CartoGraph: Graph-Based Reference Answers for Map Visual Question Answering Benchmarks

Rene Dorsch[1], Daniel Henselmann[1] and Andreas Harth[1,2]

[1]Fraunhofer IIS, Nordostpark 84, 90411 Nuremberg, Germany
[2]Friedrich-Alexander-Universität Erlangen-Nürnberg, Lange Gasse 20, 90403 Nuremberg, Germany

Abstract

In a map visual question answering benchmark, each reference answer must match the content shown in its associated map image. Benchmark constructors obtain such answers by interpreting the map image manually, which scales poorly to complex maps, or by extracting them from the geospatial source data, which can include content that the image does not show. We present CartoGraph, a map-production system that generates map images together with their reference answers. While producing a map, CartoGraph records in a map-content graph which source features appear in the map image and how they are drawn. CartoGraph renders the map image from this graph and question-specific queries derive the reference answers from the same graph. Because the image and the answers share one recorded map content, each answer is restricted to the content its map image shows and requires no manual interpretation. An initial implementation for the OpenStreetMap Standard Map, evaluated across 200 map images at zoom levels 16 and 17, demonstrates that the map image and the reference answers for street-name enumeration and spatial-relation questions can be derived from the same map-content graph.

Keywords

Visual Question Answering, Geospatial Knowledge Graph, Benchmark, Map

1. Introduction

Benchmarks evaluate systems on a defined test set. The test set in a map visual question answering (map-VQA) benchmark consists of test items that combine (i) a map image, (ii) a question about the map image and (iii) a reference answer.

The map image is the visual input to the evaluated system. The map image depicts geographic features based on their geometries and properties through display elements such as points, lines, symbols and labels. A map-production system produces the map image by determining which content from the geospatial source data appears and how it is represented (Figure 1). The number and meaning of the visible display elements describe the complexity of a map image [1]. The complexity of a map image can therefore vary along a continuum from low to high.

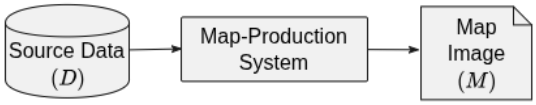


Figure 1: A map-production system transforms geospatial source data D into a map image M.

The question about the map image specifies which information the evaluated system must retrieve. A question may ask for represented features, their properties or spatial relationships between them. Questions are created manually from the map images or generated from structured data (e.g. templates)

First International Workshop on Spatial Intelligence and Reasoning Enabled by Knowledge Graphs and Foundation Models (SPARK 2026), in ISWC 2026 Workshops Joint Proceedings, October 25–26, 2026, Bari, Italy
rene.dorsch@iis.fraunhofer.de (R. Dorsch); daniel.henselmann@iis.fraunhofer.de (D. Henselmann); andreas.harth@iis.fraunhofer.de (A. Harth)
0000-0001-6857-7314 (R. Dorsch); 0000-0001-6701-0287 (D. Henselmann); 0000-0002-0702-510X (A. Harth)
© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073
published 2026-11-05

CEUR-WS.org/Vol-4313/SPARK-paper3.pdf

for the construction of a test set. Regardless of the questions construction method, each question must refer to information that is available in its associated map image.

The reference answer to the question and the map image defines the expected response. A change to the question or map image can change the reference answer, even when the geographic area remains unchanged. Benchmark constructors obtain reference answers by either (i) inspecting the map image manually or with machine assistance, or (ii) by extracting information from the source data used to produce the map.

The first strategy [2, 3, 4, 5] restricts the reference answers to the content represented in the map image, but requires each map image to be inspected manually. Figure 2 illustrates this task for the question "What are the names of all streets shown on the map?" To construct the reference answer, a benchmark constructor must identify every visible street label, associate each label with a street geometry and determine whether clipped or partially visible elements count as shown. Maps can vary in complexity as shown in Figure 2. Among the three zoom levels, the map image at zoom level 17 represents more visible elements and is therefore more complex than the maps at zoom levels 16 and 15. Greater complexity requires the benchmark constructor to identify more labels and resolve more boundary cases. The greater complexity limits scalability and increases the risk of omissions and inconsistent decisions.

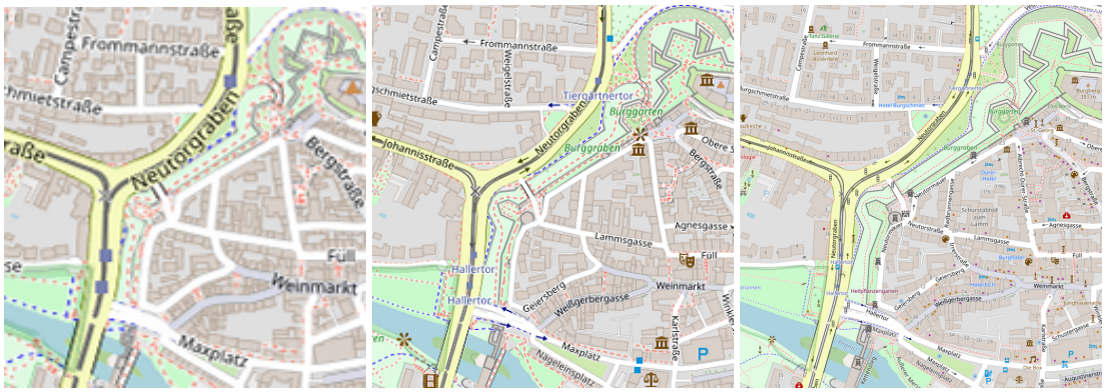


Figure 2: OpenStreetMap (OSM) map extracts around Neutorstraße in Nuremberg at zoom levels 15, 16 and 17 (from left to right). Changes in zoom level alter the represented content and its degree of map complexity.

