Universiteit Leiden

Visualisation of historical mobility patterns of former Leiden University Professors (1575-1812)

Oscar Boonstra (s3320634)

Supervisors:
Richard M.K. van Dijk & Wessel Kraaij

Bachelor Thesis– Informatica

Leiden Institute of Advanced Computer Science (LIACS)
liacs.leidenuniv.nl

July 1, 2026

Abstract

This thesis contributes to the Linking University and City project. This is a project where data scientists and historians collaborate in creating a platform for visualising and analysing historical data about the city of Leiden and Leiden University. In this thesis, we design and evaluate multiple visualisations for historical mobility patterns of Leiden University Professors from the period 1575-1812. The research question is *how can historical spatio-temporal mobility data of Leiden University professors be visualised, and to what extent do the resulting visualisations support the analytical tasks identified through Munzner's Nested Model?* To this end we build upon earlier work on an OCR- and AI-driven pipeline that converts historical text documents into a standardised format like JSON. We improve and extend upon this pipeline by using a more recent AI model to improve conversion accuracy and automatically store the data in a database. This extension was necessary in order to obtain usable data for the visualisations, but the main contribution of this thesis lies in the design, implementation, and evaluation of the visualisation system. We design and implement multiple visualisations following Munzner's Nested Model for the evaluation of visualisations, which provides a theoretical evaluation framework for visualisations. The end result is a multi-view visualisation implementation for viewing historical spatio-temporal life courses of Leiden University professors from the period 1575-1812. Each visualisation provides a different perspective of the data and helps answer a subset of the analytical questions identified through Munzner's Model, but all visualisations combined collectively support the identified analytical tasks.

Contents

1 Introduction

Visualising historical life courses of people can provide insights into how people and groups of people migrated from one location to another through time. We define a life course as the chronologically traversed path of life events, moving from location to location during someone's life. Visualisations can help researchers get a better understanding of the data they possess. The raw textual or numerical data is often hard to interpret for most people[23], while visualisations of data in graphs, diagrams and maps convey information faster and are often easier to interpret[29]. The main subject and research question of this thesis will be how historical life course events and the mobility of a large number of people can be best visualised geographically. Or more concretely: how can historical spatio-temporal mobility data of former Leiden University professors from the period 1575-1812 best be visualised, and to what extent do the resulting visualisations support the analytical tasks identified through Munzner's Nested Model? We discuss Munzner's Nested Model in section 2.2.

1.1 Linking University, City and Diversity

This bachelor's thesis project contributes to an ongoing and longer-running project called "Linking University and City", hereafter referred to as LUC. This is an interdisciplinary project where the Leiden Institute of Advanced Computer Science (LIACS) data science cluster of Leiden University collaborates with historical institutes like Historisch Erfgoed Leiden, Historisch Leiden in Kaart and also with historians from Leiden University. Multiple bachelor's thesis projects and master's thesis projects have already contributed to this project. The goal of the project is to create a platform/website where historians can interact with data related to the history of Leiden University and the city of Leiden. It will help enable research into the composition of the academic community of Leiden University throughout history and show how this community interacted with the city of Leiden. It will help reveal new possible insights, like trends and patterns in the data [25]. There is interest in the visualisation of the mobility and genealogical data to aid in this research.

To this end, data scientists and computer scientists contribute to this project in numerous ways. One being the processing and converting of historical texts and documents to a digitised form using, for example, OCR and turning it into a standardised format like JSON or a database. An example of this type of contribution is a bachelor's thesis by Zahra Abedi called Enriching Historical Records: An OCR and AI-Driven Approach for Database Integration[1]. In this project, a collection of volumes containing notarial information about former Leiden University professors was converted into text using OCR, and in turn, that text was converted to JSON and into a database using generative AI. Another way students and scientists contribute to this project is in the data visualisation and analysis of these digitised texts and documents, with the former being the topic of this thesis. Previous work by Liam van Dreumel has laid the groundwork for visualisations for this platform[27], and as of the time of writing this thesis, there is an accessible work-in-progress website[24] where some features and options to explore data are already accessible. Using filters, users can interact and view tables and numerous graphs of historical data related to Leiden University professors and students, an example of such a graph can be seen in figure 1. This thesis continues to build upon both the works of Zahra Abedi and Liam van Dreumel, which will both be discussed in more detail in the related work section of this thesis. A more comprehensive

mission statement of the LUC project can be found on the official LUC research project page on the Leiden University website[26].

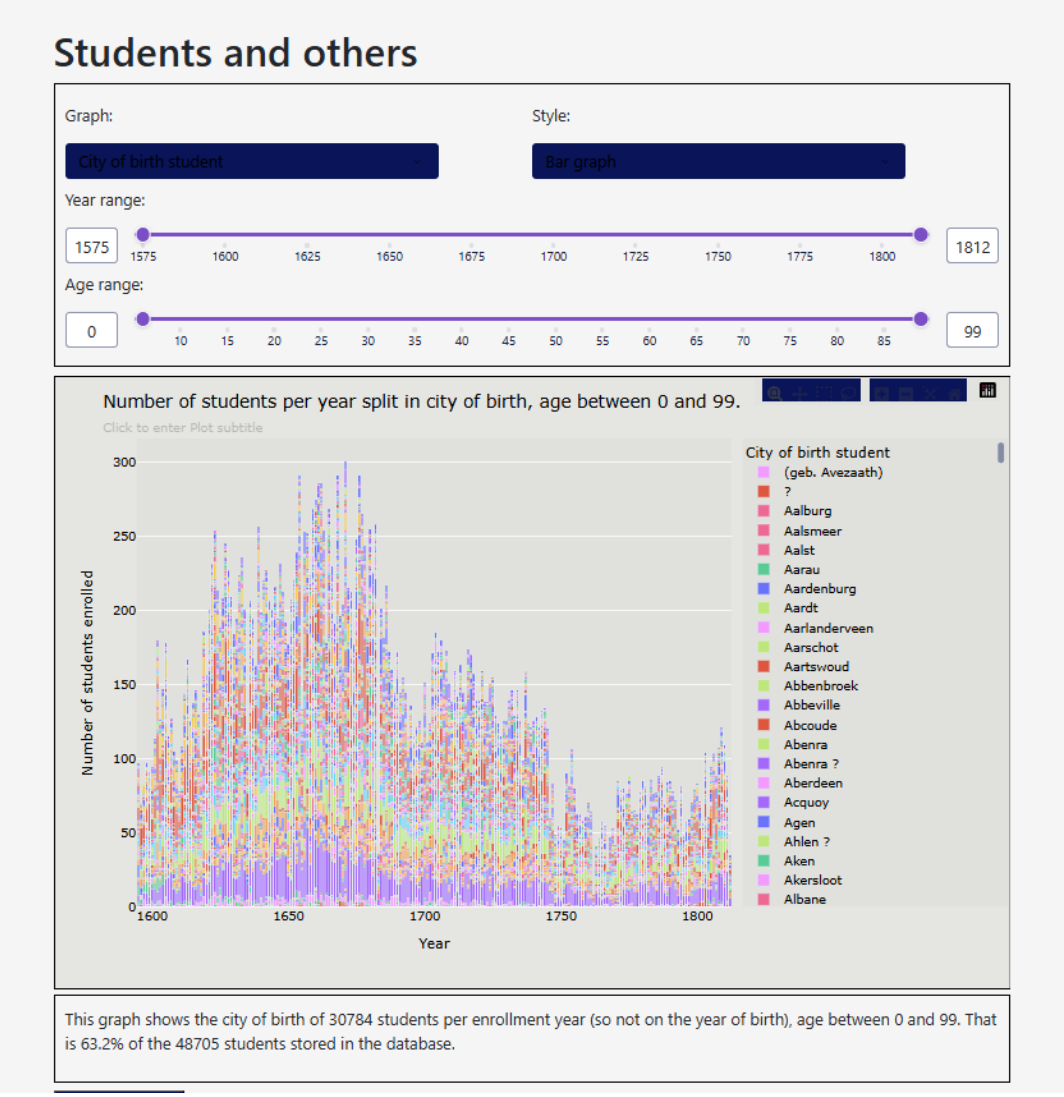


Figure 1: An arbitrary example graph from the LUC platform. Showcasing the number of student enrolments per year based on the city of birth.

1.2 Project Goal

This thesis will contribute to the LUC project by developing multiple visualisations for displaying the spatio-temporal data of former Leiden University professors from the time period 1575-1812. A dataset containing the information of 391 professors from historical notarial documents about their life trajectories, including but not limited to career and education events detailing both location and time, is in the possession of the university. This dataset will be discussed in more detail in section 3.1. Because the dataset contains the locations and times of these events, it becomes possible to construct a life course for these professors. The historians participating in the LUCD

project would like to have a visualisation of all these life courses. The visualisations are then to be integrated into the platform as a plugin. Figure 2 provides an overview and summary of the platform architecture and explains how visualisation plugins fit into the whole picture.

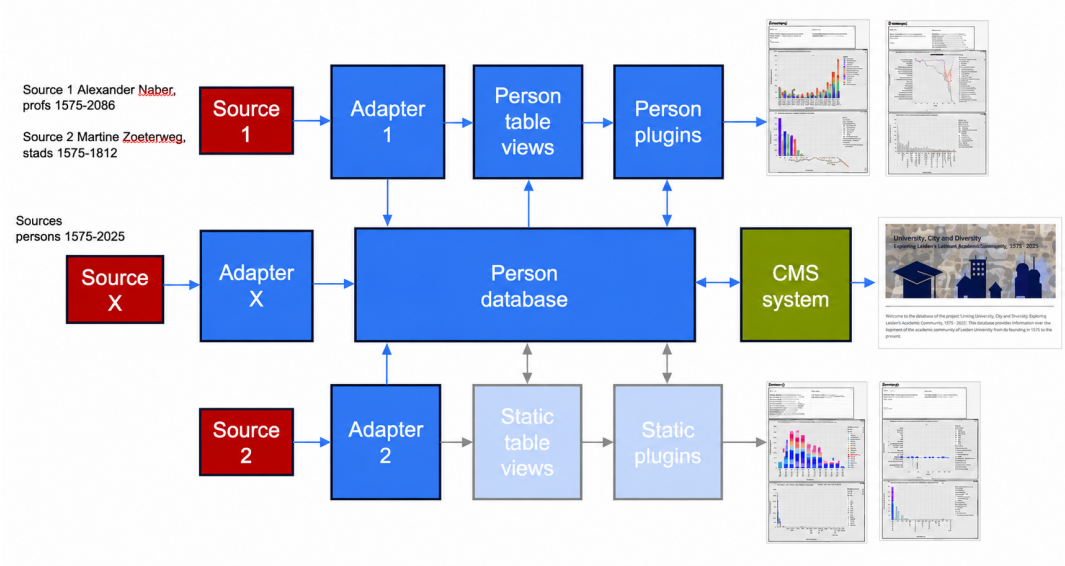


Figure 2: The architecture of the website is based on a WordPress CMS system. At the centre, we can see the person database, which is fed by multiple sources, coloured in red. These sources are historical texts and document which first need to be digitised, cleaned and formatted by adapters before the data that is contained within them can be added and joined with the database. In turn, visualisations like graphs, tables, and maps provide an interface for the user to the database. These visualisations are separate plugins that can be connected and fed by the database. Both adapters and visualisation plugins are independent from one another, allowing for a modular architecture, supporting the relatively easy removal and addition of modules.

1.3 Thesis overview

After this introduction section, this thesis will continue first with the background section, related work section, approach section, experiments and results section, finishing with the conclusions and further research section. In the background section, we will further elaborate on some key ideas and terms regarding visualisations of spatio-temporal data of people. The related works section will focus on previous work that has contributed to the project. This will focus on both Zahra Abedi's and Liam van Dreumel's work and how this thesis uses and builds upon it. In the approach section, we discuss the methodology of how the visualisations were developed, some prerequisites that had to be done beforehand, and how the visualisations and the final results will be evaluated. The experiments and results section will showcase the final result, and show the integrated visualisations and updated results from Zahra Abedi's pipeline using a more recent version of OpenAI's GPT. In the final section, we will conclude and address any shortcomings that can be addressed in potential future work.

2 Background

As introduced before, the dataset that needs to be visualised contains information about Leiden University professors from old biographical documents. It contains information about their life, from education to career events. Because the education and career events contain locations, often city names and time frames, and also often a date in the form of a year or period of years, we can plot these events on a geographic map. This is a naive approach that could visualise how a single individual moves from place to place, but for more than 1 individual, this would quickly become a convoluted web of lines crossing over one another. Furthermore, it offers limited support for discerning patterns or broader trends in the data. We are dealing with a problem where the data is spatial and temporal in nature; people migrate from place to place over time. Visualisations are thus needed that can show the relation between time and location, and should be able to show the movement too. The visualisations should also be able to clearly show trends in the life courses of all. For example, what were the most visited locations during a certain time period, or what life course routes were common in a certain period? If questions like these can be answered by the visualisations, it can help researchers discover patterns, and in turn allow them to find potential reasons for it by comparing it to known historical events or trends. In the next subsections, we will discuss different types of visualisations and their literature.

2.1 Types of visualisations for Spatio-Temporal Movement Data

There are numerous existing types of visualisations that can visualise spatial and temporal relations or flows. With each of them having its own strong points and weak points. Because different visualisation types emphasise different aspects of the data, a single best visualisation is unlikely to exist for all analytical tasks. This motivates the use of a multi-view approach in this thesis. In A Nested Model for Visualisation Design and Validation[15], Munzner argues that certain approaches are more suitable for specific data types or analytical tasks than others. Not only are the characteristics (data type, dataset size, dimensions, attributes, range and relational structure) of the data important, but the analytical tasks users need and want to perform also play an important role in deciding what types of visualisations are suitable and support the execution of those tasks.

2.1.1 Dot Maps

Dot maps plot data in the form of dots on a geographic map, an example can be seen in figure 8. It is therefore very close, if not the same as looking at a regular geographic map; cities or places can be regarded as datapoints that are plotted on the map, in accordance with their actual spatial location. The events from our dataset can be plotted in the same way based on their location. Dot maps are good at showing the spatial distribution and concentration of data[11], but when used naively, they cannot convey other dimensions such as time or mobility. Colours, symbols, animation or size can be used to encode additional information. In this thesis, dot maps are therefore useful as a baseline geographic representation of where events occurred, but they require additional interaction or linking to other views to support temporal analysis.

2.1.2 Space and Time: Space Time Cube

Geographic movement data often combines two spatial dimensions with one temporal dimension. A space-time cube represents this by using the horizontal plane for geographic space and the vertical axis for time, see figure 16 for an example of the space time cube. A space-time cube is a 3D graph and was first introduced by Hägerstrand[10] in 1970 as a way to analyse and visualise human activities, movement, and interactions across both geographic space and time simultaneously. It is a plot with 3 axes where the horizontal x-axis and y-axis represent space and the vertical z-axis represents time. This way it becomes possible to plot locations over time, and for a person to track through time which places they have visited. The vertical z-axis represents temporal progression, while movement can be interpreted from the sequence of connected events. This is highly relevant to the visualisation issue of this thesis. Modernised versions of the space-time cube have been introduced numerous times. An example is an reinterpretation by Kraak[12] using the modern geographic information system (GIS). The strengths of a space-time cube is that it can simultaneously show spatial and temporal relations. However, for some tasks, 2D representations can perform just as well if not even better[13]. A disadvantage is that many different crossing traces can become visual clutter, which is difficult to interpret.

2.1.3 Flows and Flow Maps

A flow is the movement of objects, information or people from location A to location B over time. An important property of a flow is that quantity plays an important role. As a result, flows represent not only the existence of movement between locations, but also the magnitude and direction of that movement, for example, in the form of migration volumes, transportation intensity, or communication frequency[2]. Flow maps are a type of visualisation specifically designed to represent such movements geographically by connecting origin and destination locations using lines or arrows whose visual properties can encode characteristics such as direction, quantity, or time[8]. Because flows contain information about origin, destination, and quantity, multiple individual movements can be aggregated into larger flows. Aggregation makes it possible to represent overall movement intensity between locations while reducing the complexity and clutter caused by displaying every individual movement separately[2, 8]. In this thesis, individual ordered event sequences are treated as life-course trajectories, while repeated movements between the same or similar locations by different professors can be aggregated into flows. The origins of flow maps can be traced all the way back to 1837 to engineer and cartographer Henry Drury Harness, but arguably one of the most famous flow maps was created a little later in 1869 by Charles Joseph Minard, who refined the technique in a visualisation of Napoleon's Russian campaign, visualising geographic movement, direction, quantity (army size) and even temperature[6], see figure 3. Computational approaches to flow maps emerged much later, for example, through the work of Tobler in 1987[22]. A more abstract type of flow visualisation are Sankey diagrams, which are not based on geographic maps. These diagrams do not show geographical space, but rather only show the flows from point A to point B, where A and B do not necessarily have to be physical places. Sankey diagrams are ideal for showing flows in processes or energy flows[20].

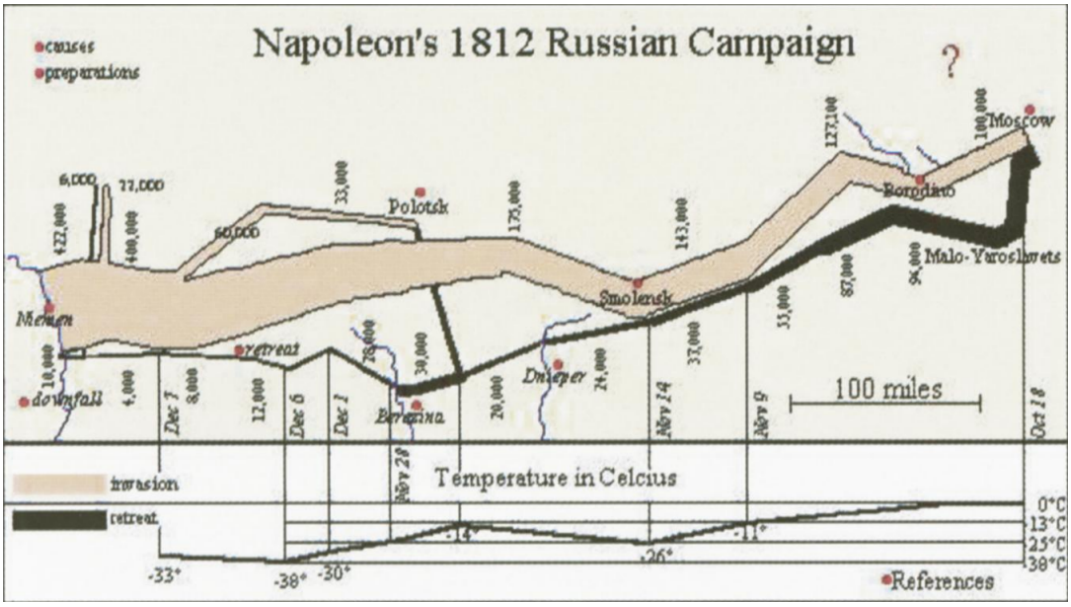


Figure 3: A re-vision of Minards flow map by Michael Friendly[6]. Army size is encoded by the thickness of the flow, while location is encoded like on a regular map. Time is encoded from right to left and geographically aligned.

