Curio: A Dataflow-Based Framework for Collaborative Urban Visual Analytics

Gustavo Moreira, Maryam Hosseini, Carolina Veiga, Lucas Alexandre, Nicola Colaninno, Daniel de Oliveira, Nivan Ferreira, Marcos Lage, and Fabio Miranda

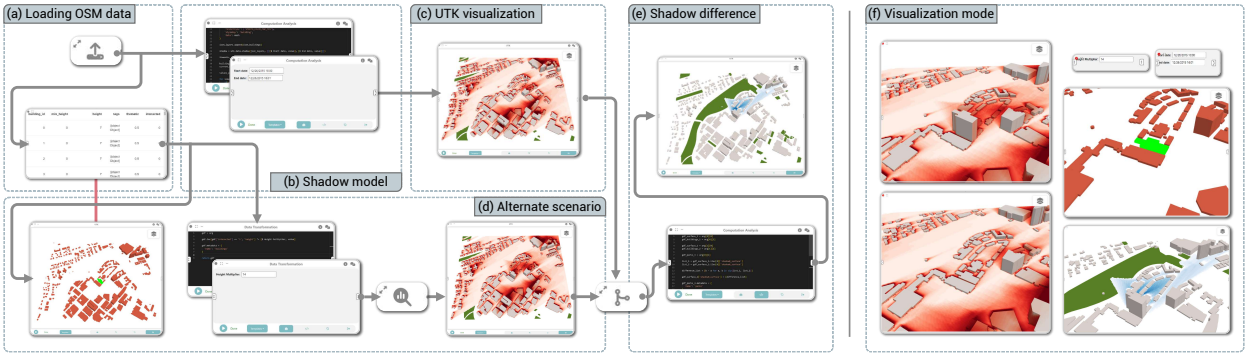

Fig. 1: Using Curio to build a dataflow to analyze the shadow impact of a proposed building in Boston. (a) The user first adds a node to download OpenStreetMap data for the region of interest, (b) then accumulates shadow data for a period of the day, and (c) visualizes the results in a 3D map. (d) An alternate scenario is created by leveraging Curio's interaction node, allowing the user to select buildings in the 3D map. (e) The difference in shadow between the current and alternate states is then visualized. (b,d) With Curio, it is also possible to create annotations in the code to facilitate the creation of GUI elements. (f) The final dataflow can be exported as a standalone visualization that includes 3D maps and GUI elements.

Abstract—Over the past decade, several urban visual analytics systems and tools have been proposed to tackle a host of challenges faced by cities, in areas as diverse as transportation, weather, and real estate. Many of these tools have been designed through collaborations with urban experts, aiming to distill intricate urban analysis workflows into interactive visualizations and interfaces. However, the design, implementation, and practical use of these tools still rely on siloed approaches, resulting in bespoke systems that are difficult to reproduce and extend. At the design level, these tools undervalue rich data workflows from urban experts, typically treating them only as data providers and evaluators. At the implementation level, they lack interoperability with other technical frameworks. At the practical use level, they tend to be narrowly focused on specific fields, inadvertently creating barriers to cross-domain collaboration. To address these gaps, we present Curio, a framework for collaborative urban visual analytics. Curio uses a dataflow model with multiple abstraction levels (code, grammar, GUI elements) to facilitate collaboration across the design and implementation of visual analytics components. The framework allows experts to intertwine data preprocessing, management, and visualization stages while tracking the provenance of code and visualizations. In collaboration with urban experts, we evaluate Curio through a diverse set of usage scenarios targeting urban accessibility, urban microclimate, and sunlight access. These scenarios use different types of data and domain methodologies to illustrate Curio's flexibility in tackling pressing societal challenges. Curio is available at urbantk.org/curio.

Index Terms—Urban analytics, urban data, spatial data, dataflow, provenance, visualization framework, visualization system.

1 Introduction

The growing availability of urban data in the past decade has led urban experts from a diverse range of disciplines to increasingly adopt data-driven methodologies for scientific inquiry and policy formation [32, 35]. Such data is derived from a multitude of sources, including government databases [5], sensor networks [4], street-level images [8], and transportation systems [74]. By capturing various aspects of city life, infrastructure, and the environment, data analysis can offer valuable insights to address pressing challenges related to housing [9], transportation [19], accessibility [59], climate [38], disasters management [54], and public health [39]. To effectively analyze urban data, experts must tap into a variety of resources and develop analytics workflows capable of handling the diversity and complexity inherent to urban datasets. One popular approach is the design of urban visual analytics applications. In collaboration with urban experts, visualization researchers and practitioners construct applications that untangle intricate urban analytics workflows into intuitive, interactive visual representations. This enables experts to explore, understand, and gain insights from data. Alternatively, urban experts have extensively used computational notebooks (e.g., Jupyter, Observable) to implement their analytics workflows, leveraging a rich ecosystem of urban science libraries [70]. Both approaches have their strengths and weaknesses.

On the one hand, urban visual analytics applications tackle research and engineering challenges to enable the interactive visual analysis of

- *Gustavo Moreira and Fabio Miranda are with the University of Illinois Chicago. E-mail: {gmorei3,fabiom}@uic.edu.*
- *Maryam Hosseini is with the University of California, Berkeley, and the Massachusetts Institute of Technology. E-mail: maryamh@berkeley.edu.*
- *Carolina Veiga is with the University of Illinois Urbana-Champaign. E-mail: carolvfs@illinois.edu.*
- *Lucas Alexandre, Daniel de Oliveira, and Marcos Lage are with the Universidade Federal Fluminense. E-mail: {danielcmo,mlage}@ic.uff.br.*
- *Nicola Colaninno is with the Politecnico di Milano. E-mail: nicola.colaninno@polimi.it.*
- *Nivan Ferreira is with the Universidade Federal de Pernambuco. E-mail: nivan@cin.ufpe.br.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

data. Designing and constructing these applications, however, is a laborious and time-consuming process, requiring visualization researchers and practitioners to account for the inherent complexity of urban data and the diverse needs of experts. Such a complex development process often leads to the creation of one-off monolithic pieces of software that are difficult to extend and adapt to handle new data or use cases. On the other hand, computational notebooks support literate computing [34] by intertwining code, comments, and visualizations. However, they offer limited visualization capabilities, being largely constrained to static and simple views that need to be manually specified [6, 66]. Furthermore, notebooks have been increasingly criticized for encouraging bad programming habits, leading to unexpected execution order, fragmented code, poor versioning, and lack of modularization [22, 51, 56, 69].

To bridge these gaps and facilitate the creation of urban visual analytics workflows, we propose Curio, the Collaborative Urban Insights Observatory. Curio is a web-based visualization framework that allows different users, such as visualization researchers, practitioners, and urban experts, to collaborate in the design and implementation of urban visual analytics workflows. Such collaboration takes place in a shared interface that uses an intuitive dataflow diagram where users externalize their workflow design decisions through a series of urban-specific computing modules. Unlike previous dataflow works that were designed for scientific visualization [7, 23, 55] or tabular data [71], Curio was specifically designed considering common urban data analytics tasks. Its modules enable users to execute a comprehensive series of tasks seamlessly. These tasks include loading 2D and 3D spatial data that describe the built environment (e.g., street networks, parks, buildings), accessing open-data APIs, cleaning, transforming, and filtering data, as well as creating plot- and map-based interactive visualizations that can be connected through linking and brushing.

Curio contains a series of pre-defined, reusable modules and also allows for the creation of user-defined ones. These modules can be specified at three abstraction levels: programming using Python, specifying visualizations using Vega-Lite [60] or our previously introduced urban-specific grammar [49], or through GUI elements. Following such an approach allows users with different expertise to collaborate and quickly iterate over different design choices, offering functionalities that meet the expectations of both experts and visualization researchers. Importantly, rather than a compartmentalized development approach that confines visualization researchers and urban experts within the boundaries of their respective domains, Curio introduces a common canvas in which they can articulate their intent while maintaining awareness of the contributions and artifacts generated by their collaborators.

Moreover, to ensure the tracking of dataflow modules, Curio is a provenance-aware framework that records the evolution of artifacts throughout the collaborative process. It provides a history of modifications, iterations, and contributions made by each user. In doing so, Curio facilitates the exploration of alternative development paths, allowing users to easily revisit and revert changes if necessary. Curio also supports reproducibility through the sharing of complete workflows or specific modules. Users can save and export self-contained descriptions of the workflow and share them with collaborators or other audiences. Curio is available at urbantk.org/curio. Our contributions can be summarized as follows:

- We present a provenance-aware dataflow that supports urban visual analytics through the combination of modules.
- We present Curio, a web-based framework that supports the asynchronous collaborative creation of urban visual analytics dataflows.
- We present a series of usage scenarios created in collaboration with urban experts, highlighting Curio's flexibility in addressing several domain problems.

2 Background

Urban analytics workflows usually require multidisciplinary teams with skills in all stages of the data lifecycle [14, 35, 72]. For example, urban experts (e.g., architects, urban planners, climate scientists) collect data from various sources or generate new data using simulations. Second, they clean and transform the data, using general [2] or urban-specific [15] data management solutions. Third, they use analytical and modeling approaches [70] to extract features, identify trends, analyze correlations, etc. Finally, experts create visualizations to gain insights into the data and better understand urban problems or phenomena. The complex nature of urban analytics presents key challenges that increase the analytical bottleneck for urban experts. For instance, the heterogeneity and volume of urban data require preprocessing techniques to ensure consistency and usability across the processing pipeline. Urban datasets exhibit considerable variation in format and scale, ranging from street-level imagery to 3D building geometries. Moreover, the dynamic nature of urban environments means that data is constantly changing, requiring analyses to be updated so that insights do not become outdated before they can be leveraged. Lastly, urban environments introduce multifaceted relationships among data elements, necessitating multiple iterations and refinements of the data pipeline to handle spatial and temporal dependencies.

