FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

A Procedural Framework for the Immersive Visualization of Multidimensional Flow Networks

José António Pereira Martins



Mestrado em Engenharia Informática e Computação

Supervisor: Prof. Rui Rodrigues

July 28, 2026

© José Martins, 2026

A Procedural Framework for the Immersive Visualization of Multidimensional Flow Networks

José António Pereira Martins

Mestrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

President: Prof. Alexandre Valle de Carvalho
Examiner: Prof. João Moreira
Member: Prof. Rui Rodrigues

July 28, 2026

Resumo

As Analíticas Imersivas têm como objetivo melhorar a análise de dados ao colocar o analista dentro de um ambiente tridimensional. No entanto, a construção destas visualizações continua reservada a especialistas, e os dados de fluxo abstratos, como transações financeiras, viagens ou tráfego de rede, são definidos pelas suas ligações e não necessariamente por uma posição fixa, não trazendo qualquer geometria inerente que os situe num mundo 3D. Os sistemas existentes ou dependem de dados que já trazem coordenadas, ou tratam o ambiente gerado como um contentor passivo, ou são ferramentas de propósito único.

Esta dissertação propõe uma framework procedural que transforma qualquer conjunto de dados de fluxo baseado em eventos num mundo de dados navegável em Realidade Virtual (VR), no qual a própria geração procedimental é o canal analítico. Isto é, o relevo do terreno e a densidade da vegetação codificam quantidades por nó definidas pelo analista, tipicamente o fluxo de entrada e de saída de cada nó, ao longo do tempo. O ambiente é gerado a partir dos dados e configurado parcialmente através de um ficheiro JavaScript Object Notation (JSON) que acompanha o conjunto de dados, sem alterações de código nem modelação manual.

O protótipo, construído em Unity, dispõe os nós com um algoritmo de forças, gera o terreno a partir de fórmulas com biomas, água e vegetação, anima os fluxos como partículas que viajam ao longo de curvas entre origem e destino, e trata o tempo como uma função pura do instante normalizado, navegável com reprodução, pausa e salto arbitrário, a taxas de imagem adequadas à Realidade Virtual.

Como o contexto ambiental é um canal separável, a sua contribuição pode ser medida por ablação. A avaliação tem três vertentes: o desempenho do sistema; uma apreciação por especialistas com a framework SCORE, que quantifica a diegese do ambiente; e um estudo de utilizadores que cruza o ambiente, básico contra procedimental, com a complexidade do conjunto de dados, utilizando um sistema público de partilha de bicicletas amostrado em duas escalas, cujas viagens são tratadas como fluxo abstrato, deixando de lado a geografia das estações.

Num estudo com vinte participantes, cada um a realizar todas as condições, o ambiente procedimental melhorou a correção das tarefas e reduziu a carga cognitiva percebida face a uma cena abstrata equivalente, com o benefício de carga a crescer à medida que o conjunto de dados se adensa. A melhoria na correção foi unânime nas tarefas que o terreno codifica. O benefício foi específico desse canal. As tarefas lidas no terreno melhoraram nitidamente, enquanto uma tarefa de controlo, não codificada pelo ambiente, e o canal de vegetação não mostraram efeito, o que indica que o ganho vem da informação que os elementos gerados transmitem e não de uma cena globalmente mais rica. O painel de especialistas classificou a condição procedimental como tendo maior diegese, concentrada nas dimensões que o relevo determina, e dezoito de dezanove participantes preferiram-na. Os resultados assentam numa amostra modesta e num único domínio.

Palavras-chave: Analíticas Imersivas • Geração Procedimental de Conteúdos • Visualização de Fluxos • Realidade Virtual • Terreno Orientado por Dados

Abstract

Immersive Analytics (IA) aims to improve data analysis by placing the analyst inside a three-dimensional environment. Yet authoring these visualizations remains the preserve of experts, and abstract flow data, such as financial transactions, trips or network traffic, is defined by its connections, not necessarily by a fixed position, and so brings no inherent geometry with which to place it in a 3D world. Existing systems either depend on data that already carries coordinates, treat the generated environment as a passive container, or are single-purpose tools.

This dissertation proposes a procedural framework that turns any event-based flow dataset into a navigable Virtual Reality (VR) data-world, in which the procedural generation itself is the analytical channel. That is, the terrain relief and the vegetation density carry analyst-defined per-node quantities, typically each node's incoming and outgoing flow, over time. The environment is generated from the data and configured partially through a JavaScript Object Notation (JSON) file accompanying the dataset, with no code changes and no manual modeling.

The prototype, built on Unity, lays the nodes out with a force-directed algorithm, generates the terrain from formulas together with biomes, water and vegetation, animates the flows as particles traveling along curves between origin and destination, and treats time as a pure function of the normalized instant, navigable through play, pause and arbitrary seeking, at frame rates suited to VR.

Because the environmental context is a separable channel, its contribution can be measured by ablation. The evaluation has three strands: the system's performance; an expert assessment with the SCORE framework, which quantifies the environment's diegesis; and a user study crossing the environment, basic against procedural, with dataset complexity, using a public bike-sharing dataset sampled at two scales, whose trips are treated as abstract flow with their station geography set aside.

In a within-subjects study with twenty participants, the procedural environment improved task correctness, with every participant improving on the tasks the terrain encodes, and lowered perceived cognitive load relative to an otherwise identical abstract scene, with the load benefit growing as the dataset densified. The benefit was specific to the channel the relief encodes. Tasks read from the terrain improved sharply, while a control task the environment does not encode and the vegetation channel showed no effect, which indicates that the gain comes from the information the generated features carry and not from a generally richer scene. The expert panel rated the procedural condition higher on diegesis, concentrated in the dimensions the relief drives, and eighteen of nineteen participants preferred it. The findings rest on a modest sample in a single domain.

Keywords: Immersive Analytics • Procedural Content Generation • Flow Visualization • Virtual Reality • Data-driven Terrain

The United Nations Sustainable Development Goals

The United Nations Sustainable Development Goalss (SDGs) provide a global framework for achieving a better and more sustainable future for all. It includes 17 goals (https://sdgs.un.org/goals) to address the world's most pressing challenges, including poverty, inequality, climate change, environmental degradation, peace, and justice.

This dissertation is a methodological contribution at the intersection of procedural content generation and immersive analytics. Its impact on the SDGs is therefore mostly enabling and indirect. It does not tackle a societal challenge on its own, but it lowers the cost and the expertise required to analyze complex movement data in immersive environments, and it is evaluated on data drawn from sustainable urban mobility. The goals most relevant to the work are SDG 9, SDG 11 and SDG 4.

The specific Sustainable Development Goals mentioned have the following names:

SDG 9 Build resilient infrastructure, promote inclusive and sustainable industrialization and foster innovation

SDG 11 Make cities and human settlements inclusive, safe, resilient and sustainable

SDG 4 Ensure inclusive and equitable quality education and promote lifelong learning opportunities for all

SDG	Target	Contribution	Performance Indicators and Metrics
9	9.5	Advances scientific research and technological capability by contributing a novel method that turns abstract, coordinate-free flow data into navigable immersive 3D environments, together with an empirical evaluation of its analytical value.	Research outputs (a working prototype, an expert SCORE assessment and a controlled user study); novelty with respect to prior immersive-visualization systems.
	9.c	Increases access to advanced analytical technology by expressing all dataset-specific configuration as portable data rather than code, so that an immersive analysis can be set up without specialized 3D-modeling or programming skills.	Reduction in expertise and time needed to deploy an immersive visualization for a new dataset; number of distinct domains served by the same pipeline without code changes.
11	11.2	Supports the analysis of sustainable urban mobility. The system is evaluated on a public bike-sharing dataset and helps analysts perceive spatial and temporal demand patterns, such as busy stations and peak periods, relevant to planning and operating transport systems.	Applicability to mobility datasets; analysts' accuracy and response time when identifying mobility patterns, measured directly by the study tasks.
4	4.4	May support data literacy and analytical training, since a naturalistic, immersive representation can make complex relational data more understandable and engaging than abstract charts.	Perceived cognitive load and task performance from the user study; lower load and higher accuracy would indicate improved understandability.

Acknowledgements

I would like to thank my supervisor, Professor Rui Rodrigues, who guided this dissertation from the first ideas to the final version. He has been a great teacher since the early years of the course, always available and willing to help, and a big part of how I got to this point. This work owes a lot to his advice and his patience.

My thanks also go to everyone who took part in the user study, and to all the people who helped this dissertation reach the finish line in one way or another. The evaluation in this work would not have been possible without their time and their willingness to help.

More generally, I want to thank everyone who was part of my years at the university. Many people crossed my path along the way and helped shape it, including some I am surely forgetting as I write this, and I am grateful to all of them.

To everyone I shared time with at NIAEFEUP, the people I laughed and worked with, and who gave me a way to step away from the course when I needed it. Those moments meant more than they probably realise.

To my friends António, Cardoso, Dan, Guida and Tomás, who were there from day one. We have a lot of good memories together, the nights at each other's places, the games, and the long talks that had nothing to do with the course and everything to do with keeping me sane. They were there on the bad days and the good ones. I was lucky to have them, and I won't forget it.

Finally, and most importantly, to my parents. None of this would exist without them. They gave me the chance to study and to do what I wanted, and they were behind me on every choice I made, the good and the bad, without ever asking for anything back. Everything I have done, this dissertation included, started with them. Thank you for everything, always, since long before any of this. I hope that, after all this time, you are proud of who I have become.

José Martins

Declaração de Honra

Declaro que a presente dissertação é de minha autoria e não foi previamente apresentada e avaliada nesta ou em outra instituição. As referências a outros autores (afirmações, ideias, pensamentos), assim como a utilização de trabalhos publicados respeitam escrupulosamente as regras da atribuição, e encontram-se devidamente indicadas no texto e nas referências bibliográficas, de acordo com as normas de referenciação. Tenho consciência de que a prática de plágio ou de autoplágio constituem ilícitos académicos, conforme estabelecido no Código de Ética da U.Porto.

Declaro, ainda, que utilizei ferramentas de Inteligência Artificial como apoio ao longo da elaboração da presente dissertação, em tarefas como pesquisa, organização de ideias e revisão de texto e de código. Dada a natureza geral e transversal dessa utilização, não a assinalo em cada passagem específica. Assumo a integral responsabilidade por todo o conteúdo, que foi objeto de cuidada verificação para deteção de erros, imprecisões, atribuições infundadas ou outras formas de enviesamento de conteúdos e argumentos, bem como de determinação de autoria.

José António Pereira Martins



"It's the kind of thing that makes you glad you stopped and smelled the pine trees along the way, you know?"

Gabbro, Outer Wilds

Contents

1 Introduction **1**
- 1.1 Motivation . . . 2
- 1.2 Problem . . . 2
- 1.3 Research Questions . . . 3
- 1.4 Hypothesis . . . 4
- 1.5 Document Structure . . . 4

2 State of the Art **5**
- 2.1 Technologies of Procedural Generation . . . 5
  - 2.1.1 Stochastic Generation and Noise . . . 5
  - 2.1.2 Grammars and Rules . . . 6
  - 2.1.3 Constraint Satisfaction and Tessellation . . . 8
- 2.2 Design Principles in Immersive Analytics . . . 8
  - 2.2.1 Spatial Metaphors and Data Mapping . . . 9
  - 2.2.2 Validating Coherence and Presence . . . 10
- 2.3 Immersive Analytics of Abstract and Network Data . . . 10
- 2.4 Existing Intersections: PCG Applied to Visualization . . . 13
  - 2.4.1 Geospatial and Urban Reconstruction . . . 13
  - 2.4.2 Virtual Archives and Museums . . . 14
  - 2.4.3 Navigation in Solution Spaces . . . 14
  - 2.4.4 Scientific and Ecological Visualization . . . 15
- 2.5 Gap Analysis . . . 16

3 A Procedural Framework for Flow Visualization **19**
- 3.1 Scoping: From Arbitrary Data to Flow Data . . . 19
- 3.2 The Data-World Concept . . . 20
- 3.3 Flow Data Model . . . 22
- 3.4 Mapping Strategies . . . 23
  - 3.4.1 Terrain Elevation as a Configurable Aggregate . . . 24
  - 3.4.2 Flows as Animated, Sized Events . . . 28
  - 3.4.3 Identity Through Color and Form . . . 29
  - 3.4.4 Time as a Navigable Axis . . . 29
- 3.5 Requirements . . . 29
- 3.6 Summary . . . 30

4 Implementing the Data-World **31**
- 4.1 Architecture . . . 31
  - 4.1.1 Timeline and Playback . . . 31

4.1.2 Configuration Model . . . 32
4.1.3 Subsystem Communication . . . 32
4.1.4 Pipeline Overview . . . 32
4.2 Technologies and Platform . . . 33
4.3 Data Ingestion and Configuration . . . 34
4.4 Spatial Layout and Node Representation . . . 34
4.5 The Procedural Terrain . . . 35
4.5.1 From Formula to Relief . . . 35
4.5.2 Peak Shape . . . 36
4.5.3 Biomes, Textures, Water and Vegetation . . . 37
4.5.4 Ablation Support . . . 39
4.6 Flow Visualization . . . 39
4.6.1 Size from a Per-Record Formula . . . 40
4.6.2 Visual Aggregation . . . 40
4.6.3 Color, Trails and Reuse . . . 41
4.6.4 Trajectories over Animated Relief . . . 42
4.7 Timeline and Interaction . . . 42
4.8 Performance Engineering . . . 43
4.9 Verification . . . 44
4.10 Datasets and Observed Limits . . . 45