The second strategy [6, 7] avoids the manual interpretation and supports programmatic construction, but assumes that the source data and the map image contain the same content. A map-production system violates this assumption when it selects, simplifies, aggregates, displaces or omits source data before visualization in a map image, as shown in Figure 3. The source data may therefore contain named streets whose labels are not represented in a map image. For instance, given the map images in Figure 2, a street label shown at zoom level 17 may be absent at zoom level 15 even though both map images are produced from the same source data. Deriving the reference answer for the map at zoom level 15 directly from that source data may therefore include a street label that the evaluated system cannot observe in the map image.

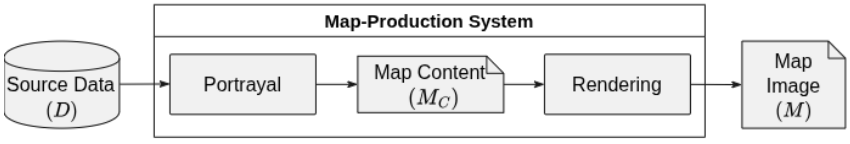


Figure 3: The map-production process transforms source data D into map content before rendering the map image M. The map content contains the display elements that retain from the source data.

Neither strategy therefore constructs reference answers that are both scalable and restricted to the actual map content. From these limitations, we derive three requirements for map-VQA test-set construction:

- **Scalability:** The test set must be constructible for maps with different degrees of map complexity without manual annotation.
- **Consistency:** The test set construction procedure must apply the same rules across test items and return the same reference answer for the same map image and question.
- **Map-content fidelity:** Each test item of the test set must contain a reference answer restricted to the geographic features, their properties and the spatial relationships represented in its map image.

All three requirements point to the same missing artifact, the map content. The map content comprises the display elements that the map-production system retains between the source data and the map image (Figure 3). Map-production systems determine this content internally but do not expose it. The first strategy recovers this content from the drawn map image, whereas the second approximates it through the source data from which it was produced. In a machine-readable and queryable form, the map content would support programmatic answer derivation while restricting each answer to the content represented in the map image.

We present CartoGraph, a map-production system that generates a map image together with its reference answers from the map content. CartoGraph takes geospatial source data together with a description of which map to produce and how to draw it. It selects and resolves the map-specific representations of geographic features, records the resulting display elements in a map-content graph and renders the map from that graph. Question-specific queries derive reference answers from the same graph. This shared representation makes the answers programmatically reproducible while restricting them to the content represented in the generated map. We implement the approach for the OSM Standard Map and demonstrate it with street-name enumeration and spatial-relation questions. The resulting answers are defined relative to the recorded source data and the description used to produce the map rather than to geographic reality. We evaluate CartoGraph using 200 map images paired with their final map-content knowledge graphs. The test set covers the OSM Standard Map at zoom levels 16 and 17. We discuss how shared map-and-answer derivation addresses the three requirements for benchmark construction and examine its limitations.

We focus in this paper on the construction of reference answers for map images. We do not evaluate question-generation strategies or the performance of map-VQA systems.

The remainder of this paper is organized as follows. Section 2 reviews existing approaches to map-VQA benchmark construction. Section 3 models the transformation from geospatial source data to map content, and Section 4 presents CartoGraph. Section 5 instantiates CartoGraph using the OSM-based example. Section 6 discusses the benefits and limitations of the approach, and Section 7 concludes the paper.

2. Related Work

We first introduce relevant map types and dimensions used to compare map-VQA benchmarks. We then review existing benchmarks and compare them with CartoGraph.

Most of the reviewed benchmarks contain either reference maps or thematic maps [8]. Reference maps, such as OSM Standard Maps [9], provide location information and represent geographic features (e.g. roads, buildings, water bodies) accurately and in detail. Thematic maps instead emphasize how a selected phenomenon, such as population density, is distributed across space, and represent it using graphical variables such as color or symbol size. Maps that combine both types or fall outside them we classify as mixed.

We compare map-VQA benchmarks along the following dimensions:

- **Map type:** Identifies whether a benchmark contains reference maps (Reference), thematic maps (Thematic) or mixed (Mixed).
- **Map complexity:** Indicates the range of map complexity, from low to high, that the benchmark's map images cover, based on the number and meaning of the visible display elements.

- **Reference-answer source:** Identifies whether an answer is derived from the map image (Map image), the geospatial source data (Source data) or a machine-readable representation of the map content selected during the map-production process (Map content).
- **Reference-answer construction:** Indicates whether people author the answers manually (Manual), people refine machine-generated drafts in a machine-assisted process (Machine-assisted) or software derives the answers programmatically from encoded data or rules (Programmatic).

Reference or mixed maps appear in MapEval-Visual, MapReader and MapVerse. MapEval-Visual [2] uses Google Maps map images with a low degree of map complexity and expert-created questions and answers. MapReader [5] also uses Google Maps map images, while GPT-4o generates an initial set of questions and answers that two experts subsequently refine. MapVerse [3] uses real-world map images from diverse sources and creates question–answer pairs manually, which two experts independently verify.

