Organizing temporally vague Raster Data in Cloud Environments for machine-learning Applications

Tobias Werner[1], Thomas Brinkhoff[2]

[1] Jade University of Applied Sciences, Germany - tobias.werner@jade-hs.de
[2] Jade University of Applied Sciences, Germany - thomas.brinkhoff@jade-hs.de

Keywords: Raster Data Management, Temporal Vagueness, Spatio-Temporal Data Management, Sovereign Cloud.

Abstract

Although geospatial time series derived from historical remote sensing and topographic maps provide critical insights into land-cover evolution, their heterogeneous structures and temporal vagueness complicate interoperability for machine-learning applications. The performance of geospatial access in cloud environments depends particularly on the formats and services used. Furthermore, raster datasets are characterized by the large amount of data, which requires efficient access for reading and writing. This paper outlines common approaches to organizing spatio-temporal raster data for use in cloud environments. It also proposes a concept for modelling temporal vagueness based on object storage. This concept provides the basis for interfacing with traditional ISO 8601 requirements.

1. Introduction

Classifying land cover using historic remote sensing imagery and topographic maps is a rapidly evolving area of study. Time series are a valuable source of information on these changes (Dorozynski et al., 2025). They form the basis for studies and are a tool for analyzing spatial processes. Time series can be used by machine-learning applications to evaluate the development of settlement areas, changes in agricultural structures, changes in forest type composition, and the spatial dynamics of coastlines and flooding zones. The research project 'The Temporal Change of Geospatial Data' is part of the Gauss Centre of the Federal Agency for Cartography and Geodesy. Its objective is to develop automated methods for classifying, storing and analyzing geospatial data of varying age and quality, to enable time series analysis of geospatial data.

The combination of spatio-temporal data from different sources typically requires the processing of heterogeneous data structures, which is challenging when using standard procedures. Furthermore, modeling spatial data management systems for historical data is challenging: In comparison to modern data acquisition, the processing of historical maps often involves temporally vague data. Therefore, we develop a raster data management system to organize temporally vague raster data in cloud environments for machine-learning applications. This paper explores commonly used data formats and systems that are suitable for use in cloud environments.

2. Related Work

Large amounts of spatio-temporal data can be provided by services (Baumann et al., 2016). Several standards have been established, especially for remote sensing, to provide large amounts of spatial data (Di and Yu, 2023). In the current era of machine-learning, managing such data is challenging and is often achieved through cloud-oriented approaches (Hogade and Pasricha, 2023). Furthermore, current trends, such as the Internet of Things, are leading to an increase in spatio-temporal data. This requires efficient systems, platforms and interfaces for analysis and further processing (Liang et al., 2024). The more data sources that are involved, the greater the potential variation in data quality. Therefore, concepts are needed to address the uncertainties in spatio-temporal data (Pawar et al., 2024).

2.1 Spatio-temporal Data Platform

A large number of open-source projects already exist for managing and providing spatio-temporal data. Typically, such projects focus on different aspects and provide different levels of support for cloud environments. This section describes typical software for managing temporal raster data that can also be run within container environments.

GeoNode[1] is an integrated open-source platform that combines proven GIS components, such as GeoServer, PostGIS, and OpenLayers, to manage and publish geospatial data via standard interfaces like Web Mapping Service (WMS) and Web Coverage Service (WCS). The platform's advanced web interface facilitates comprehensive administration and data workflow management, encompassing functions such as upload, metadata editing, styling, and publication. Fine-grained roles and permissions provide detailed access control. GeoNode is a comprehensive platform that offers a suite of interactive mapping tools, including dashboards and GeoStories. The software also supports temporal attributes, thereby enabling processing temporal raster datasets. Docker Compose[2] files and Helm charts[3] are available for deployment in orchestrated cloud environments, such as Kubernetes. Figure 1 illustrates the spatial and temporal capabilities of the GeoNode user interface for evaluating climate indices in northern Germany. The digital map shown contains a time slider that can be used to browse spatio-temporal datasets.