5 Expert Assessment and Ablation Study **47**
5.1 System Performance . . . 48
5.2 Expert Assessment with SCORE . . . 49
5.3 User Study Design . . . 50
5.3.1 Factorial Structure . . . 51
5.3.2 Pilot Study and Design Refinement . . . 51
5.3.3 Datasets: One Domain at Two Scales . . . 52
5.3.4 Tasks . . . 53
5.3.5 Measurement Instruments . . . 54
5.3.6 Procedure . . . 56
5.3.7 Participants . . . 57
5.4 What the Comparison Can and Cannot Show . . . 57
5.5 Results . . . 58
5.5.1 SCORE Expert Assessment . . . 58
5.5.2 Task Correctness . . . 60
5.5.3 Response Time . . . 61
5.5.4 Perceived Cognitive Load . . . 61
5.5.5 Simulator Sickness . . . 62
5.5.6 Subjective Preference and Open Feedback . . . 62
5.6 Discussion . . . 63
5.7 Limitations of the Study . . . 64

6 Conclusions and Future Work **66**
6.1 Summary of Contributions . . . 66
6.2 Research Questions Revisited . . . 67
6.3 Limitations . . . 68
6.4 Future Work . . . 68
6.5 Closing Remarks . . . 70

References **71**

A Study Dataset Configuration File **78**

B The Configuration Formula Language **82**
B.1 Expression Grammar 82
B.2 Per-Node Terms 83
B.3 Per-Record Flow Formula 83
B.4 Worked Examples 84

C SCORE Expert-Assessment Materials **85**
C.1 Dimensions and Rubric 85
C.2 Procedure 85
C.3 Form Items 86
C.4 Scoring 86

D User Study Protocol and Instruments **87**
D.1 Briefing and Consent 87
D.2 Counterbalancing 87
D.3 Pre-Questionnaire 88
D.4 Per-Condition and Inter-Condition Questionnaires 88
D.5 Answer Sheet and Log Format 89

List of Figures

2.1 *Archipelago*: terrain from geolocation history . . . 6
2.2 Rule-based modeling of complete streets . . . 7
2.3 Procedural urban forestry . . . 8
2.4 Data-driven generative glyphs . . . 9
2.5 Immersive scatterplots and parallel coordinates in VR . . . 11
2.6 An immersive factory generated from CI data . . . 12
2.7 Immersive origin-destination flow maps . . . 12
2.8 Procedural reconstruction of a historical city . . . 13
2.9 A virtual art gallery generated from a query . . . 14
2.10 Immersive exploration of a generative solution space . . . 15
2.11 A data-driven virtual forest . . . 16
2.12 Positioning of this thesis within the reviewed landscape . . . 18

3.1 The generated data-world for a flights dataset . . . 21
3.2 The two environmental-context conditions . . . 22
3.3 The relief and vegetation channels animating over time . . . 25

4.1 The pipeline as built . . . 33
4.2 A node rendered as a uniform pillar . . . 35
4.3 Overlapping peaks combine by maximum, not sum . . . 36
4.4 Altitude-based biomes on the relief . . . 37
4.5 Vegetation on the grass band of a hill . . . 39
4.6 Visual aggregation of flows into bundles . . . 40
4.7 A flow in transit . . . 41
4.8 Routing a flow curve over the relief . . . 42
4.9 The minimal floating control panel . . . 43
4.10 The graph-density limit on the taxi dataset . . . 46

5.1 The within-subjects 2×2 design . . . 51
5.2 The two complexity levels . . . 53
5.3 The session a participant goes through . . . 56

List of Tables

2.1 The reviewed families against the three gap properties 18

3.1 The canonical flow-data model 23
3.2 The marks and channels that convey the data 24

4.1 Structural and dynamic metrics of the datasets 45

5.1 System performance per dataset 48
5.2 The two study subsets 53
5.3 SCORE diegesis profile, basic versus procedural 59
5.4 Correctness per condition and task 60
5.5 Paired comparisons of the procedural against the basic condition 61
5.6 Perceived cognitive load per condition 62

B.1 Per-node term fields 83
B.2 Per-node formula options 83

C.1 The five SCORE dimensions and their anchors 85

D.1 The four counterbalancing sequences 88
D.2 Answer-sheet template 89

List of Acronyms

AI	Artificial Intelligence
BSP	Binary Space Partitioning
CAVE	Cave Automatic Virtual Environment
CGA	Computer Generated Architecture
CI	Confidence Interval
CPU	Central Processing Unit
CSV	Comma-Separated Values
EBNF	Extended Backus-Naur Form
FMS	Fast Motion Sickness Scale
GAN	Generative Adversarial Network
GIS	Geographic Information System
GPU	Graphics Processing Unit
HMD	Head-Mounted Display
IA	Immersive Analytics
JSON	JavaScript Object Notation
L-System	Lindenmayer System
MST	Minimum Spanning Tree
NASA-TLX	NASA Task Load Index
NLP	Natural Language Processing
NPC	Non-Player Character
PCG	Procedural Content Generation
RL	Reinforcement Learning
SDG	Sustainable Development Goals
SSQ	Simulator Sickness Questionnaire
URP	Universal Render Pipeline
VR	Virtual Reality
WFC	Wave Function Collapse

Chapter 1

Introduction

Immersive Analytics (IA) was developed with the goal of exploring new approaches to data visualization that create more engaging and immersive experiences. The aim is to support analytical reasoning in the emerging field of big data [1], through more natural and intuitive ways of interacting with information [2]. Skarbez et al.[3] suggested that IA is also built on the idea that the combination of human and machine intelligences can lead to insights that are not possible to reach alone, especially when working with large amounts of data. However, building these environments still requires design expertise and manual modeling, which makes immersive analysis impractical for everyday use.

One way to avoid this is to generate the environment automatically, instead of building it by hand. This is what Procedural Content Generation (PCG) explores, for virtual environments and virtual content in general. Togelius et al.[4] defined PCG as the algorithmic generation of game content with limited or no direct human contribution, a category that spans not only levels, maps and quests but also textures, visual effects, animation, audio, narrative and game rules, excluding only the engine itself and the behavior of Non-Player Characters (NPCs). Procedural generation is therefore not confined to the shaping of geometry. It reaches the visual, temporal and multimodal facets of a virtual world. This breadth matters here, because the environment this thesis generates conveys data not through terrain form alone but also through color, biomes, vegetation and animated change over time. This approach allows experts to significantly reduce the time and manual effort required to produce complex content. Moreover, game designers improve their understanding of design, as they are required to encode the rules of world-building instead of manually placing assets. Finally, improved PCG methods lead to a greater understanding of the design process, which leads to even more advanced PCG methods. This highlights the evolutionary nature of PCG, which explains why, after four decades of existence, it remains a major research area [4, 5].

The applicability of PCG can be extended beyond games, as exemplified by Atlas [6], an approach using PCG methods for procedural data visualization. However, while Atlas focused on the generation of 2D charts, the intersection between IA and PCG in the context of 3D or spatial environments has potential to be explored, particularly for visualizing highly interconnected data.

1.1 Motivation

The growing volume and complexity of data require new approaches for its analysis. Different research fields are concerned with the development of such methods. For data visualization, data must follow a strict format and users usually require transformation libraries or interactive tools to process it, which requires some expertise in programming or in specific tools [7]. For the specific case of IA, expertise in designing the visualization is necessary [8]. At the same time, PCG techniques offer an efficient way to create 3D environments automatically from structured data. This overcomes the need for a designer, which reduces both the costs and time required to use immersive methods for data visualization [5]. Instead, the technical requirement shifts from artistic modeling to the use of an automated generative system, allowing analysts to focus on the data itself instead of the construction of the world [5].

The objective of this thesis is to design a framework that turns an event-based flow dataset into a navigable Virtual Reality (VR) environment, built automatically from the data and configured without code, in which the procedural generation itself is the analytical channel. While the potential for PCG-IA spans across all data types, this research strategically focuses on event-based flow datasets. This focus is justified by the fact that flow data presents some of the most significant visual challenges in the field, and provides a high-value entry point to test procedural mappings within the constraints of an MSc dissertation. The motivation for this thesis arises from the recognition that PCG has the potential to advance the entire field of IA by introducing automation and scalability. To achieve this, it is necessary to first understand how principles from PCG can be adapted to develop a framework that translates data into meaningful 3D immersive analytical environments. Simultaneously, the system must be designed to accurately map data into geometry in a way that the structure, attributes, and relationships in the original dataset are maintained. Lastly, it is essential to evaluate whether such a procedurally generated environment actually improves the analyst's ability to read spatial and temporal flow patterns, and at what cognitive cost, when compared to a conventional abstract 3D baseline.

1.2 Problem

Despite the advances in IA and PCG, a solution that can automate the translation of tabular data into 3D analytical environments through the intersection of these fields is not yet a standard approach.

Currently, the creation of 3D analytical environments relies on advanced software that typically requires expertise, or on human designers specialized in 3D Modeling, visualization design and immersive interaction [8]. This makes IA an impractical option for daily use. An analyst with a high-dimensional dataset cannot simply visualize it inside an immersive environment. This bottleneck intensifies if the dataset in analysis is often updated, as manual modeling is significantly slower than the rate of incoming data. Consequently, the high cost and time necessary for IA reserve its use for specific cases instead of an everyday, functional tool for analysis.

While PCG is known to be effective in the creation of complex 3D Worlds, its application has not yet been extended to data-driven generation for analysis. This is challenging, as it implies more than just generating 3D geometry. For its use in this kind of generation, the system must produce output that is coherent with the input data, ensuring that the original data is clearly identifiable. Atlas has focused on 2D representation of data, which proves that this generation is possible [6]. However, for 3D generation, there are new challenges to be addressed, such as the navigation within the environment and the management of large-scale datasets, where the increased amount of data can obstruct readability. Moreover, the use of three dimensions for abstract data is itself contested. It has long been criticised for introducing occlusion and perspective distortion without a clear analytical gain [9]. IA reopens this question from a different starting point, yet recent surveys observe that almost no framework supports data transformation inside the immersive pipeline and that authoring remains restricted to experts [10].

The central problem this thesis aims to address is the general lack of automated "procedural engines" for IA capable of translating structured data into interactive and semantically correct 3D environments, thereby making IA accessible and scalable for analysts to use. While the ultimate goal of this intersection is a universal framework for arbitrary data, establishing such a foundation requires first solving the mapping problem for a highly complex topology. Therefore, this thesis focuses specifically on event-based flow data. By targeting a data structure that inherently suffers from severe visual clutter and temporal occlusion in 2D formats, this work uses flow networks as the ideal, high-stress environment to demonstrate and validate the semantic capabilities of procedural generation in 3D spaces.

1.3 Research Questions

To solve the problems identified in Section 1.2, this thesis aims to answer the following Research Questions (RQ):

(RQ1) Framework: How can principles from Procedural Generation be adapted to process and structure multidimensional flow data, integrating topological attributes with optional semantic and kinematic fields, for the automatic creation of meaningful 3D immersive analytical environments?

- This question concerns the system architecture: how to design a flexible rule-based pipeline that ingests dynamic flow datasets and builds the 3D scene automatically. It is answered by the framework itself.

(RQ2) Environment Quality: Does mapping flow attributes to procedural geometry and an environmental context yield visualizations with higher diegesis than equivalent abstract 3D representations?

- This question concerns the visual design. It is assessed by an expert evaluation with the SCORE framework, checking whether the generated environment becomes part of the data presentation instead of a container for it.

(RQ3) Analytical Utility: How does a procedurally generated environmental context affect task performance and perceived cognitive load as dataset complexity grows?

- This question is answered by a controlled user study, an ablation that compares task performance and cognitive load with and without the generated environment, across two levels of dataset complexity.

1.4 Hypothesis

This thesis hypothesizes that by applying and adapting PCG techniques, it is possible to develop a framework that automates the generation of interactive 3D analytical environments directly from event-based flow data, producing data-worlds that are generated by algorithms yet remain semantically coherent and analytically useful.

Furthermore, it is hypothesized that embedding the data in a procedurally generated environment, one whose own features carry per-node quantities computed from the flows, improves the analyst's ability to identify spatial and temporal flow patterns, and does so without increasing cognitive load or breaking the sense of presence, when compared to an equivalent abstract 3D environment that lacks the generated context. The analytical channel under test is the procedural generation as a whole, not any single feature. This means that the terrain relief and the vegetation density are the two channels this work realizes, and the same mechanism could carry further quantities through additional generated features. The separable nature of this environment, which can be removed while leaving every other visual channel intact, is what makes this contribution measurable through a controlled comparison and not just asserted.