Thematic maps are the focus of MapQA, MAPWise and MapIQ. MapQA [7] combines existing and generated choropleth map images and derives question–answer pairs from their underlying source data. MAPWise [4] generates choropleth map images from socioeconomic data, after which experts manually construct question–answer pairs from predefined templates. MapIQ [6] generates choropleth map images, cartograms and proportional-symbol map images and extracts their reference answers programmatically from the geospatial source data. A human expert subsequently reviews the completed question–answer dataset as a quality-control step.

Table 1 summarizes the reviewed benchmarks along the four dimensions.

Table 1
Comparison of map-VQA benchmark construction approaches.

Benchmark	Map type	Map complexity	Reference-answer source	Reference-answer construction
MapEval-Visual [2]	Reference	Low	Map image	Manual
MapReader [5]	Reference	Low	Map image	Machine-assisted
MapVerse [3]	Mixed	Low to high	Map image	Manual
MapQA [7]	Thematic	Low	Source data	Programmatic
MAPWise [4]	Thematic	Low	Map image	Manual
MapIQ [6]	Thematic	Low	Source data	Programmatic
CartoGraph	Reference	Low to high	Map content	Programmatic

In summary, existing benchmarks use either the map image or the source data as their reference-answer source. CartoGraph instead uses the map content and is, among the reviewed approaches, the only one that constructs reference answers programmatically for reference maps. It records this content in a machine-readable graph that is used both to render the map image and to answer questions. Because the graph replaces per-image interpretation, CartoGraph supports programmatic reference-answer construction for reference maps with varying degrees of complexity.

3. Modeling the Map-Production Process

Deriving reference answers from map content requires understanding how geospatial source data is transformed into a map image. We must therefore identify the stage at which a map-production system has made content and layout decisions and after which the resulting elements shown in the map image are still available in a machine-readable form. This section reviews (i) how existing models divide the map-production process, (ii) synthesizes them into a three-stage model that names the information required in each stage and (iii) shows how the model is realized in an existing rendering stack.

3.1. Existing Models of the Map-Production Process

Existing models describe map production as a sequence of stages, each with defined inputs and a defined output. Cuthbert introduced a model for map production with the four stages selection, generation, rendering and display [10]. The selection stage returns the source data to be displayed, the generation stage turns this data into display elements, the rendering stage constructs a map image from these elements, and the display stage makes the map image visible. A display element is a graphical record produced for a feature, such as the casing of a road or the text of its name. The StyleInfo and DisplayInfo govern these stages. The StyleInfo input describes how the source data is represented in the map image, whereas the DisplayInfo input describes the spatial reference system, the extent and the scale of the display. Because the two are separate, the same StyleInfo can be used with different DisplayInfo to display different areas of the same source data. The OGC Web Map Server specification adopts these four stages and defines the display elements that the generation stage passes to the rendering stage [11].

ISO 19117 divides the same sequence into two stages, performed by a portrayal engine and a rendering engine [12] (see also Figure 3). The portrayal engine applies portrayal functions, which map source features of the source data to references of the symbols that should be displayed on the map image. The rendering engine draws these symbol references into the map image using the corresponding symbol definitions. The portrayal engine therefore covers the selection and generation stages, and its symbol references play the role of the display elements above. The rendering engine covers the rendering stage, whereas the display of the map image lies outside the standard. A portrayal catalogue collects the portrayal functions and symbols so that they can be used independently of the source data. The standard also names the decisions that adjust symbols relative to one another portrayal finishing and excludes their rules, such as generalization, the resolution of overprinting and displacement rules, from its scope.

The models agree in the sequence of the stages but differ in where they place the decisions that require more than one display element. ISO 19117 excludes them from its scope as portrayal finishing, Cuthbert assigns the ordering of styles to the StyleInfo but leaves clipping unassigned [10], and the Web Map Server specification fixes only the drawing order [11]. Label placement and collision handling appear in none of the models, although these decisions determine which display elements the map image finally shows. Consequently, no model exposes a machine-readable artifact that records the map content after these decisions, which we call the final map content in the following.

3.2. Three-Stage Model of the Map-Production Process

Based on these models, we divide the map-production process into rule application, display-element resolution and map-image rendering, as shown in Figure 4.

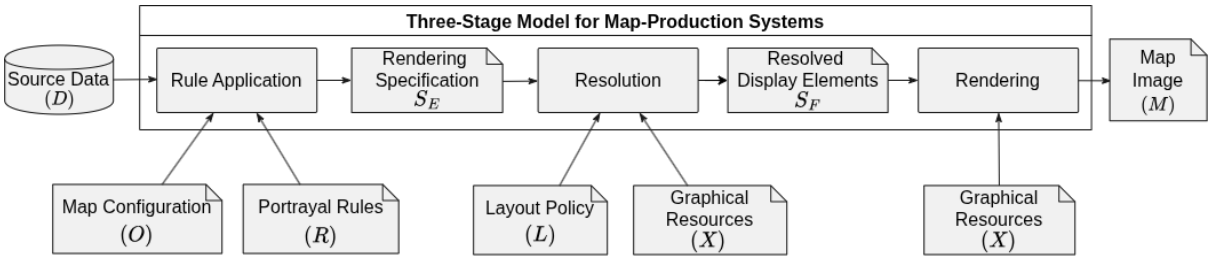


Figure 4: General model of a map-production system with two decision stages and a concluding map-image rendering

Written as a sequence of transformations, the three stages are:

$$D \xrightarrow[O,R]{f_{\text{rule}}} S_E \xrightarrow[L,X]{f_{\text{resolve}}} S_F \xrightarrow[X]{f_{\text{render}}} M,$$

where D denotes the geospatial source data and M the map image. The stages differ in the scope of the

decisions they make and in the information that governs these decisions. The map configuration (O) applies to the process as a whole and specifies the map extent, coordinate reference system, output size and format, and identifies the map product to be produced.