To tackle some of these challenges, for more than a decade [14, 16, 73], researchers across disciplines have been creating end-to-end applications that encompass many of the aforementioned steps and seek to tightly integrate interactive visualizations and analytics capabilities. The design of these applications easily leads to rather large codebases. For example, our previous applications [18, 19, 46, 47, 49, 57] have approximately 100,000 lines of low-level code that took 6 to 18 months and multiple researchers with different computer science expertise to design and develop, from initial conversations with experts to operational versions. The iterative process of design and development, often including multiple rounds of feedback and revisions with experts from multiple disciplines, contributes to the expansion of the codebase.

With such a lengthy and complex process, development is susceptible to various pitfalls. These may include neglecting best development practices, failing to maintain records of intermediate artifacts, and opting to reinvent the wheel rather than adapting existing codebases (including ones created by urban experts). Such a scenario stiffens visualization research as well as the creation of practical tools for urban experts. From a visualization standpoint, given the complexity of these systems, iterating over the design space of urban visualizations is difficult, as the implications of changes need to be carefully weighed against their impact on the existing codebase. The result is a less fluid design process, rarely considering alternative design choices once a commitment has been made. From a practical standpoint, such laborious design and development lead to the creation of bespoke and monolithic pieces of software. And these applications are rarely made public. In fact, in a recent survey of open urban planning tools [70], only one was originally published by the visualization community [46], despite the abundance of contributions by the community tackling planning problems.

This scenario is not unique to urban visual analytics [68], but the complexities of urban data and tasks make the situation more pronounced in urban contexts. A framework that tackles the aforementioned challenges answers several recent calls from the visualization and urban research communities. First, it would support the collaborative creation of urban and geospatial analysis pipelines [14, 66, 75]. Second, it would lower the barriers to the creation of reproducible artifacts that can be easily shared and made public [75]. Third, it would facilitate the creation of transparent and modular visual analytics modules [31, 67]. Finally, it would support the creation of provenance-aware pipelines that can be used to recover past states [3, 11, 58, 75], including visualizations discarded in the design process [1].

3 Related Work

In this section, we review research related to various aspects of this work. In particular, we review challenges in urban visual analytics, construction tools for urban visual analytics, and collaborative visual analytics systems.

3.1 Urban visual analytics

A common usage scenario for urban visual analytics tools is to help untangle the complex dynamics within a city by analyzing its present conditions through visual summaries or mining algorithms [13, 19, 37]. Other applications involve analytical tasks that consider the 3D nature

of the city and, for this reason, include visualizations of 3D environments [48]. This makes the design and implementation of these tools considerably more difficult due to more complex data management, rendering, and integration between physical and thematic data [49, 50]. Urban visual analytics tools also provide critical support for assessing the potential impacts of various changes, ranging from policy to infrastructure and urban development, i.e., *scenario planning*. Such analyses can help evaluate and prepare for different possible future conditions. To do so, these tools employ modeling techniques to simulate the effect of changes in the city and use visualization to quantitatively and qualitatively evaluate these new scenarios [18, 42, 45].

The examples mentioned above showcase the diversity and complexity of urban visual analytics systems. These tools are sophisticated pieces of software designed through collaborations with urban experts. Their development encompasses intricate data processing and integration with modeling and simulation techniques, employing advanced visualization designs and interactions. Curio builds on previous efforts, including our own [19, 45–47, 49], to facilitate the creation of urban visual analytics applications through urban-specific dataflows. These dataflows can be translated into standalone applications that can be shared and reproduced by urban experts.

3.2 Construction tools for urban visual analytics

Urban visual analytics systems rely on several toolkits, frameworks, and authoring tools to implement their visualization requirements [17]. Mei et al. [44] presented a design space and named these tools as construction tools. In our previous work [49], we proposed a grammar-based construction tool for urban visual analytics. This tool relies on an urban-specific visualization grammar, requiring users to write JSON specifications.

Computational notebooks offer a highly flexible approach to data analysis, making them a popular environment for urban studies and analyses [64]. However, notebooks still carry a number of shortcomings (e.g., unexpected execution order, fragmented code, and poor versioning and modularization [22, 51, 56, 69]).

Dataflow visualization frameworks provide a powerful solution to the problems mentioned above, by providing intuitive interfaces to specify not only visualizations but also an entire analytical pipeline. For example, Bavoil et al. [7] proposed VisTrails, which enabled the specification of visualizations via a dataflow diagram based on VTK [24]. A core component of VisTrails was the use of provenance graphs to record the process of constructing the dataflow. However, the closeness between VTK classes made the dataflow specifications in Vistrails low-level and hard to create. Also, the interaction between the specified views is limited. Other examples of visualization dataflow frameworks include ExPlates [30] and VisFlow [71]. However, previous frameworks are limited in the data types and visualizations they support, preventing them from being used for urban analytics workflows that involve modeling, simulation, spatial data, and 2D and 3D map-based visualizations. Curio builds on these experiences and uses a dataflow approach to enable the easier creation and modification of urban analytics pipelines, while supporting the provenance of the users' creations. Curio also supports the integration of custom modeling and visualization capabilities.

3.3 Collaborative visual analytics systems

Collaboration is essential in projects that leverage data, including the ones targeting urban problems. In these scenarios, experts collaborate in the process of exploring data to extract insights. In urban analytics, this generally involves processing, modeling, and visualizing data, as well as predictive analysis for scenario planning. Therefore, there is a need for effective tools that support this complex collaborative process. Wang et al. [64] highlighted that collaboration is desirable, but it is not efficiently facilitated by widely used computational notebooks, such as Jupyter notebooks, for several reasons. For instance, it is difficult to maintain intermediate products and an understanding of the exploration process. Another observation is that maintaining the provenance of the development can also help in the collaboration process.

As discussed by Isenberg et al. [29], providing support for collaboration is neither a trivial task nor usually considered when designing visual analytics systems. This is especially true in urban visual analytics, with few systems reporting collaboration as one of their design goals, which has been suggested as an interesting research direction [14, 36]. Some works are exceptions to this. For example, Lukasczyk et al. [40] proposed a web-based map tool that enables synchronous and asynchronous collaboration between multiple users. Sun et al. [62] proposed a 3D collaborative environment for urban design. All these works, however, have different objectives compared to Curio, as they focus on specific urban analytics workflows. In other words, they do not allow for the construction of analytics pipelines, but rather provide a fixed interface with pre-defined analytics scenarios. Finally, ArcGIS, a general GIS tool, supports collaboration via shared workspaces and synchronization mechanisms. Curio supports collaboration through various features. For example, Curio's modules have levels of abstraction to support users with different programming backgrounds. Also, users can add and share comments on the dataflow diagram to facilitate knowledge sharing and the creation of data narratives. Finally, through provenance, users can explore the different versions of the dataflow produced during the collaboration.

4 Curio's Design Goals

Curio's overarching goal is to facilitate the flexible creation of urban visual analytics dataflows. Curio's design goals were motivated by our previous contributions (e.g., [19, 45, 47, 49]), as well as meetings with three urban experts (co-authors of this paper). In these meetings, we were able to better understand their workflows and how visualization tools and computational notebooks were used by them. This process also shed light on the practical applications of one of our recently proposed toolkits, the Urban Toolkit (UTK) [49], helping identify key pain points and directly informing our design goals. Our goals are also influenced by recent works highlighting the need to support collaborative workflows [14, 66, 75], modular visual analytics components [31, 67], and provenance [3, 11, 58, 75]. In the following sections, we refer back to these goals when describing Curio.

DG1. Collaborative visual analytics. Urban visual analytics problems are complex and inherently multidisciplinary. For this reason, they are addressed by analysts from different domains (e.g., architecture, engineering, visualization) working in teams. To enable this, Curio should support collaboration in the creation of urban visual analytics dataflows. The framework should offer features that allow users to contribute to shared dataflows, increasing user engagement in the design process. Additionally, Curio should facilitate an iterative design methodology, encouraging continuous feedback and adjustments.

DG2. Flexibility and modular design. As mentioned, urban experts come from diverse disciplinary backgrounds, each with unique methodologies, data requirements, and analytical needs. To avoid reinventing the wheel with each new urban visual analytics application, Curio should support a modular component design. It should enable users to easily add, remove, or modify modules to meet their specific needs and preferences. This flexibility extends to the preservation of user-created modules, ensuring their reusability in future projects by allowing users to save and load desired functionalities. Moreover, recognizing that urban analytics involves experts from diverse fields, Curio should support shared component templates. The framework should also ensure compatibility with a broad spectrum of data formats and standards.

DG3. Reproducibility and provenance of modules. In a collaborative environment, provenance can enhance efficiency and transparency among collaborators by facilitating users' understanding of the evolution of a dataflow and identifying the sources of data and transformations. Moreover, provenance can also support the collection of design alternatives created in the collaborative process, including ideas that were ultimately set aside in favor of other solutions. Curio should then support the provenance of modules created when users interact with the framework. It should also support the visualization of provenance data and enable users to easily revert to specific versions of modules, facilitating the exploration of their evolution over time. The framework should also facilitate the reproducibility of modules and dataflows.

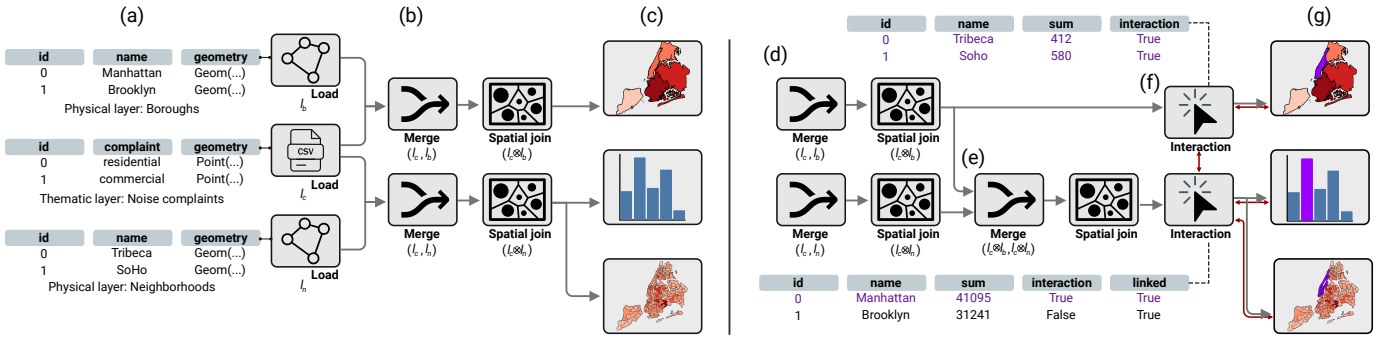

Fig. 2: Illustration of the key concepts of the Curio dataflow model. (a) A thematic layer l_c and physical layers l_b and l_n are loaded. (b) Spatial joins between l_c and l_b ($l_c \otimes l_b$), and l_c and l_n ($l_c \otimes l_n$) are computed. (c) The results of the joins are visualized. (d) To support linked views, the Curio dataflow makes use of interaction nodes. (e) $l_c \otimes l_b$ and $l_c \otimes l_n$ are further joined, creating a link between them. (f) Interaction nodes augment the previously joined layers, propagating information when a user selects or brushes elements in a visualization (shown in (g)).