2.1.4 Visualising a 3rd Dimension

This section briefly touches upon how to encode additional variables such as time, event type, or density. In geographic visualisations this can be achieved through 3D representations, extrusion, or height-based encodings. For example, the vertical axis can be used to represent time, quantity, or density. This makes it possible to visualise multiple dimensions simultaneously within a single representation. However, adding an additional dimension also introduces additional complexity. While for example in 3D visualisations it can increase information density and reveal patterns that may be difficult to observe in two-dimensional views, they can also introduce issues such as occlusion, perspective distortion, and navigation difficulties[29]. Interpreting depth on a flat display can furthermore be cognitively demanding for users. Because of this, the effectiveness of 3D visualisations strongly depends on the analytical tasks users need to perform[15]. Solutions for occlusion and navigation can be the introduction of interactivity in 3D graphs; rotating graphs, zooming in and out, moving around can allow the user to focus on a specific part of the graph or view parts and sides of the graph more clearly that were previously hidden. Animation shows the change of states and essentially represents the passing of time between those states[3]. Animation can represent temporal change by showing different states of the data sequentially. Other ways to represent a third dimension like time can be through the use of colours[29], where every colour represents a different time frame.

2.1.5 Scalability: Aggregating & Edge Bundling

Because our dataset contains information about 391 professors where each professor has numerous events tied to them, plotting all this amount of data can lead to visual clutter. Visualisations may become difficult to interpret and important patterns can become obscured. One common solution

is aggregation. Rather than visualising every individual movement separately, multiple trajectories can be grouped together into larger aggregated flows, like seen in the flow maps, based on shared properties such as origin, destination, geographic proximity, or temporal similarity[2]. Aggregation can reduce visual complexity while simultaneously revealing broader movement patterns that may otherwise remain hidden in dense visualisations. Another important technique is edge bundling. Edge bundling attempts to visually group similar or spatially close trajectories together into bundles or bigger flows. By visually combining related paths, edge bundling reduces overlap and clutter while improving the readability of movement patterns and network structures[9]. There are multiple methods for edge bundling: some approaches rely on geometric proximity, where edges that are spatially close are attracted towards each other, while others make use of hierarchical or force-directed techniques to iteratively construct bundles. The resulting visualisations can reveal larger movement structures and dominant routes that may be difficult to identify in unbundled representations. However, edge bundling also introduces abstraction and distortion. Individual trajectories may become more difficult to distinguish, and exact spatial paths can become less precise due to the bundling process. As a result, edge bundling represents a trade-off between preserving detailed individual information and improving the readability of larger overall patterns.

2.2 Evaluating Visualisations

Although visualisation quality partly depends on users and tasks, evaluation can be structured through established criteria and frameworks[18]. Visualisations are good if the viewer can extract as much information as possible without getting confused or overwhelmed. User-based evaluation is a common used evaluation technique for not only evaluations, but essentially everything that will end up being used [14]. By letting users try out or perform tasks from that what needs to be evaluated, a common truth or objective evaluation is extracted from all the separate subjective experiences. This approach is commonly used and works, but requires volunteers or paid workers who can participate in the evaluation[14]. A full user-based evaluation with historians however was beyond the scope of this bachelor thesis due to time constraints and the limited availability of domain experts. Another approach is the ICE-T approach; a heuristic approach to value-driven evaluation[28]. The framework divides visualisation value into four components: Insight, Confidence, Essence, and Time. These components are then broken down into guidelines and smaller low-level heuristics that can be rated by evaluators. It differs from user-based evaluation in that it relies on expert inspection. The involved experts are often experts in visualisation or human computer interaction instead of domain experts. These experts inspect and evaluate the visualisations using the established heuristics.

Alternatively a theoretical framework for the evaluation of visualisations by Tamara Munzner also exists[15, 16]. In this framework, which will be discussed in more detail in the next section, visualisations are evaluated against a set of established criteria. This framework was chosen as the main evaluation method of this thesis because it allows us to use a theory-driven approach for visualisation evaluation. However the advantage of it being a theory-driven approach does not mean it can completely replace user- or expert-based evaluation. Yet is does provide us with a way to reason about our evaluations in the absence of those user- and expert-based evaluations. The Nested Model is structured and helps us translate the domain problem into concrete data/task requirements and helps us choose specific visual encodings.

2.2.1 A Nested Model for Visualisation Design and Validation

This framework describes visualisation design and evaluation as a process consisting of multiple nested levels, where decisions made at higher levels influence the lower levels[15]. The model distinguishes between four main levels: domain problem characterization, data and task abstraction, visual encoding and interaction design, and algorithm design. The highest level focuses on understanding the domain problem. At this level, the goals of the visualisation need to be established, the characteristics of the dataset determined, and the needs of the target users need to be analysed. For this thesis we are dealing with historians as the domain users. Therefore the needs and wants of the historians need to be analysed; what do they want from the visualisations? What questions do they want to answer using the visualisation? (e.g. Where were professors living, what cities were popular in de the academic world during a certain period?) In addition the dataset is to be analysed, we are dealing with spatial and temporal data. The second level concerns the abstraction of data and tasks. Real-world domain problems are translated into more abstract concepts closer to the data structures and analytical tasks that visualisations can support. User goals are translated into analytical tasks (comparison, filtering, pattern detection) and are abstracted to form a bridge between the user and eventual visual implementation. The third level focuses on visual encoding and interaction techniques. At this level, decisions are made regarding how abstract data should visually be represented. Examples include the use of flow maps, dot maps, timelines, animation, or space-time cubes. Interaction techniques such as filtering, zooming, or selecting trajectories also belong to this level. Finally, the lowest level concerns algorithm design and implementation efficiency. This includes the technical implementation of visualisation techniques and ensuring that the system performs adequately for the size and complexity of the dataset. Munzner's model is visualisation-specific because it evaluates not only the software implementation, but also the translation from domain questions into abstract visual tasks and from those tasks into visual encodings and interactions.

This framework allows us to determine different concrete criteria per level which our visualisations need to adhere to. This allows the implementation to be evaluated systematically against explicitly defined criteria. An important aspect of the nested model is that evaluation takes place at every level. Problems discovered at lower levels may originate from incorrect assumptions or design choices made at higher levels. As a result, the framework provides a structured way to reason about the strengths and weaknesses of visualisation approaches without the need of using a large-scale empirical user evaluation, making it useful for a thesis of this scale. It provides a way to reason about visualisations, however this does not completely remove the need for user or expert validation. In fact each level needs to be validated separately as each level introduces a hazard according to Munzner, and user- or expert-based evaluations remain one of the possible ways to validate a level. In order for the Domain level to be validated correctly, the domain questions and requirements should line up with the needs of the domain users. In our case the historians. For the abstraction of data and tasks level to be validated, the tasks that the visualisation provides should be helpful to answering the domain level requirements. For the visual encoding and interaction level to be validated, the chosen visualisations should be justified and help solve the tasks established in the abstraction level. The final algorithm and implementation level is correctly validated if the implementation works and performs fast enough. For each of these levels there are multiple ways to validate the evaluation, of which user- or expert-based evaluation is an option.

3 Related Work

In this section we will discuss two related works that have directly contributed to the LUC project, Enriching Historical Records: An OCR and AI-Driven Approach for Database by Zahra Abedi[1] and Visualisation tools to support historical research on a linked dataset about Leiden University by Liam Dreumel[27]. Both works are bachelor thesis projects that contributed to the LUC project, and this thesis acts as a continuation. We also discuss the source material for our dataset.

3.1 Source Material: Leidse hoogleraren en lectoren 1575–1815 DOUSA 80 0211-17

The source material used in this thesis consists of the *Leidse hoogleraren en lectoren 1575–1815 DOUSA 80 0211-1–7* volumes by A.A. Bantjes and L. van Poelgeest. This series of volumes consists out of 7 volumes and combined contain the information of 391 professors and Leiden University administrative workers from the period 1575-1815 spread out across approximately 700 pages. Each entry contains information about a person's life and academic career, such as birth and death information, education, appointments, career events, family relations, and particularities. We will refer to the source material as the B&P data from now on. However not every entry in the B&P data is as complete and there are some issues. For some professors, family relations are described in detail, with the names, birthdates, place of birth, date of death, place of death all listed for their relatives, but for others this can be partially incomplete or entirely missing. In addition, for events most often both the location and date are listed, but sometimes one of them or both are missing. Sometimes there is also uncertainty about a location and multiple possible locations are listed. These issues have to be taken into account when we create visualisations for this data and when we eventually use these visualisations to reach conclusions. It means that visualisations using this data will contain the inaccuracies introduced by this dataset through either missing or uncertain information.

For this thesis, the most important parts of the source material are the events that contain both a location and a date or time period. Birth, education, career, and death events can be used to reconstruct life courses of professors. These life courses form the basis for the visualisations developed in this thesis. By ordering these events chronologically and linking them to geographic coordinates, it becomes possible to visualise where professors lived, studied, worked, and moved throughout their lives.

However, the original source material is not directly usable for computational analysis or visualisation. The information is stored in historical text documents, while the visualisations requires structured data that can be queried by for example person, event type, location, and time. The conversion of the source material into a structured format such as JSON, and later into a relational database, is therefore an important prerequisite for this thesis. Without this conversion step, the data could not easily be used to construct the life-courses for our visualisations. The conversion of these historical texts to JSON has already been mostly done through the work of Zahra Abedi, which will be discussed in the next section.

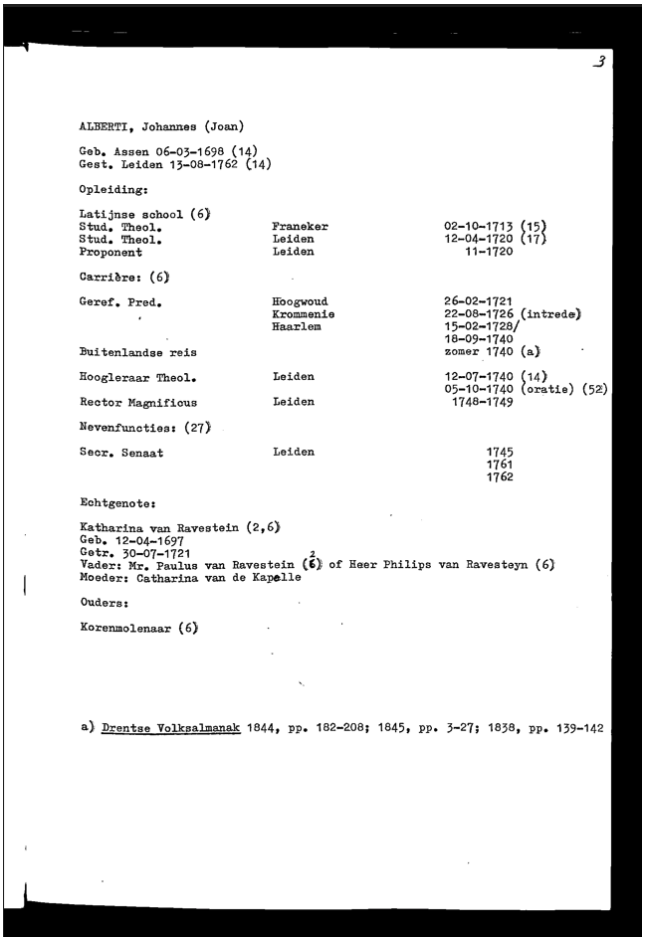
3.2 Enriching Historical Records: An OCR and AI-Driven Approach for Database

The bachelor project of Abedi was based on the study by historians A.A. Bantjes and L. van Poelgeest which was discussed in the previous section. In her work she created an OCR and AI-driven pipeline to convert the original physical typescript of the B&P volumes into the computer readable format JSON. First, images of the original documents were cleaned and processed to make them more clear before using OCR and Tesseract and they were converted into digital text. This text was then re-interpreted by ChatGPT 3.5 which was prompted to convert this digital text into a structured JSON format, see figure 4. ChatGPT 3.5 is only able to process text and unable to directly receive the images, hence the prerequisite of applying OCR first separately. Abedi suggested that perhaps with newer GPT versions the OCR step in its entirety could be skipped. Using GPT 3.5 the JSON extraction from OCR text in this way achieved an average accuracy of 63% and, based on correct manually converted text, 65%[1]. Accuracy in the JSON conversion here is measured through a field-by-field comparison, where each key-value pair in the generated JSON is compared to its equivalent in a manually created correct JSON by Abedi. Accuracy was calculated as the percentage of correct values for each key across all tested JSON files.

For this thesis and our visualisations, this pipeline by Abedi essentially needs to be extended by first, passing the JSON data into a database, and after the data is in the database, it can then be filtered by using different types of queries in order to more easily construct different kinds of visualisations. The source material consists of seven volumes, but in Abedi's repository only volume 1 and other selected entries from the other volumes for evaluation were converted. One of the objectives of our study was to convert all volumes of B&P and attempt to further increase the accuracy, since the quality of the visualisations depends on the quality and completeness of the converted data.

3.3 Visualisation tools to support historical research on a linked dataset about Leiden University

In this thesis multiple visualisation approaches to display historical data were analysed and implemented on the LUC platform. Of particular interest for this thesis were the visualisations of mobility. Van Dreumel's work suggests that different visualisation types support different aspects of historical data exploration, which supports the multi-view approach taken in this thesis. Geographic maps seem to be well suited for visualising mobility and migration, while timelines and graphs seem a good option to visualise chronological development[27]. The visualisations constructed are currently available on the platform and were constructed using Python tools like Ploty Dash, GeoPandas, Plotly and Network X. Because the platform is based on a Plotly Dash implementation, visualisations in this thesis need to be constructed in a way that is supported by the Plotly Dash implementation. In Liam van Dreumel's work a geographic visualisation of students was also already implemented. It included a heat map, line map, MP Heat map, MP scatter map and animated map for showcasing the geographical origin of students. This thesis builds on this context by focusing more specifically on professor mobility, individual life-course trajectories, recurring movement routes, temporal change, and possible co-presence.



3

ALBERTI, Johannes (Joan)

Geb. Assen 06-03-1698 (14)
Gest. Leiden 13-08-1762 (14)

Opleiding:

Latijnse school (6)		
Stud. Theol.	Franeker	02-10-1713 (15)
Stud. Theol.	Leiden	12-04-1720 (17)
Proponent	Leiden	11-1720

Carrière: (6)

Geref. Pred.	Hoogwoud	26-02-1721
	Krommenie	22-08-1726 (intrede)
	Haarlem	15-02-1728/ 18-09-1740
Buitenlandse reis		zomer 1740 (a)
Hoogleraar Theol.	Leiden	12-07-1740 (14)
		05-10-1740 (oratie) (52)
Rector Magnificus	Leiden	1748-1749

Nevenfuncties: (27)

Secr. Senaat	Leiden	1745
		1761
		1762

Echtgenote:

Katharina van Ravestein (2,6)
Geb. 12-04-1697
Getr. 30-07-1721
Vader: Mr. Paulus van Ravestein (2) of Heer Philips van Ravesteyn (6)
Moeder: Catharina van de Kapelle

Ouders:

Korenmolenaar (6)

a) Drentse Volksalmanak 1844, pp. 182-208; 1845, pp. 3-27; 1838, pp. 139-142


```
{
  "FirstName": "Johannes",
  "LastName": "ALBERTI",
  "Affix": null,
  "Gender": "Man",
  "second_names": [
    "'Joan' in Johannes (Joan)"
  ],
  "alternative_last_names": [],
  "education": [
    {
      "subject": "Latijnse school",
      "location": null,
      "date": null,
      "source": "6"
    },
    {
      "subject": "Stud.Theol.",
      "location": "Franeker",
      "date": "1713-10-02",
      "source": null
    },
    {
      "subject": "Stud.Theol.",
      "location": "Leiden",
      "date": "1720-04-12",
      "source": null
    },
    {
      "subject": "Proponent",
      "location": "Leiden",
      "date": "1720-11",
      "source": null
    }
  ],
  "careers": [
    {
      "job": "Geref. Pred.",
      "location": "Hoogwoud",
      "date": "1721-02-26",
      "source": null,
      "is_side_job": 0
    },
    {
      "job": "Geref. Pred. (intrede)",
      "location": "Krommenie",
      "date": "1726-08-22",
      "source": null,
      "is_side_job": 0
```


Figure 4: On the left hand side is a cleaned up image of the original text, on right hand side is the partial output of the pipeline into a structured JSON format which needs to be visualised. The structure of the fields in this JSON data was chosen and created by Abedi and was part of the prompt for GPT 3.5. This structured JSON data is easier to work with for visualisation or analyses purposes.

Another relevant contribution to the LUC project is the thesis by De Boer[4], which focused on interactive and explorative visualisations for researchers with unexpected questions. De Boer's work aimed to move beyond predetermined graphs by allowing users to create visualisations based on the available data, including tabular, geographical and genealogical visualisations. The objective of this bachelor thesis project is to design and evaluate geographic visualisations of the life courses of the professors in the B&P dataset. These visualisations need to be able to support the discovery of trends, patterns and help answer questions researchers might have. Before we can start working on the visualisations, some prerequisites need to be fulfilled.

4 Approach

In this section we will discuss the approach and everything that has to be done before the visualisations can be implemented in the final platform. We will discuss this in terms of multiple

stages beginning with the extension of the OCR and AI pipeline and analysing formatted JSON by Zahra Abedi[1] and extra work that has been done to facilitate this. This is an important prerequisite before we start working on the visualisations. The next step is the prototyping stage for visualisations, in where we develop multiple visualisations outside the platform. A prerequisite for the prototyping stage is to first establish criteria according Munzner's Nested Model for evaluation of visualisations. It is essential for evaluation following this model, that before development on visualisations are started, clear criteria are established for the visualisation model which can judged during the evaluation phase. After prototyping, the visualisations need to be integrated into the actual platform. The final stage is evaluating using the previously established criteria, this stage will not be discussed in this section, but will be discussed in detail in section 5.5.