As a running example, consider the road and the label *Neutorstraße* in the zoom-level-17 map (M) in Figure 2 (right), whose configuration (O) specifies the shown extent, coordinate reference system and output dimensions and format, and selects the OSM Standard Map at zoom level 17. We follow a source road feature carrying this name in the OSM source data (D) through the three stages to explain how its geometry and name become visible display elements.

The rule-application stage (f_{rule}) decides for each source feature independently of every other how that feature would be represented. It applies the portrayal rules (R), which select source features for the map and derive candidate display elements from them. For each element, the rules specify the transformed and generalized geometry, the symbolization, the label text and the drawing priority. The resulting artifact (S_E) collects these display elements and represents the expected map content. Because the interactions between display elements are not covered at this stage, S_E can only over-approximate the map content. In the running example, D supplies the road geometry and its name, and R produces candidate road and label elements in S_E for the map selected by O; the label has not yet been placed.

The display-element resolution stage (f_{resolve}) resolves the candidate display elements into the elements the map image shows. Its decisions depend on interactions between display elements. The stage places labels and symbols, resolves collisions, determines the drawing order and decides how to treat elements that cross the image boundary. Two further types of information govern these decisions. The layout policy (L) collects the placement, collision, drawing-order and boundary policies. The graphical resources (X) are the assets referenced by the portrayal rules, such as fonts, icons, line patterns and fill patterns. They determine the space each display element occupies and its visual appearance. The resulting artifact (S_F) records each resolved display element and represents the final map content. For the *Neutorstraße* label, this stage uses the font in X to determine the space occupied by the label and the policies in L to choose a placement along the road, recording the result in S_F.

The map-image rendering stage (f_{render}) draws the resolved display elements into the map image. For the drawing, it requires the drawable content of the graphical resources X, such as glyph outlines, icon bitmaps and pattern tiles. No content or layout decision remains at this stage. Because S_F is expressed in the image coordinates established by O, the stage also requires no further map configuration. The stage must resolve the same X as display-element resolution, because substituting a font between the stages would invalidate the computed placements and S_F would no longer describe M. Rendering then draws the road and the placed *Neutorstraße* label from S_F into M.

The three stages reorganize the stages of the models from Section 3.1. Rule application combines selection and generation, whereas display-element resolution and map-image rendering divide rendering, because the displayed content is fixed before the drawing begins. In ISO 19117 terms, rule application corresponds to the portrayal engine, whereas display-element resolution and map-image rendering correspond to the rendering engine. The information governing the stages has counterparts as well. The map configuration O plays the role of the DisplayInfo of Cuthbert's model, and the portrayal rules R play the role of its StyleInfo and of the ISO 19117 portrayal functions. The candidate display elements in S_E correspond to the display elements and symbol references that the models exchange between their stages, and the ISO 19117 portrayal catalogue, which collects portrayal functions and symbols, corresponds to R and X taken together. The existing models cover only parts of the layout policy L and provide no counterpart for the final map content S_F, as Table 2 shows.

The boundary between the first two stages follows the scope of a decision rather than the type of the operation. Simplifying the geometry of one feature therefore belongs to rule application, whereas displacing an element because another element occupies its position belongs to display-element resolution, although cartographic practice calls both operations generalization. Neither the StyleInfo nor the ISO 19117 portrayal functions restrict a rule to a single source feature, so the restriction to feature-local decisions is a commitment of our model rather than one inherited from the existing models.

Table 2
Coverage of the three-stage-model terms in existing models of the map-production process. A full counterpart covers the role of the term, a partial counterpart covers isolated aspects, and '–' marks a missing counterpart.

	Cuthbert [10]	WMS [11]	ISO 19117 [12]
source data D	full	full	full
map configuration O	full	full	partial
portrayal rules R	full	full	full
layout policy L	partial	partial	–
graphical resources X	full	full	full
expected map content S_E	full	full	full
final map content S_F	–	–	–
map image M	full	full	full

3.3. The Three-Stage Model in Practice

Together, the portrayal rules (R), the layout policy (L) and the graphical resources (X) form the map-product specification, which is independent of the map configuration O. The OSM Standard Map [9] and the OSM Transport Map [13] are examples of map products defined by such specifications. A scale or zoom level belongs to the map-product specification rather than to the map configuration. The Digitale Topographische Karte 1:100 000 (DTK100) [14] defines its content for a single scale, whereas the OSM Standard Map defines its rules per zoom level, so a map configuration selects a zoom level only by identifying the specification that applies at it.