Also, the geOrchestra[4] spatial data infrastructure (SDI) integrates well-known GIS components into a single platform and offers a comprehensive web interface for resource management and administrative tasks. Unlike GeoNode, particular steps require a deeper understanding of the underlying GIS tools. For

[1] https://github.com/GeoNode
[2] https://docs.docker.com/compose/
[3] https://helm.sh
[4] https://github.com/georchestra

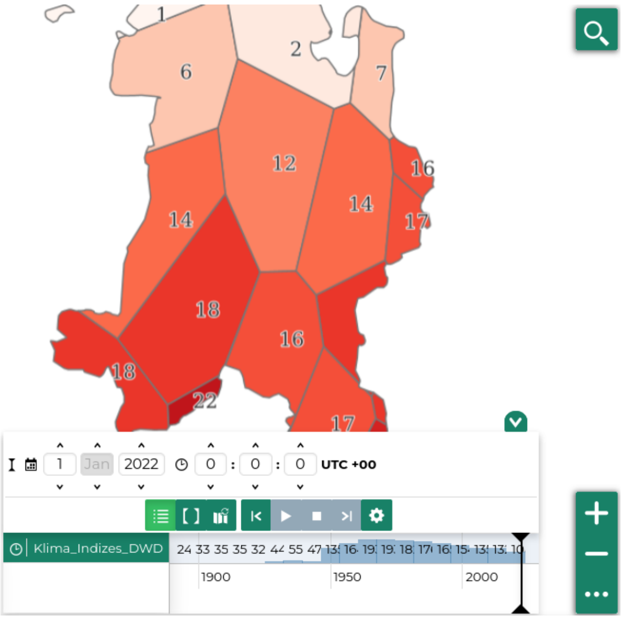

Figure 1. Illustration of the temporal capabilities in the user interface of GeoNode (voronoi diagrams calculated based on climate indices from Deutscher Wetterdienst)

example, adding time-enabled raster data usually requires manual configuration of GeoServer. GeOrchestra’s architecture is based on loosely coupled services, and an official Helm chart is available for cloud deployments.

The EOxServer[5] framework is designed to serve Earth observation data through standardized interfaces, with a strong focus on the Earth Observation profile of the Web Coverage Service (WCS). Docker files are provided to support containerized deployments of EOxServer.

OPeNDAP[6] specifies a model for server-side subsetting and delivering data via the Data Access Protocol (DAP). The focus is on accessing large amounts of data efficiently, without the need to download entire datasets. It is often bundled with file formats, such as NetCDF, which do model temporal data as an additional dimension. The Hyrax Server[7] is tailored to OPeNDAP and is available as a containerized deployment option.

The THREDDS[8] data server is designed for cataloging and aggregating large data sets. It supports multiple services beyond OPeNDAP, including WMS and WCS. As a data proxy, THREDDS can filter data and generate subsets on demand. For example, it can produce NetCDF outputs.

2.2 Temporally Vague Data

Historical raster datasets often comprise scanned maps and other analog imagery, for which the acquisition date or period of validity is only approximately known, for example historical German topographic map sheets. Therefore, concepts for modeling temporally vague raster data are required. One typical approach uses temporal primitives to define vague and fuzzy instants and periods (Bejaouia et al., 2007). Geospatial data formats and services mostly make use of ISO 8601 to encode temporal properties, which typically requires the specification of precise dates. In addition, there are many options for adding temporal attributes, such as adding metadata in the file itself or using additional metadata files, which complicates uniform processing. Although ISO 8601–2 added support for fuzzy temporal properties, this is not usually supported when dealing with spatio-temporal data. Maintaining the interoperability of existing, standard geospatial web interfaces while leveraging modern cloud services requires balancing requirements and techniques from both areas.

[5] https://github.com/EOxServer
[6] https://github.com/OPENDAP
[7] https://github.com/OPENDAP/hyrax-docker
[8] https://github.com/Unidata/tds

3. Geospatial Cloud Access

Spatial information science is well known for its rich set of formal standards, which are driven by the Open Geospatial Consortium (OGC) and complemented by widely adopted, de facto community standards (Chowdhury et al., 2024). These specifications are essential for developing interoperable spatio-temporal middleware. We have examined typical data management systems, storage concepts, and access interfaces, for their suitability in dealing with vague temporal properties and interoperability. Each of the examined components covers the capabilities to create, read, update, and delete (CRUD) raster data, which is essential in the machine-learning domain (Werner and Brinkhoff, 2025). For the integration into our spatio-temporal architecture, the following systems have been examined in more detail.