4.1 Extending the pipeline

The OCR and AI-driven pipeline by Abedi used ChatGPT 3.5 to convert the original texts into useable JSON format[1]. Like mentioned before an average accuracy of 63% - 65% was achieved with 10% of the data being annotated manually and homogeneously distributed over the entire dataset. This means that in the JSON output there are quite a few mistakes or empty fields. This includes the omission of certain information and wrongly converted text. In some cases the wrong fields were filled in with the wrong text, e.g. the date was filled in the location field or location was filled in the subject field. Furthermore the formatting of dates also seemed to be inconsistent and vary from entry to entry, sometimes the YYYY-MM-DD format was used, other times the MM-DD-YYYY format was used. In addition to wrongly or inconsistently converted data, a lot of text was also not converted and completely skipped. Furthermore the volumes that were converted to JSON using this pipeline was only limited to volume 1 and some entries used as sample for evaluation from the other volumes. It is thus important for the visualisations of this project to make sure all the professors in the OCR text are converted to JSON this time. To this end we are using Zahra Abedi's pipeline again using a more recent model of Open AI's GPT. The motivation for using GPT 5 and not another competitive model like Claude or Gemini is that this would require the minimum amount of adjustments to Abedi's pipeline. Changing the GPT model was achieved by changing a single variable in Abedi's code. ChatGPT's API supports the passing of pre-defined formats for the produced output. This was all already implemented in the code. Switching to a different model like Claude would require more code to be rewritten. By sticking with GPT 5, changing API calls, rewriting structured output handling, and possibly changing the prompt format could be avoided. In addition this allows us to more directly compare the results with one another as we keep the original pipeline and updated pipeline as identical as possibles. Next to converting all the OCR text to JSON and fully digitising these volumes into a standardised form, the expectation is that by running a more recent model like GPT-5, the accuracy of the JSON conversion will also increase. While there are not many direct comparisons between GPT-3.5 and GPT-5, comparisons between GPT 3.5 and 4 claim GPT 4 performs much better[5] and GPT 5 performs better than GPT 4[17], it thus seems reasonable to expect that GPT-5 may perform better than GPT-3.5 on structured extraction tasks. As such to acquire the full converted data, we will rerun the pipeline using GPT-5. The specific API version that been used is snapshot: gpt-5-2025-08-07.

After aquiring the JSON data, the data is to be stored in a proper database. The LUC project makes use of monetdb, an aim here is to create a database based on a central table for persons, and seperate tables for other attributes that are related to persons. Querying from this database should be relatively simple and fast for the visualisations. To get all the data from JSON to a database, an automatic Python script was written for this. The next step in the pipeline is creating the visualisations from the data that is now accessible in the database. The code is implemented in the LUC repository. Generative AI was used to help and accelerate development throughout the project. It was used as a programming aid mainly for debugging, exploring new possible code structures and generating code fragments. Thanks to this increase in productivity, it was possible to implement more visualisations and other auxiliary features. The final implementation decisions, including code structure, database structure, visualisation design and evaluation, all remained the responsibility of the author. All generated code was reviewed, modified and tested before being adapted and included into the codebase. Please refer to the Appendix for the documentation of the code.

4.2 Establishing Design and Implementation Criteria

Like discussed before, the nested model describes 4 levels of abstraction and criteria that we need to establish, from to top to bottom, they are: Domain Problem & Characterization, Data & Task Abstraction, Visual Encoding & Interaction Design, Algorithm & Implementation. We will start clarifying these from top to bottom in regards to this thesis, starting with the domain problem and characterization.

4.2.1 Level 1: Domain Problem & Characterization

At this level, the purpose of the visualisations and their intended users are defined. The overall goals of the visualisation are established here and can later be refined into more concrete and measurable criteria in the lower levels of Munzner's Nested Model. As discussed previously, the primary end users are historians and researchers seeking insight into historical data, particularly in relation to geographic movement and spatio-temporal patterns.

Using the dataset, life courses of Leiden University professors must be reconstructed from educational, career, and other life events associated with locations and time periods. The visualisations should support the exploration of where professors were born, lived, studied, and worked throughout their lives, while also enabling the discovery of collective mobility patterns and flows within the dataset. A broader goal of the visualisations is to support the discovery of clusters and trends in the data. Instead of treating this as a separate question, this goal is addressed through the more specific questions below, such as identifying academic hubs, recurring flows, and temporal change.

Based on these goals, several concrete research and analysis questions can be formulated that the visualisations should support. The domain questions listed below were initially decided on based on the characteristics of the data, and were afterwards presented to the Leiden Institute of History to validate the historical relevance of these questions.

1. Where did professors move throughout their lives?

2. Which cities or regions acted as important academic hubs in certain periods?

3. Are there recurring geographic mobility patterns between universities or cities in certain time periods, in other words, were certain cities academically connected?
4. How did the mobility of professors change over time?
5. Which professors were present in the same place, and could potentially have met?
6. Do mobility patterns differ between faculties?

4.2.2 Level 2: Data & Task Abstraction

The domain questions established in the previous section need to be translated into abstract visualisation concepts. In other words, this level defines what kind of data, tasks, and operations are required to answer each question.

- **Question 1: Where did professors move throughout their lives?**
  To answer this question, the complete sequence of locations associated with a professor must be reconstructed and visualised as a spatio-temporal trajectory or life course.
- **Question 2: Which cities or regions acted as important academic hubs during certain periods?**
  This task requires identifying spatial clusters and highly connected locations in the dataset. The number of visits, arrivals, or departures associated with cities or regions should be analysable over time.
- **Question 3: Are there recurring geographic mobility patterns between universities or cities?**
  This task focuses on identifying common flows and repeated movement patterns between locations in order to determine whether certain cities were strongly academically connected.
- **Question 4: How did the mobility of professors change over time?**
  The visualisations should support the comparison of temporal mobility patterns and changes in movement behaviour across different historical periods.
- **Question 5: Which professors were present in the same place during overlapping periods of time?**
  The visualisations should make it possible to identify potential encounters or overlaps between professors at shared locations and time periods.
- **Question 6: Do mobility patterns differ between faculties?**
  This task requires comparing subsets of professors based on faculty. The visualisations should support filtering or selecting professors by faculty in order to inspect whether different faculties show different spatial distributions, mobility routes, or temporal patterns.

4.2.3 Level 3: Visual Encoding & Interaction Design

At this level, the abstract data and tasks defined in the previous level are translated into concrete visual encodings and interaction techniques. The goal is not only to choose suitable visualisations, but to ensure that each visual design decision directly supports one or more of the previously established analytical tasks. The main visualisation techniques used in this thesis are therefore linked back to the domain questions from Level 1.

- **Dot map for tracing individual lifepaths**
  The dot map primarily supports Question 1, because it allows the locations associated with a professor to be plotted geographically. A professor's events can be shown as points on the map, while movements between consecutive events can be represented by connecting lines. This makes it possible to reconstruct and visually follow an individual life course. The dot map also partly supports Question 2, because concentrations of points can indicate locations where many events occurred.

- **Timeline for chronological life-course reconstruction**
  The timeline supports Question 1 and Question 4. While the dot map shows where events occurred, the timeline shows when they occurred and in what order. This makes it possible to inspect the chronological structure of a professor's life course. When used together with the map, the timeline helps connect spatial movement to temporal development.

- **Location-time visualisation for hubs, trends, and possible overlap**
  The location-time visualisation supports Question 2, Question 4, and Question 5. By combining location and time, this visualisation can show which places were important during specific periods, whether certain locations became more or less prominent over time, and whether multiple professors were associated with the same place during overlapping periods. This makes it useful for identifying temporal trends and possible co-presence.

- **Space-time cube for spatio-temporal patterns**
  The space-time cube supports Question 1, Question 2, and Question 4. It represents geographic space on the horizontal plane and time on the vertical axis, allowing spatial and temporal movement to be shown in one visualisation. This makes it possible to inspect trajectories through both space and time, while also making concentrations of events in certain places and periods visible.

- **Flow maps and Sankey diagrams for recurring movement patterns**
  Flow maps and Sankey diagrams primarily support Question 3, because they visualise repeated movements between locations. Individual life-course movements can be aggregated into larger flows, and the strength of a connection can be encoded using visual properties such as line thickness or flow size. These visualisations also support Question 2, because locations with large incoming or outgoing flows may indicate important academic hubs. When combined with temporal filtering, they also support Question 4 by making it possible to compare flows across different time periods. Flow maps and Sankey diagrams primarily support Question 3, because they visualise repeated movements between locations. Individual life-course movements can be aggregated into larger flows, and the strength of a connection can be encoded using visual properties such as line thickness or flow size. These visualisations also support Question 2,

because locations with large incoming or outgoing flows may indicate important academic hubs. When combined with temporal filtering, they also support Question 4 by making it possible to compare flows across different time periods. In addition, faculty filtering through the people tab supports Question 6, because it allows mobility patterns of different faculty groups to be compared.

4.2.4 Level 4: Algorithm & Implementation

At this level, the visual encoding and interaction requirements are translated into concrete implementation and algorithmic decisions. The implementation should support interactive exploration of spatial-temporal data while remaining responsive and scalable for larger datasets. Furthermore the implementation needs to work in the LUC Plotly Dash and a monetdb database codebase.

- **Efficient storage and querying of spatial-temporal data**
  Since the visualisations rely heavily on filtering by person, location, and time period in order to plot life courses, the database structure should support efficient querying of spatial-temporal event data. Furthermore in order to construct a lifepath of a person, all events of a single person should be easy to be queried. A relational database structure in monetdb centred around, a central persons table with separate events and locations table was therefore chosen.

- **Interactive geographic rendering**
  To visualise trajectories and flows on geographic maps, rendering libraries capable of displaying interactive map-based visualisations are required. The implementation should support plotting large numbers of points and trajectories while maintaining responsiveness.

- **Support for temporal interaction**
  Since users must be able to explore data over time, the implementation should support dynamic filtering using year sliders, timelines, and interactivity through animation. Queries and rendering updates should occur efficiently to maintain usability.

- **Linked and coordinated views**
  Multiple visualisations should be connected through shared application state. Selecting a person, location, or time period in one visualisation should update other visualisations accordingly.

- **Scalability and clutter reduction**
  Since simultaneously displaying many trajectories can lead to visual clutter and performance issues, aggregation and filtering techniques should be implemented. The ability to only select certain professors and hiding the rest reduces the amount of data that needs to be displayed.

- **Support for multiple visualisation techniques**
  The implementation should remain flexible enough to support different prototype visualisations, including flow maps, timelines, and space-time cube representations.

- **Automated data processing pipeline**
  To handle the large amount of OCR-derived JSON data, automated scripts are required to convert, clean, standardise, and insert data into the database. This includes standardising date formats and linking city names with coordinates.

- **Support for faculty-based comparison**
  Since one analytical question concerns differences between faculties, the implementation should support filtering or selecting professors based on faculty. This makes it possible to compare whether different faculty groups show different mobility patterns, hubs, or flows.

  All the described criteria across the 4 levels, will form the basis of the implementation and its evaluation. In the next section we will discuss results from the re-run of the OCR and AI-driven pipeline using GPT-5 and also showcase and discuss the implemented visualisations.

5 Results & Implementation

5.1 Re-running the OCR and AI Pipeline

To acquire all the converted JSON data, the pipeline by Abedi discussed in section 3.2 was re-run using GPT 5 using specifically snapshot version gpt-5-2025-08-07. In addition to just running the full pipeline, the same exact experiment was run as described in Abedi's thesis[1]: a subset of person entries were manually converted by Abedi to JSON as ground truths that can be compared to the output of GPT-3.5, average scores of accuracy are calculated based on 5 runs to account for the variance in output produced by LLM's like Chat GPT. This exact same experiment was re-run for this thesis, now using GPT-5, see figure 6 for the results and a comparison with GPT-3.5 results. Figure 5 shows the complete pipeline and how it is extended.

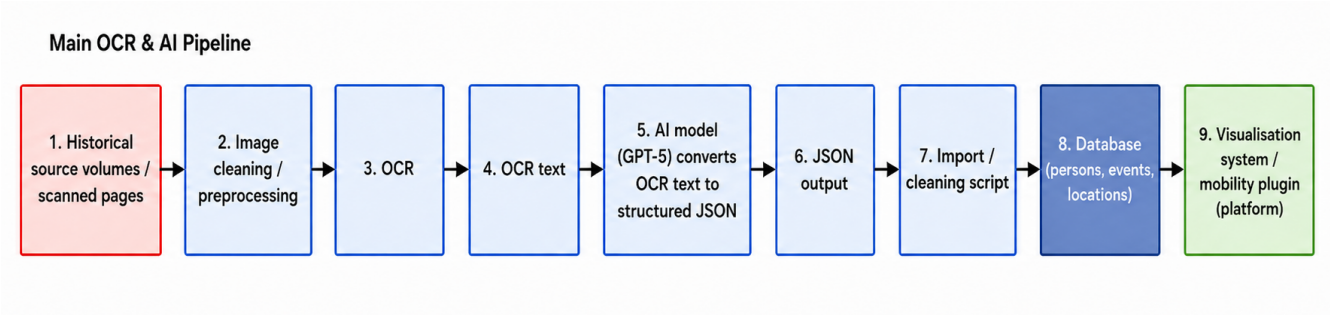


Figure 5: Step 1-4 are exactly the same as in Zahra Abedi's pipeline, however in step 5 we us GPT-5 instead. Step 6 remains the same, but steps 7-9 are the extension implemented by us.

Figure 6 show that on average GPT-5 scores much better than GPT 3.5; on average GPT 5 achieves an accuracy of 80.5% for correct text versus 65.0% achieved by GPT 3.5. For OCR text it is similar, with 79.3% for GPT-5 and 62.7% for GPT 3.5. The average accuracy across most subcategories is also higher for GPT-5, however for careers and particularities they are quite similar with GPT 3.5 perhaps even outperforming GPT-5. Another noteworthy discovery that can be made from this figure is that the variance in performance between runs with GPT-5 is much lower than with GPT 3.5. In other words, GPT-5 seems to produce similar output between runs, suggesting that it might be more consistent in its outputs generation compared to GPT 3.5 where the range between minimum and maximum average accuracy is much larger for most categories. For this thesis and the implementation of visuals of life courses, the categories main_person, education and

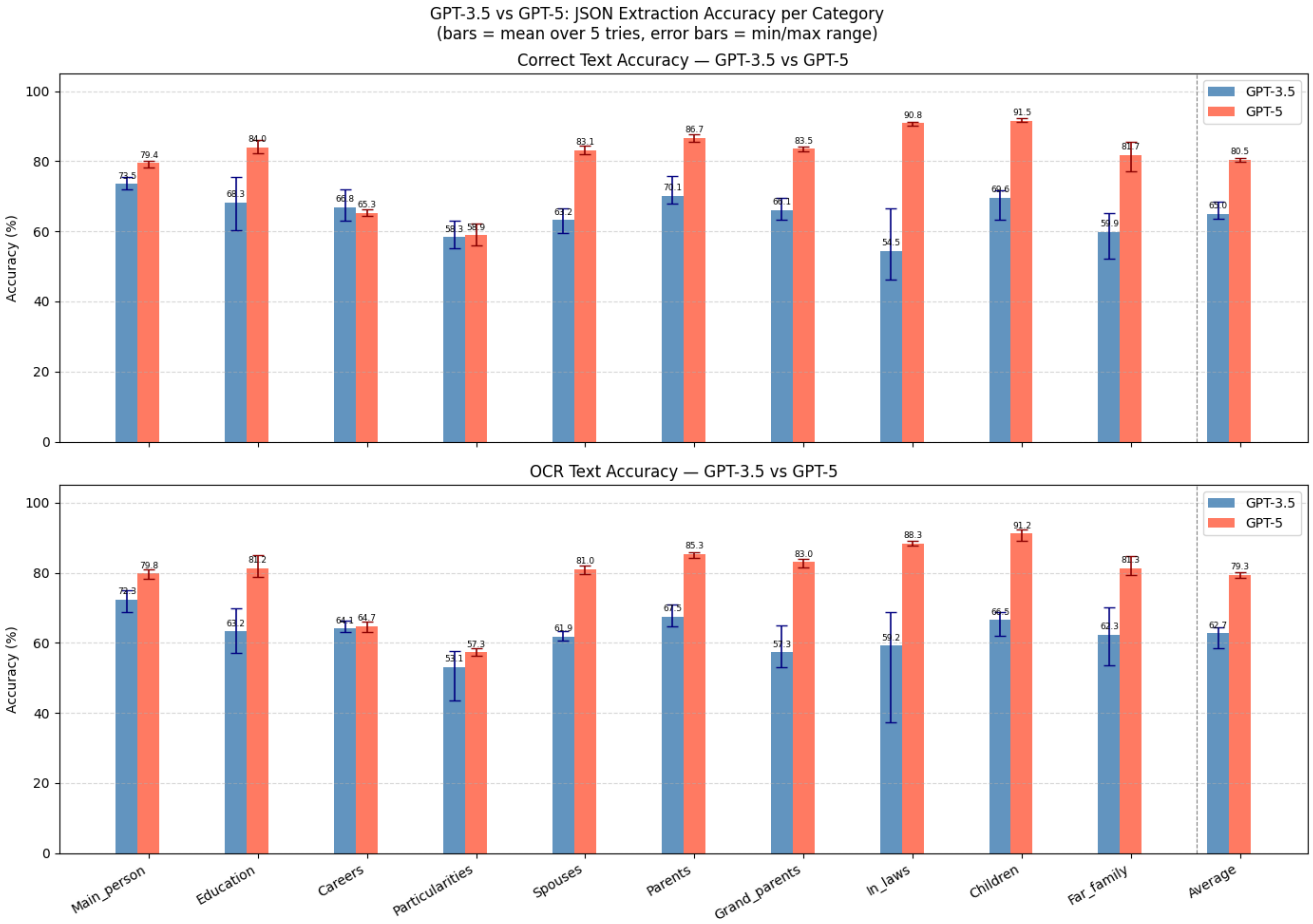


Figure 6: One test was conducted on correct text and one on OCR text. For each subcategory across 5 runs the average accuracy percentage is plotted, with a range indicating the minimum and maximum accuracy in percentage across 5 runs.

careers are of particular importance. These contain the information about the date and location of events related to education, career, birth and death. Table 24 shows the structure of the career and education fields. For main person and education GPT-5 does seem to yield better results, but for careers it seems that it is comparable. Within this evaluation setup, GPT-5 produced higher average accuracy scores than GPT-3.5 for most categories and showed lower variation between runs. Although the GPT-5 output improves the usability of the data, it does not remove the need to treat the visualisations as exploratory rather than historically definitive. To produce reliable historical data, the JSON needs manual post-correction which was beyond the scope of this project.

5.2 Database Implementation

According to the earlier established Munzner's Nested model criteria called Efficient storage and querying of spatial-temporal data, a database implementation should support the easy and efficient querying of data in order to construct life courses. We can define a life courses as a series of events

ordered chronologically and belonging the same person. The complete structure of the career and education field containing all the subfields of the JSON data can be found below in table 1.