1.5 Document Structure

Beyond this introduction, the document is organized into five chapters. Chapter 2 reviews the state of the art in procedural generation and in the immersive visualization of abstract and flow data, and identifies the gaps this thesis addresses. Chapter 3 presents the proposed approach, the data-world concept, the flow data model and the mapping strategies used to turn a dataset into a navigable environment. Chapter 4 details the system that realizes this approach. Chapter 5 describes the two-part evaluation, combining an expert assessment with the SCORE framework and a controlled user study. Chapter 6 summarizes the contributions and outlines future work.

Chapter 2

State of the Art

The intersection of PCG and IA creates a new area of study regarding how to experience complex data. While traditional methods for visualizing data often rely on abstract graphics, such as charts and dashboards, immersive approaches want the users to feel presence within the data environment. This chapter reviews the technologies that make this possible, the fundamental design principles involved, and the applications currently present in the literature. It concludes by identifying the critical gaps that this thesis aims to address.

2.1 Technologies of Procedural Generation

PCG is broadly defined as the algorithmic creation of content with limited or no direct human input [5, 11], and its method families have been organized in dedicated taxonomies and surveys that distinguish, among others, search-based and experience-driven approaches [12, 13]. The capability to generate coherent virtual worlds from data relies on a specific set of algorithms that facilitate the transition from manual to automated creation. In this review, the technologies are divided into three primary categories: stochastic generation, grammar-based systems, and constraint satisfaction algorithms.

2.1.1 Stochastic Generation and Noise

Natural environments are rarely uniform. Instead, they usually present a structured form of randomness. To replicate this effect in virtual environments, coherent noise functions such as Perlin Noise or Simplex Noise are commonly used. Unlike white noise, which is completely unpredictable and results in visual chaos, coherent noise ensures that spatially adjacent points have correlated values.

While gradient noise functions are computationally efficient, they often lack the realism required for high-fidelity simulations[14]. To address this limitation, recent studies have moved beyond simple noise. For instance, methods simulating hydraulic and thermal erosion on large-scale terrains have been proposed to model river formation and sediment transport based on physical laws, not mathematical probability [15]. Furthermore, addressing the gap between low-resolution

global data and the high-fidelity expectations of VR often requires advanced methods that add footprint-level random details to topographic surfaces [16].

In the specific context of data visualization, stochastic functions serve as more than visual tools. They act as a structural framework for data representation. This is evident in projects like *Archipelago*, where geolocation history modulates the generation of virtual islands [17], illustrated in Figure 2.1. Here, noise provides the topographical baseline, while user data sculpts the specific features, effectively turning the terrain's roughness and elevation into a semantic representation of a digital footprint. Similarly, data from mobile sensors has been utilized to seed generation functions for game levels, creating terrain tracks that reflect the roughness of real-world roads [18]. These examples demonstrate that well-parameterized noise becomes a medium for visualizing statistical distributions in continuous space.

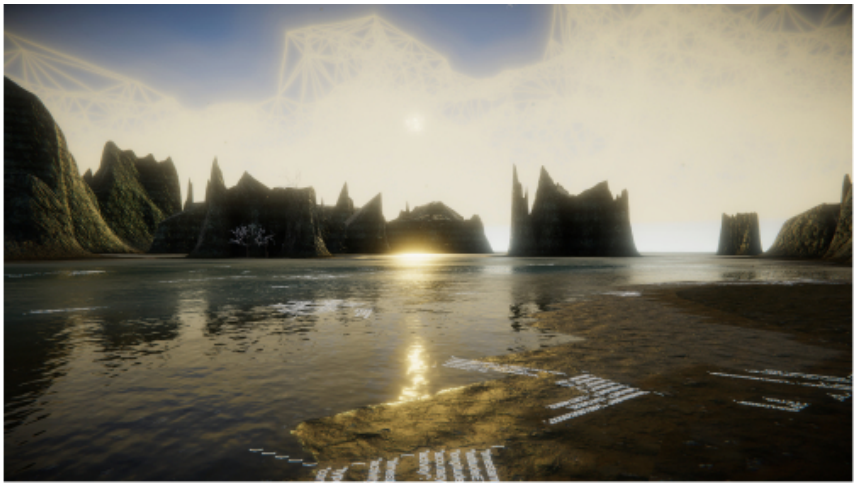


Figure 2.1: *Archipelago*: virtual islands whose topography is modulated by the user's geolocation history, so that personal movement data sculpts a navigable terrain instead of just decorating it. Source: [17].

More recently, the field has also incorporated machine learning, in particular Generative Adversarial Networks (GANs), for terrain generation [19]. Unlike explicit algorithmic design, these data-driven approaches learn the statistical features of real elevation data to create natural landscapes, such as "Stone Forest" landscapes or volcanic terrains for VR simulations, proving that procedural methods can be tuned to represent distinct geological phenomena [20, 21].

2.1.2 Grammars and Rules

While random noise is effective for continuous surfaces, the generation of structured objects, such as buildings, road networks, or layouts, requires a deterministic approach based on formal rules. Consequently, random noise becomes insufficient for translating data into meaningful structures. Technologies like Lindenmayer Systems (L-Systems) and Shape Grammars allow for the description of complex structures through iterative rewriting rules.

In urban modeling, the standard technology remains Computer Generated Architecture (CGA). Recent surveys indicate a transition in this field from purely geometric rules to semantic-aware integration [22]. CGA rules allow the automatic extrusion and texturing of buildings from 2D footprints, such that the generated architecture respects historical styles or planning constraints without the need for manual modeling [23, 24]. This parametric approach is also used in infrastructure design, where rules can adjust street widths and lane configurations in real-time to facilitate urban planning scenarios [25], shown in Figure 2.2.

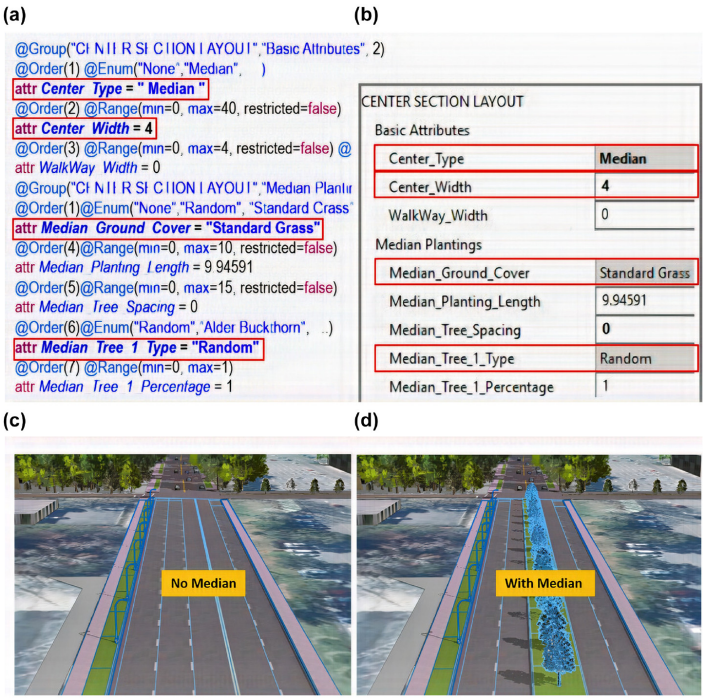


Figure 2.2: Rule-based (CGA) modeling of "complete streets": the same parametric grammar, driven by the attribute panels (a, b), produces a street without (c) and with (d) a planted central median. Source: [25].

A significant evolution in grammar-based generation is the capacity to be easily updated over time. Realistic settlements are rarely the result of a single set of rules, usually tending to evolve. Timeline-based approaches simulate the historical growth of settlements over different architectural periods, layering styles to create cities that evolve [26]. Similar shape grammars have been applied to digital architectural heritage to ensure that generated forms coherently represent specific historical eras [27, 28].

Beyond static structures, the literature presents dynamic and agent-based grammars. Swarm Grammars, for example, combine the rigor of the L-Systems' structure with the adaptability of multi-agent systems, allowing agents to build the environment as they traverse it [29]. Furthermore, the accessibility of these complex tools is being improved through systems that integrate Natural Language Processing (NLP), allowing users to describe layouts in plain language that are then translated into formal grammar rules [30].

2.1.3 Constraint Satisfaction and Tessellation

To ensure that a generated world is not only visually consistent but also coherent with the data and navigable, the intersection between random grammatical methods and constraint satisfaction algorithms is the option that the literature presents.

The Wave Function Collapse (WFC) algorithm is valued for its ability to maintain local consistency, ensuring that every element of a generated world is compatible with its neighbors. Some studies have highlighted the utility of WFC in mixed-initiative tools, where it helps reduce the workload of designers by automatically resolving adjacency constraints [31]. Other spatial partitioning methods, such as Binary Space Partitioning (BSP) trees, are used to create infinite or scrolling 3D levels, enforcing connectivity constraints in real-time [32].

In VR, navigation imposes strict physical constraints. Generated spaces must not only be coherent with the data but also structurally correct, as demonstrated by frameworks generating architecturally consistent mazes [33]. Moreover, the movement limitations of the user's play area present a challenge. Adaptive virtual environments address this by using procedurally generated portals and corridors to transform large worlds into small physical spaces [34, 35].

Graph-based methods provide another layer of structural control. Reinforcement Learning (RL) has been applied to procedural graph generation, where agents learn to construct graphs that satisfy complex structural constraints that are difficult to encode in static grammars [36]. Similarly, Minimum Spanning Trees (MSTs) are used to generate indoor level layouts, ensuring reachability and efficient corridor networks [37]. On a larger scale, tessellation techniques are used to reconstruct landscapes from remote sensing images, dividing agricultural fields or urban blocks using Voronoi-like structures to mimic natural spatial divisions [38, 39]. These structural methods are often complemented by procedural placement models that follow precise spatial limits to populate environments with vegetation in plausible locations [40], depicted in Figure 2.3.

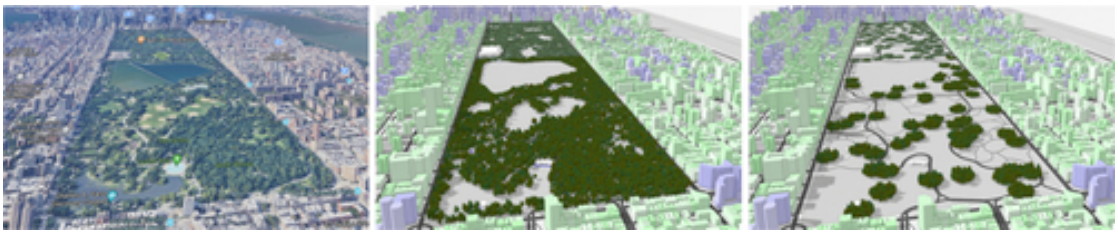


Figure 2.3: Procedural urban forestry: from a real city block (left), vegetation is placed according to spatial and ecological constraints (center) to generate a plausibly populated 3D scene (right), rather than scattered at random. Source: [40].

2.2 Design Principles in Immersive Analytics

The transition from raw data to virtual environments requires more than algorithms. It demands design principles that ensure the user can read and understand the environment while maintaining a sense of immersion.

2.2.1 Spatial Metaphors and Data Mapping

Mapping is the process of translating data variables into the sensory properties of virtual objects. In IA, this mapping expands beyond color and size to include spatial, physical, and behavioral properties.

Direct mapping strategies align the visualization structure strictly with the source dataset. For example, ecological data can be mapped directly to a virtual forest, where variables like biomass density or tree species determine the 3D geometry [41, 42]. This allows scientists to "walk" through their data. In contrast, abstract approaches generate glyphs inspired by organic forms, where multiple data types control the morphology of the object, such as the shape of diatoms [43], as can be seen in Figure 2.4. The success of these systems depends on the clarity of the metaphor. The user must intuitively understand that a larger tree or a rougher terrain corresponds to a higher data value, minimizing the need to constantly consult a legend.

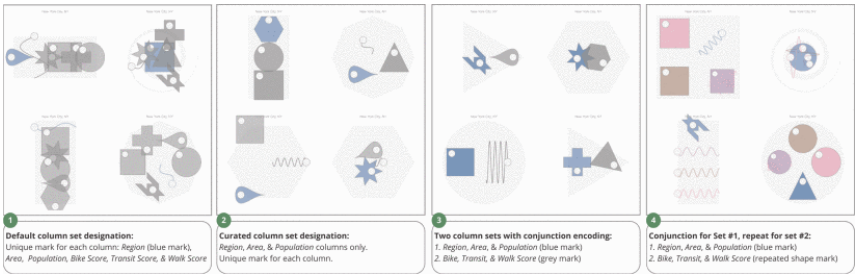


Figure 2.4: Generative glyphs in which the columns of an urban-mobility dataset (region, area, population, and bike, transit and walk scores) are each assigned a distinct mark, so that a record's combined attributes determine the form of a single composite glyph. The metaphor carries the values through shape instead of a legend. Source: [43].