3.1 PostGIS Raster

PostGIS[9] Raster is an extension that adds capabilities for managing and processing raster data to the PostgreSQL database management system. It is used in various applications like analyzing spatio-temporal terrain models (Liu et al., 2025). For this purpose, the database system is primarily extended by the data type raster and methods for its processing. For modeling multispectral data, for example satellite imagery, such data type makes use of bands. A typical approach to load existing raster datasets into the database is to utilize the PostGIS tool `raster2pgsql`. The example in Listing 1 cuts bands 1, 2, 3 of the GeoTIFF file `dop_1_2.tiff` into 512x512 tiles and creates a corresponding SQL command for appending the results to the `raster_column` of table `raster_table`.

```
raster2pgsql -s 25832 -b 1,2,3 -t 512
    x512 -f raster_column -M  -Y  -a
    dop_1_2.tiff raster_table >
    command.sql
```

Listing 1. Using raster2pgsql to import a GeoTIFF file into PostGIS

In the context of database fundamentals, temporal attributes can be added by appending datetime columns. However, modeling temporal vagueness is challenging and requires further elaboration of concepts in terms of structure and queries. Furthermore, advanced cloud environments are characterized by the primary utilization of HTTP(S). Depending on the infrastructure, applications like machine-learning may not have a direct database connection. Thus, a layer that provides database capabilities through a REST API, such as PostgREST[10], is necessary. This

[9] https://postgis.net/
[10] https://docs.postgrest.org

layer can be used to encapsulate queries, such as preparing a raster for machine-learning purposes. For instance, server-side augmentation can be achieved by chaining the PostGIS functions ST_Resize, ST_Reskew, and ST_Resample.

3.2 Cloud Optimized GeoTIFF (COG)

The COG[11] is a cloud-native data raster format that is based on the commonly used GeoTIFF. In contrast to its predecessor, it organizes data in a certain structure that proposes a basis for satisfying the technical requirements of HTTP range queries. Therefore, it’s a well-known and efficient approach for accessing raster data in cloud environments like performing land cover queries (McQuade et al., 2025).

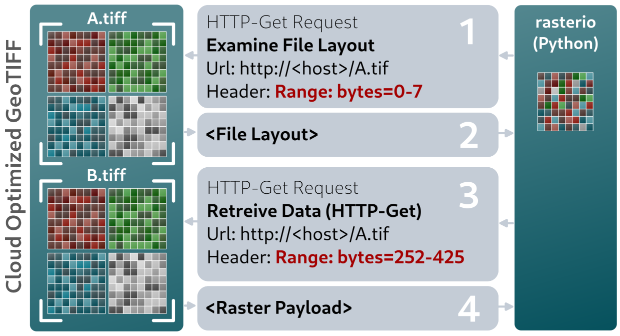

Figure 2. Conceptual illustration of querying data from a Cloud Optimized GeoTIFF

Figure 2 illustrates the main concept of COG. The raster datasets `A.tiff` and `B.tiff` were served via an HTTP server. First, a COG-compatible client, for example, the Python library Rasterio[12], performs an HTTP request that contains a header to read a range of bytes. The header contains metadata that Rasterio can use to examine the file layout. With this information, Rasterio can translate spatial queries into the corresponding byte ranges in `A.tiff` and perform HTTP range queries to retrieve the data.

Although temporal information can be added via metadata according to the PostGIS Raster standard, modeling temporal vagueness requires further elaboration. Unlike a database approach, which offers flexible data models, adding further data structures to COG is limited. Furthermore, machine-learning applications usually process large amounts of data. As illustrated, comprehensive datasets, such as satellite imagery, consist of multiple COG files. Performing queries requires knowledge of the affected files.

3.3 Zarr

Zarr[13] is a cloud-native data format that has recently become popular. It utilizes arrays to organize data as cubes. Due to its efficient access capabilities, it is often compared to the previously described COG to share mass raster data. One of the main differences is its file structure. Whereas COG follows a one file per tile approach, a Zarr datacube typically consists of many files (Open Geospatial Consortium, 2022). The use of COG and Zarr particularly benefits from the capabilities of S3-compatible object storage, such as scalability and organizational structures.