Because career and education events both have a similar structure in the JSON data, it is possible to easily store them in the same single table, which has been dubbed events in figure 7. At the moment we distinguish 4 event types which are stored in the event_types tables: birth, education, career and death. Birth and death events are also stored in the table, provided they are accompanied with a date and location. Locations are stored in the location table and referenced by a foreign key in the event table, this allows us to avoid repeat entries of location and allows for queries for selecting all events with certain location_id. Constructing a life courses and querying all the events of an individual can be done by selecting all the events where the person_id matches. A python script that automatically imports all the JSON files into this database was also made to speed things up. This script also enriches the location data by adding longitude and latitude values through the use of geopy[7]. For most locations this yields the correct coordinates, but sometimes if multiple cities or towns use the same name, the wrong one gets chosen. This happens in particular for cities who have a same named counterpart in the United States of America. In addition sometimes the location name might be misspelled or contain multiple locations, preventing accurate coordinate enriching. Because automatic geocoding can introduce incorrect coordinates, the resulting maps should be interpreted as exploratory visualisations rather than fully verified historical-geographic reconstructions. Our current visualisations uses contemporary maps, which may differ from the situation in the time periods in the dataset. Only events that have both a time and location with coordinates, get plotted in the visualisations.

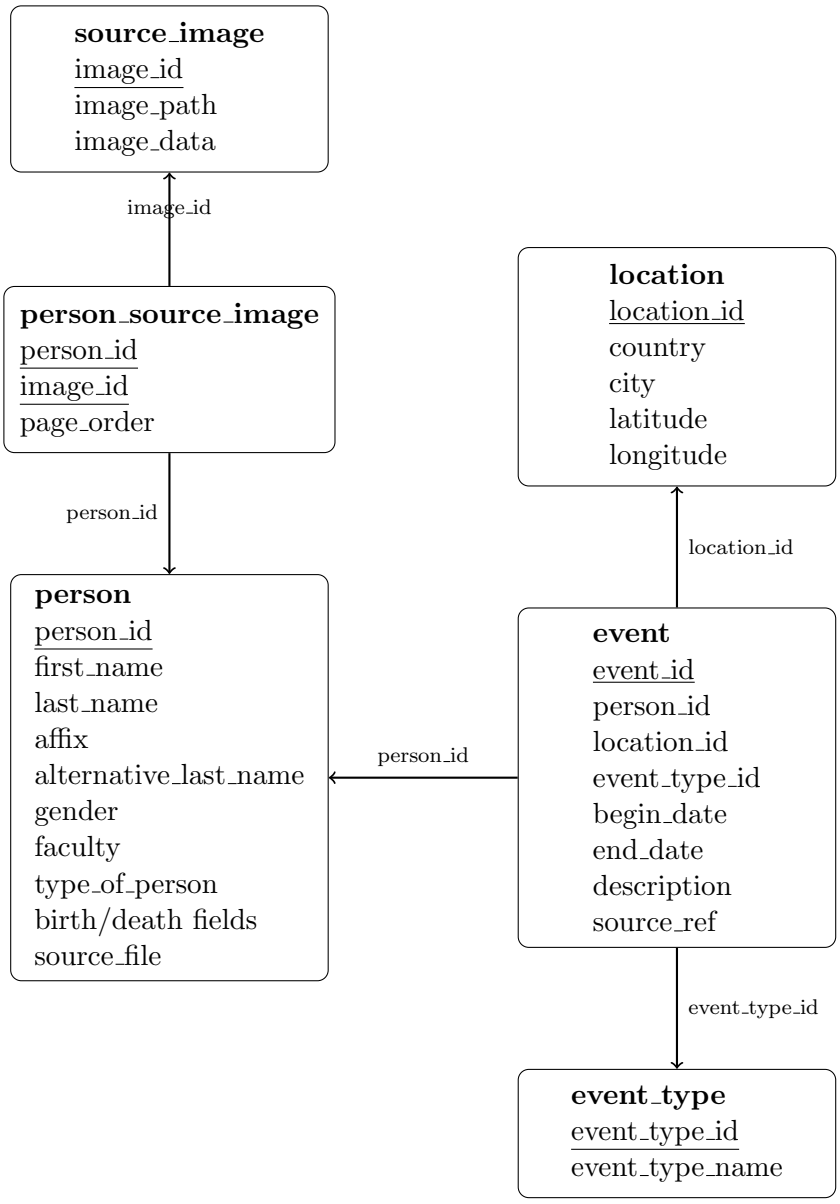


Figure 7: Overview of the LifePaths database schema. Underlined fields are primary keys, and arrow labels indicate foreign keys. Source images are linked to people through the junction table `person_source_image`, allowing one person to be associated with one or more scanned source pages or allowing one page to be associated with one or more persons.

5.3 Implemented Visualisations

Multiple visualisations have been implemented and each is able to showcase different aspects of the data and thus help in answering different types of questions. In this subsection we will discuss the implemented visualisations and what kind of information they are able to show. The evaluation of how well these visualisations meet the previously established criteria will be discussed in section 5.5. The implementation is available in the LUCD GitHub repository.[1]

5.3.1 Dot Map

The dot map is able to plot all events of all professors geographically as life courses, see figure 8. Events belonging to a specific professor can be selected as a trace, where a highlighted line connects the events with one another, see figure 9. This visualisation is able to show the locations where professors moved to and lived. Displaying all events from all professors also gives an indication of around which location(s) events are clustered, which can suggest potential important hub areas for academic activity. The dot map does not directly aggregate the number of visits, arrivals or departures associated with cities or regions. Any hubs can therefore only be identified by looking at where many events are clustered closely together. Anti-cluttering techniques for this visualisation are options such as selecting only specific types of events or only showing events and life courses of selected people.

[1] Source code repository

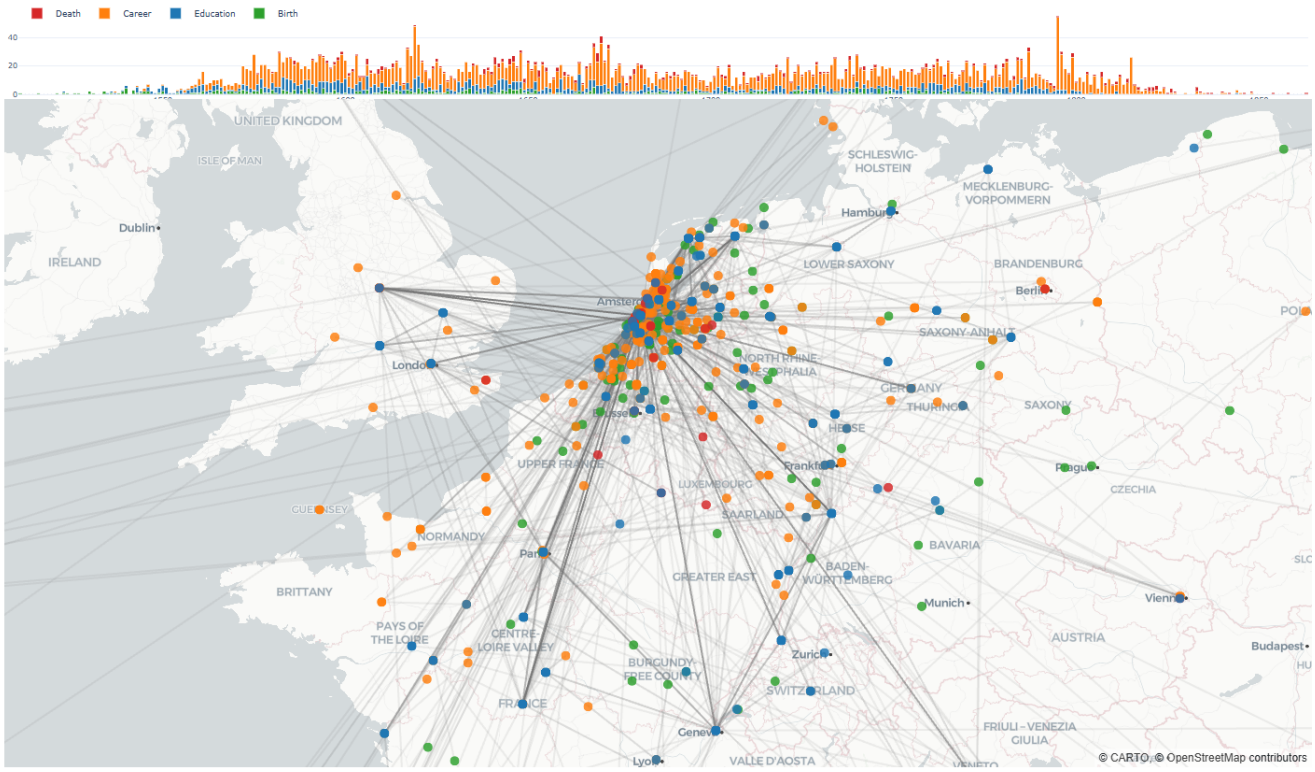


Figure 8: This image shows all the event data from all professors plotted at once. It gives an idea of the locations that are involved and connects events from a life course of a professor with lines. Thicker lines indicate that this specific point-to-point route has been traversed more often than thinner lines. Green dots represent birth events, blue dots education-related events, orange dots career-related events and red dots indicate place of death. The map is zoom-able and can be explored freely. This map is from OpenStreetMap and is a contemporary map. Above the map the amount of events of each type can be seen per year.

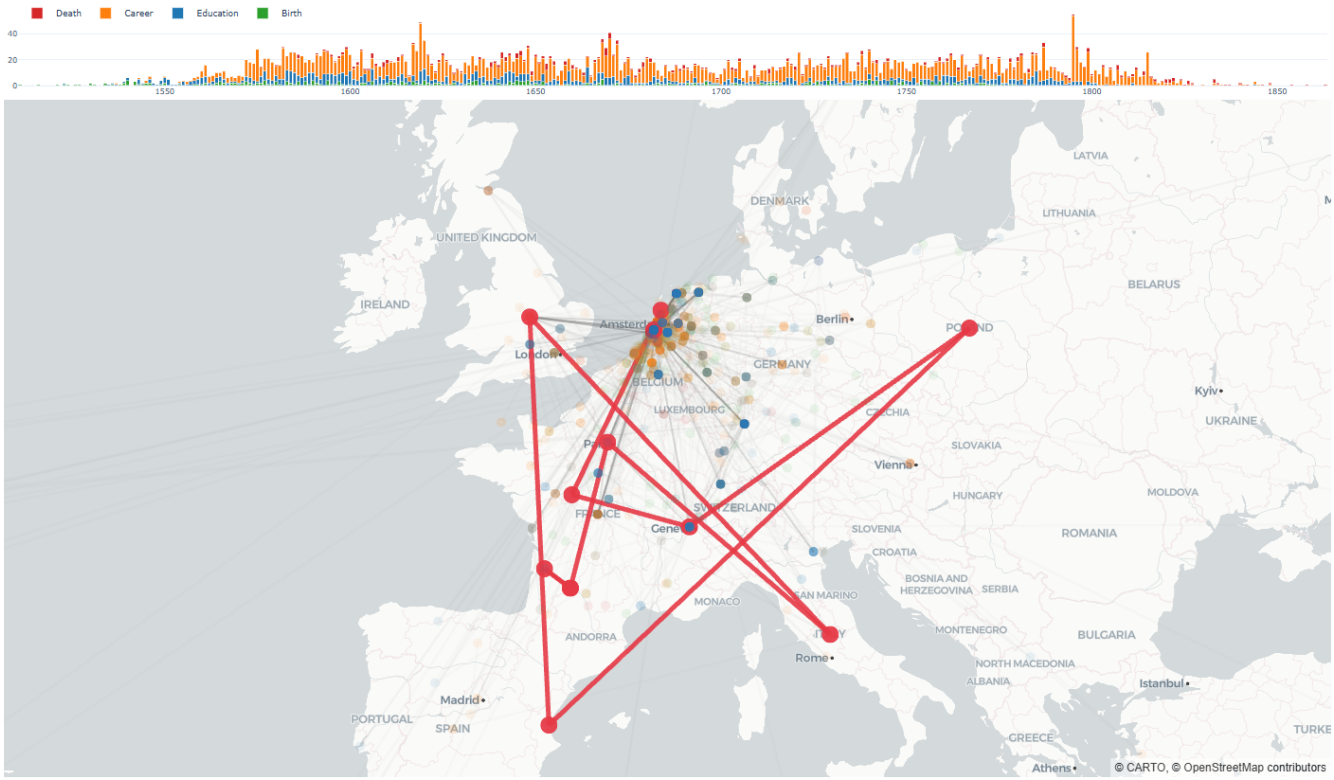


Figure 9: In this specific example the life course of Josephus Scaliger has been plotted and highlighted in red. It shows that he has been all over Europe in the span of his life. It is possible to highlight the life course of more than one professor, in which case they will be highlighted with different colours.

5.3.2 Timeline

The timeline is a common way to plot events over time. It was implemented mostly as a tool for comparison, to see if different professors lived in roughly the same time period. It can therefore help with confirming and visualising a small amount of individuals and seeing if time-wise they lived in the same or overlapping periods, see figure 11. If no professors are selected, it will show all events over the years, see figure 10. Hovering over an event displays more information about that event, see figure 12. The timeline is therefore not suitable as a stand-alone visualisation for geographic mobility, but it is useful in combination with the other visualisations.

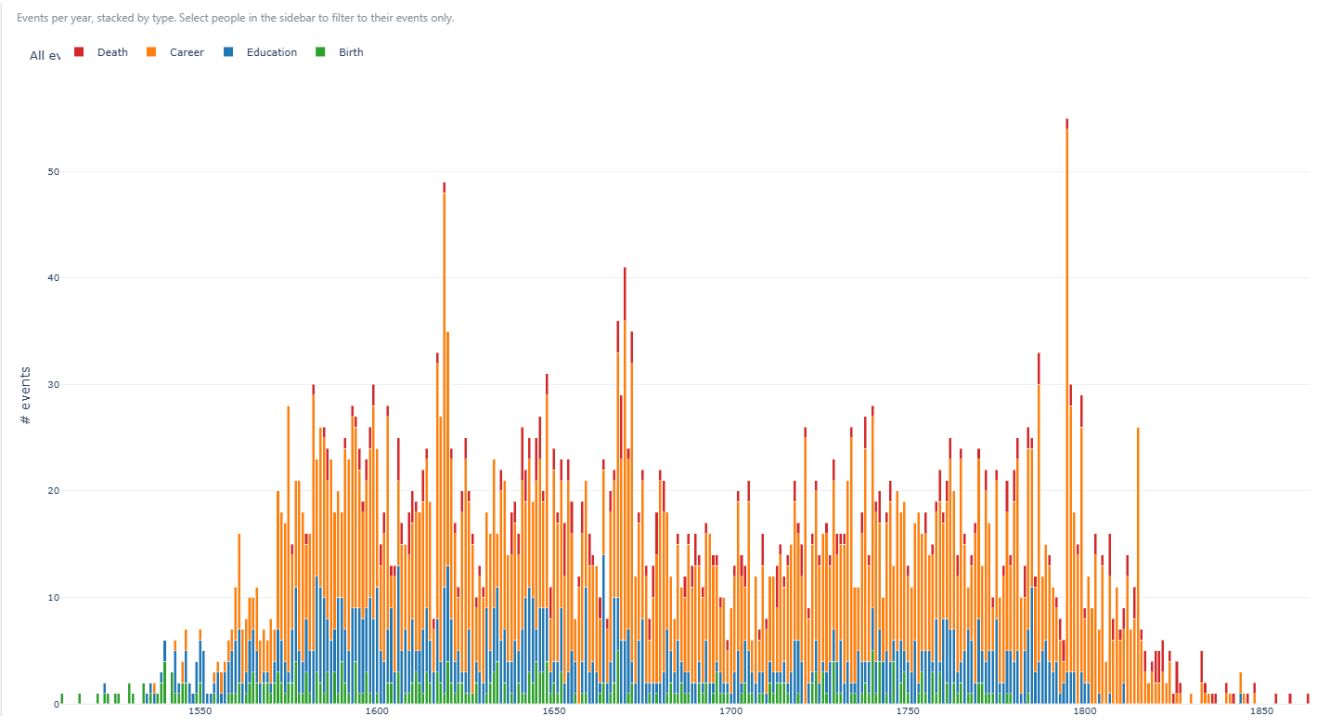


Figure 10: Events are plotted and coloured by type for each year, creating a bar plot that shows the distribution of events per year. Perhaps it is possible to discern patterns like years of plagues or disasters by looking at how many death events occurred in a year. However, with a dataset of only 394 professors this might not be accurate.

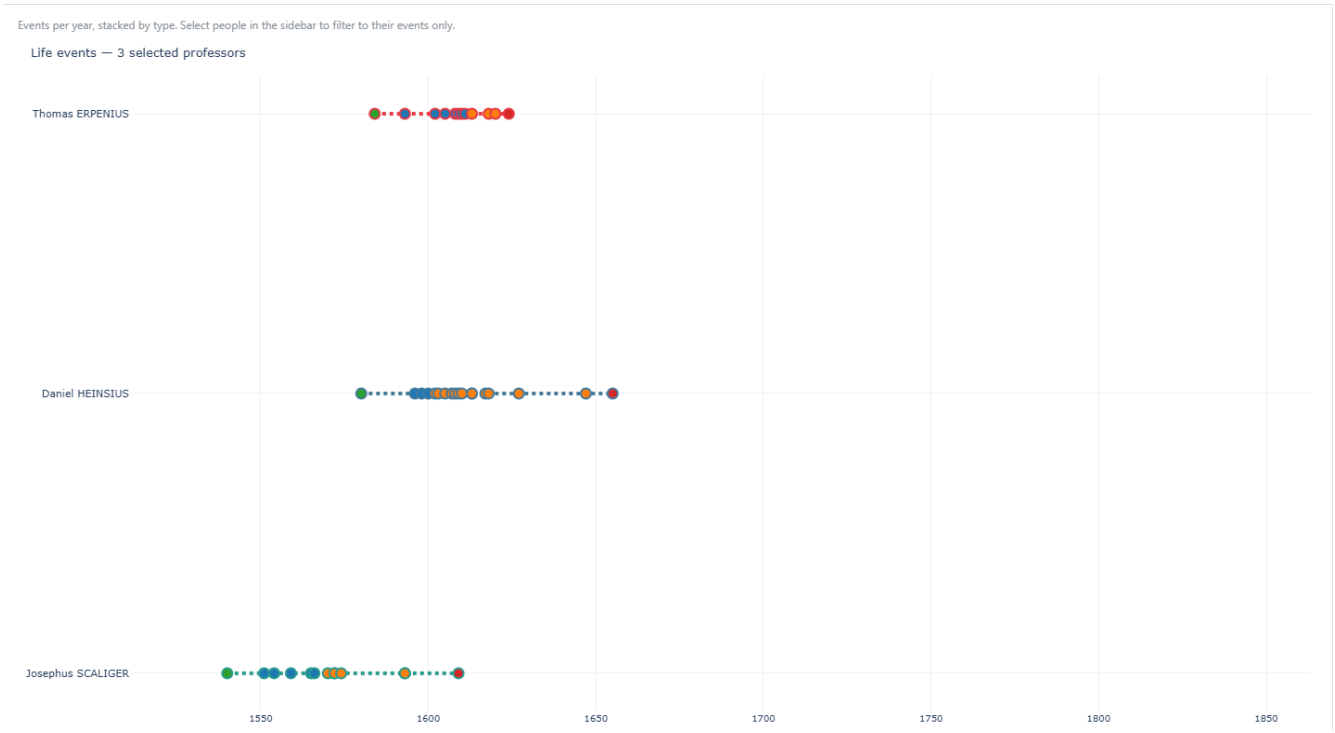


Figure 11: When one or more professor(s) are selected it will display their timeline(s). In this example Josephus Scaliger and two of his students [30] are plotted. In this visualisation it can be seen that all three lived during overlapping periods and that the students started their education in Leiden when Scaliger had already started working in Leiden.