In practice, a map-product specification is rarely provided as a single artifact. OpenStreetMap Carto [15] illustrates this distribution for the OSM Standard Map. The import configuration and the CartoCSS stylesheets provide the portrayal rules R, the tile scheme supplies the map configuration O for each requested tile, the placement parameters together with Mapnik's collision handling and the configured layer order specify the layout policy L, and the fonts, icons and patterns referenced by the stylesheets provide the graphical resources X. The layout policy is thus only partly declarative, because its collision-handling part exists only as Mapnik's behavior. Mapnik then produces the tiles by placing labels, resolving collisions, clipping elements at tile boundaries and drawing the resolved elements.

The three-stage model identifies S_F as the point at which all content and layout decisions are made, but a map-production system does not need to retain S_F. The map image shows the final map content but retains no machine-readable description of the resolved display elements. Deriving reference answers from the final map content therefore requires a system that records these elements and the outcomes of the decisions that produced them.

4. CartoGraph

We introduce in this section CartoGraph, a map-production system that follows the three-stage model of Section 3. CartoGraph takes geospatial source data (D), a map configuration (O) and a map-product specification (R, L, X) as input and records the display elements and their resolution outcomes in a map-content graph. A map-image renderer derives the map image from this graph. Question-specific graph queries can be applied during test set construction or added later to derive reference answers. Because each query operates on the map-content graph used to produce the map image, additional questions can reuse the existing graph and map image while remaining restricted to the final map content.

CartoGraph realizes the three stages as five operations, which construct the intermediate graphs and derive the map image and reference answers:

$$D \xrightarrow{\text{lifting}} G_I \xrightarrow[O_i, R_G]{f_{\text{rule}}} G_{E,i} \xrightarrow[L_G, X_G]{f_{\text{resolve}}} G_{F,i} \begin{cases} \xrightarrow{f_{\text{render}}} M_i \\ \xrightarrow{q_j} A_{i,j} \end{cases}$$

The five operations define the notation used in the chain:

1. **Knowledge-graph construction ($D \rightarrow G_I$):** The lifting operation turns the source data D into its knowledge-graph representation G_I. It represents source-feature identifiers, types, properties and geometries without adding map-product-specific information. It encodes each geometry as a spatial literal so that subsequent operations can evaluate spatial predicates and functions. Separate source features remain separate graph resources even when they share a property value, such as a street name, and the resulting graph preserves links to the corresponding source features in D.
2. **Expected map-content construction ($G_I \rightarrow G_{E,i}$):** The rule-application function takes the map configuration O_i of map i and the portrayal-rule representation R_G as additional inputs. Because O_i selects a map extent within D, several maps can be derived from the same source data and its knowledge-graph representation. R_G instantiates the portrayal rules R as graph queries that select the source features from G_I whose types and properties satisfy the rule conditions within this extent. The link from O_i to the map-product specification determines which portrayal rules apply. For instance, for a product such as the OSM Standard Map, which defines its rules per zoom level, this link includes the zoom level. The queries materialize the label strings, geometries, styles and drawing priorities specified by the portrayal rules, while linking each selected element to its source feature and supporting rule. The resulting graph $G_{E,i}$ instantiates S_E and remains an over-approximation because it does not yet record placement, collision or clipping outcomes.
3. **Final map-content construction ($G_{E,i} \rightarrow G_{F,i}$):** The resolution function takes the layout-policy representation L_G and the graphical-resource representation X_G as additional inputs. L_G encodes the placement, collision, suppression, clipping and drawing-order constraints of the layout policy L, and X_G provides the fonts, icons, and patterns of X together with the identifiers used to reference them from the graph. The resulting graph $G_{F,i}$ instantiates S_F and records whether each considered display element was placed, suppressed or clipped. These outcomes allow a question query to distinguish, for example, a label removed through collision handling from one that remains partially visible at the map boundary.
4. **Map derivation ($G_{F,i} \rightarrow M_i$):** The map-image rendering function consumes the placed display elements in $G_{F,i}$ together with their final geometries, styles, graphical resources and drawing order. It draws these elements into the map image M_i without making further content-selection or layout decisions.
5. **Reference-answer derivation ($G_{F,i} \rightarrow A_{i,j}$):** The graph query q_j formalizes question Q_j and retrieves the reference answer $A_{i,j}$ from $G_{F,i}$. Together, the map image, question and reference answer form the test item $T_{i,j} = (M_i, Q_j, A_{i,j})$. Different questions and their queries can derive several test items from the same graph and map image.

Continuing the *Neutorstraße* example, G_I records the source road feature and its name, $G_{E,i}$ records its candidate road and label elements, and $G_{F,i}$ records their resolved geometries and placement outcomes. The renderer draws the placed label from $G_{F,i}$, while a street-name query retrieves *Neutorstraße* from that same placed label. If resolution instead suppressed the label, its name would contribute neither a drawn label nor an answer to this query, even though it remains in G_I.

5. Case Study

We implement an initial version of CartoGraph for the OSM Standard Map to examine whether it can derive a map image and reference answers for four question types from a shared map-content graph. It realizes the map-content graph in RDF and every graph query in SPARQL, using CONSTRUCT queries for the portrayal rules and SELECT or ASK queries for the questions. We first show for one map product the configuration through the five operations. Afterwards we present the construction of reference answers for a test set using map images at zoom levels 16 and 17 and their final map-content graphs.

5.1. Worked example

The selected case uses OpenStreetMap source data for an area in Nuremberg, Germany and a map configuration that identifies the OSM Standard Map at zoom level 17. We formalize the street-name enumeration and spatial-relation questions from Section 1 as the question queries q_1 and q_2. Figure 5 summarizes the five operations, whose intermediate graphs and outputs we report below.