[11] https://cogeo.org/
[12] https://rasterio.readthedocs.io
[13] https://zarr.dev/

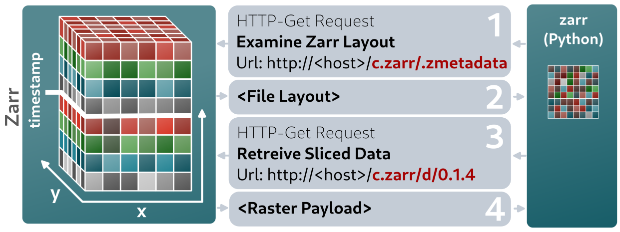

Figure 3. Conceptual illustration of querying data from a Zarr dataset

Figure 3 illustrates the key principles of Zarr. First, the metadata is queried to examine the layout of the data cube for identifying dimensions, compression, and further properties. Depending on version and structure, the `.zmetadata` file typically contains consolidated metadata. In comparison to COG, Zarr uses multiple files for representing raster data. Due to chunking, each dimension groups values into chunks. Such chunks can be referenced via indices. The illustrated retrieval of data accesses the three dimensional cube `c.zarr` at indices 0, 1, 4. However, adding vague temporal attribution requires additional concepts. Especially gridded structures, such as Zarr, are characterized by equally distributed intervals.

3.4 Rasdaman

Rasdaman is a project focused on managing and serving spatio-temporal raster datasets via standardized interfaces. It utilizes the OGC Web Coverage Service to provide a standardized, interoperable interface that can be used by compatible WCS clients (Baumann et al., 2018).

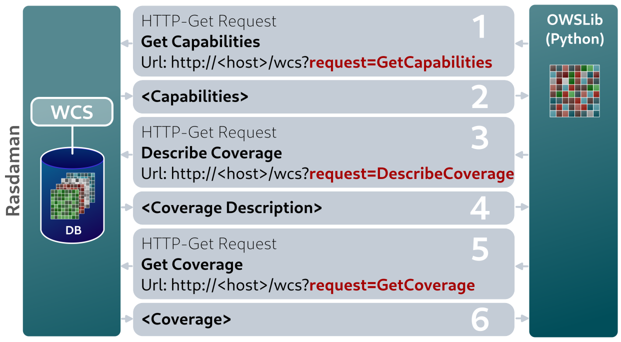

Figure 4. Conceptual illustration of querying data from a Web Coverage Service

Figure 4 illustrates the main concept of such an interface. First, the client queries the capabilities of the WCS server. The resulting capabilities document lists the provided datasets and their general technical properties. Next, a detailed description of the target dataset is queried, forming the basis for querying the coverage. Furthermore, Rasdaman is a reference implementation of the Transactional Web Coverage Service (WCS-T). Additional concepts have been added to enable the creation, modification and deletion of temporal raster data.

4. Architecture

Based on the investigation of the four approaches presented in the previous section, we designed an architecture for managing temporally vague raster data in cloud environments. The architecture developed is shown in Figure 5 and makes use of the layered structure in cloud environments. It consists of the following main components:

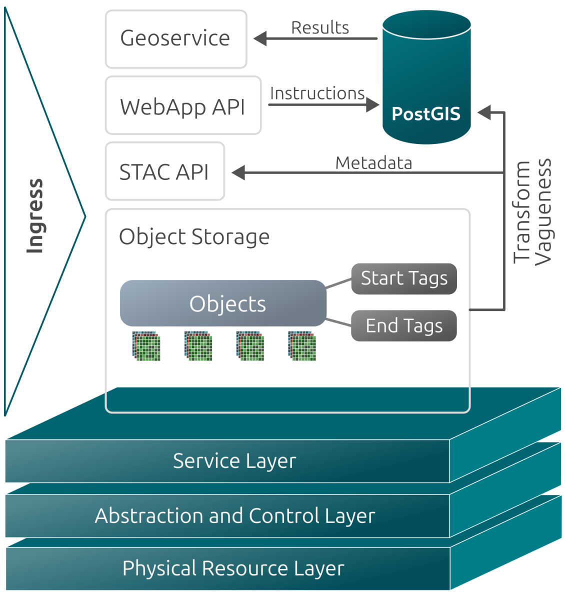

Figure 5. Simplified architecture for managing temporally vague raster data based on the Cloud Computing Reference Architecture in (National Institute of Standards and Technology, 2011)

- The ingress provides access to the data management system and routes requests to the relevant components.
- The object storage acts as a centralized, scalable storage solution for spatio-temporal raster data.
- The Spatio-Temporal Asset Catalogue (STAC) API allows spatio-temporal queries and provides an interface for listing raster data that satisfies a given query.
- PostGIS Raster as part of the PostGIS database enables the application of applying custom map algebra and the provision of processed raster resources.
- The WebApp API instructs raster calculations and is responsible for configuring appropriate Web Map Services and Web Coverage Services.