5 A Dataflow for Urban Visual Analytics

This section introduces the main aspects of Curio's dataflow to support the creation of urban-specific workflows and applications, with the aforementioned design principles in mind. We first formally introduce Curio's dataflow model (Section 5.1), followed by the supported types of urban data (Section 5.2). We then present the dataflow modules (Section 5.3) and detail how Curio supports data transformation and interactions (Section 5.4).

5.1 The Curio dataflow model

With Curio, an urban expert can design and implement workflows by organizing a sequence of steps in a diagram, with steps connected through data dependencies. These steps compose a dataflow that may include several operations, such as data generation, transformation, management, analysis, and visualization. In this section, we introduce the main elements of Curio's dataflow model: its smallest unit of interest (data layer), its two classes of computing modules (data node and interaction node), and connections (data dependency and interaction dependency). Our dataflow model formalism extends the one presented by Ikeda et al. [28].

In Curio, the smallest unit of interest is a data layer l. A data layer is characterized by k attributes (columns) and m records (rows) and can be represented as $l = \{r_1, ..., r_m\}$, where each record r_i is a tuple $r_i = (a_{i,1}, ..., a_{i,k})$, where a are data attributes. We denote a set of layers as L. These data layers can represent a variety of urban data, such as sensor and raster data, images, and building geometries. The different types of urban data supported by Curio are discussed in Section 5.2.

A data node is a module that will represent an operation on top of a data layer, such as data cleaning, transformation, and visualization. These can include pre-defined algorithms or procedures defined by the user. A data node $n = (L_i, L_o)$ is then defined as a tuple consisting of input data layers L_i and output data layers L_o. Each data node takes as an input and produces zero or more data layers, depending on the type of operation. Curio supports different types of operations so that the framework can be aligned with urban experts' workflows. These steps are detailed in Section 5.3.

A data dependency φ connects the flow of data between two data nodes. It is defined as $\varphi = (L_{flow}, n_{source}, n_{target})$, with data layers L_{flow} and two data nodes: n_{source} (source node) and n_{target} (target node). These must satisfy the condition $L_{flow} = n_{source}.L_o \cap n_{target}.L_i$. The directed dependencies of a dataflow diagram express the connection between two nodes, with the target node having access to the data processed in the source node.

To support user interactions and linked views, Curio's dataflow model introduces an interaction node and an interaction dependency. An interaction node n^i is a special node that will receive data layer l as an input and will output a data layer l' that is equal to l but with an extra attribute $a_{i,k+1}$ that specifies whether the set of records R were selected or not by the user. In Figure 2(d), both tables have an *interaction* column that specifies whether that record was selected by the user. Furthermore, an interaction dependency ε defines an interaction flow between (1) a data node and an interaction node or (2) two interaction nodes. For example, if a dataflow contains two linked plots (defined by two data nodes), each will have data and interaction dependencies to an interaction node; the data dependency will be responsible for passing along data, while the interaction dependency will be responsible for updating l with respect to the records R selected by the user. It is defined as $\varepsilon = (R, n^i, n)$. An interaction dependency can only exist between two nodes if there is a data dependency between them. Curio's interactions are detailed in Section 5.4.

Consequently, a Curio dataflow is a composition of multiple nodes, connected by data and interaction dependencies. Such dataflow can be represented as a tuple $F = (N, N^i, L, \phi)$, comprising data nodes N, interaction nodes N^i, data layers L, and data dependencies ϕ. Interaction dependencies since they only change visualization properties, are not part of F. This ensures that F is a directed acyclic graph with respect to the movement of data along the dataflow. By design, this is a rather flexible dataflow model. As outlined in Section 5.3, it supports the creation of urban-specific nodes to handle different parts of a workflow. Following such a model, Curio supports the collection of historical information regarding dataflow executions for further analysis, i.e., provenance data [25]. Our provenance approach is detailed in Section 6.4.

5.2 Urban data layers

Curio supports a variety of urban data layers, covering a wide range of applications and use cases across different urban domains. Each type of data node, discussed in Section 5.3, is restricted to certain input and output layers. In our previous work [18, 49], we divided these layers into *thematic* and *physical* data layers. Thematic data layers correspond to values over 2D or 3D space, such as sociodemographic data over 2D regions or sunlight access values over 3D building surfaces. Physical data layers correspond to the built or natural environment, such as buildings, road networks, or regions of interest in a city, such as neighborhoods and parks. In Curio, we extend this to also include street-level imagery data, an increasingly popular source of data for urban analyses [8] – which we call the *image layer*. Next, we detail these data layers, mentioning key urban datasets as examples.

Thematic layers. Two types of thematic layers are supported: point and grid layers. These can hold univariate or multivariate data. A point layer is used to represent events or features associated with specific locations. Examples include taxi pickups and drop-off positions, environmental observations (air quality monitoring stations, temperature sensors), or positions of noise complaints and crime incidents. Each data point is a data layer record. A grid layer (or raster layer) is used to represent data aggregated over a fine-grained grid that covers a region of the city. The grid cells hold data values, such as rainfall volumes or simulated temperatures. Each grid cell is a data layer record.

Physical layers. Three types of physical layers are supported: 2D and 3D mesh layers, and network layer. A 2D mesh layer is used to represent geographical boundaries and surfaces in 2D, such as lots,

neighborhood areas, water bodies, or any zone that requires delineation over a flat plane. Here, each geographical area (e.g., neighborhood) is a data layer record. A 3D mesh layer extends the capabilities of the 2D mesh by adding a third dimension to represent the spatial properties of features, such as buildings. In this case, each building is a data layer record. A network layer is designed to represent linear features that form networks, such as roads and sidewalks. Here, each network segment (e.g., a street between intersections) is a data layer record.

Image layer. An image layer is used to represent street-level image data, such as data from Google Street View or Mapillary. Each entry in this dataset contains, at the very least, a position and an associated image. For images captured by car-mounted sensors (such as Google Street View), this layer can also contain information like the time of capture and whether the camera was facing the left or right side of the street. In an image layer, a tuple with position and image constitutes a data layer record.

5.3 Dataflow nodes

Curio's dataflow nodes support common data operations performed in urban workflows. While the framework has pre-defined operations for each one of these nodes, it is reasonable to assume that, due to the diversity of urban workflows, they are not comprehensive. As such, we present the nodes as *templates* that have defined inputs and outputs, but that can be modified, stored for later use, and shared with other users. We categorize these nodes into broad categories: data loading, data wrangling & transformation, analysis & modeling, visualization, and interaction. When describing them, we indicate whether they can have zero, one, or more sources and targets in a data dependency.

Data loading nodes. Curio provides functionalities to load data layers from different sources. These nodes are not the target in any data dependency, but they can be the source in one or more. An OSM node considers an address or bounding box covering a region of interest and uses that to produce physical layers from OpenStreetMap, including 2D mesh layers, 3D mesh layers, and network layers. A raster node and NetCDF node load raster data (e.g., from satellite imagery) and NetCDF files (e.g., from WRF simulations). An open data node uses the Socrata Open Data API to access common datasets made available by cities. A generic file node loads a data file and produces an output with the corresponding layer.

Data wrangling & transformation nodes. These nodes are responsible for cleaning and transforming data layers. They can serve as both the source and target in one or more data dependencies. For data cleaning, Curio provides two basic nodes: the remove duplicates and remove missing values nodes. For data transformation, Curio provides the ability to perform statistical normalization, group-by's, and spatial joins.

Analysis & modeling nodes. These nodes are available for analytics and simulation tasks. They can be the source and target in one or more data dependencies. For modeling, Curio provides a node to compute sunlight access. This node leverages our previous work [45] and takes as input one or more physical layers (e.g., buildings, parks) and outputs a 3D mesh layer with corresponding sunlight access values at each vertex. Curio also provides a topology node, offering connections with the Topology Toolkit [63], facilitating the extraction of features shown to be useful in urban visual analytics [46].

Visualization nodes. Curio has four types of visualization nodes. They can serve as both the source and target in one or more data dependencies. For a quick overview of the data, we provide a table node that takes as input one or more thematic layers. For 2D visualizations, we provide a Vega-Lite node that also takes as input one or more thematic layers. In this node, the user can directly edit the Vega-Lite specification. For 3D visualizations, a UTK node takes as input one or more thematic layers and one or more physical layers, using our previously proposed UTK [49]. Once a data dependency is connected to a UTK node, it automatically creates a basic specification, taking into account the spatial extension of the data given as an input. For image visualizations, an image node takes as input one or more image layers. This node will display input images in the form of a mosaic gallery, similar to our previous work [47].

Interaction node. This type of node will orchestrate interactions between views. It takes as input one or more data layers, and outputs one or more data layers. Next, this node is discussed in detail.