Metaphor also operates at a second level, not only between data and visual object but between the author and the generator itself. Khaled, Nelson and Barr propose four design metaphors that describe the possible relationships between a designer and a procedural generator, casting it as a *tool*, a *material*, a *designer* in its own right, or a domain *expert*, each implying a different balance of control, effort and predictability over the generated result [44]. This authoring perspective bears directly on a system meant to be configured by non-programmers. The mapping the generator applies has to remain legible and controllable by the analyst, and not only by whoever implemented it.

User preference studies in immersive environments suggest a tendency toward intuitive interactions that try to mimic real-world behaviors [45], reinforcing the idea that data mappings should align with physical expectations. This is particularly important in training simulations, such as procedurally generated colonoscopy models, where the mapping is literal and the geometry must behave exactly like human anatomy to provide valid training data [46].

2.2.2 Validating Coherence and Presence

The validity of an immersive analytic environment depends on the maintenance of "presence", the sense of being there. Theoretical frameworks like Heterotopias position virtual spaces as discursive playgrounds, where the internal logic of the world is fundamental for learning [47]. If the generated environment presents logical errors, such as impossible architecture or strange lighting, the immersion is broken, the trust in the data diminishes and the interpretability becomes harder.

Interactivity is a key point in maintaining this coherence. By using game engines for historical reconstruction, users can interact with the space instead of just viewing it, validating their understanding through action [48]. The literature points out that procedurally generated environments do not inherently break presence if the generative rules are consistent with the user's expectation of that reality.

2.3 Immersive Analytics of Abstract and Network Data

A long-standing objection in the visualization community is that the third dimension is harmful, not beneficial, for data that has no inherent spatial form. Munzner argues that adding depth to abstract data introduces occlusion, perspective distortion and increased interaction cost, with little analytical gain [9]. IA revisits this objection from a different starting point. Instead of a 3D projection shown on a flat screen, the analyst is placed inside a stereoscopic environment where head motion, depth perception and direct manipulation may offset the classical drawbacks. Whether immersion actually reverses the objection is still an open empirical question, as a recent survey makes explicit when it asks how network layouts, scatterplots and parallel coordinates in immersive environments can be designed to improve upon classic desktop setups [10]. Concrete realizations of exactly these idioms already exist. Moutinho, for example, reconstructs 2D scatterplots and parallel coordinates inside a virtual reality environment, asking whether immersion eases the reading of multidimensional abstract data that has no natural spatial form [49]. As can be seen in Figure 2.5, the plots are transposed into VR yet stay suspended in space, not embedded in any environment.

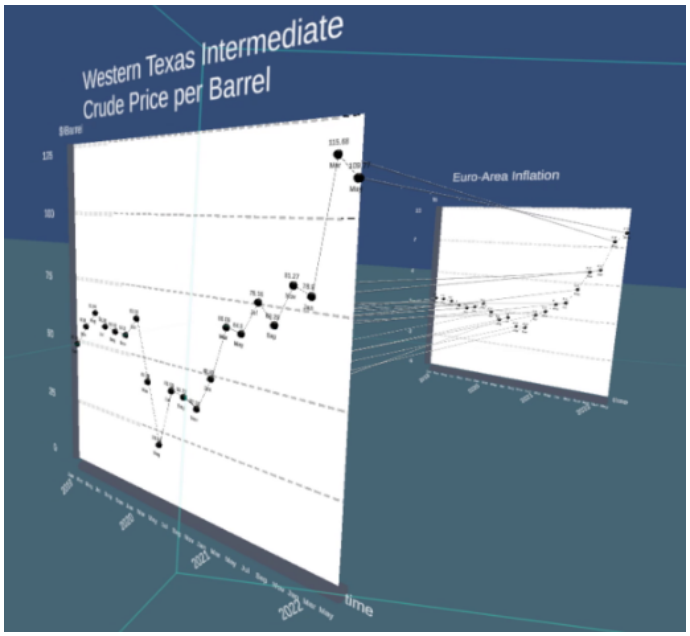


Figure 2.5: Immersive 2D scatterplots and parallel coordinates rendered in a virtual reality environment: standard abstract-data idioms transposed into VR, with the data suspended in space rather than embedded in an environment. Source: [49].

For network data specifically, the immersive literature is small but consistent. Ware and Franck showed, already in the 1990s, that viewing information networks in a head-coupled stereoscopic display lets users trace substantially larger graphs than an equivalent 2D diagram [50]. More recent work confirms and refines this result with current hardware. Kwon et al. study layout, rendering and interaction for immersive graphs and report that a spherical 3D layout, paired with a matching interaction technique, yields faster and more accurate task solving than 2D layouts as graphs grow larger [51]. Cordeil et al. compare Cave Automatic Virtual Environment (CAVE) and head-mounted display setups on collaborative graph-analysis tasks such as path finding, finding the two media broadly equivalent and favoring the cheaper, more accessible Head-Mounted Display (HMD) [52]. A common thread runs through all of these systems, however. The graph is treated as a figure suspended in empty space. The volume around and beneath the nodes carries no information, and the layout encodes only connectivity, not the magnitude or temporal behavior of the underlying flows.

A different way of giving abstract data a spatial presence is to surround it with a built environment, not a layout. In recent work within the same group as this thesis, Medina Pereira maps continuous-integration and delivery pipelines to an immersive factory, where each workflow becomes a floor and its individual runs become blocks placed upon it [53], shown in Figure 2.6. This is closer in spirit to the present proposal, since the analyst inhabits a place, not reading a diagram. The distinction, returned to in Section 2.5, is that the factory is a fixed architectural metaphor that houses the data, whereas this thesis generates the environment from the data and makes its relief carry the magnitudes.

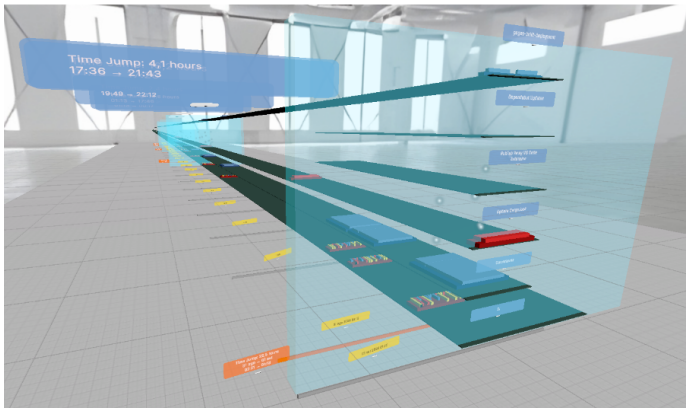


Figure 2.6: An immersive "factory" generated from continuous-integration data: each workflow becomes a floor and its runs become blocks placed along it. The environment houses abstract data through a fixed architectural metaphor, in contrast with the data-generated terrain proposed in this thesis. Source: [53].

When the relational data is also a flow, that is, directed and timestamped movement between entities, the closest immersive work places the flows on pre-existing geography, drawing origin-destination arcs over immersive maps or globes [54], illustrated in Figure 2.7. Such systems inherit their spatial frame from the real world, which is appropriate for geographically anchored data but unavailable for abstract flow datasets whose nodes are stations, accounts or network hosts, not coordinates. The broader analysis of movement data likewise assumes a spatial substrate and a temporal axis as given [55]. The question of how to generate a spatial frame, and an information-bearing environment, for flow data that carries no coordinates of its own remains unaddressed.



Figure 2.7: Origin-destination flows drawn over pre-existing geography in an immersive environment, with arcs raised above a 3D globe to separate overlapping routes. The spatial frame is inherited from the real world, which abstract flow data does not provide. Source: [54].

Finally, the same survey that motivates the immersive turn also identifies a practical barrier that this thesis targets directly. It observes that hardly any framework supports data transformation procedures in immersive environments, and calls for authoring environments that make immersive visualization accessible to non-experts without advanced programming skills [10]. Existing immersive systems tend to be built per study, by their authors, against a single fixed dataset. A configurable pipeline that ingests an arbitrary flow dataset and emits a navigable environment, with no modeling and no code, is precisely the gap between the controlled experiments above and a tool an analyst could actually use.

2.4 Existing Intersections: PCG Applied to Visualization

The application of PCG in data visualization is diverse. The current literature can be categorized into four main domains, depending on the type of input data and the purpose of the generated space.

2.4.1 Geospatial and Urban Reconstruction

This category represents the most mature application of PCG in visualization, where geographic data (GIS) generate digital cities.

Historical reconstruction projects often prioritize accuracy based on maps, using procedural generation to transform 2D data into 3D visualizations [23], depicted in Figure 2.8. Similar data-driven modeling techniques have been applied to old architectures, using semantic rules to reconstruct heritage sites with cultural fidelity [28].

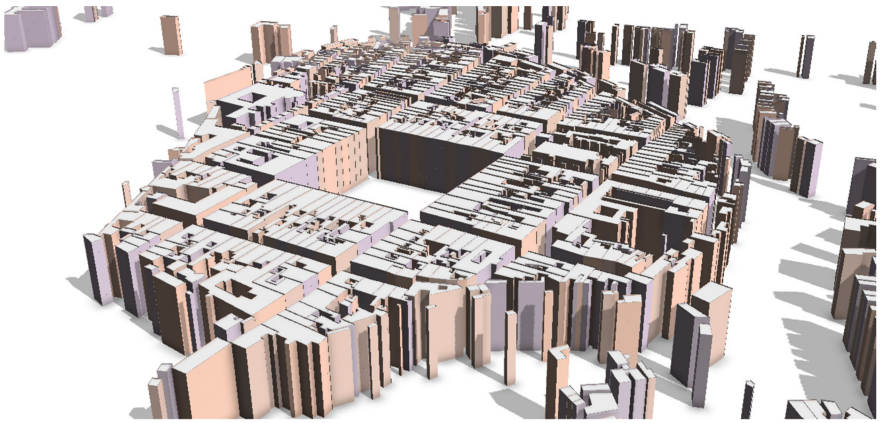


Figure 2.8: Procedural 3D reconstruction of a historical city, with buildings extruded and textured from 2D cadastral footprints. The generated space faithfully reconstructs reality but does not itself encode any further data. Source: [23].

The concept of "Semantic Cities" is born from these reconstructions, ensuring that the generated city understands its own sociological structure. Frameworks have been proposed where the semantics of a zone (e.g., commercial vs. residential) guide the geometry and the behavior of the agents [24, 56]. This layer can be extended to simulate dynamic scenes, such as virtual construction sites generated directly from OpenStreetMap data for safety training before construction [57].

Distinct from reconstruction, PCG is extensively used to generate training data. Large virtual city datasets are generated to train deep learning models for object detection, demonstrating the need for statistical fidelity where the virtual world must match the variability of the real world [58]. Efficiency is also a concern for real-time applications, where procedural methods must be optimized for limited hardware [59]. Search-based procedural generation has been used to create road networks specifically to test and break self-driving car software [60].

2.4.2 Virtual Archives and Museums

In this domain, PCG generates architectural containers for information. In these cases, the generated space serves as a canvas for selected content.

Systems have been developed to generate virtual art galleries based on user search queries [61, 62], shown in Figure 2.9. If a user searches for a specific art movement, the system creates a virtual room of appropriate dimensions to house the results. This concept has been scaled to an infinite museum capable of housing thousands of paintings [63]. While technically impressive, the architecture in these systems often remains simple, with similar corridors that do not change according to the data shown.

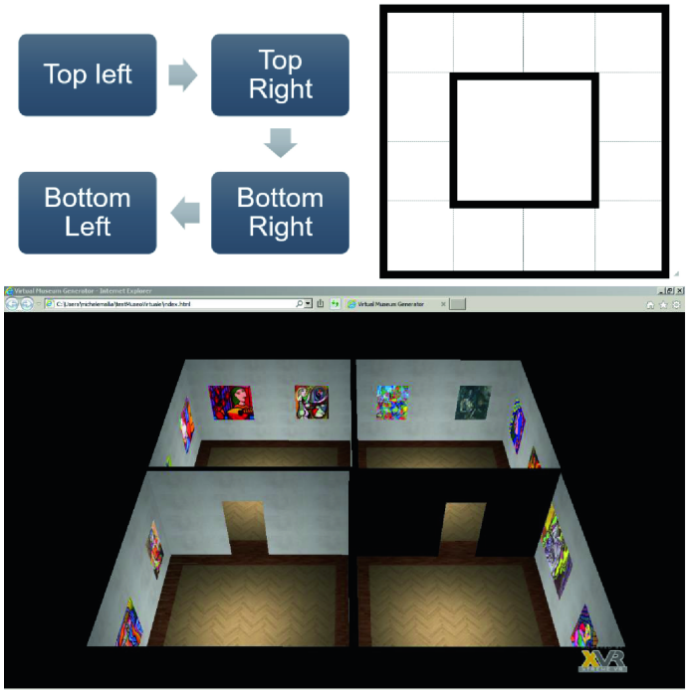


Figure 2.9: A virtual art gallery generated from a user query: the retrieved works are distributed across procedurally arranged rooms (top) to produce a navigable exhibition space (bottom). The environment is a container for the content, not a representation of it. Source: [61].

Multimedia archives in mixed reality have also been explored using spatial organization concepts to create layouts that facilitate connection between documents [47]. Variations include generating social virtual spaces that map onto real-world scanned rooms, allowing the virtual museum to adapt to the physical furniture of the user's environment [64].