COGs and Zarr datasets can be natively extended by key-value pairs (tags). As Figure 6 illustrates, we use these tags to store the known temporal components of timestamps; unknown components are omitted. Each component is represented by its own tag, such as century, year, month and day. This allows us to define and evaluate vague instants as well as fuzzy intervals platform-independently. In order to satisfy ISO 8601 in spatial interfaces, we perform a transformation by evaluating those vague instants and intervals and expanding the dates to their temporal extremes. The spatial and temporal extent is then added to geoservices, such as the STAC API.

5. Application

The current trend is to use cloud resources to perform classification by machine-learning applications. Sovereign data management solutions and self-managed infrastructures are becoming

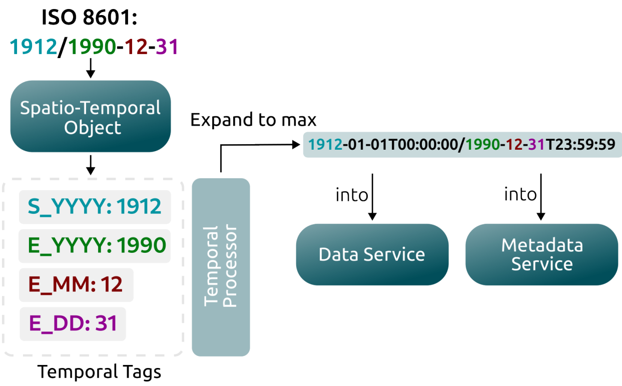

Figure 6. Modeling temporal vagueness using temporal tags on object storage

increasingly popular as soon as sensitive data is processed. Independent environments for cloud computing are also being promoted by governmental efforts, such as IPCEI Next Generation Cloud Infrastructure and Services (IPCEI-CIS) (European Commission, 2023). Other advantages of not using public cloud providers include flexible, customizable infrastructure and avoiding budget overruns. The proposed architecture for accessing temporally vague raster data is intended for use in sovereign cloud environments. It offers an interoperable and extendable solution for managing and processing sensitive databases. Adding fuzzy temporal properties on object levels provides flexible access and a uniform definition across multiple data formats. The use of client-side frameworks to access S3 object storage directly, for example, the Virtual Storage Interface of GDAL (vsis3), allows efficient raster processing. Results can be found and added without major obstacles using desktop GIS software, such as QGIS. Especially machine-learning applications benefit by performing exact queries on vague raster to find temporally suited datasets. By the integration of the PostGIS database, raster preprocessing steps such as reprojection and map algebra can be applied on the server without transferring data.

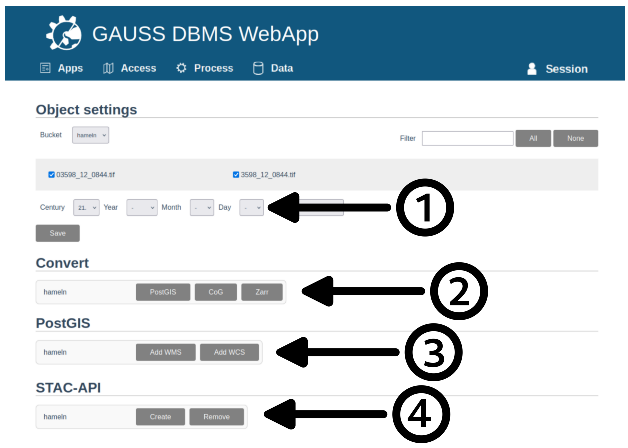

Figure 7. Excerpt of the developed web application for data administration

5.1 Web interface

Figure 7 illustrates the prototype of a web application for managing temporal vague raster data within the proposed architecture. To update the temporal attribution of uploaded datasets, date components can be set in step 1. Next, the spatio-temporal data can be converted into various formats. In particular, the proposed cloud-friendly formats COG and Zarr. Importing into the PostGIS raster-enabled database is also available. Based on PostGIS raster datasets, OGC web services, such as WMS and WCS, can be created in step 3. Finally, spatio-temporal extents and further technical metadata are derived in step 4 to integrate them into the STAC API.