5.4 Interactions

Interactions between visualizations are fundamental for enabling deep exploration of intricate collections of urban data. For this reason, Curio's dataflow includes an interaction node that propagates metadata produced when a user selects or brushes elements in a visualization. This propagated information can be used to update the visualizations connected to the node. Figure 2 illustrates an interaction example across a bar chart and multiresolution spatial visualizations (one visualization at a neighborhood level and another one at a borough level). The user loads a point layer l_c with noise complaints. The layer contains two attributes: position and complaint type. The user also loads two 2D mesh layers representing areas, l_n and l_b. These layers have two attributes each: area name and 2D mesh of the area. First, two data transformation nodes perform a spatial join between l_c and l_n, and l_c and l_b, using the location of the noise complaints and areas from the 2D mesh layers. Then, the data is visualized in three separate views.

In order to support linked views, an interaction node is added. This interaction node adds an extra *interaction attribute* that contains a Boolean value depending on whether a particular data record is selected or not. The previous visualization nodes are then connected to the interaction node through three new interaction dependencies. It is important to note that this approach selects data records, irrespective of the type of data layer. Given a user selection, the different visualization nodes will be responsible for appropriately selecting the data rows and propagating the information to the interaction node. One can also notice an interaction dependency between the two interaction nodes. That connection allows the propagation of interactions between data layers in different resolutions (e.g., neighborhood and borough levels). For this to be possible, an extra attribute must be added to each record of one of the nodes, indicating how that record relates to the records of the other node. Curio's templates automatically include that extra attribute.

6 The Curio Framework

In this section, we present the Curio framework, an implementation of the Curio dataflow model. Curio is composed of a user interface (detailed in Section 6.1) and a backend infrastructure (Section 6.2). Figure 3 presents Curio's interface and Figure 4 its architecture. Then, we present the collaborative aspects of the framework (Section 6.3), followed by how it supports provenance (Section 6.4), and implementation details (Section 6.5).

6.1 Visual interface

Curio's visual interface is the central element of the framework. It is divided into two modes: a workspace mode, where users collaboratively create their dataflows; and a visualization mode which streamlines the dataflow into a visual analytics interface.

6.1.1 Workspace mode

In the workspace mode, Curio provides an infinite canvas in which users assemble their own dataflows by inserting and connecting nodes. This mode is composed of the canvas, the node panel (with the list of available nodes), a visualization mode toggle button, and a user information panel. When a new dataflow is created, the canvas is initially empty. Nodes can be created by selecting them from the node panel, and they can be resized, repositioned, and deleted. Once on the canvas, a node is composed of four elements, shown in Figure 3(left): a header, displaying the type of the node; a body, showing one of four *node facets*; a footer, showing six buttons (run, template selector, GUI mode, programming mode, provenance mode, output mode); and handles on the left and right sides of the node. Nodes can either be shown in full detail or collapsed into an icon. An edge can be created by dragging a path between two handles from different nodes. Edges can also be repositioned and deleted.

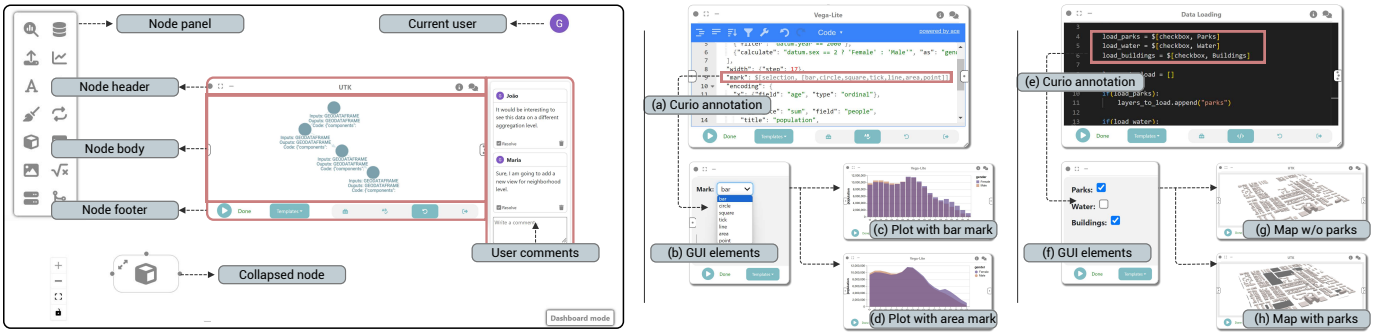

Fig. 3: Left: Main elements of Curio's interface. Center, Right: Connecting facets from the same node. Center: A drop-down menu listing mark attributes is created through an annotation in a Vega-Lite specification. Right: A checkbox is created, but now through an annotation in Python code.

Node facets. Curio's nodes have four facets. The selected facet is displayed in the body of the node: programming facet, GUI facet, provenance facet, and output facet. In the programming facet, the node's implementation source file is made available to the user and can be freely customized according to their needs. Depending on the node type, either a Python code or visualization grammar specification is displayed and is editable by the user. In the GUI facet, visual interface elements, such as drop-downs, sliders, and checkboxes, are provided to the user, allowing them to adjust parameters and configure the behavior of the node operation. For example, in an OpenStreetMap data loading operation, the user can define the physical layers of interest (e.g., parks, streets, buildings). In the provenance facet, a tree is displayed with the history of versions of that particular node (Figure 3(left)). Since nodes can be customized by the user, every time a new version of the node is executed, its version is stored. This approach allows the user to roll back to previous versions at any point of the dataflow construction. Finally, the output facet displays the data produced by the node, either as a visualization (for visualization nodes) or a formatted output (for all other nodes).

Connecting different facets of the same node. Though Curio comes with a host of pre-defined nodes, as outlined in Section 5.3, users can still edit nodes' implementations and make parameters available through GUI elements. This link between the programming facet and the GUI facet is done through *annotations*. Curio's backend interprets the code (either Python or a grammar specification) and searches for specific characters that indicate the beginning and end of an annotation. An annotation will have the format $\$[type, parameters...]$, where *type* can be one of several GUI elements (checkbox, drop-down, slider, date), and *parameters* are the parameters passed to the element for their construction. The Curio interpreter will translate users' interaction with the GUI elements into Python code and grammar specifications. Once the translation is done, Python code will be sent to the backend, and grammar specifications will be run by Vega-Lite or UTK interpreters. Figure 3 exemplifies this procedure, for both grammar (a,b,c,d) and Python code (e,f,g,h). In the grammar example, the user defines a Vega-Lite specification (a) and creates an annotation that will replace the mark property with one of seven marks displayed to the user through a drop-down selection (b). When the user interacts with the GUI element, the node is updated (c,d). In the Python example, the user modifies a Python code loading OpenStreetMap data and creates an annotation (e) that will replace three Boolean values with the values of three checkboxes (f). Similarly, when the user interacts with the GUI element, the node is updated (g,h). This design feature allows users to expose desired functionalities to a GUI while retaining the ability to modify and extend the code as needed.

Node templates. Each type of node provides users with several pre-defined *node templates*. For example, if the user wants to visualize a scatter plot, they can create a Vega-Lite visualization node and use the scatterplot template. Additionally, the framework supports storing and retrieving node templates created or updated by the user. For example, if the user customizes the pre-defined scatterplot template and transforms it into a bubble chart, a new template can be saved in the framework, building an ever-growing template library that can be used by other users in the future. These pre-defined node templates can also be updated using annotations to expose common parameters (such as marks and colorscales for Vega-Lite) through GUI elements. We note that, since Curio's nodes represent common operations performed in urban workflows, node templates enable Curio to be adapted for various domain applications and workflows. For example, new data loaders can be added to handle data formats popular in different areas (e.g., OpenStreetMap data, WRF simulations, NASA's SRTM data), and multiple analysis nodes, such as simulation and machine learning techniques, can be added to the system. In combination with annotations, templates enable users with different backgrounds to collaborate in workflow construction and data analysis tasks by adding new features to the system and receiving feedback, feature requests, and bug reports. For example, these features may help developers understand the needs of domain experts during the implementation of new templates. It also may enable domain experts to create a workflow sketch based on pre-defined node templates and ask for custom features. Additionally, they may allow multiple developers to discuss implementation details or different experts to discuss the workflow steps and results.

Interactive and linked visualizations. Curio provides visualization nodes for 2D plots, supported by the Vega-Lite node, and 3D urban visualizations, supported by the UTK node. Both of these nodes support a diverse set of user interactions, such as picking, brushing, panning, and zooming. For data exploration, Curio allows the sharing of interaction states between different visualization nodes, as first outlined in Section 5.4. These will be handled by interaction nodes, which are responsible for enriching elements of shared layers with descriptions of the state of the interactions. In other words, if an element is selected in one visualization node, that information is updated in the interaction node and propagated to all connected visualizations, allowing for interactive updates.

6.1.2 Visualization mode

After defining a dataflow, Curio provides support for a *visualization mode*, where all nodes and edges are hidden, with the exception of nodes selected by the user to compose a visual analytics interface. To add a node to the visualization mode, the user can *pin* a node to the interface. The nodes selected to compose the visual analytics interface can be resized and organized on the screen without losing their original dataflow order and relationships. This feature is especially important for collaboration. Programmers can use a mix of node templates, Python code, and visualization grammars, and reorganize the elements to present an end-to-end application to urban experts. Figure 1(f) shows the visualization mode.

6.2 Backend infrastructure

Curio's backend infrastructure contains a server, a data manager, and computing sandboxes. The server is responsible for accepting and responding to HTTP requests from the web-based frontend. It handles user interactions related to the creation and maintenance of dataflows, and orchestrates the communication between the frontend, data manager and computing sandboxes. The data manager is responsible for storing all datasets loaded and produced during the execution of the

dataflow, maintaining the persistence of the dataflow state, and managing the provenance database that tracks of all actions performed during the dataflow construction. Finally, the computing sandboxes control the execution of the Python code implemented in the dataflow nodes. For security reasons, Python code is executed in a Docker container.