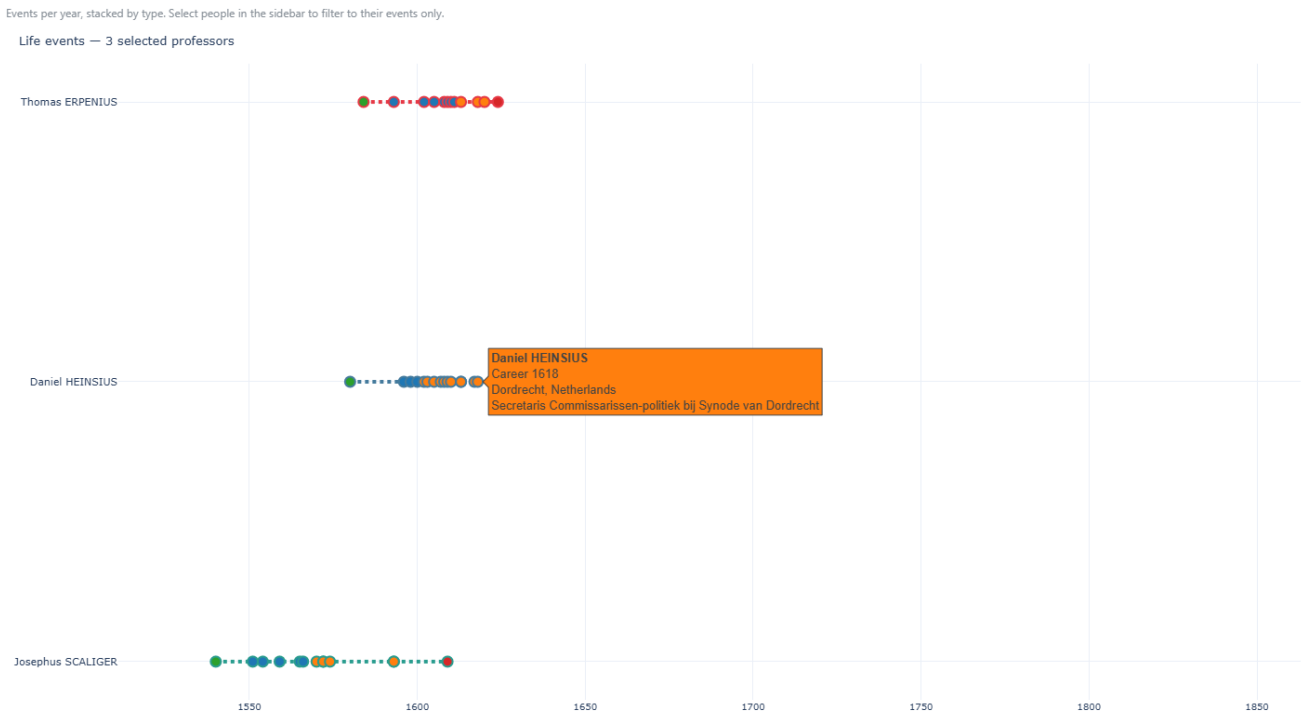


Figure 12: Hovering over an event displays the information that is available about that event.

5.3.3 Location x Time Timeline

This visualization plots location against time. Locations here are cities, regions or countries. It is able to visualize where a person is over time without the need of a geographical map representation. Selecting multiple people plots multiple different life courses. Since this visualisation clearly shows time compared to location, it is able to show whether professors were recorded in the same location during the same period. In other words, this visualisation can help with identifying possible co-presence. This visualisation could also indicate which cities were hubs in certain periods; if many different life courses converge on the same location, this could be an indication of that. However, plotting many different life courses in this visualisation, like with the dot map, leads to visual clutter of many lines crossing one another, where it becomes difficult to discern larger patterns in the data. Figure 13 shows the visualisation with all data for the top 20 locations, while figure 14 and figure 15 show the same selected professors as in figure 11.

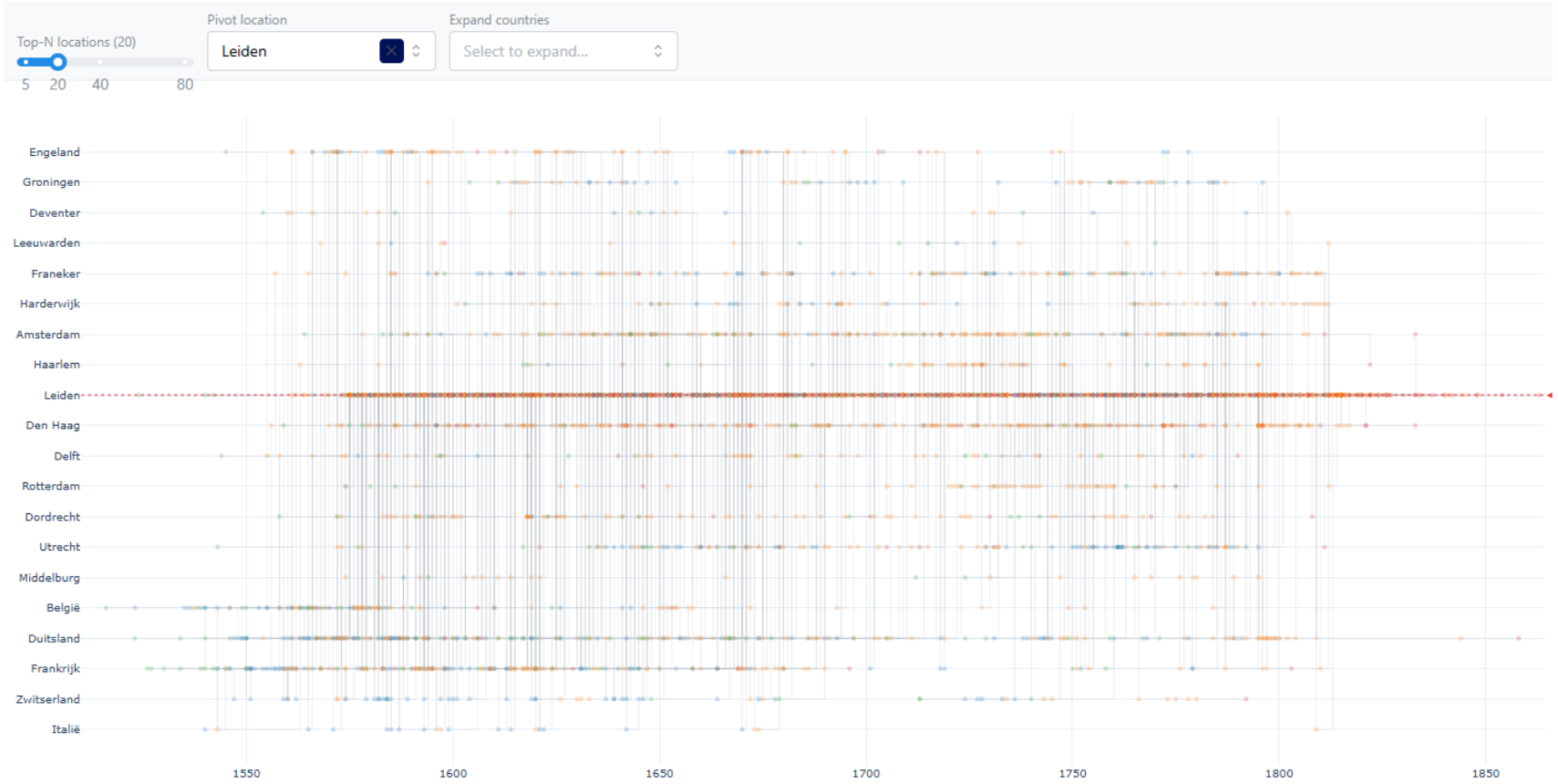


Figure 13: This plot shows all the data in the dataset for the top 20 locations with the most events. There are three selection features in the top left corner. The first one is the ability to only show the top-N locations, this helps reduce visual clutter by removing locations that do not have many events. The second option is the ability to select a pivot location as point of comparison or reference. In this case Leiden was selected, all the locations listed above Leiden are geographically north of Leiden and are ordered from furthest north to closest to Leiden. All the locations below Leiden are listed in exactly the same way except now to the south of Leiden. The choice for the north-south order was based on the fact that there is not much to the west of Leiden in terms of events in this dataset, except England, and some outliers in the Americas. A different pivot sorting on either west-east or something else can be implemented later if it makes sense. The final option is expand countries. Because there are a lot of different locations in the data, by default foreign locations are combined under one category, this being the contemporary nation state where they are located in. This reduces visual clutter, however if detailed information is important, it is possible using this option to show all locations in that country.

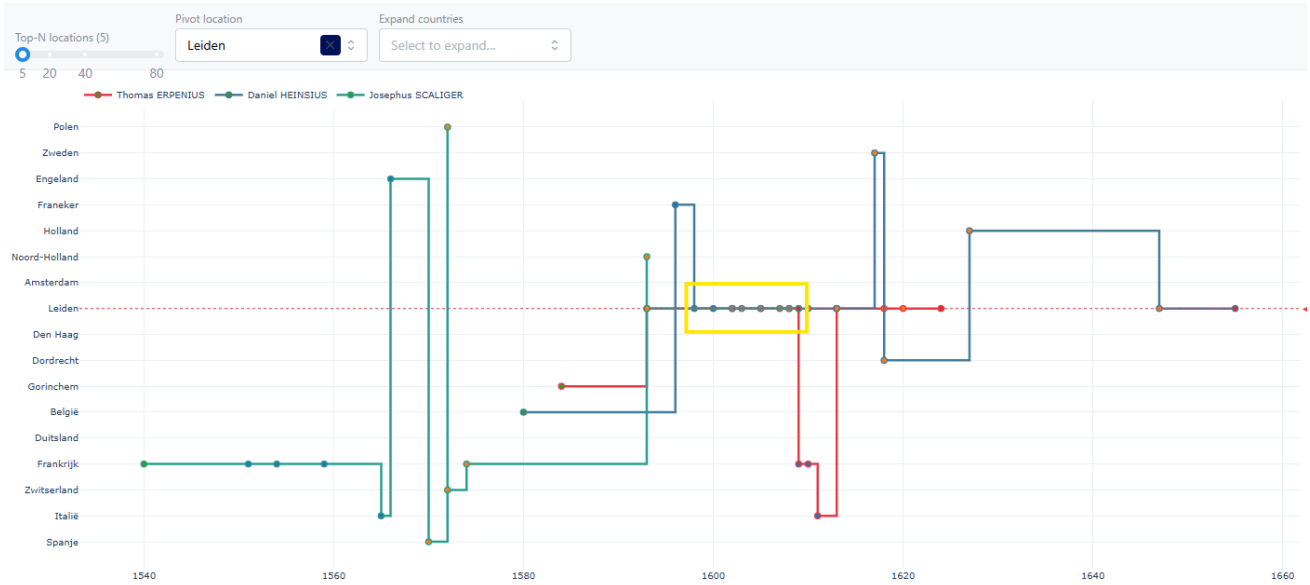


Figure 14: In this example we plot the same professors as in figure 11. This visualisation is able to show how professors move from location to location over time. As a result it becomes possible to compare and see whether professors were recorded in the same location during the same time. The data suggests that in the time period 1598–1609, highlighted by the yellow box, Scaliger and his two students were all present in Leiden at the same time. Note that when professors are selected, the top-N nodes are overridden to make sure that all the locations visited by the selected professors are displayed.

Figure 15: In this example we have the same situation as in figure 14, but this time we have zoomed in on the period of possible co-presence and turned on the toggle for showing overlaps. This automatically highlights periods where people were recorded in a location at the same time.

5.3.4 Space Time Cube

The Space Time Cube, like the Location x Time timeline, plots location against time. What is different here is that the location is represented in two dimensions: longitude and latitude, like on a geographic map such as the previously discussed dot map in figure 8, instead of one categorical location name. Plotting this against time creates a three-dimensional visualisation, where the x- and y-axis represent coordinates of locations and the z-axis represents the continuation of time. Events get plotted based on their location and time. Essentially this is a dot map but with the dimension of time also accounted for. It is able to visualise where professors moved throughout their lives, but like the dot map it also suffers from visual clutter when a lot of life courses are plotted. With this visualisation it is also possible to identify regions of possible academic importance; if there are many events in a certain area, this can be an indication of that. Periods of possible co-presence can also be identified from this visualisation. Figure 16 shows all European locations, while figure 17 and figure 18 show selected professors and highlighted overlaps. From a usability point of view, the space-time cube has both advantages and disadvantages compared to time-based slices. The main advantage of the space-time cube is that it integrates geographic space and time into a single visualisation. This makes it possible to inspect life courses as continuous paths through both space and time, instead of separating movement into different temporal snapshots. However, the space-time cube is also more difficult to interpret than a set of time-based 2D slices. Because it uses a 3D representation, users have to deal with perspective, depth, occlusion, and navigation. When many trajectories or points are shown at the same time, parts of the visualisation may hide behind other parts. The user may need to rotate, zoom, or pan the cube in order to understand the data. It is useful for showing the combined spatial and temporal structure of the data, but

time-based 2D slices may be more suitable depending on the analytical task that needs to be performed[13].

Figure 16: This is an example of the space time cube. All the data from locations within Europe are plotted. It is possible to interact with the plot by tilting it, scrolling it and zooming in and out.

Figure 17: Like with the previous visualisations we have plotted Scaliger and his two students. We can see where they have been and when. This visualisation also supports the automatic highlighting of periods of possible co-presence, see figure 18.

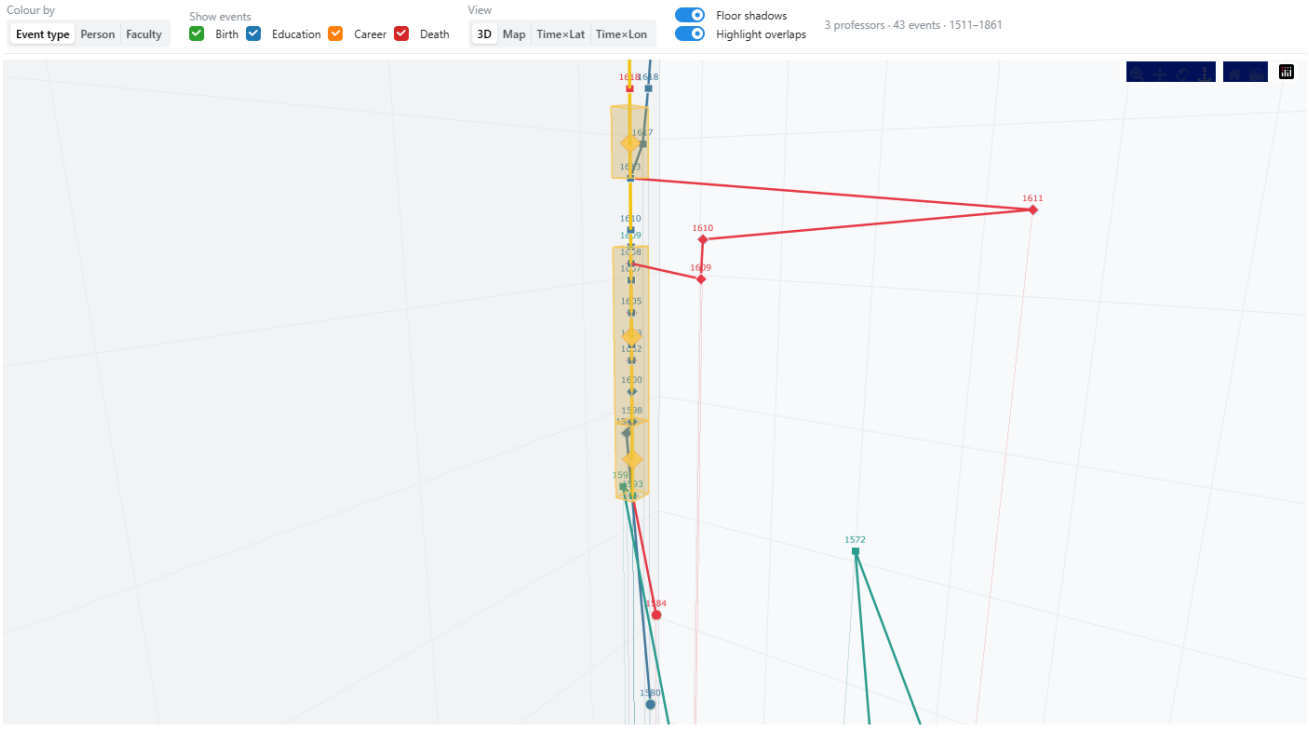


Figure 18: This is a zoomed-in version of figure 17. The periods of possible co-presence are highlighted with yellow boxes. These align with the highlighted periods of possible co-presence as seen in figure 15.

5.3.5 Flow Maps and Sankey Diagram

The implementation contains multiple flow maps and a Sankey diagram, all based on the idea of visualising flows. Instead of plotting individual paths between locations for each person and event, they are aggregated into a single path, i.e. a flow. This not only helps reduce visual clutter, but can also reveal patterns in mobility. The Sankey diagram gives a process-like overview of this without a geographic map, while the flow maps use a geographic map as background. Since flow maps do not automatically show time, extra features for this were implemented. Flows can be coloured based on the period in which the first, median or last occurrence of someone following that route takes place. Alternatively, flows can also be deconstructed into multiple coloured flows that represent the sub-flow of a certain period. Additional filters that combine flows from different locations in a contemporary nation state into a single flow can reduce visual clutter. Moreover, most locations abroad only have small flows where only one or two people traverse that route. By displaying only the top-N nodes, most flows coming from abroad would be regarded as insignificant. By aggregating foreign locations at the country level, international mobility remains visible without overwhelming the visualisation with many individual locations. This shows that movements to and from locations outside the Netherlands still form a substantial part of the dataset, which would otherwise be filtered away by the minimum flow size filter. Displaying only the top-N nodes or selecting a minimum size of a flow can help leave out clutter and only show the most important patterns. However, by aggregating life courses from multiple professors, individual details about professors are more difficult to discern from this visualisation. This visualisation is good for gaining insights into general mobility patterns through time, but is less useful for visualising data for individuals or a small amount of people. Figure 19 shows the Sankey diagram, while figure 20 shows the geographic flow map. Figure 21 and figure 22 show the temporal modes.