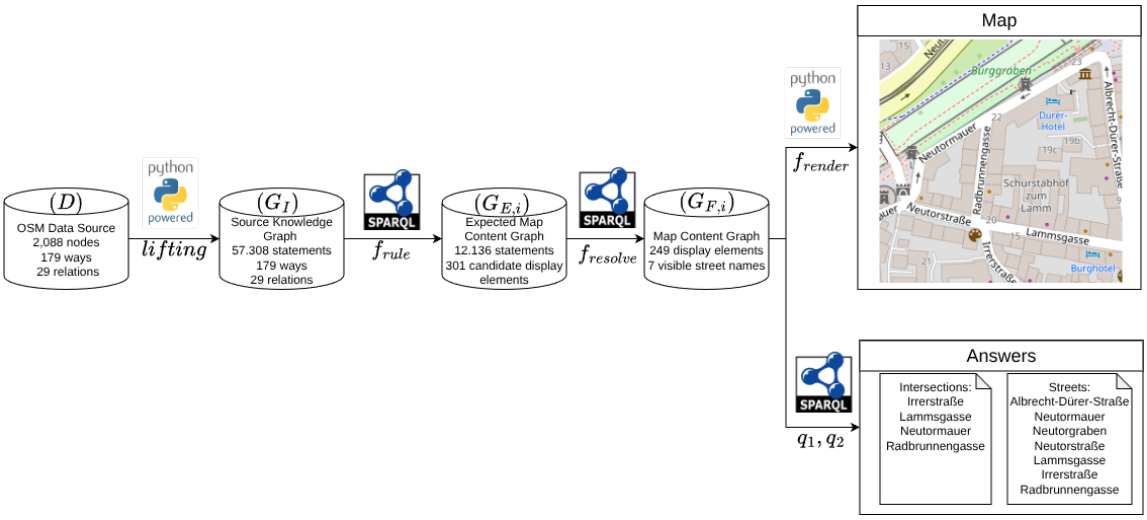


Figure 5: Implementation of CartoGraph for the OSM Standard Map at zoom level 17.

1. **Knowledge-graph construction ($D \rightarrow G_I$):** The input D is source data from OSM for an area in Nuremberg and contains 2,088 nodes, 179 ways and 29 relations in the source OSM XML file. A Python script implemented with RDFLib lifts the source features into RDF, preserves their source-feature identifiers and tags, and encodes the geometries of nodes and ways as GeoSPARQL WKT literals in OGC CRS84 (WGS 84 longitude–latitude order). The operation produces the geospatial knowledge graph G_I with 57,308 RDF statements and preserves links to the source features for the subsequent map-production operations.
2. **Expected map-content construction ($G_I \rightarrow G_{E,i}$):** This operation takes G_I together with the map configuration O_i and the portrayal-rule representation R_G as input. The configuration O_i specifies the map extent in OGC CRS84 from $49.45562800607982°$ to $49.457413352792216°$ N and from $11.07147216796875°$ to $11.07421875°$ E, Web Mercator as the map projection and the output dimensions (256×256 pixels), and identifies the OSM Standard Map at zoom level 17 as the map-product specification. The portrayal-rule representation contains 77 SPARQL CONSTRUCT queries. The operation produces $G_{E,i}$ with 12,136 RDF statements describing 301 candidate display elements. Each candidate display element remains linked to its source feature and supporting rule. These results show which source features satisfy the portrayal rules, but they remain an over-approximation because the resolution decisions have not yet been applied.
3. **Final map-content construction ($G_{E,i} \rightarrow G_{F,i}$):** This operation takes $G_{E,i}$ together with the layout-policy representation L_G and graphical-resource representation X_G as input. The operation produces $G_{F,i}$ and records the resolution outcomes. The graph contains 249 display elements retained for rendering. We group these elements by the geometry to which they are anchored, including labels anchored to curves, surfaces or points. The groups contain 110, 97 and 42 elements, respectively, of which 52, 39 and 5 are clipped at the image boundary. The graph separately records placement outcomes for 110 labels. Of these, 17 have at least one placed instance, 90 are suppressed and three have no valid placement candidates. The placed label instances are already included in the 249 display elements and are not counted again. The map-content graph therefore distinguishes placed, suppressed and clipped elements and provides the machine-readable record of the final map content.