6.3 Collaboration

A central aspect of Curio is its capability to facilitate the collaborative development of dataflows. Curio offers a user registration feature via a panel located at the top right part of the interface. This enables registered users to collaborate asynchronously on shared dataflows. To increase collaboration among users with varying levels of programming expertise, Curio's nodes have both programming and GUI facets. The GUI facet presents configuration parameters for each node, making it more accessible and user-friendly for users without a programming background. This strategy can effectively transform a dataflow into a GUI application, similar to tools commonly utilized by urban planners, climate professionals, and social scientists. Also, Curio has a comment feature for nodes (Figure 3(left)). This allows urban experts and visualization researchers to discuss the addition of new features, the customization of operations, or even the creation of completely new node templates within a dataflow. This approach documents the collaboration through comments, clarifying the definition and tracking of requirements, and increasing the usefulness of the constructed dataflow. Lastly, since Curio maintains the version history of all nodes, the user can test different variations of the nodes that were created during the collaboration at any time.

6.4 Provenance

Curio contains a provenance database to track versions of nodes. This database has a schema that encompasses two primary levels: the dataflow level, which includes specifications and executions of the dataflows, and the node level, which details the specifications and executions of individual nodes. Next, we detail the database, highlighting its classes. At the dataflow level, the database stores information regarding the name of the dataflow and its nodes, which consume and produce layers. Each layer consists of multiple attributes. Throughout the collaborative process, several versions of the same dataflow can be generated, for instance, by adding or removing nodes or by modifying the source code of specific nodes. Consequently, the user may execute a transaction (such as adding a node) that results in the creation of a new version of the dataflow. Each time a version of the dataflow is executed, a new instance of dataflow execution is generated, associated with the specification of a dataflow. At the node level, nodes' executions are recorded by the node execution class, along with instances of layers consumed and produced, represented by the layer instance class, and their values stored in the attribute value class. These schema classes can be linked with PROV constructs to generate a provenance graph in the W3C PROV standard [21]. The user is mapped to a PROV *agent*; nodes, layers, and attributes to PROV *entities*; and dataflow execution and transformation execution to PROV *activities*.

6.5 Implementation

Curio's frontend was developed using React.js. React Flow was used to implement the main visual components. A React component was created for each data node, containing the logic for parsing and displaying the data. Vega-Lite and UTK were used for visualizations.

The backend server was implemented using Python and Flask. SQLite was used to store provenance data, given its simplicity and portability. Another important part of the backend is the sandbox to run Python code. These containers were configured with the most common libraries used for urban analytics. Additionally, a new serverless version of UTK was developed, allowing external data from the workflow to be integrated into the framework. Beyond its visualization capabilities, UTK's Python library was also used to load and parse OpenStreetMap data. To demonstrate the usefulness of these functionalities, we created a set of default template nodes that can be extended or completely redefined by the user. Finally, user login and registration are managed through Google OAuth.

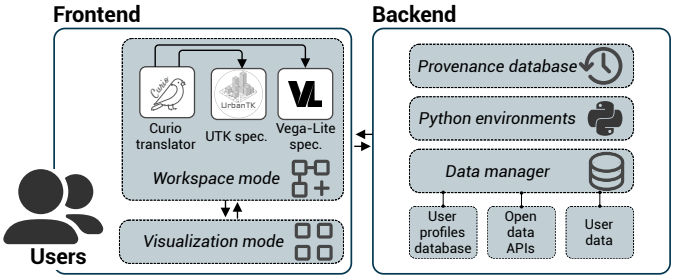

Fig. 4: Curio's frontend and backend components.

7 Usage Scenarios

In this section, we present a set of usage scenarios that demonstrate Curio's flexibility in creating dataflows to tackle pressing urban issues. The scenarios were created in collaboration with three urban experts: two urban planners with experience in urban accessibility (E1) and urban microclimate (E2), and one climate scientist (E3). All of them hold PhDs. First, we engaged with them in a series of one-hour interviews during which we asked them to present some of their common workflows, from data collection to visualization. We then collaboratively built a series of dataflows using Curio. In this process, the urban experts were responsible for creating data analysis and modeling nodes, while visualization researchers handled data wrangling and transformation, and visualization nodes. This collaboration occurred asynchronously, using Curio's comment feature to track changes and updates. The scenarios tackle distinct challenges by integrating a diverse array of data, including images, 3D models of buildings, weather simulations, and sociodemographic data, drawn from various urban settings, such as Boston, Chicago, and Milan. The cases highlight the steps of the dataflow construction process. We direct the reader to the supplementary video for an overview of Curio in action, as well as to Curio's webpage for step-by-step overviews of the scenarios.

7.1 Expert-in-the-loop urban accessibility analyses

While urban livability and quality of life are highly dependent on well-designed public spaces, for a large group of urban dwellers, particularly those with mobility or vision impairments, these spaces remain out of reach. Thus, it is crucial to quantify the degree to which essential destinations are reachable by people with different levels of mobility [43]. In this context, the existence, quality, and surface material of sidewalks are major determinants of destination accessibility, specifically for elderly and wheelchair users [10]. Despite their importance, few cities around the world maintain such spatial catalogues [12]. Recent advances in computer vision and the availability of street-level images have paved the way for low-cost, high-accuracy data collection on various built environment features, including sidewalk paving materials.

In this scenario, we illustrate how Curio can facilitate the workflow of CitySurfaces [26], an active learning framework with expert-in-the-loop for the semantic segmentation of surface materials. Active learning seeks to maximize accuracy while minimizing the number of required labeled data. By iteratively identifying and labeling the most informative or representative images, active learning reduces the number of necessary labeled instances to attain performance comparable to that achieved by labeling a large, randomly selected training dataset all at once [27]. Since CitySurfaces tackles a challenging problem of in-the-wild texture segmentation with high within-class variation and between-class similarity, the training process should be carefully overseen by an expert to reduce systematic biases and identify patterns of failure and their spatial distribution [26].

We use Curio to automate the main part of the CitySurfaces workflow. An overview of the dataflow is presented in Figure 5. E1 begins by importing their training procedure into a new analysis & modeling node (a). Curio's provenance feature allows the expert to analyze several versions of their training procedure, also highlighted in (a). After training, E1 creates an analysis & modeling node with the procedures to calculate the difference between the two highest prediction probabilities in the Softmax layer (b). Using Curio's collaborative functionalities, a

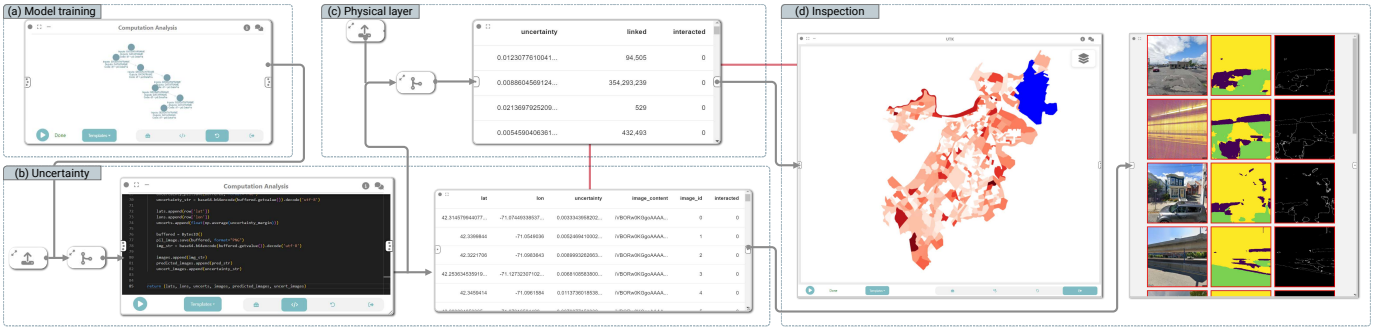

Fig. 5: Using Curio to facilitate expert-in-the-loop inspection of a computer vision model. (a) The user begins by training the model. Provenance information is stored, allowing them to revert to previous versions of the model or explore different training parameters. (b) New nodes are created to load unseen image data and compute the uncertainty of predictions. (c) A physical layer describing neighborhoods in Boston is loaded. (d) An interactive visualization is created, enabling experts to analyze prediction uncertainty across neighborhoods in Boston.

visualization researcher creates a data node to load the image data and joins it with Boston's neighborhoods (c). A UTK node is then added to the dataflow with a spatial map showing the distribution of prediction uncertainties across Boston's neighborhoods (d). These nodes are then used to identify potential shortcomings with the model, requiring new labeled data. The sorted mosaic of images also helps identify patterns of failures where the model had the most difficulty classifying. This signals the need to sample more images with similar light, shadow, and built environment conditions. Given that dense labeling of images is an expensive endeavor, Curio facilitates a more targeted approach by identifying specific conditions that can guide the labeling process.

Importantly, the creation of this Curio dataflow allows the expert to easily iterate over different stages, making use of interactive visualizations that link map and image gallery. This case illustrates how Curio can significantly streamline the expert-in-the-loop process by creating a custom dataflow that integrates the sampling strategy into the main training pipeline, automatically computing the uncertainty metric. Furthermore, Curio enables the visualization of uncertainty maps to guide new sampling at each stage. Tasks that were done in isolation in the original paper [26] are now modularized into an easy-to-use and understandable dataflow.

7.2 What-if scenario planning

Urban development projects, particularly those involving the construction of high-rise buildings, pose significant challenges to the ecological balance and social fabric of cities. In Boston, the introduction of such developments has sparked serious debates [52]. At the heart of the controversy is the potential shadow cast by the new development on the Emerald Necklace, a cherished chain of parks in Boston. This situation has catalyzed community responses, leading to a petition urging the City of Boston to enforce and possibly extend its shadow protection policies [65]. Advocates for the parks argue that the new development threatens the vitality and accessibility of these valued green spaces. The profound ecological and societal ramifications of shadow accumulation on public green spaces demand a sophisticated, user-centric approach to urban planning and environmental stewardship.

We use Curio to create a dataflow that computes the shadow impact of the proposed buildings. Figure 1 presents an overview of the dataflow. With this dataflow, we can compute the shadow impact of the proposed buildings at different times and seasons to support evidence-based environmental impact analysis in Boston. The dataflow begins with a visualization researcher creating an OSM node that loads OpenStreetMap data from the region of interest (a). A sunlight access node connected to the previous node computes the sunlight access (b), which is then visualized in a UTK node (c).