5.2 Evaluation

Figure 8 depicts a boxplot of the performance of creating COG datasets on integrated object storage. The x-axis shows the number of concurrent working agents used to create the COG datasets. The y-axis displays the time taken to process the entire test dataset. The proposed agents work on a queue to read, convert and write tiles individually. Evaluation was performed on a single node with 16 CPU cores using a raster dataset consisting of 275 tiles (216 MB, three bands, 512 x 512 pixels, 0.2 m/pixel). The boxplot shows that COG writing processes in object storage benefit from working in parallel. Similar results can be expected when using further cloud-friendly data formats like Zarr on object storages, depending on the model within the used data structure. Thus, writing classification results in machine-learning can result in massive speed increases when working concurrently. By contrast, WCS-T-based systems may be limited when performing parallel write operations.

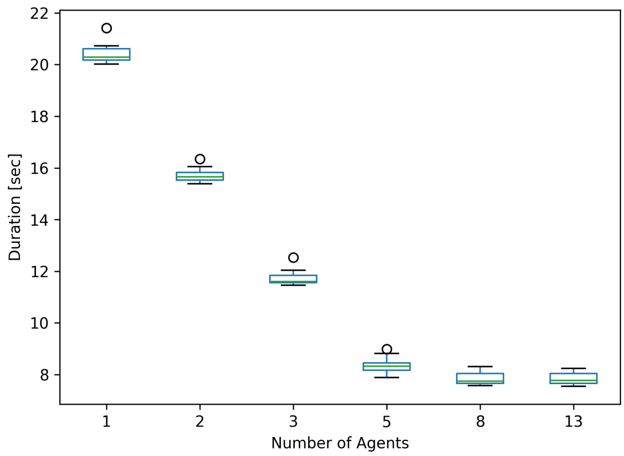

Figure 8. Time consumption of concurrent creation of COG on S3 object storage

5.3 Usage of STAC API

The STAC API uses collections to organize spatio-temporal items within catalogs. These collections and items have metadata for describing their temporal and spatial properties. An example search is illustrated in Listing 2.

```
https://<host>/pycsw/stac/collections/
    hameln/items?datetime=2000-01-01T00
    :00:00Z/2020-01-31T23:59:59Z&bbox
    =9.2,52.0,9.5,52.2
```

Listing 2. Using the STAC API to find datasets

This example searches for all items in the `hameln` collection whose temporal interval is between `2000-01-01T00:00:00` and `2020-01-31T23:59:59`, and whose location is within the bounding box of `9.2, 52.0, 9.5, 52.2`. Performing the shown query as a HTTP GET request returns a list of documents in JavaScript Object Notation (JSON), which makes it straightforward for any client software to utilize. Based on the expansion of the temporal vague data in Section 4, the STAC API shown returns possible candidates within the given interval. However, it does not return candidates with a temporally overlapping interval. Advanced temporal and spatial query conditions, such as overlap, can be applied using the Common Query Language (CQL2). This OGC standard provides support for filters and functions, including support for the Dimensionally Extended 9-Intersection Model (DE-9IM) (Open Geospatial Consortium, 2024).

6. Conclusion and Outlook

This paper has outlined selected approaches and highlighted aspects that are relevant when choosing data formats and systems for organizing vague temporal data in cloud environments. Prominent open-source software for managing spatio-temporal data is often already containerized, providing a solid foundation for running in cloud environments. However, providing geospatial access to temporally vague data is challenging. Our approach of using broadly applied object storage in cloud environments to model temporal vagueness by tags can be applied to widely used, cloud-friendly spatial data formats. Expanding such temporal properties makes them compatible with the requirements of ISO 8601, thereby enhancing their interoperability. This allows geoservices and metadata systems, such as the STAC API, to connect to large amounts of spatio-temporal data via cloud-oriented storage systems.

Further work is required to validate our approach to modeling temporal vagueness in cloud environments. In particular, while the proposed CRUD-oriented approaches appear to meet the requirements of machine-learning applications, a more detailed evaluation using practical datasets is needed. Moreover, the proposed concepts involve the use of data structures with temporal capabilities. Leveraging these abilities in conjunction with the developed architecture could enhance efficiency.