4. **Map derivation ($G_{F,i} \rightarrow M_i$):** The map-image rendering operation takes the map-content graph $G_{F,i}$ together with the external font and icon files whose paths and hashes are recorded in the graph. A Python script retrieves the placed display elements with their resolved geometries, label text, styles, graphical-resource references and drawing order from the graph, verifies and loads the referenced assets, and draws these elements without making further content-selection or layout decisions. The operation produces a 256×256 pixel map image M_i containing the visible street names *Albrecht-Dürer-Straße*, *Irrerstraße*, *Lammsgasse*, *Neutorgraben*, *Neutormauer*, *Neutorstraße* and *Radbrunnengasse*. The label for *Albrecht-Dürer-Straße* is clipped at the image boundary but remains accepted because approximately 92.8% of the area of its bounding rectangle lies inside the image, exceeding the layout policy's 90% threshold. The image demonstrates that the map can be drawn directly from the same graph used to derive the reference answers.
5. **Reference-answer derivation ($G_{F,i} \rightarrow A_{i,j}$):** This operation takes $G_{F,i}$ and the two question-specific SPARQL SELECT queries q_1 and q_2 as input. Both queries filter for elements recorded as visible in the map-content graph. The query q_1 retrieves the visible street names, while q_2 retrieves the street names whose road display elements, with their geometries clipped to the map extent, intersect the road display elements of *Neutorstraße*. These element geometries derive from the source roads and are associated with street labels that have placed instances recorded in $G_{F,i}$. Consequently, q_2 evaluates neither unmodified source geometries nor the bounding rectangles of label characters. The first query produces $A_{i,1}$, which contains {*Albrecht-Dürer-Straße*, *Irrerstraße*, *Lammsgasse*, *Neutorgraben*, *Neutormauer*, *Neutorstraße*, *Radbrunnengasse*}. The second query produces $A_{i,2}$, which contains {*Irrerstraße*, *Lammsgasse*, *Neutormauer*, *Radbrunnengasse*}. Because both answers are derived from $G_{F,i}$, they remain restricted to the content recorded for the generated map.

5.2. Test set and query setup

The test set extends the worked example to 200 map images of the Nuremberg area, Germany, using the OSM Standard Map at zoom levels 16 and 17. Each test item combines a map image, a question about the map image and a reference answer. Each map image is paired with its final map-content graph $G_{F,i}$, which records the resolved map content used to render the map image and derive reference answers. A map image can occur in multiple test items with different questions, so the number of map images does not determine the number of test items. The map images, map-content graphs and question-specific SPARQL queries, and reference answers are available via GitHub[1].

The test set covers four question types: (i) street-name enumeration, (ii) street intersection, (iii) label position within the map and (iv) relative direction between labels. The worked example illustrates the first two question types. For each map-content graph, we query relevant display elements to instantiate the question-specific queries and derive reference answers from the same graph. Reusing these queries applies the same answer criteria across map images.

All query executions across the 200 map-content graphs completed without errors or per-map manual intervention after the portrayal rules and queries had been configured.

6. Discussion

This section discusses how CartoGraph addresses the three requirements, presents two properties that the case study makes explicit and examines the limitations of representing automated and semi-manual map-production systems.

CartoGraph addresses scalability, map-content fidelity and consistency through the reusable map-product specification and the shared map-content graph. CartoGraph supports benchmark-construction scalability because portrayal-rule queries can be reused across map images of the same map product, and question-specific queries derive reference answers without per-map manual annotation. It also

[1]See https://github.com/wintechis/CartoGraph

supports map-content fidelity, because the derived reference answers remain restricted to the final map content recorded in $G_{F,i}$ rather than on all source features (G_I). Reusing question-specific queries across applicable map-content graphs applies explicit answer criteria consistently across test items. The evaluation covers 200 map images at zoom levels 16 and 17, but remains limited to the implemented OSM Standard Map rules.

The case study surfaces two properties of reference answers derived from map content.

First, a formal query is more precise than the natural-language question it answers. The question Q_j specifies what to retrieve, whereas the query q_j defines the elements, geometries and relations that constitute the answer. In the spatial-relation question, reading *intersects* as GeoSPARQL `sfIntersects` returns the four streets in $A_{i,2}$, whereas reading it as `sfCrosses`[2] returns none of them, because the four streets meet *Neutorstraße* at a shared endpoint. We therefore publish q_j with each test item, which makes the selected interpretation explicit in a way that an annotator's judgment is not.

Second, the display elements of a map image are not necessarily connected to the geographic features they represent. The portrayal rules transform individual source features into display elements, but OSM splits a way whenever a tag changes and a `name` labels a way rather than a street, so $G_{F,i}$ contains no resource for an individual street. Questions about rendered content, such as the two case-study questions, remain unaffected, whereas questions about geographic features require a grouping of the rendered segments. For the question *How many streets are shown on the map?*, counting one street per name gives 11 streets, whereas counting connected pieces that share a name gives 19, because the map boundary cuts streets that continue outside it. No single grouping removes this dependence, because map rendering and geographic-feature identification serve different tasks.

CartoGraph reproduces an intended map product only to the extent that its production decisions can be represented in R_G, L_G and X_G. Two properties of existing map-production systems limit this representation.

First, an automated system such as OSM Carto [15] distributes its portrayal rules, layout policy and graphical resources across stylesheets, configuration files, assets and renderer behavior. Translating these components into CartoGraph's representations can omit or alter a production decision.

Second, semi-manual map-production systems may include map-specific decisions that are not expressed as reusable rules. For example, the German surveying authority (ADV) produce the Digital Topographic Maps (DTK) through a semi-automated process. For the DTK100 map product, the production system first generates a map image, after which cartographers manually reposition display elements to improve readability. Within the three-stage model, these adjustments belong to display-element resolution and their outcomes must be recorded in $G_{F,i}$ before the final map image and reference answers are derived. Recording the adjustments in $G_{F,i}$ preserves the shared derivation of the map image and reference answers, but the individual changes to the map images limits fully programmatic test-set construction.

An incomplete translation or an omitted manual adjustment can produce a map image and reference answers that remain internally consistent but do not faithfully reproduce the intended map product. The links from candidate display elements to their supporting portrayal rules help trace errors in R_G, but they cannot recover decisions that were never encoded. Future work should quantify deviations from the intended map product by comparing CartoGraph's outputs with those of the reference implementation across map configurations and specifications and should examine how manual adjustments can be captured as provenance-bearing updates to $G_{F,i}$.