2.4.3 Navigation in Solution Spaces

In these applications, PCG is combined with Artificial Intelligence (AI) to generate thousands of design variants. The visualization component serves as a navigation interface for this "space of possibilities."

Frameworks like V-Dream use VR to explore the solution space of generative design, where the user navigates through clusters of design options instead of a single world [65], illustrated in Figure 2.10. Similarly, AI tools generate procedural design suggestions for environments which the user can accept or refine [66].



Figure 2.10: A user in virtual reality reaching into V-Dream, an interface for exploring the solution space of a generative-design problem by moving among the generated options instead of inspecting a single result. Source: [65].

Hybrid approaches expand on this by proposing partnerships where human experts train the generative design assistant, creating a feedback loop that refines the solution over time [67].

These concepts are extended to the lifecycle of buildings, with "Digital Circular Twins" using visual programming to evaluate designs based on economy principles [68]. Genetic algorithms and L-Systems are also used to generate game rooms and rule sets, allowing designers to explore different gameplay possibilities [69, 70].

2.4.4 Scientific and Ecological Visualization

This category aligns most closely with the concept of this research, the "Abstract-to-World," where complex scientific data drives the creation of a natural environment.

One of the main articles presents an approach that involves visually exploring climate change models through generated forests. In these systems, every tree represents real data regarding biomass, species, and age, creating a direct mapping from scientific data to a natural immersive

environment [41, 42], as can be seen in Figure 2.11. Other procedural generation methods support this by driving plant distribution via soil wetness and sun exposure maps instead of random scattering [71].



Figure 2.11: A data-driven virtual forest in which each tree's species, size and age encode forest-inventory data, letting scientists walk through climate-change projections rather than read them from a chart. This is the work most closely related to the present proposal, a natural environment that is the data. Source: [42].

At the micro-scale, procedural generation is used to animate the dynamics of cellular microtubules based on measurement data, representing behavior that cannot be filmed directly [72]. Furthermore, there exists generation of biological models for medical training, where procedural content is strictly driven by physical constraints [46]. Other contributions offer methods to generate specific ecological niches, such as rivers or Mediterranean environments, from single images or parameters [73, 74].

In disaster management, Digital Twins are used to visualize wildfire propagation, combining real-time data with procedural terrain models to support decision-making [75]. Here, the world is a functional replica of reality, augmented by predictive data layers.

2.5 Gap Analysis

The literature reviewed above converges on three gaps, which this thesis takes as the starting point for the approach it proposes.

Dependency on spatial data. Most systems that generate a coherent world from data, whether the urban reconstructions of Section 2.4.1 or the immersive flow maps of Section 2.3, assume input that already carries geographic or physical coordinates. Frameworks that accept purely abstract, tabular flow data and synthesize a spatial frame from it, instead of borrowing one from the real world, are all but absent.

Container versus content, and the authoring barrier. Where the environment is generated, it tends to be a passive container, architecture that accommodates data without representing it. The environment itself is rarely an analytical channel. The immersive pipeline factory discussed in Section 2.3 is a recent instance, a generated building that houses the data through metaphor while its floors and walls encode no magnitude [53]. Compounding this, the immersive-analytics literature reports that almost no framework supports data transformation inside the immersive pipeline, and that authoring remains restricted to experts [10]. There is therefore a double opportunity: to make the generated environment carry the data, and to make that generation configurable by a non-programmer.

Limited generalization, and the empty environment of immersive networks. Scientific and ecological systems are typically single-purpose, tied to one data type and not repurposable. The immersive-network studies of Section 2.3, although general in method, treat the graph as a figure in empty space and never use the surrounding environment as a bearer of magnitude or temporal behavior. No general methodology combines the configurability of visualization grammars with the immersion of procedural environments to turn an arbitrary flow dataset into a navigable data landscape.

These three gaps together motivate the approach this thesis proposes to explore: generating the immersive environment as the visualization, directly from abstract event-based flow data, and exposing the whole mapping as portable configuration instead of code. Table 2.1 sets the reviewed families against the three properties the gaps concern. Several generate a coherent world and a few let it carry the data, but the combination this work sets out to explore, a carrier environment built from abstract, coordinate-free input with the mapping exposed as portable configuration, is one the surveyed families leave open.

Figure 2.12 places these families on the two axes the gaps define. The upper-right region they leave vacant is the one this thesis sets out to occupy.

Table 2.1: The reviewed families of work against the three properties the gap analysis concerns: whether the input data is abstract (carries no spatial coordinates of its own), whether the generated environment carries the data as an analytical channel instead of just containing it, and whether the mapping is configurable by a non-programmer. Each property is met by some families; the combination of all three is what this thesis sets out to provide.

Family (representative work)	Input data	Environment role	Abstract input	Carrier env.
Urban reconstruction [23]	Spatial (GIS)	Container	No	No
Virtual museums [61]	Collection/query	Container	No	No
Solution-space navigation [65]	Design space	Interface	No	No
Ecological forests [42]	Spatial/scientific	Carrier	No	Yes
Immersive networks [51, 52]	Abstract graph	Empty space	Yes	No
Immersive flow maps [54]	Spatial (geographic)	Borrowed	No	Partly
Pipeline factory [53]	Abstract	Container	Yes	No
This thesis	**Abstract flow**	**Carrier**	**Yes**	**Yes**

All listed immersive systems are configured per study against a fixed dataset; this thesis additionally sets out to expose the mapping as portable, non-programmer configuration (gap 3).

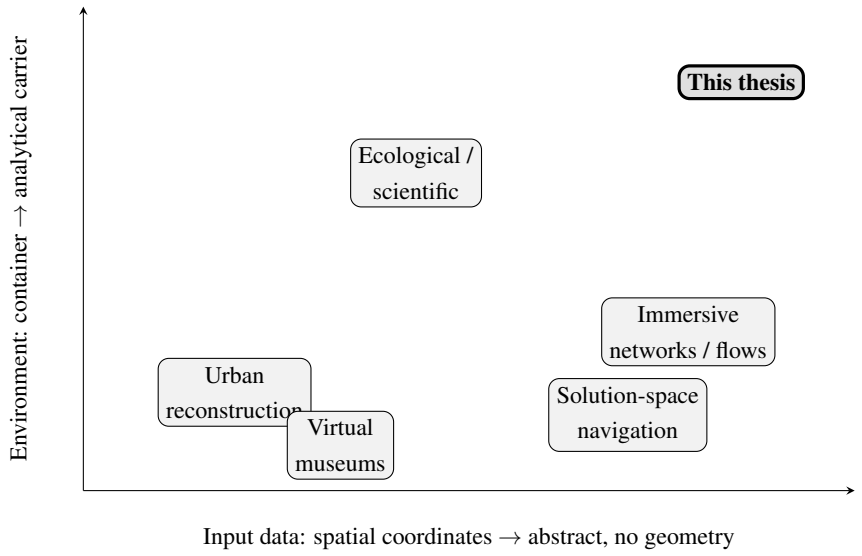


Figure 2.12: Positioning of this thesis within the reviewed landscape. The horizontal axis runs from input data that already carries spatial coordinates to abstract data with no inherent geometry (gap 1); the vertical axis from environments that just contain the data to environments that carry it as an analytical channel (gap 2). The existing families cluster away from the upper-right region; this thesis sets out to occupy it, generating an information-bearing environment directly from abstract flow data, and additionally to expose the whole mapping as portable configuration (gap 3).

Chapter 3

A Procedural Framework for Flow Visualization

The objective of this thesis, introduced in Chapter 1, is to design a framework that turns an event-based flow dataset into a navigable VR environment, built automatically from the data and configured without code, in which the procedural generation itself is the analytical channel. This chapter presents the conceptual proposal that pursues it, responding to the gaps identified in Chapter 2 and realized by the system described in Chapter 4. It justifies the decision to focus on event-based flow data, introduces the central concept of the *data-world* (an immersive environment generated from, and semantically bound to, the dataset it visualizes), and derives from that concept the visual mappings and the requirements that guided the implementation.

3.1 Scoping: From Arbitrary Data to Flow Data

The long-term ambition behind this research is a general procedural engine capable of translating arbitrary tabular data into immersive analytical environments. The gap analysis in Section 2.5, however, showed that no theoretical foundation currently exists for such a general mapping into 3D environments. Existing systems either depend on data that already carries spatial coordinates, treat the generated space as a passive container for the data, or are single-purpose tools that cannot be repurposed. Attempting such a framework directly would therefore mean designing mappings for every data topology at once, with no literature to ground any of them, which is incompatible with the rigor expected of the validation.

Instead, this thesis narrows the scope to event-based flow data, where each record is a directed, timestamped event between two identifiable entities, such as a financial transaction between countries, a flight between airports, a trip between bike-sharing stations, or a packet between network hosts. This choice is justified on three grounds:

- **Flow data is a high-stress test case.** Flow datasets combine network topology, time, direction and magnitude in a single structure, and are notorious for visual clutter and temporal occlusion in conventional 2D representations. A procedural mapping that remains readable

under these conditions provides stronger evidence for the approach than one validated on simpler topologies.

- **Flow data needs no inherent geometry.** A flow dataset is defined by its connections, not by any position in space. Some flow datasets do carry coordinates (airports and bike stations have known locations), but the approach deliberately sets them aside and generates the spatial layout from the connectivity instead. This directly addresses the first identified gap, the dependency of existing systems on pre-existing spatial data, and contrasts with the geospatial and urban-reconstruction systems of Section 2.4.1, which require coordinates the data may not have.
- **Flow data is ubiquitous.** Finance, transportation, communications and logistics all produce event-based flow data with at least a common set of features (origin, destination, timestamp, optional attributes). A single mapping may therefore serve many domains, which partially recovers the generality that the narrowed scope appears to sacrifice.

The narrowing is one of scope, not of architecture. As Chapter 4 details, every dataset-specific aspect of the system is expressed as a portable configuration rather than code, so the framework itself remains a candidate foundation for other data topologies in future work.

3.2 The Data-World Concept

The central proposal of this thesis is to treat the dataset not as content to be placed inside a virtual world, but as the seed that generates the world itself. The second gap identified in the literature, the separation between container and content (generated architecture that houses data without representing it), is addressed by inverting the relationship, so that every salient feature of the generated environment is a function of the data and the environment as a whole becomes the visualization.

In concrete terms, the analyst who enters the environment inhabits a procedurally generated landscape in which:

- the nodes of the flow network are spatially arranged so that their proximity reflects the connectivity structure of the data;
- the terrain relief carries an analyst-defined per-node quantity computed from the flows, with each node resting on a hill whose height reflects, at every instant of the playback, what is happening in the data at that moment;
- the vegetation density carries a second analyst-defined per-node quantity, independent of the relief, so that two complementary measures (typically incoming and outgoing flow) can be read at once;
- the flows themselves are visible events, small spheres with trails traveling along curves between origin and destination, sized by an analyst-defined function of the record's attributes;

- the remaining environmental layers (altitude-based biomes, water) amplify the visibility of these channels while making the scene read as a place rather than a diagram.

What carries the data is therefore the procedural generation itself, not one privileged feature. The relief and the vegetation are the two channels realized here, chosen because each maps a fundamental per-node aggregate (incoming and outgoing flow) onto a feature that is pre-attentively read as magnitude, more height and more greenery both signifying "more." The same generative machinery could in principle drive further features as additional channels – a direction left to future work (Chapter 6).

This claim is about the environment, not about everything inside it. The terrain, biomes, water and vegetation are generated from the data and make up the world the analyst moves through. The pillars that mark the nodes and the spheres that carry the flows stay abstract on purpose. Giving them a literal form, birds or vehicles for the moving flows, would suggest a meaning the data does not have, and on a dataset of network traffic or money transfers that would mislead more than help. Calling the result a data-world therefore means the world is generated from the data, not that every object in sight belongs to it.

The environment in Figure 3.1 is generated from a flights dataset, one of the datasets the system was applied to, where each node is an airport and each flow a flight between two airports.