To support what-if scenarios, they add a custom data transformation node that receives the OSM data and changes the height of a selected building, generating an alternate scenario (d). The shadow difference is shown in (e). Such an application is made available to E1 in the visualization mode (f), along with GUI elements created through annotations (b,d). With the elements, they can interactively change the height of buildings using the GUI, and visualize the impact on the public space. Importantly, all of the steps in this pipeline can be transparently accessed by stakeholders who wish to investigate the low-level functionalities. E1 notes that the proposed buildings will add a considerable amount of shadow to the park. Given Curio's flexibility, E1 can quickly change the period and duration of shadow accumulation to gain a more comprehensive view of the proposed buildings' shadow impact.

By empowering researchers and stakeholders to conduct detailed before-and-after analyses of shadow impact assessments, Curio takes a proactive approach to sustainable urban development by ensuring that the development's trajectory honors its commitment to ecological integrity and community well-being. The visualization mode can be made available to community residents, increasing visibility and engagement on this topic.

7.3 Visual analytics of heterogeneous data

As global temperatures continue to rise, urban areas are becoming increasingly susceptible to severe heat incidents, making sustainable active transportation modes less attractive [33]. Outdoor thermal comfort and microclimate conditions in the urban environment are prime factors influencing the use of public spaces and can significantly impact the willingness to walk and bike [53]. In this scenario, we highlight how Curio can be used to create dataflows for micro-scale environmental and human-scale analysis. The objective is to demonstrate how urban planners and decision-makers can effectively use the framework to incorporate multiple datasets, including high-resolution microclimate variables, to assess heat stress levels, particularly for vulnerable populations. Figure 6 provides an overview of the dataflow.

E2 first loads high-resolution mean radiant temperature data, stored as a TIFF file, using a grid layer node. Then, using a file node, E2 loads air temperature, relative humidity, and wind speed data from an ERA5 hourly meteorological dataset. E2 then creates a custom analysis & modeling node (a) that takes the previously loaded data as input and computes the Universal Thermal Climate Index (UTCI) [20], a human biometeorology parameter used to assess human well-being in outdoor environments.

To study the impact of UTCI on vulnerable populations, particularly older adults, they create a new flow that loads sociodemographic data for adults over 65 at the neighborhood level and spatially join the UTCI data in raster format with the sociodemographic data (b). The results are visualized by adding a UTK map (c). The visualization researcher then creates a linked scatterplot using a Vega-Lite node (d). The map and scatterplot are linked through an interaction node, allowing for the analysis of outliers of concern, i.e., regions that have large populations of older adults and high UTCI.

To highlight Curio's flexibility in being adapted to other scenarios, we presented the dataflow to another urban expert (E3), who was responsible for porting the dataflow to Chicago (e). E3 only needed to change two data loading nodes to generate similar visualizations. Further analyses could use a similar dataflow to identify individual routes that may expose vulnerable populations to higher temperatures.

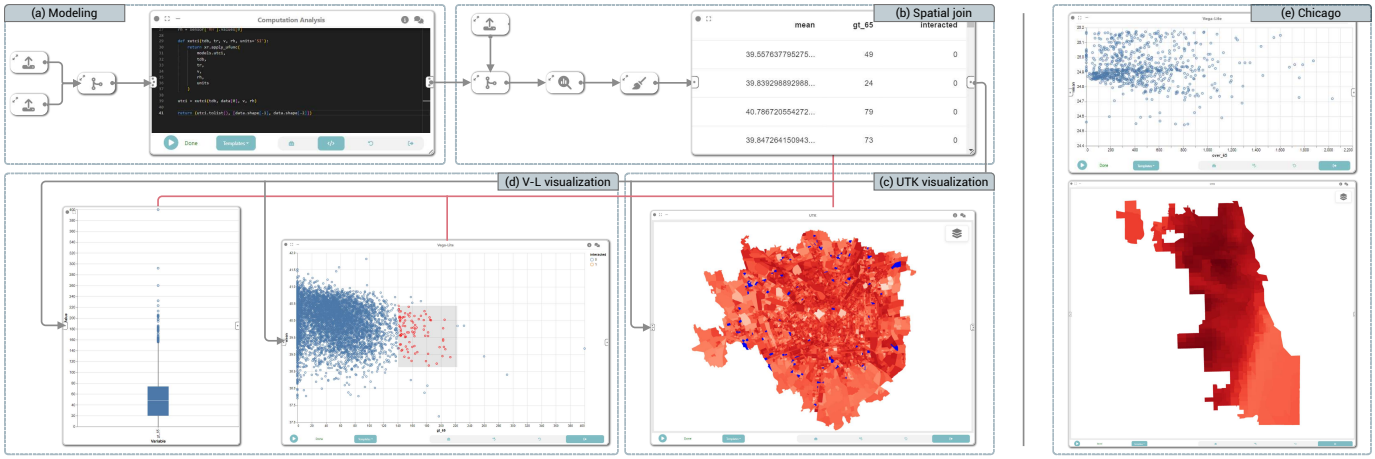

Fig. 6: Using Curio to create visualizations leveraging multiple datasets. (a) The user loads weather data and computes the UTCI, (b) followed by a spatial join with a physical layer describing neighborhoods in Milan. (c,d) The user then creates linked visualizations that highlight neighborhoods with high UTCI and a large population of older adults. (e) By changing two nodes, the user can create a similar visualization using data from Chicago.

7.4 Experts' feedback

After the creation of the dataflows, we met with the experts once more to get their perspectives regarding Curio. We conducted semi-structured interviews, asking them about Curio's strengths and limitations. The experts responded positively to the ability to have a more transparent view of the dataflow steps. Compared to their usual workflow, E1 mentioned that the ability to interactively create a dataflow through a diagram model aligns with how they typically conceptualize their pipeline, describing it as a positive "*alternative to having a series of Python scripts lying around.*"

Regarding comparisons with existing approaches, E2 mentioned that "*the diagram interface resembled ArcGIS Pro's model builder*", but added that "*Curio offers more visualization options, and Python code integration is much easier.*" E2 praised the diagram interface, saying that "*we are used to working with visual programming, specifically those of us working extensively with Rhino's Grasshopper or ArcGIS Pro's model builder, which makes working with Curio a breeze. Also, it does not tie the user to one specific software like Rhino and is much faster.*" E2 also provided positive feedback about the ability to edit Python code, noting that, unlike off-the-shelf tools, "*you can optimize operations if you want to.*" E3 was particularly impressed by the collaboration capabilities, stating that Curio's "*main strength is the collaboration.*"

Regarding limitations, E2 mentioned that it would be important to have the ability to "*drag and drop CSV files from a folder onto the canvas, instead of creating a node to load files.*" Related to this point, E2 also mentioned that purely relying on grammars for the creation of visualizations might be cumbersome for some, as it requires experts to "*learn yet another software stack.*" The same expert was hesitant regarding adoption, mentioning that "*any framework needs a strong support system, with extensive documentation, examples, and an active user community.*"

7.5 Reflection on design goals

We now reflect on the initial design goals that guided our efforts. Regarding DG1 (collaboration), Curio supports asynchronous collaboration and includes several features to facilitate it. We believe that one of the most important collaborative features of the framework is the ability to link various facets of a node through annotations. This approach allowed for a much easier exploration of the parameter space in the second usage scenario. Regarding DG2 (flexibility), Curio supports the creation and connection of computing modules, as well as storage and retrieval of previously created ones. This was positively received by urban experts, particularly because they could easily visualize how the different components of their code connected with each other. Curio's dataflow design enables it to address a wide range of urban-specific problems. As tool builders, we believe that a dataflow approach can lead to more modular and interoperable visual analytics systems. Concerning DG3 (reproducibility and provenance), Curio records changes made to a node in a provenance database. This database serves not only as a historical repository but also as a resource to visualize the evolution of a node over time. Users can leverage these visualizations to navigate through the different states of a node, enabling them to roll back to specific versions as needed.

8 Conclusions

In this paper, we introduced Curio, a web-based framework designed to enhance the creation and execution of urban visual analytics workflows. By providing an intuitive, shared platform with a robust suite of urban-specific computing modules, Curio simplifies the integration, analysis, and visualization of complex urban data, supporting a broad spectrum of urban analytics tasks. Curio is built with inclusivity in mind, accommodating users across various levels of expertise. This flexibility ensures that both domain experts and visualization researchers can contribute effectively, iterating on design choices to produce solutions that align with their specific requirements and expectations.

To anchor collaborative efforts in a robust framework, Curio records each step in the dataflow's evolution. This not only safeguards the contributions of all participants, but also empowers users to explore alternative development trajectories, revisit previous states, and share comprehensive or partial dataflow with ease, thus enhancing reproducibility and transparency. Curio offers a robust, scalable platform for ongoing collaboration and innovation in urban visual analytics.

Limitations. First, Curio's collaborative features are restricted to asynchronously commenting nodes, reusing and extending node templates, and browsing previous versions and states of nodes. Second, it inherits the limitations of Vega-Lite and UTK. For example, it does not support audio or video data, two important types of urban data. Third, Curio tracks modifications made by the user during the construction of a dataflow, but it does not track user interactions within visualization nodes. Fourth, our current mechanism for coordination between visualizations is data-driven, meaning it is not possible to synchronize visualization states. Lastly, our current implementation uses in-memory data structures to store and process datasets and does not perform rendering optimization operations.

Future work. In future work, we plan to support synchronous collaboration sessions. Moreover, although our evaluation methodology finds precedent in similar frameworks [41, 71], we intend to perform a more in-depth evaluation by engaging experts across multiple domains to better assess our system's usability and learnability. With respect to provenance features, we also plan to extend Curio to support the provenance of interactions within visualization nodes. Regarding the support of data types, we aim to explore approaches to integrate other complex data, such as audio and video, into the framework. We also plan to revisit the use of dataflows as an approach for visualization education [61], focusing on urban data and societal problems.