7. Conclusion

Manually constructing reference answers for complex maps requires substantial effort and may introduce annotation errors. To support this task, we defined a graph-based model of the map-production process that creates a shared machine-readable representation for the reference answer and the map image. A map-image renderer derives the map image from this representation, while question-specific graph

[2] See https://docs.ogc.org/is/22-047r1/22-047r1.html

queries can subsequently derive one or more reference answers from the same resolved content. We instantiated this model in CartoGraph and evaluated it in a case study with the OSM Standard Map. The case study demonstrated shared map-and-answer derivation and automated reference-answer construction for a test set covering 200 map images at zoom levels 16 and 17.

These results support automated reference-answer construction within the tested configurations, but remain limited to the implemented OSM Standard Map rules and evaluated question types. Future work should examine additional map products and question types, compare the required effort with manual annotation and validate the final map content against established map-production systems. A further direction is to generate questions from the map-content graph so that map images, questions and reference answers can be constructed through one programmatic pipeline.

Acknowledgments

This work was funded by the Agentur für Innovation in der Cybersicherheit GmbH (Cyberagentur) as part of the HEGEMON project. The Cyberagentur did not interfere in the research process and its results.

Declaration on Generative AI

During the preparation of this work, the authors used Grammarly for: Grammar and spelling check. Further, the authors used OpenAIs GPT 5.6 to improve the writing style, by generating different versions of sentences. After using these services, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] D. Fairbairn, Measuring map complexity, The Cartographic Journal 43 (2006) 224–238. doi:10.1179/000870406X169883.

[2] M. L. Dihan, M. T. Hassan, M. T. Parvez, M. H. Hasan, M. A. Alam, M. A. Cheema, M. E. Ali, M. R. Parvez, Mapeval: A map-based evaluation of geo-spatial reasoning in foundation models, in: Forty-second International Conference on Machine Learning, New Orleans, 2025. URL: https://openreview.net/forum?id=hS2Ed5XYRq.

[3] S. Bhat, H. Khandelwal, T. Kataria, V. Gupta, Mapverse: A benchmark for geospatial question answering on diverse real-world maps, in: IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2026, Tucson, AZ, USA, March 6-10, 2026, IEEE, 2026, pp. 8168–8178. doi:10.1109/WACV61042.2026.00788.

[4] S. Mukhopadhyay, A. Rajgaria, P. Khatiwada, M. Shrivastava, D. Roth, V. Gupta, MAPWise: Evaluating vision-language models for advanced map queries, in: Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies, 2025, pp. 9348–9378. doi:10.18653/v1/2025.naacl-long.473.

[5] Y. Zhang, W. Zhang, Z. Zeng, K. Jiang, J. Li, W. Min, W. Luo, Q. Guan, J. Lin, W. Yu, MapReader: a framework for learning a visual language model for map analysis, International Journal of Geographical Information Science 39 (2025). doi:10.1080/13658816.2025.2455112.

[6] V. Srivastava, F. Lei, S. Mukhopadhyay, V. Gupta, R. Maciejewski, MapIQ: Evaluating multimodal large language models for map question answering, arXiv (2025). arXiv:2507.11625.

[7] S. Chang, D. Palzer, J. Li, E. Fosler-Lussier, N. Xiao, Mapqa: A dataset for question answering on choropleth maps, in: NeurIPS 2022 First Table Representation Workshop, 2022. URL: https://openreview.net/pdf?id=znKbVjeR0yI.

[8] B. Rystedt, F. J. Ormeling (Eds.), The World of Maps, International Cartographic Association, Stockholm, 2014. URL: https://icaci.org/publications/the-world-of-maps/.

[9] OpenStreetMap Wiki contributors, OpenStreetMap Carto, 2026. URL: https://wiki.openstreetmap.org/wiki/OpenStreetMap_Carto.

[10] A. Cuthbert, User Interaction with Geospatial Data, Technical Report OpenGIS Project Document 98-060, Open GIS Consortium, 1998. URL: https://web.archive.org/web/20030323015739/http://www.opengis.org/wwwmap/98-060.pdf.

[11] A. Doyle, OpenGIS Web Map Server Interface Implementation Specification, Technical Report OpenGIS Project Document 00-028, Open GIS Consortium, 2000. URL: https://docs.ogc.org/is/00-028/00-028.pdf?utm_source=chatgpt.com.

[12] International Organization for Standardization, ISO 19117:2012: Geographic information—Portrayal, Technical Report ISO 19117:2012, International Organization for Standardization, 2012. URL: https://www.iso.org/standard/46226.html.

[13] OpenStreetMap Wiki contributors, Transport Map, 2026. URL: https://wiki.openstreetmap.org/wiki/Transport_Map.

[14] Arbeitsgemeinschaft der Vermessungsverwaltungen der Länder der Bundesrepublik Deutschland (AdV), Digitale Topographische Karte 1:100 000 (DTK100), 2026. URL: https://www.adv-online.de/de/produkte/digitale-topographische-karte-1100-000-dtk100.

[15] OpenStreetMap Carto contributors, OpenStreetMap Carto: A General-Purpose OpenStreetMap Mapnik Style in CartoCSS, 2026. URL: https://github.com/openstreetmap-carto/openstreetmap-carto.