When a session begins, the headset detects a physical anchor placed at a known geographic location. The infrastructure then calculates the spatial offset between the headset' s local tracking and the global geodetic frame. This effectively pins the entire virtual coordinate system to the Earth's surface with centimeter-level precision. This centralized registration ensures that spatial data remains consistent across different sessions and devices, even in environments where GNSS signals are unavailable.

7.2 Real-Time Visualization of Dynamic Urban Streams

The ability to perform near-field computation at the edge is what truly enables the low-latency visualization of live city states. Once anchoring is established, the platform uses the Edge-Cloud synchronization pipeline to overlay dynamic data—like real-time object detections—directly onto the user' s physical field of view. Raw sensor feeds are processed at the edge and immediately converted into object-related Spatial IDs. These are then streamed to the MR viewer via persistent WebSocket connections. By assigning these detections to discrete $1m^3$ voxels, the infrastructure offloads intensive depth-processing from the headset. This allows the Quest 3 to render complex urban scenes with minimal latency, creating an "X-ray" effect. Pedestrians, vehicles, and furniture appear as addressable volumetric entities that move in near-perfect temporal alignment with their physical counterparts. Fig. 7.2 demonstrates this capability, showing real-time detection of agents rendered as color-coded voxels in the headset's passthrough view.

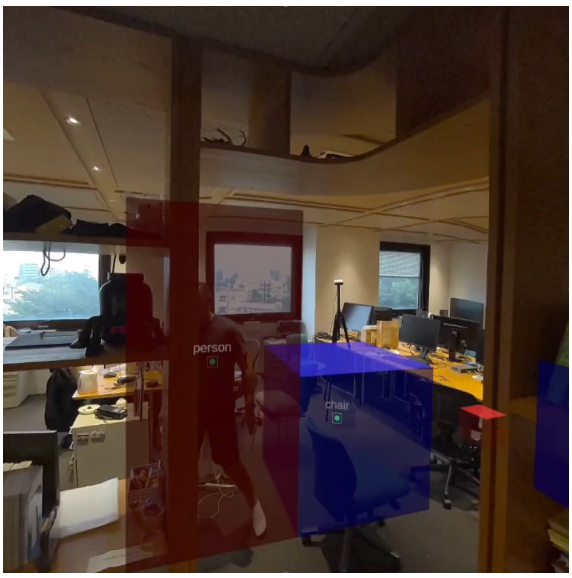


Fig. 7.2. Mixed Reality view showing real-time urban objects represented as color-coded Spatial ID voxels synced with physical counterparts.

7.3 Spatial Grid Interaction and the Voxel Paradigm

The practical utility of this infrastructure is most evident when navigating dense 3D urban environments. Selecting individual spatial units at high resolution is a significant interaction challenge; to address this, the platform introduces a "stick-and-cursor" paradigm that leverages the infrastructure' s hierarchical grid.

Users manipulate a virtual probe to highlight and select specific voxels within the

globally aligned wireframe grid, demonstrated in Fig. 7.3. The infrastructure provides immediate visual feedback as the cursor enters a discrete Spatial ID volume, ensuring the user is aware of exactly which "address" they are interacting with. This interaction model supports multi-resolution authoring: a user can select a coarse city block for broad zoning or a fine-grained, sub-meter voxel for detailed equipment labeling.

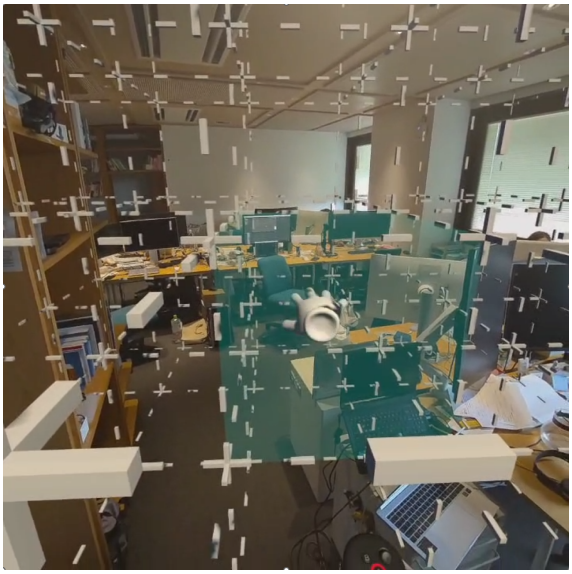


Fig. 7.3. A user performing MR authoring by selecting a specific voxel in the 3D Spatial ID grid using the stick-and-cursor paradigm.

By establishing this continuous pipeline—from initial anchoring to real-time synchronization and persistent collaborative authoring—the infrastructure successfully bridges the "Synchronization Barrier." It positions Spatial IDs as the definitive foundation for scalable and interoperable UDT.