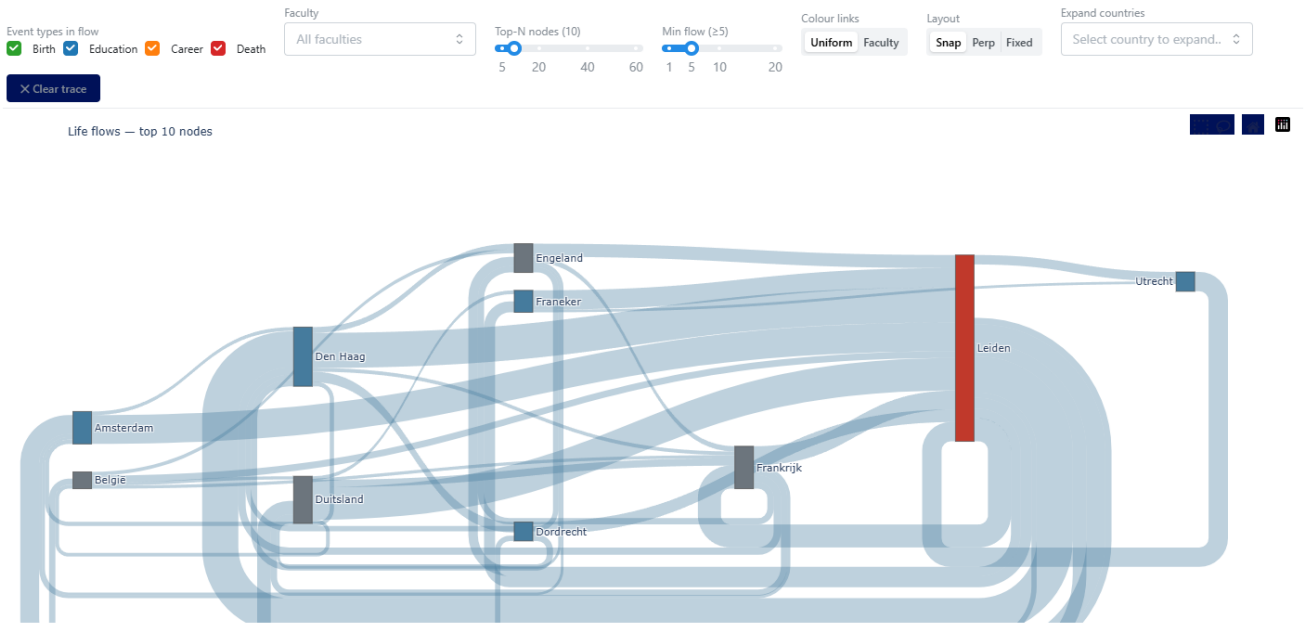


Figure 19: The Sankey diagram is able to show the flows from locations. Filtering is possible based on faculty, displaying only the top-N nodes or selecting a minimum flow requirement. Like in previous visualisations countries can be expanded. The left side of nodes are the incoming flows, while the right side are the outgoing flows

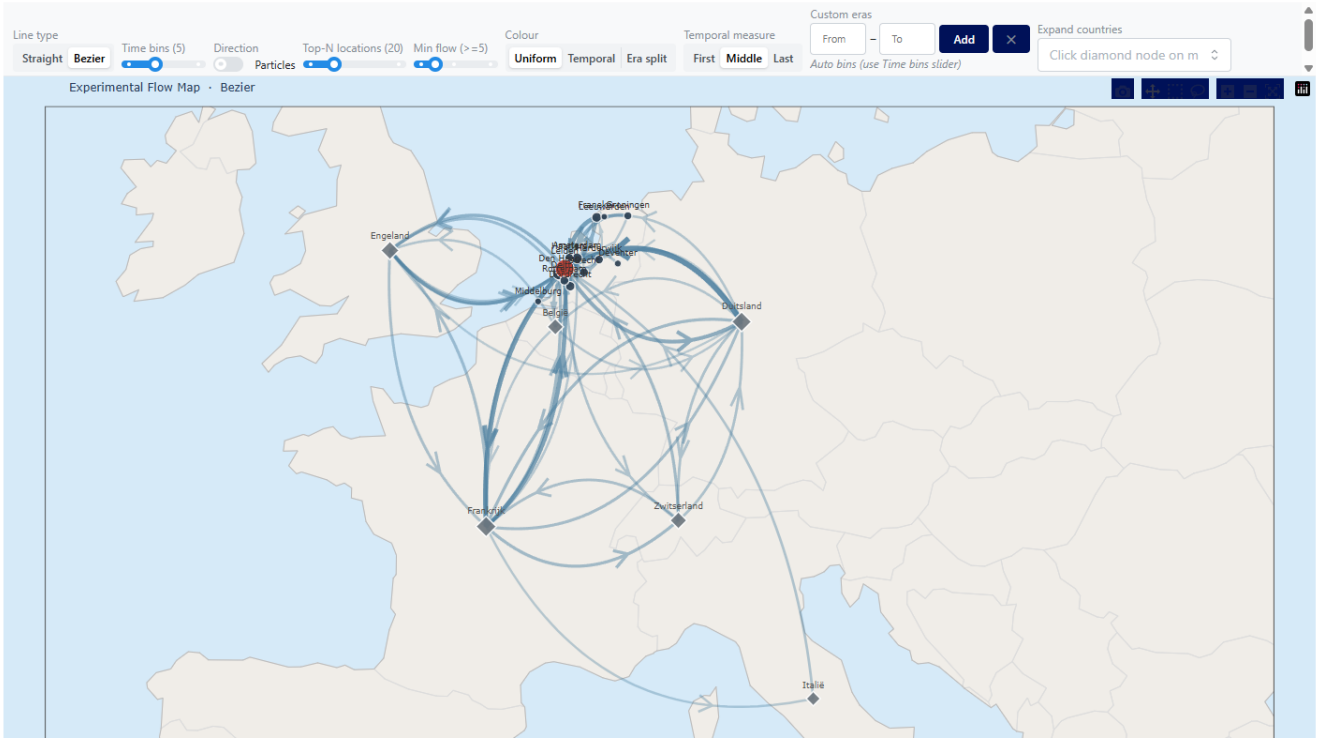


Figure 20: This flow map shows the most important flows between the top 20 locations, where a minimum flow of five is required. Instead of lines and arrows, it is also possible to toggle an animation of particles, where the direction of the movement of those particles represents the direction of movement. Here it is also possible, like with the other visualisations, to expand countries to see more detail. In addition there are two ways to visualise time. These can be seen in figure 21 and figure 22.

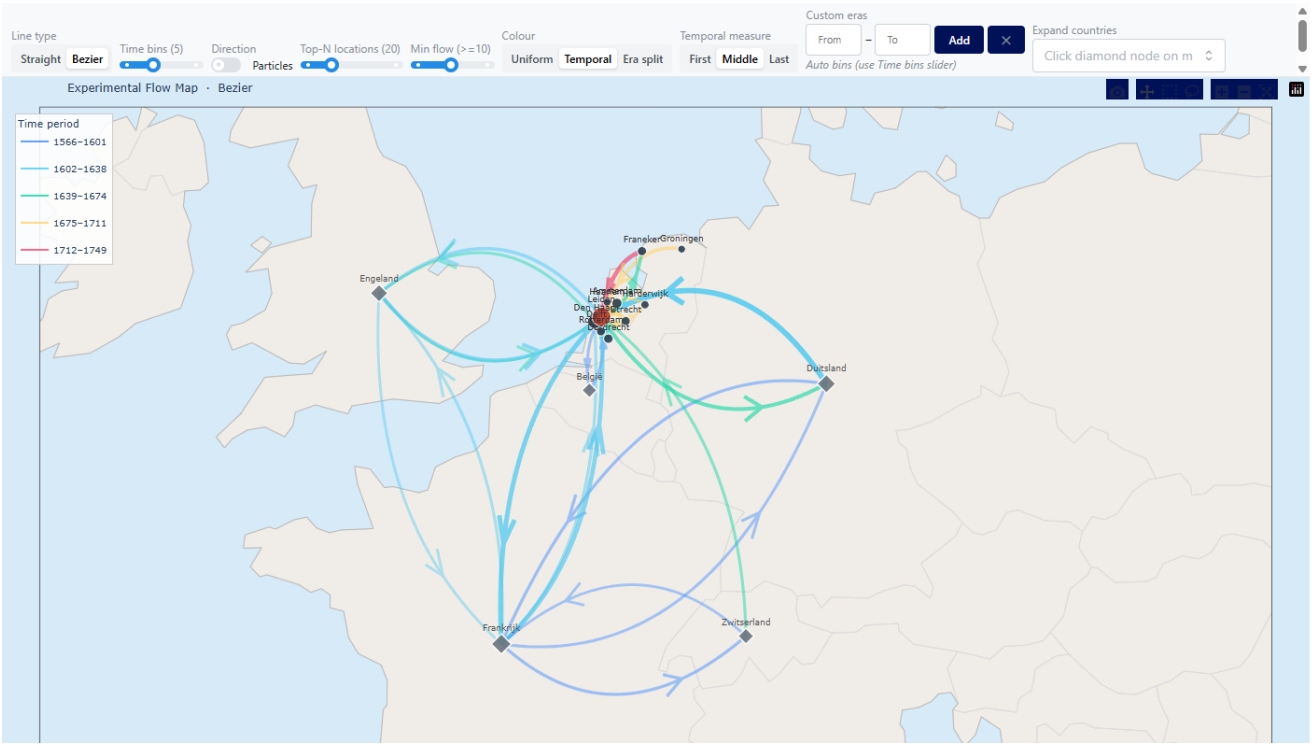


Figure 21: In this example the temporal mode is turned on and set to middle. What this does is that it will colour the lines according to a manually selected period, based on the period in which the median transition timewise in that flow falls. Alternatively, it is possible to change the temporal measure to first or last, where first means it will colour the flow according to the first transition in that flow and last according to the last time someone moved in that way. Using the time bins filter it will divide the selected time interval in equal periods, but it is also possible to define custom eras.

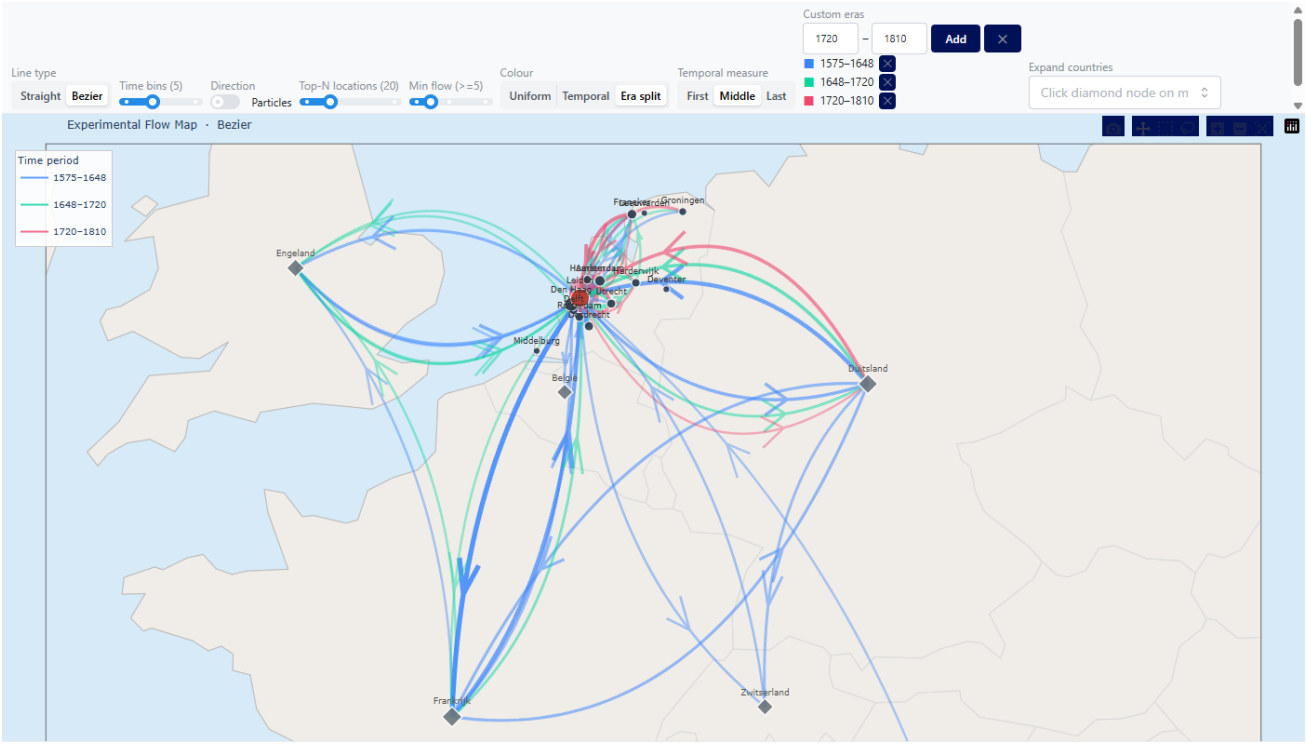


Figure 22: In this example custom eras are defined that roughly correspond with the Eighty Years' War, the Golden Age, and the downfall of the Republic. The colour mode that is selected is era split. This splits each flow in separate subflows per era. This visualisation is able to show how mobility changed over time, and how interactions between locations changed. From this specific example the data suggests that interactions with Switzerland and France mostly took place before 1648, indicated by the blue edges and arrows. Additionally, after 1720 we can see that most recorded interactions were contained within the Netherlands or with Germany.

5.4 Auxiliary implementation features

The implementation contains some extra features that support the analysis and filtering of the main visualisations. The timeline discussed in section 5.3.2 can be seen as an auxiliary feature, but since it is a type of visualisation it was decided to discuss it as a separate visualisation. The features discussed here are not visualisations themselves, but they do support the multi-view implementation. This includes the side bar where buttons to each separate visualisation are located, as well as filter options based on time, location and people. It also includes a separate table section where professor data can be displayed in table form and selected. For each professor it is possible to see a more detailed overview of their information including the source. These features are shown in figure 23, figure 24 and figure 25.

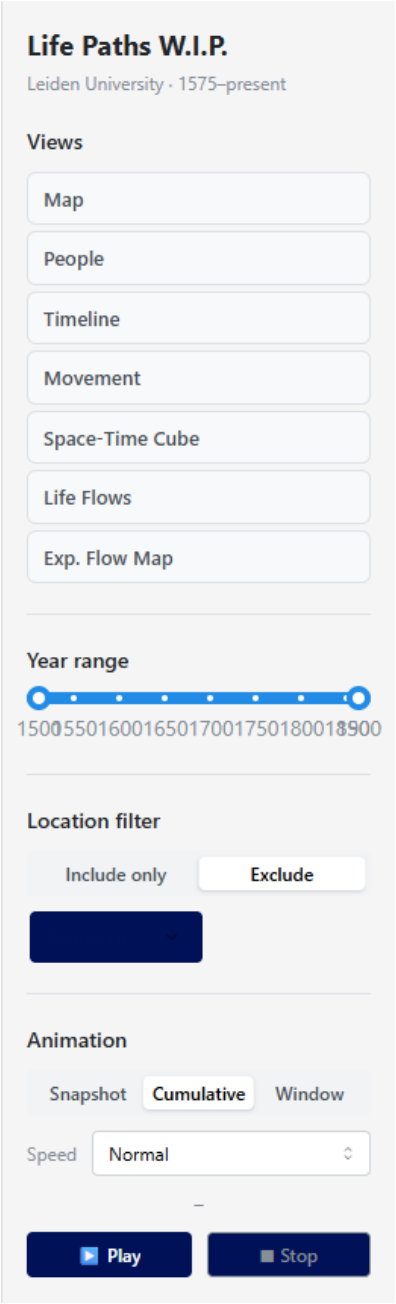


Figure 23: The side bar is always accessible on the left-hand side of the screen. It provides buttons to change visualisations and extra options and filters. These include a year range selector, location filter and an option for animation. This animation is rudimentary as it simply changes the year. Snapshot means showing only the data from the current year, cumulative means showing all data so far, and window means showing only data from a selectable window compared to the current year. All filters that are selected in this sidebar always apply to all visualisations.

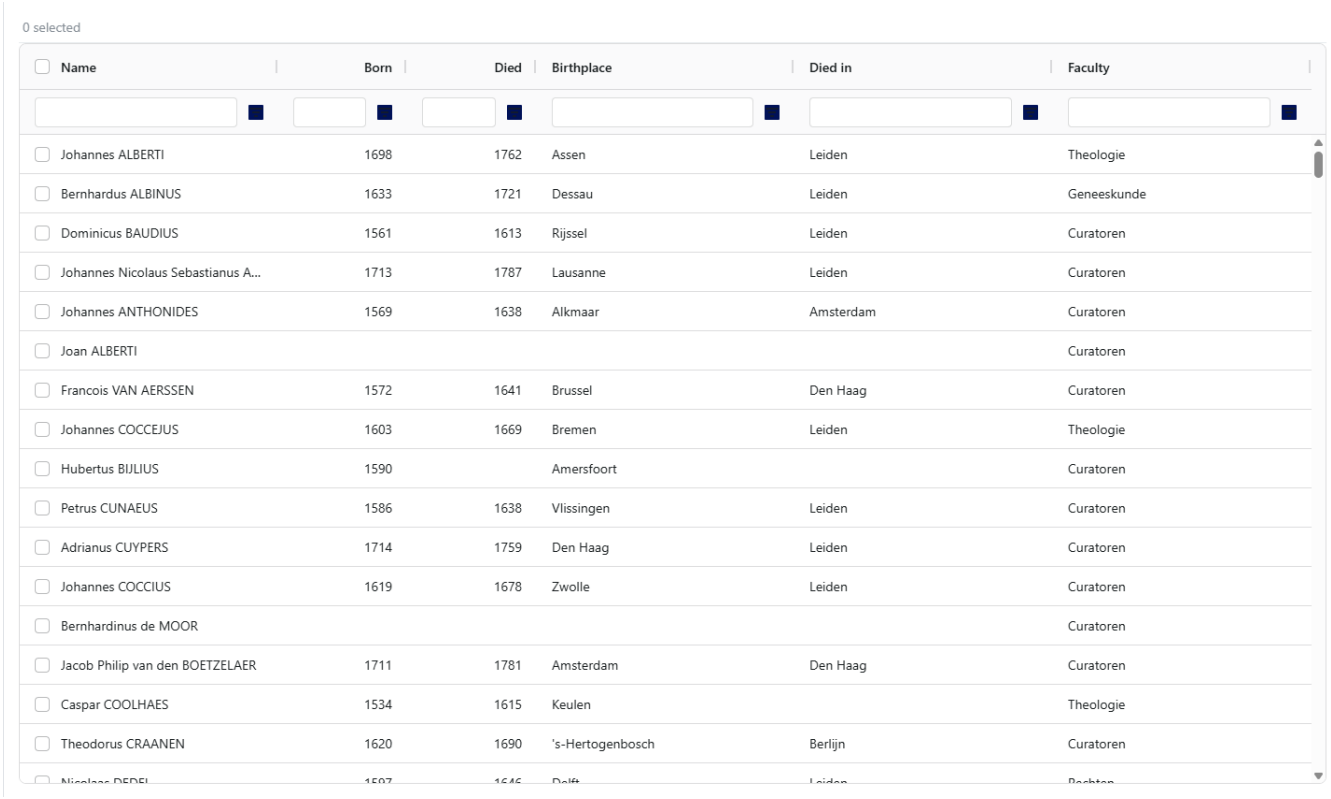


Figure 24: This page provides tabular access to the data from the dataset. From here it is possible to select professors to be visualised across all visualisations. Filtering and searching is possible on numerous categories. The faculty Curatoren seems to be a misinterpretation by GPT-5 during the pipeline phase.