7. Acknowledgements

This contribution is part of the research project 'The temporal change of geospatial data' and is funded by the Gauss Centre of the Federal Agency for Cartography and Geodesy.

References

Baumann, P., Merticariu, V., Dumitru, A., Misev, D., 2016. Standards-based services for big spatio-temporal data. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLI-B4, 691–699. doi.org/10.5194/isprs-archives-XLI-B4-691-2016.

Baumann, P., Misev, D., Merticariu, V., Huu, B. P., Bell, B., 2018. rasdaman: Spatio-temporal datacubes on steroids. *Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '18, Association for Computing Machinery, New York, NY, USA, 604–607. doi.org/10.1145/3274895.3274988.

Bejaouia, L., Bédard, Y., Pinet, F., Salehi, M., Schneider, M., 2007. Logical consistency for vague spatiotemporal objects and relations. *5th International Symposium on Spatial Data Quality SDQ 2007, Modelling qualities in space and time*, ISPRS Archives - XXXVI-2 pt C 43 WG II-7, Enschede, The Netherlands.

Chowdhury, S., Schröder, D., Ostadabbas, H., Hosseingholizadeh, M., Friesecke, F., 2024. Modernizing geospatial services: an investigation into modern OGC API implementation and comparative analysis with traditional standards in a web application. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLVIII-4/W12-2024, 11–18. doi.org/10.5194/isprs-archives-xlviii-4-w12-2024-11-2024.

Di, L., Yu, E., 2023. *Standards for Big Data Management*. Springer International Publishing, Cham, 107–133. doi.org/10.1007/978-3-031-33932-5_7.

Dorozynski, M., Rottensteiner, F., Thiemann, F., Sester, M., Dahms, T., Hovenbitzer, M., 2025. Multi-modal Land Cover Classification of Historical Aerial Images and Topographic Maps Exploiting Attention-based Feature Fusion. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences.*, 10(G-2025), 221–229. doi.org/10.5194/isprs-annals-X-G-2025-221-2025.

European Commission, 2023. Important Project of Common European Interest on Next Generation Cloud Infrastructure and Services (IPCEI-CIS) – RRF. www.bundeswirtschaftsministerium.de/Redaktion/EN/Downloads/D/decision-of-the-european-commission-regarding-the-ipcei-cis.pdf (27 March 2026).

Hogade, N., Pasricha, S., 2023. A Survey on Machine Learning for Geo-Distributed Cloud Data Center Management. *IEEE Transactions on Sustainable Computing*, 8(1), 15-31.

Liang, H., Zhang, Z., Hu, C., Gong, Y., Cheng, D., 2024. A Survey on Spatio-Temporal Big Data Analytics Ecosystem: Resource Management, Processing Platform, and Applications. *IEEE Transactions on Big Data*, 10(2), 174-193.

Liu, R., Kuper, P., Grombein, T., Naab, C., Al-Doori, M., Breunig, M., 2025. Towards Integrated Data Management and Analysis for Spatio-Temporal Digital Terrain Models. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, X-G-2025, 543–550. doi.org/10.5194/isprs-annals-X-G-2025-543-2025.

McQuade, L., Sudmanns, M., Tiede, D., 2025. Efficient aggregate land cover queries with cloud-optimized raster formats. *2025 International Conference on Machine Intelligence for GeoAnalytics and Remote Sensing (MIGARS)*, 1–4. doi.org/10.1109/MIGARS67156.2025.11231788.

National Institute of Standards and Technology, 2011. Nist cloud computing reference architecture. nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication500-292.pdf (27 March 2026).

Open Geospatial Consortium, 2022. OGC Testbed 17: COG/Zarr Evaluation Engineering Report. opengis.net/doc/PER/t17-D047 (27 March 2026).

Open Geospatial Consortium, 2024. Common Query Language (CQL2). docs.ogc.org/is/21-065r2/21-065r2.html (27 March 2026).

Pawar, S. D., Pawar, V. S., Abimannan, S., 2024. *Handling Uncertainty in Spatiotemporal Data*. Springer Nature Singapore, Singapore, 69–87.

Werner, T., Brinkhoff, T., 2025. Efficient Management of Spatio-Temporal Raster Data in Sovereign Cloud Environments for Machine Learning Applications. *AGIT Conference*, 1, 1. doi.org/10.25598/agit/2025-58.