7.4 Summary of Practical Integration

The successful integration of the MR platform proves that the proposed infrastructure is a primary enabler of city-scale spatial computing. It provides a standardized spatial reference that allows developers to move beyond isolated, application-specific silos.

Key takeaways from this implementation include:

- **GNSS-Free Accuracy:** The system demonstrates that accurate geospatial registration is achievable on consumer-grade hardware through the alignment of geodetic coordinates and physical anchors.
- **Responsive Mirroring:** The performance of the real-time visualization confirms that the infrastructure can maintain the sub-second responsiveness required to mirror physical urban states as addressable voxels.
- **Persistent Knowledge:** The ability for human-authored annotations to persist within a decoupled database transforms the digital twin into a robust, queryable cyber-physical environment where spatial knowledge is preserved across sessions and devices.

Chapter 8

Conclusion and Future Work

The development of the five-layer infrastructure presented in this research addresses the "Synchronization Barrier" that has historically limited the operational utility of UDT. By positioning the Spatial ID as the universal join-key across an edge-cloud continuum, the system successfully bridges the gap between static geospatial models and high-frequency city operations. This research has established a scalable, 4D voxel-based framework that enables a live digital mirror of the physical urban environment while satisfying the core system requirements for unified spatial referencing and real-time sensor integration.

8.1 Summary

The implementation and evaluation of the prototype confirmed that moving intensive voxelization and perception tasks to the network edge significantly reduces long-haul bandwidth requirements and ensures the sub-second response times necessary for real-time situational awareness. The use of a true 3D volumetric index allows for the precise representation of complex urban topologies, resolving the dimensional inconsistencies inherent in 2D-centric geospatial systems. Furthermore, the deployment of the MR platform proved that this infrastructure can provide a stable geospatial backbone for persistent authoring and immersive interaction, even on hardware lacking native GNSS capabilities. By establishing a continuous pipeline from initial anchoring to real-time synchronization and persistent collaborative authoring, the infrastructure positions Spatial IDs as the definitive foundation for the next generation of scalable, interoperable UDT.

8.2 Future Work

While our current database implementation successfully validates the prototype, the next phase of this research involves moving toward a more resilient, multi-layer distributed database architecture. This transition is necessary to support more robust data associations and to fully integrate temporal attributes into the Spatial ID framework.

Scaling the Indexing Backbone

Operating as an openly accessible public infrastructure presents significant technical hurdles. At high zoom levels, the volume of Spatial IDs grows exponentially as the query area expands. Future work must focus on developing stricter area constraints and refined query policies to maintain responsiveness at a national scale. This will likely require distributed computing strategies and advanced indexing methods that go beyond traditional relational models.

Sophisticated Data Ingestion and Compaction

Perhaps the most complex challenge lies in the data ingestion model itself. We need to establish zoom-level-dependent policies for data availability, aggregation, and compaction.

Determining which data points remain queryable at specific resolutions is essential for maintaining a high quality of service. Solving these issues is critical to making the Spatial ID a practical, trustworthy backbone that is compatible with existing systems while remaining ready for the future of spatial computing.

Publications

Conference Publications

1. **Spatial ID-Driven Edge-Cloud Architecture for Real-Time Urban Digital Twins**
   Yanru Chen, *Sami Brahim Djelloul, Alex Orsholits, Manabu Tsukada, Hiroshi Esaki*
   IEEE Consumer Communications & Networking Conference (CCNC2026), Las Vegas, USA, 2026

2. **Spatial ID Authoring in Mixed Reality: A Unified Platform Integrating Anchoring and Visualization**
   Sami Brahim Djelloul, ***Yanru Chen****, Alex Orsholits, Manabu Tsukada*
   IEEE International Conference on Artificial Intelligence and Virtual Reality (AIxVR), Work in Progress (WiP), Osaka, Japan, 2026

References

[1] Ben Bondaruk, Steven A Roberts, and Colin Robertson. Assessing the state of the art in discrete global grid systems: OGC criteria and present functionality. *GEOMATICA*, Vol. 74, No. 1, pp. 9–30, March 2020.

[2] K Chaturvedi and T H Kolbe. INTEGRATING DYNAMIC DATA AND SENSORS WITH SEMANTIC 3D CITY MODELS IN THE CONTEXT OF SMART CITIES. In *TC IV 11th 3D Geoinfo Conference (Volume IV-2/W1) - 20–21 October 2016, Athens, Greece*, Vol. IV-2-W1, pp. 31–38. Copernicus GmbH, October 2016.

[3] Claudio Badii, E G Belay, Pierfrancesco Bellini, Daniele Cenni, Mino Marazzini, Marco Mesiti, Paolo Nesi, Gianni Pantaleo, Michela Paolucci, Stefano Valtolina, Mirco Soderi, and Imad Zaza. Snap4City: A scalable IOT/IOE platform for developing smart city applications. In *2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, pp. 2109–2116. IEEE, October 2018.