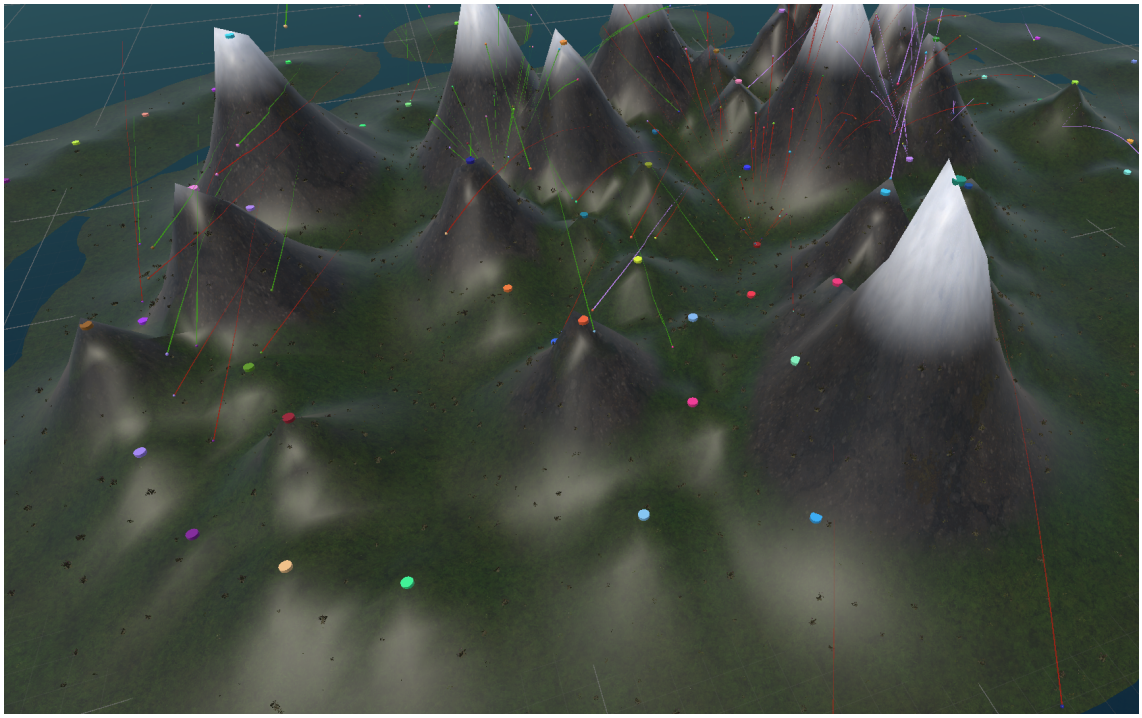
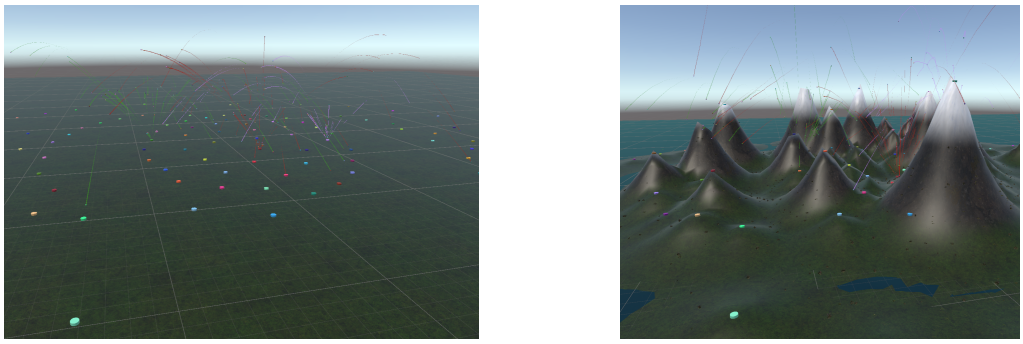


Figure 3.1: The generated data-world for a flights dataset (procedural condition). Each airport rests on a hill whose height tracks its activity at the current instant; altitude-based biomes and a water plane dress the relief; and flow spheres with trails travel along curves between airports.

The guiding intent is to give the analyst a world to inhabit instead of a chart to interpret, as Figure 3.1 shows. The motivation is not purely aesthetic. IA research argues that presence and

spatial cognition contribute to analytical reasoning [3, 8]. The ecological visualization systems reviewed in Section 2.4.4 show that natural metaphors like forests and terrain can carry scientific data legibly [41, 42]. And mappings aligned with physical intuition, where higher means more, cut the need for legends and training [45]. The terrain metaphor inherits all of these properties. Elevation is universally pre-attentively read as magnitude, and the space beneath the nodes, which a conventional node-link diagram leaves empty, becomes an information-bearing surface.

A defining property of this proposal, and the basis of its evaluation, is that the environmental context is a separable channel. The terrain (with its dependent layers, water and vegetation) can be replaced by a flat neutral plane, shown in Figure 3.2, while every other channel remains untouched. Node positions, colors, flow animation and aggregation are identical in both modes. This separability is what allows the contribution of the procedural environment to be measured empirically through an ablation design (Chapter 5), instead of being assumed.



(a) Basic (B): flat neutral plane.

(b) Procedural (P): data-driven environment.

Figure 3.2: The two environmental-context conditions, shown on the same dataset, instant and viewpoint: (a) Basic and (b) Procedural. Every other channel (node layout, colors, flows, aggregation) is identical between them; only the procedural environment is added or removed, which keeps the ablation comparison clean.

3.3 Flow Data Model

The system operates on a canonical model of event-based flow data, summarized in Table 3.1. Each record represents one directed event and carries:

- an origin and a destination node identifier (required);
- a departure timestamp (required) and an optional arrival timestamp, from which the visual duration of the event can be derived;
- an optional categorical type (e.g. carrier, transaction type);
- any number of additional attribute columns (magnitudes, scores, counts) that the analyst may reference in the visual mappings.

Table 3.1: The canonical flow-data model. Real datasets are mapped onto these logical fields by a declarative schema, without ever modifying the data.

Logical field	Required	Role
Origin	Yes	Source node identifier
Destination	Yes	Target node identifier
Departure	Yes	Event timestamp (start of the event)
Arrival	No	End timestamp; derives the event's visual duration
Type	No	Categorical label (e.g. carrier, membership class)
Attributes	No	Any number of extra columns, referenced by the mapping formulas

Real datasets never arrive in this canonical shape, and a hard premise of the approach is that the dataset is never modified. Adaptation happens entirely in a declarative schema that maps the dataset's actual column names onto the canonical fields and lists which extra columns must remain available to the mapping formulas. The timeline of the dataset, from the earliest to the latest timestamp, is normalized to the interval $[0, 1]$, which becomes the navigable time axis of the entire environment.

This model captures the common denominator of flow datasets across domains. Everything domain-specific, such as what the magnitude column means or which aggregation answers the analyst's question, is pushed into configuration, in line with the generality argument of Section 3.1.

3.4 Mapping Strategies

The translation from the data model to the generated world is organized as a set of visual channels, each with a distinct analytical role. Two principles cut across all of them.

The first is the separation between the analytical layer (what is computed from the data) and the visual layer (how the computed value is presented). Analytical quantities are defined by user-configurable formulas with clear semantics; scale transformations, normalization and legibility corrections are applied afterward, in the visual layer, without contaminating the meaning of the formula. This separation keeps formulas interpretable, since the analyst reasons about "net incoming amount," not about logarithms, while letting the system guarantee readability regardless of the raw scale of the data.

The second is that the state of the world at any normalized instant t is a pure function of t. The user must be able to play, pause and seek to any point of the timeline, forward or backward; every time-dependent channel must therefore be reconstructible from t alone, which excludes path-dependent accumulators (running averages, integrated counters) in favor of precomputed time-indexed tables.

Table 3.2 summarizes the full set, showing the data each channel carries and the visual form it takes. The last two entries carry no data of their own. The biomes make the relief easier to read

and the water gives the scene a natural setting, so both are there only to help. The sections that follow describe each channel in turn.

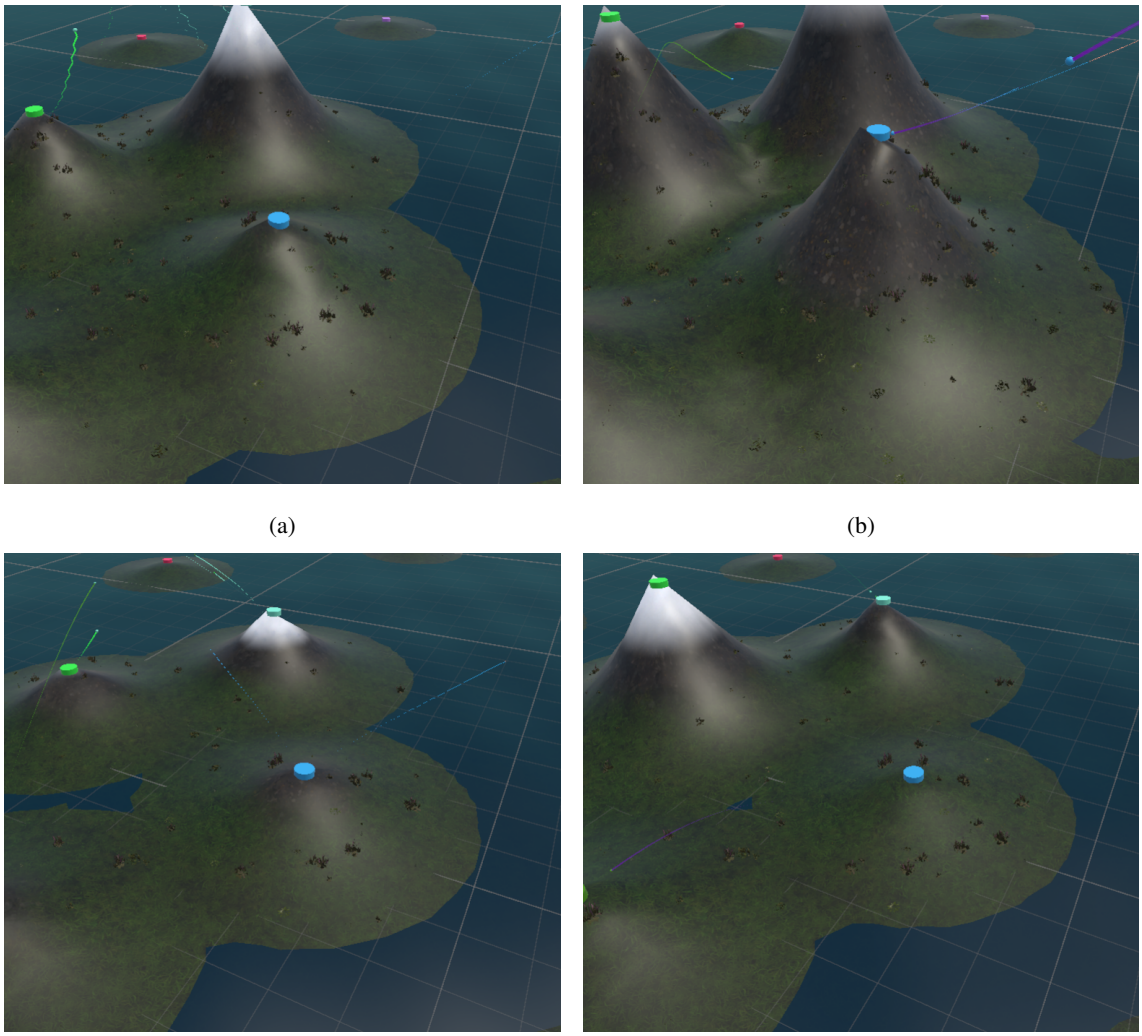
Table 3.2: The marks and channels that convey the data, following the framework of [9]. The last two rows amplify the reading and add context without encoding data of their own.

Driven by	Mark	Channel	Role
Connectivity structure	node pillar	position (XZ)	proximity shows connectivity
Node identity	node pillar	color (hue)	tells nodes apart and labels them
Incoming per-node aggregate	terrain hill	height, animated	primary analytical channel
Outgoing per-node aggregate	vegetation	density	second analytical channel
Per-flow magnitude	sphere and trail	size	flow magnitude
Origin and destination	sphere and trail	color	flow identity and direction
Altitude, from the relief	biomes	color bands	aids relief reading
Configured water level	water plane	waterline	context and natural setting

3.4.1 Terrain Elevation as a Configurable Aggregate

The terrain relief is the primary channel of the proposal, a third visual dimension that carries a per-node quantity over time. What that quantity should represent cannot be fixed in advance. For a financial dataset the question may be "how much money is entering each country?", for flights "how many landings is each airport receiving?", for another analysis "what is the difference between inflow and outflow?". The mapping is therefore defined by a formula authored by the analyst, whose named terms aggregate a chosen column over the flows entering or leaving each node, anchored to a chosen temporal reference (departure, arrival, or the in-flight interval), and combine with the four basic arithmetic operations.

Figure 3.3 shows what such a formula produces, on the bike-sharing scene of the evaluation, at four instants of one playback from a fixed viewpoint. The hill beneath each node rises and falls as its incoming trips vary, and the vegetation around the nodes follows their outgoing trips, so both quantities are read off a landscape that changes with the data. The rest of this section defines how that mapping is configured.



(a) (b)

(c) (d)

Figure 3.3: The relief and vegetation channels on the bike-sharing scene, at four instants of one playback from a fixed viewpoint (a–d). The hill beneath the central (blue) node grows and shrinks as its incoming trips rise and fall, reaching almost flat ground in (d), while the vegetation density follows each node's outgoing trips. Both channels are recomputed from the data at every instant, so the landscape itself animates with the flow.

The formula language is restricted to basic arithmetic. Sums and differences express linear combinations like totals and balances. Products and quotients express ratios and rates. More advanced operations, such as exponentials or logarithms, rarely add analytical meaning. What they provide is scale transformation, which belongs to the visual layer, where the system offers it as a separate, global option. The result is a small language that covers nearly all per-node questions an analyst asks of flow data, while remaining explainable to non-programmers.