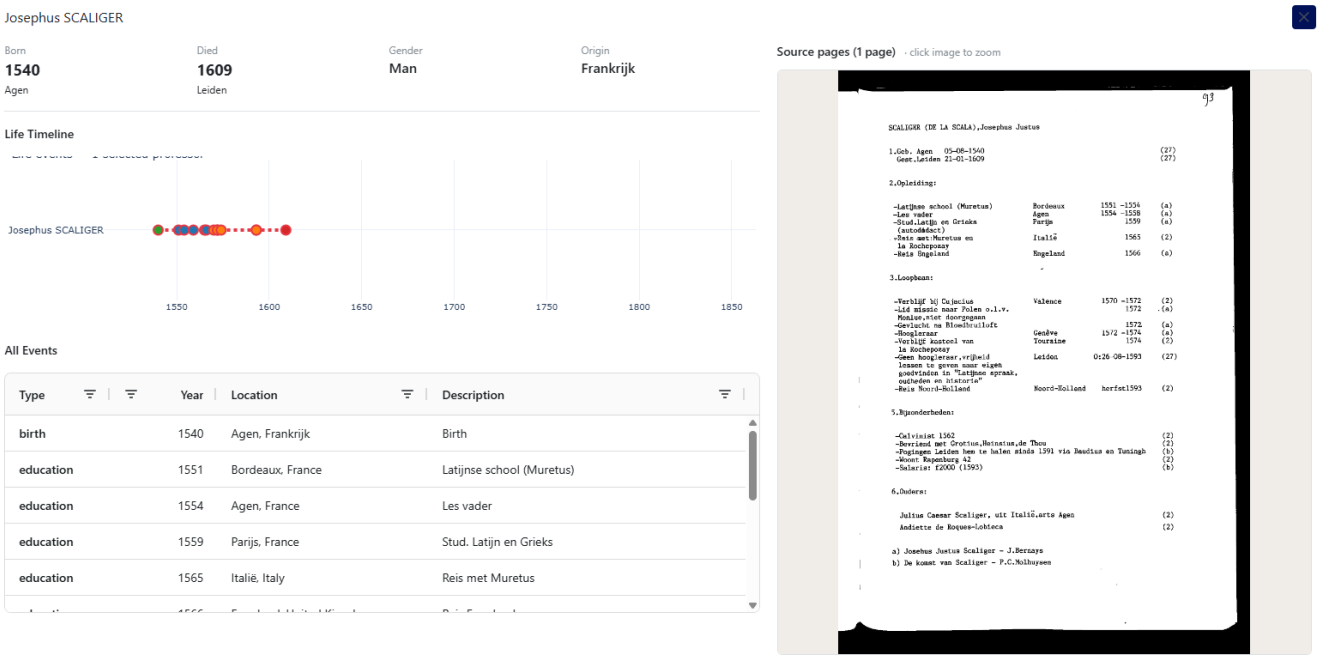


Figure 25: Once a professor has been selected, it is possible to view more data about them. This includes a mini timeline and a chronologically ordered list of all the events that belong to that professor. In addition, the original source page is also available on the right-hand side, which can be zoomed in on. This helps for faster checking if the data provided by the OCR and AI pipeline is accurate or not.

5.5 Evaluation Against the Established Criteria

In this section we will evaluate our visualisations and implementation on all the previously established criteria in Munzner's model in section 4.2. We do this by looking at each criterion individually, starting with the Algorithm and Implementation level and working our way up to the historical domain level criteria. We reason and try to validate each level by looking if all the criteria are met.

5.5.1 Level 4: Algorithm and Implementation

- **Efficient storage and querying of spatial-temporal data**
  The JSON data derived from the OCR- and AI-driven pipeline gets stored in a database built upon MonetDB. Visualisations are able to query the database easily and feel responsive, with minimal lag. Even when querying and displaying all data and life courses, the visualisations still respond fast and the initial loading is short. We consider this requirement met.

- **Interactive geographic rendering**
  Multiple visualisations like the dot map, flow maps and space time cube are capable of plotting data on a geographic map. All of these are able to plot a large amount of life courses simultaneously without strongly affecting responsiveness. We consider this requirement met.

- **Support for temporal interaction**
  Users should be able to change the time frame and interact temporally with the visualisations in order to get a sense of temporal influence on patterns. Users are able to do this through the use of year interval sliders, animation, and alternative temporal modes as seen in the flow map in figure 21 and figure 22. We consider this requirement as met.

- **Linked and coordinated views**
  Users should be able to compare multiple visualisations with one another based on selected professors, locations, or time period. The side bar and professor table as described in section 23 and figure 24 provide these options, and selected options persist across all visualisations. We consider this requirement as met.

- **Scalability and clutter reduction**
  Visualisations should be able to support a large amount of data points without turning into too much visual clutter. Options to reduce visual clutter are mostly limited to filtering options. An example is adjusting the time period, selecting a subset of people, including or excluding locations, collapsing locations outside the Netherlands into a single country, and in case of the flow maps only displaying the top-N locations or setting a minimum flow requirement. This does not remove all visual clutter, especially in visualisations where many individual life courses are plotted at once, but it does make the visualisations more usable. We consider this requirement as mostly fulfilled.

- **Support for multiple visualisation techniques**
  The implementation contains numerous visualisations such as the dot map, space time cube, timelines, flow maps and Sankey diagram. We consider this requirement met.

- **Automated data processing pipeline**
  The JSON data delivered by the OCR- and AI-driven pipeline gets automatically imported

into the database and is enriched with location coordinates using a script to automate this process. We consider this requirement met.

Most requirements of level 4 are fulfilled, with scalability and clutter reduction being mostly fulfilled rather than completely solved. Because of the above examples and arguments for how the requirements are fulfilled, we consider that the implementation is sound and performs adequately. We now move on to the next level: Visual Encoding and Interaction Design.

5.5.2 Level 3: Visual Encoding and Interaction Design

- **Tracing individual lifepaths**
  Most visualisations support the visualisation of life courses of professors. However, the Sankey diagram is not ideal for this, as it is timeless and does not show a chronological order of events. The dot map, location x time timeline, space time cube and selected-professor timeline construct life courses by plotting events as points, and connecting events in chronological order with the use of lines or edges. We consider this requirement met.

- **Identifying academic hubs**
  Academic hubs or regions can be identified by either looking at how events are clustered on the dot map and space time cube, or by looking at the Sankey diagram and flow maps, as they show the cities or locations with the most inbound and outbound traffic. This can be a sign for that location being of academic importance in relation to Leiden. Since this is based on patterns in the dataset, it should be interpreted as an indication rather than definitive historical proof. We consider this requirement met.

- **Detecting recurring moves between cities**
  The flow maps and Sankey diagrams support this. Moves made between locations by different professors in their life courses are aggregated into a single flow. The thickness of the flow corresponds to the amount of recurrence in that specific move across the data. We consider this requirement met.

- **Comparing mobility over time**
  Seeing how mobility changes over time is possible through the use of the space time cube, which plots life courses in both space and time. The flow map is also able to visualise this by using colour in the temporal and era split mode, as seen in figure 21 and figure 22. The latter is especially strong at showing how flows from point A to point B across multiple different time periods change and thus how mobility changes. We consider this requirement met.

- **Identifying possible encounters**
  This requires the ability to detect overlaps in both location and time-wise events for different professors. The space time cube and the location x time timeline both have a toggle to automatically highlight periods of possible co-presence, as seen in figure 15 and figure 18. This does not prove that professors actually met, but it does show that they were recorded in the same place during overlapping periods. We consider this requirement met.

The visualisations support the requirements of identifying and visualising the patterns established for level 3. With the requirements being met we argue that this level has been validated. We now move on to the next level: Data & Task Abstraction.

5.5.3 Level 2: Data & Task Abstraction

- **Where did professors move throughout their lives?**
  Visualisations support the construction of life courses by ordering life events chronologically and connecting them in the visualisations. For every professor it is possible to construct their life course and see where they have lived and when. This can be seen for example in figure 9, figure 14 and figure 17. We consider this requirement met.

- **Which cities or regions acted as important academic hubs during certain periods?**
  The visualisations support this either through event density on the map and through visualisations that track the amount of events and incoming or outgoing traffic. Examples of this can be seen in the dot map in figure 8, the Sankey diagram in figure 19, and the flow map in figure 20. This requirement met.

- **Are there recurring geographic mobility patterns between universities or cities?**
  Some visualisations aggregate recurrent mobility into flows, revealing which routes were more common than others. Larger flows between cities can indicate a stronger academic relation between those locations. This can especially be seen in figure 20 and figure 19. We consider this requirement met.

- **How did the mobility of professors change over time?**
  Multiple visualisations offer insights into this. The flow map with temporal colouring and era splits in particular provides good insights, as can be seen in figure 21 and figure 22. We consider this requirement met.

- **Can trends in migration patterns be identified?**
  By using the flow map and Sankey diagram, recurrent movement patterns can be identified across the dataset. By selecting professors who are suspected to have interacted with one another, or by selecting professors from a specific faculty, it is also possible to determine whether this subset exhibits migration patterns similar to the rest of the population. We consider this requirement met.

- **Which professors were present in the same place during overlapping periods of time?**
  Multiple visualisations support the automatic detection of overlapping periods and highlight these occurrences. This can be seen in figure 15 and figure 18. We consider this requirement met.

We now move to the final level, the domain problem. Here we can conclude whether our visualisations actually help answer the questions established at the beginning of the project.

5.5.4 Level 1: Domain Problem

Our visualisations should be able to help in answering the questions established in the domain problem level. We will attempt to answer these questions by providing examples of possible patterns or discoveries that can be found from the data, without trying to explain why that pattern existed, since that would be overstepping our domain of expertise and should be up to the historians to interpret and formulate theories for.

- **Where did professors move throughout their lives?**
  The implementation is able to visualise for each professor their life course in multiple different visualisations. We can for each professor see where every event has taken place in chronological order. Examples of this can be seen in figure 9, figure 14 and figure 17. We consider this requirement met.

- **Which cities or regions acted as important academic hubs in certain periods?**
  Using the flow map or the Sankey diagram we can see which cities have the most traffic professor-wise, as seen in figure 19 and figure 20. Leiden is number one, as of course all professors in this dataset at some point stayed in Leiden. Other locations that are often visited are Franeker and Heidelberg, both of which have or had a university since old times. Other locations that have a lot of traffic are Dutch cities such as Amsterdam, Utrecht and Den Haag. We consider this requirement met.

- **Are there recurring geographic mobility patterns between universities or cities in certain time periods; in other words, how did the mobility patterns of professors change over time?**
  The flow map with era split toggle enabled gives us a good way to answer this question. An example can be seen when defining custom periods like in figure 22. As explained in that figure, interactions with France and Switzerland mostly take place before 1648 in the dataset. After 1720, most recorded interactions are contained within the Netherlands or with Germany. This is an example of how mobility changes over time in the dataset. We consider this requirement as met.

- **Can clusters or trends in migration patterns be identified?**
  This can be done by using the flow map or Sankey diagram for arbitrary time intervals. Figure 20 shows a flow map in action. Throughout the entire dataset, it can be determined that most foreign migration patterns were from France, Germany and England to Leiden. While routes between the aforementioned countries themselves were less common, they did occur. We consider this requirement met.

- **Which professors were present in the same place and could potentially have met?**
  The space time cube and location x time timeline have automatic overlap highlighters as seen in figure 15 and figure 18. In these examples, we can see where and when Scaliger and his students were recorded in the same place during overlapping periods. This shows possible co-presence, but does not prove that they actually met. We consider this requirement met.

- **Do mobility patterns differ between faculties?**
  The implementation supports faculty-based filtering, making it possible to compare mobility patterns between different faculty groups. This can be used in the flow map, Sankey diagram, and table-based selection to inspect whether certain faculties show different routes, hubs, or spatial distributions. However, this requirement is only partially met, because the usefulness of faculty-based comparison is limited by inaccuracies in the AI-derived faculty data. For example, some professors were incorrectly categorised as ‘curatoren’. We therefore consider this requirement partially met.

To conclude this visualisation evaluation section using Munzner's model, we can say that our implementation supports the requirements that we established. Note that our implementation does not contain a single visualisation that meets all the requirements, indicating that from the visualisations we have implemented none were able to answer all the questions we listed on their own. However, with all visualisations combined, we were able to meet all requirements. We can therefore conclude that the implementation is sound as a multi-view visualisation system.

6 Conclusions and Further Research

Using Munzner's Nested Model, we have designed, implemented and evaluated a multi-view visualisation implementation for visualising the life courses of professors in the digitised *Leidse hoogleraren en lectoren 1575–1815 DOUSA 80 0211-1* dataset. The central research question of this thesis was how historical spatio-temporal mobility data of Leiden University professors can be visualised, and to what extent the resulting visualisations support the analytical tasks identified through Munzner's Nested Model. Based on the implemented visualisations and the evaluation in the previous section, we can conclude that historical spatio-temporal life courses can be visualised through a combination of geographic, temporal and flow-based visualisations.

Although the implementation supports the identified requirements collectively, no single visualisation was able to address every requirement on its own. This suggests that, for complex domain problems, a single visualisation may not always be sufficient to support all analytical tasks. Instead, using a multi-view approach like the one implemented in this thesis can be a more powerful way to gain insights into the data. Each view highlights a specific perspective of the data and allows users to compare different aspects of the dataset, making the implementation adaptable for different tasks and questions [19].

The use of Munzner's model and the process of trying to meet all established requirements did not lead to one single complex visualisation that could answer all questions. Instead, it led to multiple visualisations that each support a subset of those requirements. In our implementation, for example, the dot map, timeline, location x time timeline and space time cube provide ways to visualise the life courses of individual professors or small selections of professors. These visualisations are useful for focusing on smaller-scale and individual details. In contrast, visualisations like the flow map and Sankey diagram are more suitable for larger-scale analysis, because they aggregate individual life courses into broader mobility patterns across the entire dataset.

This means that mobility through time can be visualised in many different ways, depending on the task that needs to be supported. The idea that there is one visualisation technique that is simply best for displaying spatio-temporal data is therefore misleading. Instead, the best way to visualise spatio-temporal data depends on the domain problem, the nature of the dataset, and the analytical questions users want to answer. This is in line with Munzner's model for visualisation design and validation. The characteristics of the data and the desired tasks are the driving forces behind deciding which visualisations should be implemented.

6.0.1 Limitations of the evaluation

Munzner's model provided a structured way to translate the domain problem into concrete data, tasks, encoding and implementation criteria. This allowed the implementation to be evaluated systematically in the absence of an extensive user-based evaluation. However, this does not mean

that user evaluation is unnecessary. Instead, the evaluation in this thesis should be understood as a theoretical and criteria-based evaluation of whether the implementation supports the analytical tasks that were established earlier. The design, implementation and evaluation of the visualisations in this bachelor thesis project were all carried out by the author, as such this introduces substantial bias in the evaluation process. The limited scale of the project left little time for a more objective evaluation like an user- or expert-based evaluation. Munzner also relates validation to the broader distinction between formative, summative and exploratory evaluation. Exploratory evaluation is used to better understand the problem domain and the needs of the target users. In the context of this thesis, this corresponds mostly to the formulation of the historical domain questions and requirements. Formative evaluation is used during the design process to improve the system. In this thesis, this took the form of iterative author-led testing, where visualisations were adjusted when they did not sufficiently support the established requirements. Summative evaluation is used to judge whether the final system satisfies its intended goals. The final evaluation in this thesis is primarily summative, because it assesses whether the final implementation supports the criteria established for each level of Munzner's model. However, this summative evaluation is limiting because it is a criteria-based evaluation carried out by the author. Therefore, the evaluation can show that the implementation supports the established analytical tasks, but it cannot fully prove that historians would find the visualisations useful in practice. The author is aware of the bias introduced by this manner of evaluation and proposes the following future work in section 6.1.

In addition, the conclusions drawn from this evaluation depend partly on the assumption that Munzner's four-level nested model is an appropriate way to assess this type of visualization system. The effectiveness of Munzner's model is not based on empirical proof. Instead she justifies the model conceptually by comparing it to previous visualization and evaluation frameworks and by applying it analytically to existing visualization papers.

6.1 Future Work

A visualisation is only as good as the data it represents. If the dataset contains incorrect or incomplete data, the visualisations will also reflect those issues. Since our dataset was converted and generated by GPT-5 with an average accuracy of 79.3%, this means that the dataset is not completely accurate either. In fact, the important career category for visualising career events had a lower accuracy of only 64.7%. As such, any claims made about observations or patterns in the visualisations using this dataset should keep this limitation in mind. The visualisations should therefore be seen as tools for exploration and hypothesis generation, rather than as definitive historical evidence.

In order to make more academically grounded statements using these visualisations, the accuracy of the OCR- and AI-driven pipeline should be improved further. Although using GPT-5 increased the average accuracy by 16.6% compared to GPT-3.5, the current level of accuracy is still not sufficient for making strong historical claims without further manual checking. For future work, other AI models outside the latest OpenAI models, such as Claude and Gemini, could be considered too. Alternatively, because the variance in accuracy between runs in GPT-5 is low, this indicates that GPT-5 produces relatively consistent outputs. Through prompt engineering and creating better prompts, it might therefore also be possible to reach a higher accuracy. The implementation

supports the selection based on faculty, enabling the comparison of mobility between different faculties. However due to the inaccuracies in the AI-driven pipeline, a lot of professors were categorised as 'curatoren', limiting the usefulness of this feature. Improving the AI driven pipeline or manually adjusting the data might help improve the relevance of this selection option.

In addition, the implementation already allows for the comparison of GPT-5 converted professor data with the original source document. If AI cannot yield better results, manually correcting the data with the help of this comparison could be a way to improve the dataset. This would be especially useful for important fields such as career events, faculties, dates and locations, since these fields directly affect the visualisations.