[4] Linghe Kong, Jinlin Tan, Junqin Huang, Guihai Chen, Shuaitian Wang, Xi Jin, Peng Zeng, Muhammad Khan, and Sajal K Das. Edge-computing-driven internet of things: A survey. *ACM Comput. Surv.*, Vol. 55, No. 8, pp. 1–41, August 2023.

[5] Digital twin cities: Framework and global practices. Technical report, 2022.

[6] International Electrotechnical Commission and International Organization for Standardization. *Digital twin: Concepts and terminology*. iTeh Standards, Geneva, 2023.

[7] Automation systems and integration — digital twin framework for manufacturing — part 1: Overview and general principles. Technical report, Vernier, Geneva, 2021.

[8] Lalitha Abhaya and Eric Gauthier. Iso/iec/ieee 24641:mbsse, 2023.

[9] Azad M Madni, Carla C Madni, and Scott D Lucero. Leveraging digital twin technology in model-based systems engineering. *Systems*, Vol. 7, No. 1, p. 7, January 2019.

[10] U.S. Department of Energy. National definition of a zero emissions building: Part 1 operating emissions from energy use, version 1.0. Technical report, Office of Energy Efficiency and Renewable Energy (EERE), Washington, D.C., June 2024.

[11] *Smart city blueprint for Hong Kong*. Government of Hong Kong, 2020.

[12] Smart city solutions. `https://www.smartnation.gov.sg/initiatives/smart-city-solutions/`.

[13] Ministry of Land, Infrastructure, Transport and Tourism. Project PLATEAU: Japan's 3D city model project, 2024. Official website.

[14] Cabinet Office, Government of Japan. The 6th science, technology, and innovation basic plan. Technical report, Cabinet Office, 2021.

[15] MLIT. i-construction. `https://www.mlit.go.jp/tec/i-construction/`, 2025.

[16] Dawoon Jeong, Changyun Lee, Youngmin Choi, and Taeyun Jeong. Building digital twin data model based on public data. *Buildings*, Vol. 14, No. 9, p. 2911, September 2024.

[17] Kimberly Mason and Jens Schröder-Fürstenberg. Exploring the possibility of adding

DGGS support to the S-100 universal hydrographic data model. *Int. Hydrogr. Rev.*, Vol. 30, No. 1, pp. 112–123, May 2024.
[18] Google LLC. S2 Geometry Library. `https://s2geometry.io/`, 2024.
[19] Uber Technologies, Inc. H3: A Hexagonal Hierarchical Geospatial Indexing System. `https://h3geo.org/`, 2024. Version 4.x. Accessed: 2026-01-17.
[20] Shirly Stephen, Mitchell Faulk, Krzysztof Janowicz, Colby Fisher, Thomas Thelen, Rui Zhu, Pascal Hitzler, Cogan Shimizu, Kitty Currier, Mark Schildhauer, Dean Rehberger, Zhangyu Wang, and Antrea Christou. The S2 hierarchical discrete global grid as a nexus for data representation, integration, and querying across geospatial knowledge graphs. *arXiv [cs.AI]*, October 2024.
[21] Isaac Brodsky. H3: Uber' s Hexagonal Hierarchical Spatial Index. `https://www.uber.com/blog/h3/`, 2018.
[22] Ken Museth. VDB: High-resolution sparse volumes with dynamic topology. *ACM Trans. Graph.*, Vol. 32, No. 3, pp. 1–22, June 2013.
[23] Ken Museth. NanoVDB: A GPU-friendly and portable VDB data structure for real-time rendering and simulation. In *ACM SIGGRAPH 2021 Talks*, New York, NY, USA, July 2021. ACM.
[24] Francis Williams, Jiahui Huang, Jonathan Swartz, Gergely Klar, Vijay Thakkar, Matthew Cong, Xuanchi Ren, Ruilong Li, Clement Fuji-Tsang, Sanja Fidler, Eftychios Sifakis, and Ken Museth. FVDB : A deep-learning framework for sparse, large scale, and high performance spatial intelligence. *ACM Trans. Graph.*, Vol. 43, No. 4, pp. 1–15, July 2024.
[25] Obaidullah Hakimi, Hexu Liu, Osama Abudayyeh, Azim Houshyar, Manea Almatared, and Ali Alhawiti. Data fusion for smart civil infrastructure management: A conceptual digital twin framework. *Buildings*, Vol. 13, No. 11, p. 2725, October 2023.
[26] Dawoon Jeong, Taeyun Jeong, Changyun Lee, Youngmin Choi, and Dohun Lee. A study on guidelines for constructing building digital twin data. *Buildings*, Vol. 15, No. 3, p. 434.
[27] Thomas H Kolbe. Representing and exchanging 3D city models with CityGML. In *Lecture Notes in Geoinformation and Cartography*, Lecture Notes in Geoinformation and Cartography, pp. 15–31. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.
[28] Tatjana Kutzner, Kanishk Chaturvedi, and Thomas H Kolbe. CityGML 3.0: New functions open up new applications. *PFG – J. Photogramm. Remote Sens. Geoinf. Sci.*, Vol. 88, No. 1, pp. 43–61, February 2020.
[29] Houman Zarrabi and M R Doost Mohammadian. Fusion of digital twin, internet-of-things and artificial intelligence for urban intelligence. In *Urban Sustainability*, pp. 77–92. Springer Nature Singapore, Singapore, 2024.
[30] Steve Liang, Tania Khalafbeigi, and Hylke van der Schaaf. Ogc sensorthings api part 1: Sensing, version 1.1. OGC Implementation Standard 18-088, Open Geospatial Consortium, 2021.
[31] Tanzima Azad, M A Hakim Newton, Jarrod Trevathan, and Abdul Sattar. Syntactic and semantic edge interoperability. *J. Internet Serv. Appl.*, Vol. 16, No. 1, pp. 235–252, May 2025.
[32] Thunyathep Santhanavanich and Volker Coors. CityThings: An integration of the dynamic sensor data to the 3D city model. *Environ. Plan. B Urban Anal. City Sci.*, Vol. 48, No. 3, pp. 417–432, March 2021.
[33] Chih-Yuan Huang, Yao-Hsin Chiang, and Fuan Tsai. An ontology integrating the open standards of city models and internet of things for smart-city applications. *IEEE Internet Things J.*, Vol. 9, No. 20, pp. 20444–20457, October 2022.
[34] IoT Analytics. State of IoT 2025: Number of connected IoT de-