Concretely, a formula is a small block of portable data, a list of named terms, each fixing the column to aggregate (left empty when the term simply counts events), the direction (incoming or outgoing flows) and the temporal anchor, plus a single `expression` string that combines those names. Listing 3.1 shows the relief formula used for the bike-sharing dataset of the evaluation,

where a single term counts the trips arriving at each node within a sliding time window. The `expression` grammar is intentionally minimal: term names as operands, the four arithmetic operators, parentheses, unary minus and numeric literals, and nothing else; Appendix B gives the grammar and the term structure in full, with worked examples. The full configuration file in which this formula sits is reproduced in Appendix A. The same machinery, applied to a second formula, drives the vegetation density as a complementary per-node channel (Section 4.5.3). Listing 3.2 is the companion formula used in the evaluation, identical in grammar to the relief but counting the trips that leave each node, anchored to departure, so that relief and vegetation together carry two quantities, the incoming and the outgoing flow of each node. The analytical channel is thus the generated environment as a whole, not the relief alone.

Listing 3.1: Terrain relief formula for the bike-sharing dataset. A single term counts incoming trips (`field` left empty for a count) anchored to arrival time over a sliding window, and the `expression` uses it directly as the per-node elevation.

```
"formula": {
  "terms": [
    { "name": "arrivals", "field": "", "direction": "Incoming",
      "timeAnchor": "Arrival", "windowSize": 0.05 }
  ],
  "expression": "arrivals",
  "postProcessing": "ClampToZero",
  "logScale": false,
  "normalise": true
}
```

Listing 3.2: Vegetation-density formula for the same dataset, the companion to the relief of Listing 3.1. The grammar is identical; only the term changes, counting the trips leaving each node (`Outgoing`) anchored to departure, so the foliage carries the outflow while the relief carries the inflow.

```
"formula": {
  "terms": [
    { "name": "departures", "field": "", "direction": "Outgoing",
      "timeAnchor": "Departure", "windowSize": 0.05 }
  ],
  "expression": "departures",
  "postProcessing": "ClampToZero",
  "logScale": false,
  "normalise": true
}
```

Listing 3.1 can be read in plain language. It counts every trip arriving at a node and uses that count as the node's elevation. The term leaves its `field` empty, so it counts events instead of summing a column, sets its `direction` to incoming, and anchors on the arrival time, over a sliding window of five percent of the timeline. The `expression` is the term's name on its own, so the count becomes the height with nothing in between. An analyst changes only these few fields, choosing the column to aggregate, whether to read flows entering or leaving the node, which timestamp to anchor on, and how the named terms combine under the four arithmetic operators. What the language leaves out matters as much. It has no control flow and no way to call into code, so a reader who does not program can still follow what a formula computes. The scale corrections that would otherwise blur that meaning, such as a logarithm or the normalization, are separate fields applied only at the end, once the analytical value is fixed.

A single term is the simplest formula, not the most a formula can be. Since the expression is ordinary arithmetic over the named terms, several of them combine into a quantity that no single column carries. Consider a dataset of international money transfers, where each record carries the amount moved and a money-laundering risk score, and suppose the relief should raise the nodes that behave like laundering sinks. Listing 3.3 does this with four terms over those two columns: `inflow` and `outflow` sum the amount entering and leaving the node, while `riskIn` and `riskOut` sum the risk score in each direction. The expression `(inflow - outflow) * riskIn / (riskIn + riskOut)` reads as the net money a node absorbs, multiplied by the share of its flagged risk that is inbound. A node that takes in far more than it sends, with risky activity mostly on the way in, rises highest, while `ClampToZero` flattens the net senders and the logarithm tames the long tail of monetary values. What matters here is not the particular indicator but that the four arithmetic operators and parentheses, together with numeric constants, are enough to turn columns an analyst already understands into a derived measure, with nothing else in the language.

Listing 3.3: A relief formula that combines several terms, on a dataset of international money transfers. Four terms aggregate two columns, the transferred amount and a risk score, in both directions, and the expression composes them into one derived measure: the net money a node absorbs, weighted by the inbound share of its flagged risk.

```
"formula": {
  "terms": [
    { "name": "inflow",  "field": "Amount (USD)",
      "direction": "Incoming", "timeAnchor": "Departure", "windowSize": 0.01 },
    { "name": "outflow", "field": "Amount (USD)",
      "direction": "Outgoing", "timeAnchor": "Departure", "windowSize": 0.01 },
    { "name": "riskIn",  "field": "Money Laundering Risk Score",
      "direction": "Incoming", "timeAnchor": "Departure", "windowSize": 0.01 },
    { "name": "riskOut", "field": "Money Laundering Risk Score",
      "direction": "Outgoing", "timeAnchor": "Departure", "windowSize": 0.01 }
  ],
```

```
  "expression": "(inflow - outflow) * riskIn / (riskIn + riskOut)",
  "postProcessing": "ClampToZero",
  "logScale": true,
  "normalise": true
}
```

3.4.2 Flows as Animated, Sized Events

Each record (or aggregated group of records) is materialised as a sphere with a trail traveling along a curve from origin to destination during its temporal window. Its size is governed by a second analyst-defined formula, simpler than the terrain's since it evaluates over a single record with no direction or temporal anchoring, so that magnitude reaches the analyst's eye directly. A large, risky, direct transaction can be made visibly larger than the routine majority. Because many real flow datasets are heavily long-tailed, the visual layer offers logarithmic normalization, without which the typical record would appear far too small to see next to the extremes. Listing 3.4 is the size formula used in the evaluation, a single term counting the trips merged into each aggregated sphere, log-normalized and remapped to a bounded size range. It reuses the same minimal grammar and term structure as the terrain (Appendix B), dropping only the direction and temporal anchor, since a flow formula evaluates over a single record.

Listing 3.4: Per-flow size formula for the bike-sharing dataset. A single term counts the trips aggregated into each sphere (`field` left empty for a count), the `expression` uses it directly, and the result is log-normalized and remapped to the size range `[outputMin, outputMax]`. Unlike the terrain formula it has no direction or temporal anchor, since it evaluates over one record.

```
"visualFormula": {
  "terms": [
    { "name": "trips", "field": "" }
  ],
  "expression": "trips",
  "postProcessing": "ClampToZero",
  "normalisation": "Log",
  "outputMin": 0.4,
  "outputMax": 5.0
}
```

Dense datasets pose a structural problem that no per-record encoding survives. Hundreds of simultaneous spheres are computationally renderable but cognitively intractable, and quantised timestamps produce artificial synchronized bursts. The proposal adopts visual aggregation, merging records that share the same directed pair and the same temporal window into one larger sphere whose formula inputs are summed over its constituents. Aggregation is preferred over filtering or sampling because it preserves all information, with every record remaining represented, while

reducing the number of visual objects to what the scene and the viewer can sustain. An established alternative, common in two-dimensional node-link visualization, is edge bundling, which routes spatially similar connections together to reduce line crossings [76]. The present approach instead reduces clutter by aggregating events in time and by exploiting the third dimension, so that individual connections remain legible as discrete moving events instead of merging into static bundled lines.

3.4.3 Identity Through Color and Form

The spatial arrangement of the nodes is produced by a force-directed layout [77], the standard approach to drawing node-link data, so that the planar distance between two nodes reflects their connectivity, not any external coordinate; this is the same family of layout shown to support graph reading in immersive settings [51, 52]. Node identity is communicated by color, from a deterministic palette that guarantees perceptual distinctness across dozens of nodes and assigns the same color to the same node in every session. Flows reuse the same palette, with the sphere taking the destination's color and the trail the origin's, so that the question "where is this going, where did it come from?" is answered by the same visual vocabulary that identifies the nodes themselves. Nodes are rendered as uniform low pillars, categorically distinct from the moving spheres, and carry no data encoding of their own. Activity is communicated exclusively by the terrain beneath, which keeps the ablation comparison clean.

3.4.4 Time as a Navigable Axis

The timeline is navigated, not just played back. The analyst scrubs to any instant and the whole world (terrain heights, in-flight spheres, their progress along their curves) reconstructs the state corresponding to that instant. This requirement, combined with the pure-function-of-time principle, shapes the entire technical design (Chapter 4): per-node activity is precomputed into time-indexed lookup tables at load, and the set of in-flight events at any t is recomputed on demand instead of accumulated.

3.5 Requirements

The concept above, together with the constraints of VR and of the planned evaluation, yields the following requirements for the system:

1. **Dataset independence.** The system must accept arbitrary flow datasets, with any column naming and any timestamp convention, through configuration alone, never through code changes or dataset edits.

2. **Configurable semantics.** The quantities communicated by the terrain and by the flow sizes must be definable by the analyst, per dataset, as formulas over the data.

3. **Free temporal navigation.** Play, pause and seek to any normalized instant, in either direction, with the full scene state reconstructed deterministically.

4. **VR-grade performance.** Stable frame rates suitable for head-mounted displays, which calls for minimizing runtime allocation, per-frame geometry rebuilding, and object instantiation during playback.

5. **Clean ablation.** The environmental context must be removable atomically, leaving every other channel untouched, to support the comparative evaluation.

6. **Cross-channel coherence.** Colors, positions, sizes and durations must stay mutually consistent, so that the same node always has the same color everywhere and node positions follow the terrain as it animates.

7. **Non-programmer configuration.** All configuration must be expressible as portable, plain data (a JavaScript Object Notation (JSON) file accompanying the dataset), so that the system can be operated outside the development environment.

These requirements are not independent. Free temporal navigation and VR-grade performance jointly force the precomputation strategy, while clean ablation and cross-channel coherence force a single point of truth for ground height and for color. Chapter 4 describes how the architecture satisfies them simultaneously.

3.6 Summary

This chapter set out the framework this thesis proposes, one that turns any event-based flow dataset into a navigable VR environment in which the procedural generation itself is the analytical channel, with its relief and vegetation carrying the data, a data-world and not a data container. The scope was narrowed to flow data to ground the mappings in a concrete and demanding topology. The mappings keep the analytical meaning of a value separate from how it is shown, and the environment can be removed without touching any other channel, so that its contribution to analytical performance and cognitive load can be measured instead of assumed. The next chapter describes the system that realizes it.

Chapter 4

Implementing the Data-World

This chapter describes the system that realizes the approach proposed in Chapter 3. It covers the architecture and its structuring decisions, the technological platform, and the implementation of each layer of the generated world: data ingestion, spatial layout, the procedural terrain with its biomes, water and vegetation, the flow visualization, and the interaction model. It closes with the performance engineering that makes the combination viable in VR and with the empirically observed scalability limits of the approach.

4.1 Architecture

Three demands shape the architecture, and the rest of the design follows from how each is met. Time has to be freely navigable in both directions, so that the user can seek to any instant without artefacts left over from how that instant was reached. The system has to be configurable without writing code, by an analyst who is not a programmer. And its subsystems have to be replaceable in isolation, so that one part can change without rewriting the others. Each demand is answered by one of the three decisions described below, and the post-load pipeline they produce is summarized at the end of the section.

4.1.1 Timeline and Playback

A single component, the timeline orchestrator, owns the playback cursor. Every time-dependent system (terrain animation, flow spawning, visual aggregation) listens to this component instead of keeping its own clock. The decisive consequence is that the scene state at any instant is a pure function of normalized time. Seeking backward works for the same reason seeking forward does, because no subsystem holds hidden path-dependent state that would need unwinding. In practice this excludes whole classes of technically simple solutions, such as moving averages, accumulated counters, or anything else whose value depends on how the user reached the current instant, in favor of precomputed time-indexed tables and on-demand reconstruction.

When the user seeks, the orchestrator resets the cursor, returns all active visual instances to their pools, recomputes which events should be in flight at the target instant and at what progress

along their curves, and rebuilds the world state for that instant. The result is deterministic and identical regardless of the path the user took to get there.

4.1.2 Configuration Model

Every structure describing the system's behavior (the column schema, the terrain formula, the flow-size formula, aggregation and layout parameters) is modelled as plain data: numbers, strings, lists, with no references to scene objects or engine types. This makes two equivalent configuration interfaces possible: the Unity Inspector during development and testing, and a JSON file accompanying the dataset for use outside the development environment. Both edit the same internal models, and the JSON file is authoritative. Any value it defines overrides the Inspector, and a missing or invalid file halts the pipeline with a concrete error. Because the structures were designed as pure data from the start, adding the JSON loader required no refactoring.

4.1.3 Subsystem Communication

The subsystems of the world (terrain builder, flow renderer, spline cache, node factory, ablation controller, terrain animator) do not reference each other directly. They communicate through central domain events (dataset loaded, layout complete, terrain toggled) and through one coordinator that owns the order of post-load operations. Replacing a subsystem, for example migrating the flow renderer from pooled objects to Graphics Processing Unit (GPU) particles, requires only that the new implementation subscribe to the same events. The post-load ordering is concentrated in a single auditable place, not scattered across implicit dependencies.