Another important topic for future work is the visualisation of genealogical data. The LUC historians were very interested in seeing how changes in familial relations affected the life courses of professors, for example how marriage or the birth of children affected their mobility. Attempts were made to incorporate these domain requirements into the design process of the implementation, but they were ultimately dropped. This was because the dataset, while containing information about spouses, children and other familial relations, was very limited for this purpose. Dates and locations of marriage and births were often omitted. Enriching the dataset with other sources could be a possible way to make this type of analysis possible. A relevant direction for this enrichment is the work by Ter Beek[21], whose previous work for the LUC project focuses on reconstructing life courses through historical record linkage by linking person references from birth, marriage and death certificates. Such reconstructed life courses could potentially be combined with the professor mobility data used in this thesis, making it possible to analyse not only academic and career mobility, but also how familial events intersected with these movements. Future contributions could therefore expand the implementation by focusing on spatio-temporal genealogical visualisations.

A further improvement would be a user-based evaluation with historians. The current evaluation is based on Munzner's model and checks whether the implementation supports the analytical tasks that were established earlier. However, it does not test how historians actually use the implementation in practice. A future evaluation with historians could reveal whether the visualisations are understandable, whether the interactions are useful, and whether the system actually supports historical research workflows. Munzner also proposes user- or expert-based evaluation as a manner of validation. Each level should be validated, and for each level there are numerous ways to validate them, some stronger than others. For example for level 1 we have validated the domain questions and requirements by asking for feedback from the Leiden Institute of History, to make sure that they are relevant for historians. For levels 2 and 3 we have self-validated the requirements by providing examples of how the visualisations address and meet the requirements. In order to remove the self-bias and to be able to make a stronger claim about the validation of the evaluations of the requirements of levels 2 and 3, user-based or expert-based evaluations can be conducted. For level 4, stronger ways to validate the implementation could be through unit-testing for correct behavior and timing or memory benchmarks.

Finally, the current implementation still has limitations regarding scalability, uncertainty and comparison between groups. Some visualisations still suffer from visual clutter when many life

courses are shown at the same time. Future work could therefore explore additional clutter-reduction techniques such as edge bundling, better aggregation, or more advanced filtering. The implementation could also include an uncertainty visualisation, for example by showing confidence scores for events derived from the OCR- and AI-driven pipeline. This implementation of visualisations might be well suited for the questions established using Munzner's model, however in other situations when new questions arrise they might not be sufficient anymore. A generative AI prompt-based visualisation implementation that creates visualisations based on prompts might provide a way to keep the implementation more flexible for future demands. Lastly, the geographic maps used in the visualisations are based on contemporary maps, it could be interesting and provide more historical context to use maps that correspond to the time period of the displayed data.

7 Disclosure of Delegation to Generative AI

The author declares the use of generative AI tools during the project. The following tasks were partially delegated to generative AI tools under full human supervision:

- Code generation
- Code optimization
- Data cleaning

The generative AI tools used were ChatGPT 5 and Claude. Responsibility for the final manuscript lies entirely with the author. The author reviewed, tested and adjusted generated code before including it into the final code.

References

[1] Zahra Abedi, Richard M. K. van Dijk, Gijs Wijnholds, and Tessa Verhoef. Enriching historical records: An ocr and ai-driven approach for database integration. *Computational Linguistics in the Netherlands Journal 14 (2025) 401-420*, 2025.

[2] Gennady Andrienko and Natalia Andrienko. Spatial generalization and aggregation of massive movement data. *IEEE Transactions on Visualization and Computer Graphics*, 17(2):205–219, 2011.

[3] Benjamin B. Bederson and Angela Boltman. Does animation help users build mental maps of spatial information? *Information Visualization*, 1(2):83–89, 2002.

[4] Julian de Boer. Interactive and explorative visualisations for researchers with unexpected questions, 2023.

[5] Jessica López Espejel, El Hassane Ettifouri, Mahaman Sanoussi Yahaya Alassan, El Mehdi Chouham, and Walid Dahhane. Gpt-3.5, gpt-4, or bard? evaluating llms reasoning ability in zero-shot setting and performance boosting through prompts. *arXiv preprint arXiv:2305.12477*, 2023.

[6] Michael Friendly. *Visions and Re-Visions of Charles Joseph Minard*. Statistical Computing and Graphics, 2002.

[7] GeoPy Contributors. *GeoPy Documentation*, 2024. Version 2.4.1. Available at: https://geopy.readthedocs.io/en/stable/.

[8] Diansheng Guo. Flow mapping and multivariate visualization of large spatial interaction data. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1041–1048, 2009.

[9] Danny Holten and Jarke J. van Wijk. Force-directed edge bundling for graph visualization. *Computer Graphics Forum*, 28(3):983–990, 2009.

[10] T. Hägerstrand. What about people in regional science? *Papers of the Regional Science Association*, 24:7–21, 1970.

[11] A. Jon Kimerling, Aileen Buckley, Phillip Muehrcke, and Juliana Muehrcke. Placing dots in dot maps. *Cartography and Geographic Information Science*, 42(3):219–229, 2015.

[12] M.-J. Kraak. The space-time cube revisited from a geovisualization perspective. *Proceedings of the 21st International Cartographic Conference*, 2003.

[13] Per Ola Kristensson et al. The trade-offs with space time cube representation of spatiotemporal patterns. *IEEE Transactions on Visualization and Computer Graphics*, 15(4):696–702, 2009.

[14] Heidi Lam, Enrico Bertini, Petra Isenberg, Catherine Plaisant, and Sheelagh Carpendale. Empirical studies in information visualization: Seven scenarios. *IEEE Transactions on Visualization and Computer Graphics*, 18(9):1520–1536, 2011.

[15] Tamara Munzner. A nested model for visualization design and validation. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):921–928, 2009.

[16] Tamara Munzner. *Visualization Analysis and Design*. CRC Press, 2014.

[17] OpenAI. Introducing gpt-5. https://openai.com/index/introducing-gpt-5/, 2025. Accessed: 2026-05-12.

[18] Catherine Plaisant. The challenge of information visualization evaluation. *Proceedings of AVI Workshop on Beyond Time and Errors*, 2004.

[19] Jonathan C. Roberts, Hayder Al-Maneea, Peter W. S. Butcher, Rachel Lew, Gareth Rees, N. Sharma, and Ana Frankenberg-Garcia. Multiple views: different meanings and collocated words. *Computer Graphics Forum*, 38(3):79–93, 2019.

[20] Mario Schmidt. The sankey diagram in energy and material flow management. *Journal of Industrial Ecology*, 12(1):82–94, 2008.

[21] Tijmen ter Beek. A novel technique for life course reconstruction using historical record linkage, 2023.

[22] Waldo Tobler. Experiments in migration mapping by computer. *The American Cartographer*, 14(2):155–163, 1987.

[23] Edward R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, 1983.

[24] UniverCity Project. Univercity, n.d. Accessed: 2026-05-08.

[25] Universiteit Leiden. Diversere beelden van de academische geschiedenis, 2021. Accessed: 2026-05-08.

[26] Universiteit Leiden. Liacs – linking university, city and diversity, n.d. Accessed: 2026-05-08.

[27] Liam van Dreumel. Visualisation of historical university data, 2023.

[28] Emily Wall, Meeshu Agnihotri, Laura Matzen, Kristin Divis, Michael Haass, Alex Endert, and John Stasko. A heuristic approach to value-driven evaluation of visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):491–500, 2019.

[29] Colin Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann, 2012.

[30] Wikipedia contributors. Josephus justus scaliger. https://nl.wikipedia.org/wiki/Josephus_Justus_Scaliger, 2026. Dutch Wikipedia, accessed 4 June 2026.

A Appendix: Code Documentation

This section will contain a description of how the plugin is integrated into the LUC codebase and motivates certain implementation decisions. The aim of this section is to provide the reader with a way to understand the plugin so that the reader can use the plugin, maintain, extend or validate the plugin. This documentation focuses on the implementation architecture rather than providing a user-manual. It explains how the LifePaths plugin uses data, how it is stored in MonetDB, how the plugin interacts with the database and how the visualisations are constructed.

The plugin lives in two main directories in the LUCD repository:

Path	Role
`Plugins/LifePaths/`	Import scripts, schema definition, source-image tools, generated SQL, and this documentation.
`Plugins/pages/LifePaths/`	Dash page integration, LifePaths visualisations (figures), callbacks and layout.

LifePaths SQL belongs to the shared LUCD database module, and is accessed through SQL queries in:

`Plugins/data/database.py`

This makes sure there is only a single MonetDB database, containing within multiple separate database schemas.

A.1 Installation

This description describes how to install and use the plugin on a linux-based environment. The code was developed, used and tested on WSL and MACOSX. The first step is to clone the repository. At the moment, the plugin is located in the LifeLines branch of the repository. From here the simplest and fastest way to install the plugin is by running the `reset_db.sh` script located in `LUCD/Plugins/Lifepaths/reset_db.sh`. This script automatically installs the requirements from `LUCD/Plugins/Lifepaths/requirements.txt` into the virtual environment. After this the script automatically setups the database and add the new schema for the Lifepaths plugin. The JSON data is also automatically imported and locations coordinates are added through geopy. The coordinate enriching process can take a while because the data contains a lot of different locations, and each API request for locations has a delay in between in order to prevent overloading the API. The source image files are not stored in the repository and have to be acquired separately through Abedi's GitHub repository. The entire directory `bantjes_data/png_processed` serves as the input for the image importer. Its nested and interior structure should be kept. The `reset_db.sh` script should then be modified to point to the filepath of where the images are stored on the users computer. To this end `IMAGES_DIR` near the top of the file should be changed. While the script is running it is possible that it will ask for a password. This is for MonetDB, for which the default password is `monetdb`. After the `reset_db.sh` script has finished, the plugin can now be run. For Linux-based environments two scripts are provided to quickly do this: `LUCD/start_dashboard.sh` and `LUCD/start_plugins.sh`. The former will start the dashboard variant while the latter starts the plugin testbench.

A.2 Architectural Overview

LifePaths is structured as four cooperating layers.

The intended data flow is:

```
Professor JSON files / scanned source images
  -> Plugins/LifePaths/reset_db.sh
  -> schema.sql + JSONImporter.py + ImageImporter.py
  -> MonetDB database lucd, schema lifepaths
  -> Plugins/data/database.py
  -> lifepaths/data.py
  -> callbacks.py
  -> figures/*.py
  -> Dash UI
```

See figure 26 for an overview of the LifePaths architecture.

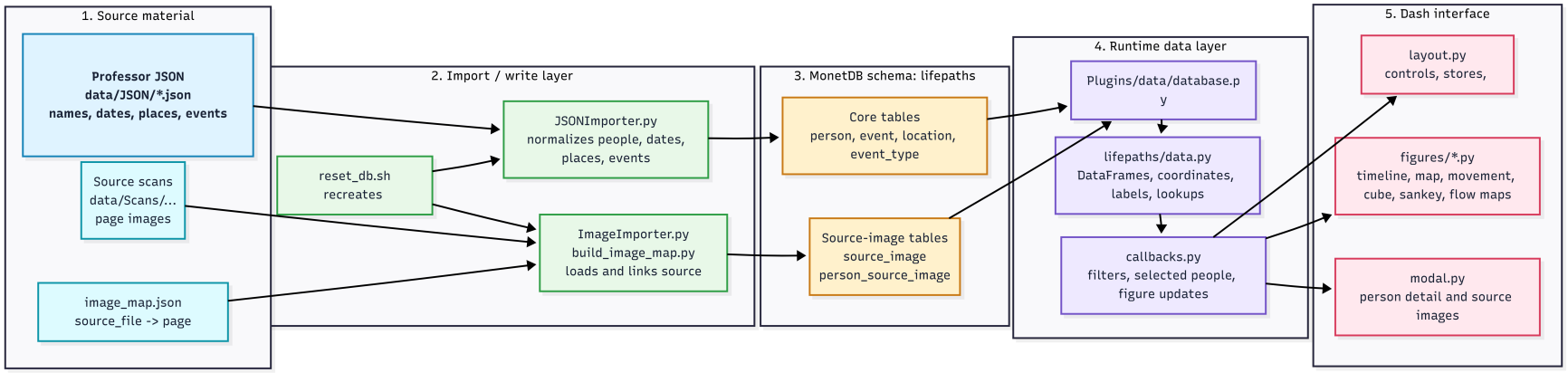


Figure 26: First professor JSON files and scanned source images are imported into the MonetDB database. The `reset_db.sh` script creates the database using `schema.sql` for the tables and relational structures, and invokes `JSONImporter.py` to import the JSON files and `ImageImporter.py` to import the source image files as binary large objects. `database.py` contains the queries used to extract the required data from the database.

A.3 Shared MonetDB Database

The LifePaths tables are isolated by schema rather than by database. The main LUCD database and LifePaths database live in the same MonetDB database while remaining independent. They do not need to join against eachother and the `reset_db.sh` script only drops and recreates the LifePaths schema. The benefits is that there is need for only one MonetDB database connection, simplifying the process of connecting to the database. For more information about the database structure please refer to section 4.2 and figure 7.

A.4 JSON Importer

`JSONImporter.py` is responsible for converting JSON files into SQL statements. It expects a directory with JSON files for input. This directory should not be nested. The default path for this directory is set to the relative path: `data/JSON`. The input format of the JSON should adhere to the format established by Abedi. The JSON filename is also stored in the person table of the database in `person.source_file`, this is later important to link the source images in the `image_map.json` later. Most fields in the JSON are optional during the import step. Only persons who miss both a first- and last name are skipped. The import script also geocodes location names using geopy and stores the latitudes and longitudes in the database. Geocoding can be disabled by using the `--no-geocode` parameter. In short, `JSONImporter.py` does the following:

- Reads one JSON file per professor.
- Normalizes and cleans the input (scalar text values, inconsistent date formats)
- Geocodes locations when enabled.
- Generates and writes SQL insert statements for every professor JSON file to `import.sql`.
- Optionally executes the generated SQL through `mclient`.

A.5 Image Importer

`ImageImporter.py` imports scanned source images into MonetDB as BLOBs in the table `source_image`. It expects a nested directory input where images are divided by volume. The expected input layout is:

```
Plugins/LifePaths/
  data/
    Scans/
      vol1/
        person/
          page_11.png
          page_12.png
      vol2/
        person/
          page_11.png
```

This structure makes the source image paths predictable during import. However, the database does not use these paths as the main identifier of an image. Each imported image receives an automatically generated `image_id`. The relative path is still stored in `source_image.image_path`, but only as metadata. This makes it possible to trace a stored BLOB back to the original scan file and also helps with debugging and MIME-type detection.

For example, the file:

```
data/Scans/vol1/person/page_11.png
```

is stored as something like:

```
source_image.image_id   = 42
source_image.image_path = vol1/person/page_11.png
```

The Dash modal does not request the image by path. It requests the image by its database identifier:

```
/lifepaths-images/42
```

The importer supports common image extensions such as `.png`, `.jpg`, `.jpeg`, `.gif` and `.webp`. Before importing new images, the importer removes the existing rows from both `person_source_image` and `source_image`. The image import should therefore be understood as a full refresh of the available source images and their links, not as an incremental update.

The images themselves are stored in `source_image`. The relationship between persons and images is stored in a separate relational table, `person_source_image`. This table links `person.person_id` to `source_image.image_id`. The file `image_map.json` is still used, but only as an import-time mapping file.

The runtime flow is:

```
Person modal opens
  -> modal.py asks data.py for images linked to person_id
  -> data.py asks database.py for linked image_id values
  -> database.py queries person_source_image
  -> modal.py creates image URLs using image_id
  -> browser requests /lifepaths-images/42
  -> callbacks.py receives the request
  -> data.py asks database.py for the image bytes
  -> database.py queries source_image by image_id
  -> Flask returns the raw image data
```

The images are therefore not embedded directly in the Dash layout. They are requested by the browser when needed. The choice to keep `image_map.json` was made because the manual linking of source pages to persons was done by hand. Editing a JSON file was quicker and more practical during development. The final runtime design still imports this manually curated mapping into a relational table, so the application uses explicit foreign-key relations while the editing process remains simple. The Dash application does not manually decode the BLOB. It retrieves the binary data from MonetDB and returns it as an HTTP response with the correct MIME type. The browser then decodes the returned bytes and displays the image.

A.6 Runtime Database Layer

Runtime database access for LifePaths is implemented in the shared LUCD database module:

`Plugins/data/database.py`

The LifePaths Dash page does not write SQL queries directly in the figure or callback files. Instead, it calls helper functions from `database.py`. The most important LifePaths database functions are responsible for loading persons, events, available locations, year bounds and source images.

Function	Purpose
`get_lifepaths_people()`	Loads person metadata used by search controls, tables and detail modals.
`get_lifepaths_events(...)`	Loads event data, optionally filtered by year, person, event type or faculty.
`get_lifepaths_year_bounds()`	Returns the minimum and maximum event years used by time sliders.
`get_lifepaths_locations()`	Loads locations with coordinates for map-based visualisations.
`get_lifepaths_source_image(...)`	Returns the binary image data for one scanned source page.

The central event query joins the `event`, `person`, `location` and `event_type` tables. This is the main query used by the visualisations, because most views need to know who an event belongs to, what type of event it is, when it happened and where it happened.

A.7 App Data Layer

The app data layer is implemented in:

`Plugins/pages/LifePaths/lifepaths/data.py`

This file loads data from `database.py` and prepares it for use by the Dash interface. The database query results are converted into pandas DataFrames. These DataFrames are then reused by callbacks and figure functions.

This separation is useful because the visualisation code should not need to know how SQL queries are written. The figure modules can work with prepared DataFrames and focus on filtering, grouping and rendering.

A.8 Dash Interface Layer

The Dash interface layer consists of the layout, callbacks and figure modules. The layout defines the visible controls and graph containers. The callbacks respond to user interaction and decide which data should be passed to each figure. The figure modules then construct the actual Plotly or Dash components.

The main files are:

```
layout.py
callbacks.py
figures/timeline.py
figures/map.py
figures/cube.py
figures/sankey.py
figures/movement.py
figures/modal.py
figures/expflowmap.py
```

The callbacks form the connection between the user interface and the visualisations. They read the state of filters such as selected persons, year ranges, event types and faculties. Based on this state, they request filtered data and pass it to the relevant figure function.

A.9 Visualisation Modules

The LifePaths visualisations are implemented as separate modules in the `figures` directory. Each module receives prepared event data and returns a figure or Dash component. This keeps the visualisation logic separate from the database logic and from the callback logic. Several visualisations use shared helper functions from `figures/shared.py`. These helpers apply common filters, build colour mappings and handle empty figures.

A.10 Notes

A few implementation details are especially important for future maintenance.

- The LifePaths schema should be reset through `reset_db.sh`, because this script applies the schema, imports JSON data and imports source images in the expected order.
- Runtime SQL should remain in `Plugins/data/database.py`. Figure files and callbacks should not contain separate SQL queries.
- The JSON importer expects one person per JSON file and does not recursively search nested folders.
- The image importer recursively imports images, but it does not decide which image belongs to which person.
- Spatial visualisations require coordinates. Events without coordinates can still exist in the database, but they are usually excluded from map-based views.