Acknowledgments

We would like to thank the reviewers for their constructive comments and feedback. This study was supported by the Discovery Partners Institute, the National Science Foundation (#2320261, #2330565, #2411223), NASA (#80NSSC22K1683), IDOT (TS-22-340), CNPq (316963/2021-6, 311425/2023-2), and FAPERJ (E-26/202.915/2019, E-26/211.134/2019).

References

[1] D. Akbaba, D. Lange, M. Correll, A. Lex, and M. Meyer. Troubling collaboration: Matters of care for visualization design study. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pp. 1–15. ACM, 2023. doi: 10.1145/3544548.3581168 2

[2] M. M. Alam, L. Torgo, and A. Bifet. A survey on spatio-temporal data analytics systems. *ACM Computing Surveys*, 54(10s):1–38, 2022. doi: 10.1145/3507904 2

[3] S. Alspaugh, N. Zokaei, A. Liu, C. Jin, and M. A. Hearst. Futzing and moseying: Interviews with professional data analysts on exploration practices. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):22–31, 2019. doi: 10.1109/TVCG.2018.2865040 2, 3

[4] L. Ang and K. P. Seng. Big sensor data applications in urban environments. *Big Data Research*, 4:1–12, 2016. doi: 10.1016/j.bdr.2015.12.003 1

[5] L. Barbosa, K. Pham, C. Silva, M. R. Vieira, and J. Freire. Structured open urban data: Understanding the landscape. *Big Data*, 2(3):144–154, 2014. doi: 10.1089/big.2014.0020 1

[6] A. Batch and N. Elmqvist. The interactive visualization gap in initial exploratory data analysis. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):278–287, 2018. doi: 10.1109/TVCG.2017.2743990 2

[7] L. Bavoil, S. Callahan, P. Crossno, J. Freire, C. Scheidegger, C. Silva, and H. Vo. VisTrails: Enabling interactive multiple-view visualizations. In *IEEE Visualization*, pp. 135–142. IEEE, 2005. doi: 10.1109/VISUAL.2005.1532788 2, 3

[8] F. Biljecki and K. Ito. Street view imagery in urban analytics and GIS: A review. *Landscape and Urban Planning*, 215:104217, 2021. doi: 10.1016/j.landurbplan.2021.104217 1, 4

[9] J. Bin, B. Gardiner, E. Li, and Z. Liu. Multi-source urban data fusion for property value assessment: A case study in Philadelphia. *Neurocomputing*, 404:70–83, 2020. doi: 10.1016/j.neucom.2020.05.013 1

[10] T. Chippendale and M. Boltz. The neighborhood environment: Perceived fall risk, resources, and strategies for fall prevention. *The Gerontologist*, 55(4):575–583, 2015. doi: 10.1093/geront/gnu019 7

[11] A. Degbelo. FAIR geovisualizations: Definitions, challenges, and the road ahead. *International Journal of Geographical Information Science*, 36(6):1059–1099, 2021. doi: 10.1080/13658816.2021.1983579 2, 3

[12] S. Deitz, A. Lobben, and A. Alferez. Squeaky wheels: Missing data, disability, and power in the smart city. *Big Data & Society*, 8(2):1–16, 2021. doi: 10.1177/20539517211047735 7

[13] Z. Deng, D. Weng, J. Chen, R. Liu, Z. Wang, J. Bao, Y. Zheng, and Y. Wu. AirVis: Visual analytics of air pollution propagation. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):800–810, 2020. doi: 10.1109/TVCG.2019.2934670 2

[14] Z. Deng, D. Weng, S. Liu, Y. Tian, M. Xu, and Y. Wu. A survey of urban visual analytics: Advances and future directions. *Computational Visual Media*, 9:3–39, 2023. doi: 10.1007/s41095-022-0275-7 2, 3

[15] H. Doraiswamy, E. Tzirita Zacharatou, F. Miranda, M. Lage, A. Ailamaki, C. T. Silva, and J. Freire. Interactive visual exploration of spatio-temporal urban data sets using Urbane. In *International Conference on Management of Data*, pp. 1693–1696. ACM, 2018. doi: 10.1145/3183713.3193559 2

[16] Z. Feng, H. Qu, S.-H. Yang, Y. Ding, and J. Song. A survey of visual analytics in urban area. *Expert Systems*, 39(9):e13065, 2022. doi: 10.1111/exsy.13065 2

[17] L. Ferreira, G. Moreira, M. Hosseini, M. Lage, N. Ferreira, and F. Miranda. Assessing the landscape of toolkits, frameworks, and authoring tools for urban visual analytics systems. *Computers & Graphics*, 123:104013, 2024. doi: 10.1016/j.cag.2024.104013 3

[18] N. Ferreira, M. Lage, H. Doraiswamy, H. Vo, L. Wilson, H. Werner, M. Park, and C. Silva. Urbane: A 3D framework to support data driven decision making in urban development. In *Conference on Visual Analytics Science and Technology*, pp. 97–104. IEEE, 2015. doi: 10.1109/VAST.2015.7347636 2, 3, 4

[19] N. Ferreira, J. Poco, H. T. Vo, J. Freire, and C. T. Silva. Visual exploration of big spatio-temporal urban data: A study of New York City taxi trips. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2149–2158, 2013. doi: 10.1109/TVCG.2013.226 1, 2, 3

[20] D. Fiala, G. Havenith, P. Bröde, B. Kampmann, and G. Jendritzky. UTCI-Fiala multi-node model of human heat transfer and temperature regulation. *International Journal of Biometeorology*, 56:429–441, 2012. doi: 10.1007/s00484-011-0424-7 8

[21] P. Groth and L. Moreau. W3C PROV - An overview of the PROV family of documents. `https://www.w3.org/TR/prov-overview/`, 2013. Accessed on: Jun 2024. 7

[22] J. Grus. I don't like notebooks. `https://conferences.oreilly.com/jupyter/jup-ny/public/schedule/detail/68282.html`, 2018. Accessed on: Jun 2024. 2, 3

[23] P. E. Haeberli. ConMan: A visual programming language for interactive graphics. In *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 103–111. ACM, 1988. doi: 10.1145/54852.378494 2

[24] M. D. Hanwell, K. M. Martin, A. Chaudhary, and L. S. Avila. The Visualization Toolkit (VTK): Rewriting the rendering code for modern graphics cards. *SoftwareX*, 1:9–12, 2015. doi: 10.1016/j.softx.2015.04.001 3

[25] M. Herschel, R. Diestelkämper, and H. Lahmar. A survey on provenance: What for? What form? What from? *The VLDB Journal*, 26:881–906, 2017. doi: 10.1007/s00778-017-0486-1 4

[26] M. Hosseini, F. Miranda, J. Lin, and C. Silva. CitySurfaces: City-scale semantic segmentation of sidewalk materials. *Sustainable Cities and Society*, 79:103630, 2022. doi: 10.1016/j.scs.2021.103630 7, 8

[27] S.-J. Huang, R. Jin, and Z.-H. Zhou. Active learning by querying informative and representative examples. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(10):1936–1949, 2014. doi: 10.1109/TPAMI.2014.2307881 7

[28] R. Ikeda, A. Das Sarma, and J. Widom. Logical provenance in data-oriented workflows? In *International Conference on Data Engineering*, pp. 877–888. IEEE, 2013. doi: 10.1109/ICDE.2013.6544882 4

[29] P. Isenberg, N. Elmqvist, J. Scholtz, D. Cernea, K.-L. Ma, and H. Hagen. Collaborative visualization: Definition, challenges, and research agenda. *Information Visualization*, 10(4):310–326, 2011. doi: 10.1177/1473871611412817 3

[30] W. Javed and N. Elmqvist. ExPlates: Spatializing interactive analysis to scaffold visual exploration. *Computer Graphics Forum*, 32(3pt4):441–450, 2013. doi: 10.1111/cgf.12131 3

[31] S. Jänicke, P. Kaur, P. Kuzmicki, and J. Schmidt. Participatory visualization design as an approach to minimize the gap between research and application. In *VisGap - The Gap Between Visualization Research and Visualization Software*. EG, 2020. doi: 10.2312/visgap.20201108 2, 3

[32] J. Kandt and M. Batty. Smart cities, big data and urban policy: Towards urban analytics for the long run. *Cities*, 109:102992, 2021. doi: 10.1016/j.cities.2020.102992 1

[33] Y. Kim and R. Brown. Effect of meteorological conditions on leisure walking: A time series analysis and the application of outdoor thermal comfort indexes. *International Journal of Biometeorology*, 66(6):1109–1123, 2022. doi: 10.1007/s00484-022-02262-w 8

[34] D. E. Knuth. Literate programming. *The Computer Journal*, 27(2):97–111, 1984. doi: 10.1093/comjnl/27.2.97 2

[35] C. E. Kontokosta. Urban informatics in the science and practice of planning. *Journal of Planning Education and Research*, 41(4):382–395, 2021. doi: 10.1177/0739456X18793716 1, 2

[36] A. Kunze, R. Burkhard, S. Gebhardt, and B. Tuncer. Visualization and decision support tools in urban planning. In *Digital Urban Modeling and Simulation*, pp. 279–298. Springer, 2012. doi: 10.1007/978-3-642-29758-8_15 3

[37] C. Lee, Y. Kim, S. Jin, D. Kim, R. Maciejewski, D. Ebert, and S. Ko. A visual analytics system for exploring, monitoring, and forecasting road traffic congestion. *IEEE Transactions on Visualization and Computer Graphics*, 26(11):3133–3146, 2020. doi: 10.1109/TVCG.2019.2922597 2

[38] W. Loibl, M. Vuckovic, G. Etminan, M. Ratheiser, S. Tschannett, and D. Österreicher. Effects of densification on urban microclimate—a case study for the city of Vienna. *Atmosphere*, 12(4):1–23, 2021. doi: 10.3390/atmos12040511 1

[39] M. Luca, G. M. Campedelli, S. Centellegher, M. Tizzoni, and B. Lepri. Crime, inequality and public health: A survey of emerging trends in urban data science. *Frontiers in Big Data*, 6:1–20, 2023. doi: 10.3389/fdata.2023.1124526 1