vices growing 14% to 21.1 billion globally. https://iot-analytics.com/number-connected-iot-devices/, 2025. Accessed: 2026-01-18.

[35] Pawani Porambage, Jude Okwuibe, Madhusanka Liyanage, Mika Ylianttila, and Tarik Taleb. Survey on multi-access edge computing for internet of things realization. *IEEE Commun. Surv. Tutor.*, Vol. 20, No. 4, pp. 2961–2991, 2018.

[36] Jianli Pan and James McElhannon. Future edge cloud and edge computing for internet of things applications. *IEEE Internet Things J.*, Vol. 5, No. 1, pp. 439–449, February 2018.

[37] Wajid Rafique, Lianyong Qi, Ibrar Yaqoob, Muhammad Imran, Raihan Ur Rasool, and Wanchun Dou. Complementing IoT services through software defined networking and edge computing: A comprehensive survey. *IEEE Commun. Surv. Tutor.*, Vol. 22, No. 3, pp. 1761–1804, 2020.

[38] Information-technology Promotion Agency (IPA). 4d spatio-temporal information infrastructure architectural guidelines. https://www.ipa.go.jp/digital/architecture/guidelines/4dspatio-temporal-guideline.html, 2023.

[39] Information-technology Promotion Agency (IPA). Definition of spatial id and spatial voxel. https://www.ipa.go.jp/en/digital/architecture-guidelines/definition-of-spatial-id-and-spatial-voxel.html, 2023.

[40] B. Miyahara, T. Kodama, and Y. Kuroishi. Development of new hybrid geoid model for japan, "gsigeo2011". *Bulletin of the Geospatial Information Authority of Japan*, Vol. 62, pp. 11–20, 2014.

[41] Alex Orsholits and Tsukada Laboratory. Spatial id dataset for the 23 special wards of tokyo (zl25), 2025. Generated July 2025. Derived from publicly available CityGML data from the Project PLATEAU dataset.

[42] Google. Photorealistic 3D tiles. https://developers.google.com/maps/documentation/tile/3d-tiles, 2023.

Acknowledgements

I would like to express my deepest appreciation to my supervisors, Professor Hiroshi Esaki and Professor Manabu Tsukada, for their unwavering support and invaluable guidance throughout my master's research. Their expertise has been instrumental in shaping both the direction and execution of this study. I am especially thankful for their willingness to challenge my ideas and push me to deepen my analytical skills.

My gratitude also goes to Alex Orsholits, with whom I had the privilege of working during my research. His perspectives and expertise were immensely helpful. I am also thankful to the laboratory staff at the Tsukada Laboratory, Professor Naren Bao, and Professor Muhammad Asad. My sincere thanks also go to my colleagues and friends at UTokyo for the fun times we shared during this journey.

I am indebted to my family, Wen Yang and Yulin Chen, for their love, understanding, and support. Their belief in my abilities has been a constant source of motivation.

At last, I owe a special thanks to climbing. It has been a constant source of well-being and joy, but its greatest gift has been the friends I met along the way. Thank you all for your presence and for being such a significant part of my life.