[40] J. Lukasczyk, X. Liang, W. Luo, E. D. Ragan, A. Middel, N. Bliss,

D. White, H. Hagen, and R. Maciejewski. A collaborative web-based environmental data visualization and analysis framework. In *Workshop on Visualisation in Environmental Sciences*. EG, 2015. doi: 10.2312/envirvis.20151087 3

[41] S. L'Yi, Q. Wang, F. Lekschas, and N. Gehlenborg. Gosling: A grammar-based toolkit for scalable and interactive genomics data visualization. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):140–150, 2022. doi: 10.1109/TVCG.2021.3114876 9

[42] Y. Lyu, H. Lu, M. K. Lee, G. Schmitt, and B. Y. Lim. IF-City: Intelligible fair city planning to measure, explain and mitigate inequality. *IEEE Transactions on Visualization and Computer Graphics*, 30(7):3749–3766, 2024. doi: 10.1109/TVCG.2023.3239909 3

[43] J. Mason, P. Turner, and M. Steriu. Universal access in urban areas: Why universal access in urban areas matters for sustainable mobility. Technical report, World Bank Group, Washington, D.C., 2017. 7

[44] H. Mei, Y. Ma, Y. Wei, and W. Chen. The design space of construction tools for information visualization: A survey. *Journal of Visual Languages & Computing*, 44:120–132, 2018. doi: 10.1016/j.jvlc.2017.10.001 3

[45] F. Miranda, H. Doraiswamy, M. Lage, L. Wilson, M. Hsieh, and C. T. Silva. Shadow Accrual Maps: Efficient accumulation of city-scale shadows over time. *IEEE Transactions on Visualization and Computer Graphics*, 25(3):1559–1574, 2019. doi: 10.1109/TVCG.2018.2802945 3, 5

[46] F. Miranda, H. Doraiswamy, M. Lage, K. Zhao, B. Gonçalves, L. Wilson, M. Hsieh, and C. T. Silva. Urban Pulse: Capturing the rhythm of cities. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):791–800, 2017. doi: 10.1109/TVCG.2016.2598585 2, 3, 5

[47] F. Miranda, M. Hosseini, M. Lage, H. Doraiswamy, G. Dove, and C. T. Silva. Urban Mosaic: Visual exploration of streetscapes using large-scale image data. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pp. 1–15. ACM, 2020. doi: 10.1145/3313831.3376399 2, 3, 5

[48] F. Miranda, T. Ortner, G. Moreira, M. Hosseini, M. Vuckovic, F. Biljecki, C. T. Silva, M. Lage, and N. Ferreira. The state of the art in visual analytics for 3D urban data. *Computer Graphics Forum*, 43(3):e15112, 2024. doi: 10.1111/cgf.15112 3

[49] G. Moreira, M. Hosseini, M. N. A. Nipu, M. Lage, N. Ferreira, and F. Miranda. The Urban Toolkit: A grammar-based framework for urban visual analytics. *IEEE Transactions on Visualization and Computer Graphics*, 30(1):1402–1412, 2024. doi: 10.1109/TVCG.2023.3326598 2, 3, 4, 5

[50] R. Mota, N. Ferreira, J. D. Silva, M. Horga, M. Lage, L. Ceferino, U. Alim, E. Sharlin, and F. Miranda. A comparison of spatiotemporal visualizations for 3D urban analytics. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):1277–1287, 2023. doi: 10.1109/TVCG.2022.3209474 3

[51] A. Mueller. 5 reasons why Jupyter notebooks suck. https://towardsdatascience.com/5-reasons-why-jupyter-notebooks-suck-4dc201e27086/, 2018. Accessed on: Jun 2024. 2, 3

[52] D. Murphy. Charles River Conservancy calls for citywide shadow protection policy Amid Longwood Place proposal. https://thebostonsun.com/2023/01/12/charles-river-conservancy-calls-for-citywide-shadow-protection-policy-amid-longwood-place-proposal/, 2023. Accessed on: Jun 2024. 8

[53] M. Nikolopoulou, N. Baker, and K. Steemers. Thermal comfort in outdoor urban spaces: Understanding the human parameter. *Solar Energy*, 70(3):227–235, 2001. doi: 10.1016/S0038-092X(00)00093-1 8

[54] R. I. Ogie, R. J. Clarke, H. Forehead, and P. Perez. Crowdsourced social media data for disaster management: Lessons from the PetaJakarta.org project. *Computers, Environment and Urban Systems*, 73:108–117, 2019. doi: 10.1016/j.compenvurbsys.2018.09.002 1

[55] S. G. Parker and C. R. Johnson. SCIRun: A scientific programming environment for computational steering. In *Proceedings of the ACM/IEEE Conference on Supercomputing*, pp. 52–es. ACM, 1995. doi: 10.1145/224170.224354 2

[56] J. F. Pimentel, L. Murta, V. Braganholo, and J. Freire. A large-scale study about quality and reproducibility of Jupyter notebooks. In *IEEE/ACM International Conference on Mining Software Repositories*, pp. 507–517. IEEE, 2019. doi: 10.1109/MSR.2019.00077 2, 3

[57] J. Rulff, F. Miranda, M. Hosseini, M. Lage, M. Cartwright, G. Dove, J. Bello, and C. T. Silva. Urban Rhapsody: Large-scale exploration of urban soundscapes. *Computer Graphics Forum*, 41(3):209–221, 2022. doi: 10.1111/cgf.14534 2

[58] M. Saha, S. Patil, E. Cho, E. Y.-Y. Cheng, C. Horng, D. Chauhan, R. Kangas, R. McGovern, A. Li, J. Heer, and J. E. Froehlich. Visualizing urban accessibility: Investigating multi-stakeholder perspectives through a map-based design probe study. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, pp. 1–14. ACM, 2022. doi: 10.1145/3491102.3517460 2, 3

[59] M. Saha, M. Saugstad, H. T. Maddali, A. Zeng, R. Holland, S. Bower, A. Dash, S. Chen, A. Li, K. Hara, and J. Froehlich. Project Sidewalk: A web-based crowdsourcing tool for collecting sidewalk accessibility data at scale. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pp. 1–14. ACM, 2019. doi: 10.1145/3290605.3300292 1

[60] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer. Vega-Lite: A grammar of interactive graphics. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):341–350, 2017. doi: 10.1109/TVCG.2016.2599030 2

[61] C. T. Silva, E. Anderson, E. Santos, and J. Freire. Using VisTrails and provenance for teaching scientific visualization. *Computer Graphics Forum*, 30(1):75–84, 2011. doi: 10.1111/j.1467-8659.2010.01830.x 9

[62] X. Sun, M. Plaudis, and Y. Coady. BUDI: Building urban designs interactively. A spatial-based visualization and collaboration platform for urban planning. In *IEEE Annual Information Technology, Electronics and Mobile Communication Conference*, pp. 888–895. IEEE, 2021. doi: 10.1109/IEMCON53756.2021.9623112 3

[63] J. Tierny, G. Favelier, J. A. Levine, C. Gueunet, and M. Michaux. The Topology ToolKit. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):832–842, 2018. doi: 10.1109/TVCG.2017.2743938 5

[64] A. Y. Wang, A. Mittal, C. Brooks, and S. Oney. How data scientists use computational notebooks for real-time collaboration. *Proceedings of the ACM on Human-Computer Interaction*, 3(CSCW):1–30, 2019. doi: 10.1145/3359141 3

[65] S. Wolf. Protect the Emerald Necklace parks - Please don't take our sunlight away. https://www.change.org/p/protect-the-emerald-necklace-parks-please-don-t-take-our-sunlight-away, 2023. Accessed on: Jun 2024. 8

[66] K. Wongsuphasawat, Y. Liu, and J. Heer. Goals, process, and challenges of exploratory data analysis: An interview study. *arXiv*, pp. 1–10, 2019. doi: 10.48550/arXiv.1911.00568 2, 3

[67] A. Wu, D. Deng, M. Chen, S. Liu, D. Keim, R. Maciejewski, S. Miksch, H. Strobelt, F. Viégas, and M. Wattenberg. Grand challenges in visual analytics applications. *IEEE Computer Graphics and Applications*, 43(5):83–90, 2023. doi: 10.1109/MCG.2023.3284620 2, 3

[68] A. Wu, D. Deng, F. Cheng, Y. Wu, S. Liu, and H. Qu. In defence of visual analytics systems: Replies to critics. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):1026–1036, 2023. doi: 10.1109/TVCG.2022.3209360 2

[69] Y. Xie. The first notebook war. https://yihui.org/en/2018/09/notebook-war/, 2018. Accessed on: Jun 2024. 2, 3

[70] W. Yap, P. Janssen, and F. Biljecki. Free and open source urbanism: Software for urban planning practice. *Computers, Environment and Urban Systems*, 96:101825, 2022. doi: 10.1016/j.compenvurbsys.2022.101825 1, 2

[71] B. Yu and C. T. Silva. VisFlow - Web-based visualization framework for tabular data with a subset flow model. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):251–260, 2017. doi: 10.1109/TVCG.2016.2598497 2, 3, 9

[72] Y. Zheng, L. Capra, O. Wolfson, and H. Yang. Urban computing: Concepts, methodologies, and applications. *ACM Transactions on Intelligent Systems and Technology*, 5(3):1–55, 2014. doi: 10.1145/2629592 2

[73] Y. Zheng, W. Wu, Y. Chen, H. Qu, and L. M. Ni. Visual analytics in urban computing: An overview. *IEEE Transactions on Big Data*, 2(3):276–296, 2016. doi: 10.1109/TBDATA.2016.2586447 2

[74] L. Zhu, F. R. Yu, Y. Wang, B. Ning, and T. Tang. Big data analytics in intelligent transportation systems: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 20(1):383–398, 2019. doi: 10.1109/TITS.2018.2815678 1

[75] P. Ziegler and S. E. Chasins. A need-finding study with users of geospatial data. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pp. 1–16. ACM, 2023. doi: 10.1145/3544548.3581370 2, 3