4.1.4 Pipeline Overview

At startup the configuration layer reads and validates the JSON file and writes its values into every component. An invalid or missing file halts the run here (Section 4.1.2). The parser then streams the unmodified Comma-Separated Values (CSV) once through the schema, validating the required fields and producing an immutable record set with the timeline bounds, and announces dataset loaded. On that event the coordinator runs a fixed ordered sequence: force-directed node layout, baking the per-node activity table and evaluating the relief at t=0, building the terrain mesh once, projecting node heights onto the surface, instantiating the node pillars with their persistent colors, and building one flow curve per directed pair. The same event also triggers, in parallel, the bakes that do not depend on that order, the flow renderer preparing its flow-size table and aggregation bundles, the vegetation system its density table, and the orchestrator sorting the events by departure. The coordinator then announces layout complete, the viewer rig, node interaction and water are placed, and playback begins.

At runtime the orchestrator is the single clock. Each frame it advances normalized time, spawns from the pool whatever events enter their temporal window, and announces the new time. The terrain animator, throttled, then evaluates the activity table at that instant, rewrites vertex heights and normals, snaps the node heights, and refreshes the spline and vegetation heights to

match. The ablation toggle can fire at any time, on which the controller swaps the terrain for a flat plane, announces terrain toggled, and the layout reprojects the node heights while the terrain, water and vegetation update accordingly. Figure 4.1 shows the five stages and the three events that connect them.

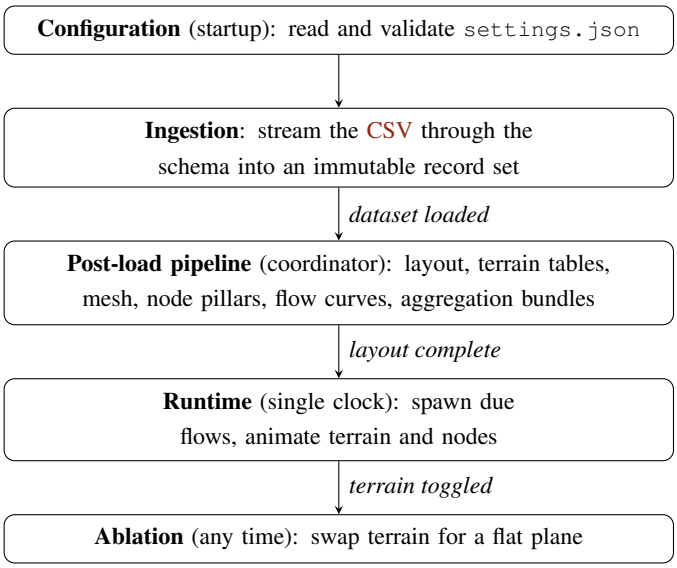


Figure 4.1: The pipeline as built: its five stages and the three domain events (in italic) that are the only cross-system coupling. Configuration and ingestion run once, the post-load pipeline prepares every time-indexed structure, and at runtime a single clock drives flow spawning and terrain animation as pure functions of normalized time. The stages are described in the text.

4.2 Technologies and Platform

The prototype is built on the Unity engine with the Universal Render Pipeline (URP), targeting OpenXR head-mounted displays, in practice the Meta Quest 3 family driven from a desktop computer. VR interaction is built on the XR Interaction Toolkit, and rendering uses Single-Pass Instanced stereo to halve the per-eye draw cost. Flow trajectories are represented with Unity's Splines package, sampled analytically at animation time. The configuration files are parsed with the Newtonsoft.Json library, chosen over the engine's built-in serializer for its fuller support of the nested objects and lists the settings format uses. All shading specific to the system (terrain biomes, water, flow trails) is implemented as custom URP shaders, since the stock materials proved either too costly or semantically wrong for these uses (Section 4.6.3).

The choice of a game engine over a visualization toolkit is a direct consequence of the proposal. The system generates a world, with terrain meshes, lighting, vegetation and locomotion, and game engines are the mature technology for exactly that combination, while still allowing every generated element to be driven from data.

4.3 Data Ingestion and Configuration

The dataset is a CSV file that is never modified. A declarative schema maps its actual headers to the canonical fields (origin, destination, departure, optional arrival and type) and lists the pass-through columns that must remain available to the formulas. Timestamp parsing accepts ISO 8601 and common formats automatically, an explicit format string for unusual cases, and Unix epoch seconds, the latter added for network captures whose fields are epochs. Relative times (seconds since the start of a capture, with no absolute anchor) are not supported. Fabricating an anchor would be an arbitrary decision, and one candidate dataset was discarded for this reason.

Each dataset is described by its own settings file, and switching datasets is a file swap, with no scene or code changes. This is the operational proof of the dataset-independence requirement.

4.4 Spatial Layout and Node Representation

Since flow data carries no inherent geometry, node positions are computed with a Fruchterman–Reingold force-directed layout [77] over the directed-pair graph, scaled to a configurable world radius. Force-directed placement makes proximity meaningful, since densely connected nodes gravitate together, and that is the property the terrain metaphor builds on. The graph dispersion is proportional to the world size, so enlarging the world genuinely spreads the nodes apart instead of leaving a central clump in empty space.

Nodes are rendered as low, wide cylindrical pillars of uniform size. The shape was chosen for two reasons. First, the flows traveling between nodes are spheres; making nodes spheres as well saturated the scene with one shape, forcing the viewer to consciously classify each object. The squat pillars separate the two categories pre-attentively. Broad static markers are nodes, and compact moving objects are flows. The pillars were flattened from an earlier tall, thin form, which was easy to miss against the relief and read ambiguously against the upright flow trails. Second, the uniform size is an experimental-design decision. An earlier version scaled nodes by accumulated importance, but that was a static parallel channel that kept encoding activity even in the ablation condition with the terrain off, polluting the comparison. Activity is communicated only by the terrain, and identity by color and an on-demand label, shown in Figure 4.2.

These markers, like the flow spheres and their trails, are intentionally non-diegetic. They are not meant to blend into the landscape but to stand apart from it, which follows from the framework's aim of remaining abstract enough for any flow dataset. A fully diegetic encoding would have to dress the data in concrete real-world objects, and that commitment risks suggesting properties the data does not carry. Keeping the data-bearing elements visually distinct, even at a cost to the immersive coherence that the expert assessment of Chapter 5 rewards, keeps the representation honest and dataset-independent.

Node colors come from a deterministic palette combining golden-ratio hue stepping, which covers the hue spectrum as uniformly as possible for any number of nodes, with a rotation through three saturation/value tiers (vivid, dark, pastel). The tiers were introduced because hue stepping

alone produces colors of identical visual energy that become hard to tell apart beyond a few dozen nodes. Assigned colors are persisted in a registry, so the same node identifier receives the same color in every session, and an analyst returning to a dataset recognizes the nodes studied before.

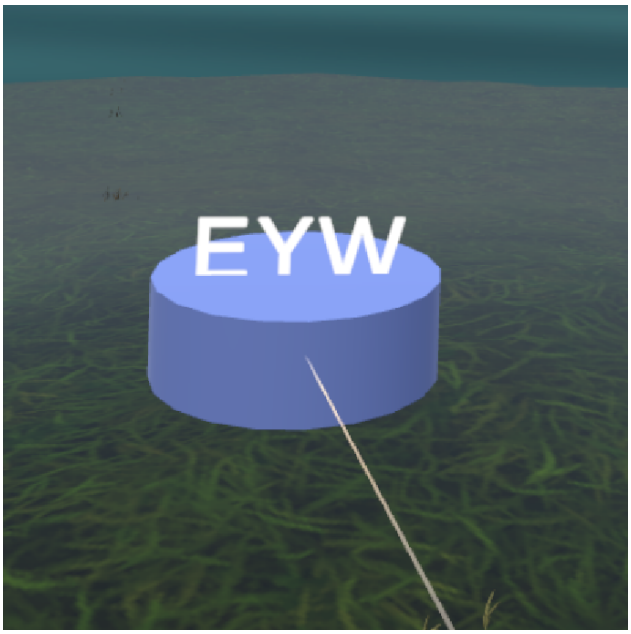


Figure 4.2: A node rendered as a uniform vertical pillar with its on-demand label revealed on hover. The pillar form is categorically distinct from the moving flow spheres, and labels are otherwise hidden, keeping the scene free of permanent text.

4.5 The Procedural Terrain

The terrain is the primary analytical channel, and building it raises a difficulty the other layers avoid. It must exist as a complete 3D mesh, yet reshape itself continuously as the timeline advances and each node's underlying quantity rises and falls. Rebuilding the mesh on every frame would be both too costly for VR and wasteful, since the vertex grid and the triangle connectivity never change and only the heights do. The whole terrain pipeline is organized around that fact. The expensive and fixed parts (the vertex grid and the triangle topology) are built once when the dataset loads, while the per-tick update touches only the vertex heights and their normals, fed by a per-node activity that has itself been precomputed for the entire timeline.

4.5.1 From Formula to Relief

The quantity carried by the relief is defined by the analyst as a formula (Section 3.4.1), built from named terms, each aggregating a column over incoming or outgoing flows with a temporal anchor, combined arithmetically. At load time, the system makes a single pass over the records and builds, for each term, a lookup table indexed by node and by discretized time interval (512 intervals by default, covering the normalized timeline). During playback, updating the terrain reduces to one

table read per term per node, the arithmetic evaluation of the formula, and a height update. This cost is independent of the dataset size and proportional only to the number of nodes and terms.

Raw formula values arrive in arbitrary scales (millions of currency units, small counts, signed differences), while the terrain works in a fixed range of world meters. After evaluation, values are normalized against the global maximum of the formula over the whole dataset, computed once, so that hill heights remain visually comparable across instants. Negative values are clamped to zero by default (such nodes read as flat ground), with absolute value as an alternative. True valleys below ground level are identified as a possible extension. Scale options (linear or logarithmic) are applied in the visual layer, after the formula, preserving the separation between analytical meaning and presentation.

One legibility correction lives explicitly in the visual layer. When the formula produces values spanning a wide range, secondary peaks can normalize to near-invisibility. A configurable minimum visual peak height compresses the output range upward so that every active node produces a discernible hill, while relative ordering is preserved. The analytical values themselves are never altered.

4.5.2 Peak Shape

Each node sits on a finite-support peak, and overlapping peaks combine by taking the maximum, not the sum, of their contributions. A sum of smooth contributions, as in the initial summed-Gaussian implementation, produces plateaus between nearby nodes that hide each node's individual value. The maximum of truncated peaks keeps every hill a separate entity, with a visible valley around it, as illustrated in Figure 4.3. A configurable falloff exponent controls peak sharpness, with higher values deepening the valleys between neighbors at some cost in naturalism. Both peak height and base radius are expressed as fractions of the world size, not absolute meters, so the relief scales with the space it occupies and remains readable when the world grows.

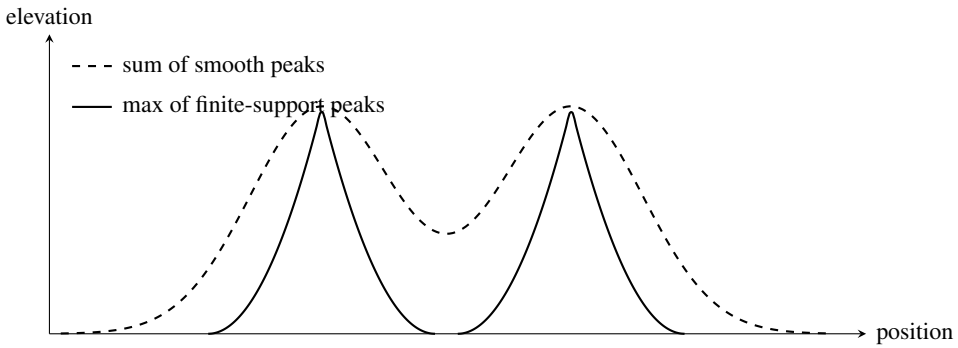


Figure 4.3: Why overlapping peaks combine by maximum, not sum. For two nearby nodes, summing smooth contributions (dashed) merges them into a single plateau that hides each node's individual value, whereas taking the maximum of truncated, finite-support peaks (solid) keeps each hill a separate entity with a clear valley between them.

4.5.3 Biomes, Textures, Water and Vegetation

The terrain geometry carries a per-node quantity, and the biome, texture and water layers above it amplify its readability and make the environment read as a place. The vegetation is the exception among these layers. As described below it carries a second per-node quantity in its own right, so it is a data channel and not dressing.

Biomes. A custom shader blends grass, rock and snow as a function of altitude, with smooth transitions, plus a slope rule that renders sufficiently steep surfaces as rock regardless of altitude. The green, gray and white triplet is read as relief without legend or training, as can be seen in Figure 4.4. A heat-map alternative was considered and rejected because warm colors carry emotional associations that could be misread as analytical meaning. The altitude bands are measured against the highest peak currently present as the terrain animates, not against a fixed ceiling. Since isolated instants rarely approach the theoretical maximum of the formula, a fixed reference left the upper bands (especially snow) almost never reached. Anchoring to the instantaneous maximum guarantees the full spectrum is always distributed across the visible relief. The most active node of each moment carries a snowy summit, intermediate peaks read as rock, and the comparison between peaks becomes categorical (color), not just geometric. The accepted trade-off is that in instants where all peaks are uniformly low, modest elevations can reach the snow band. With realistically unequal datasets this does not occur at analytically meaningful moments.

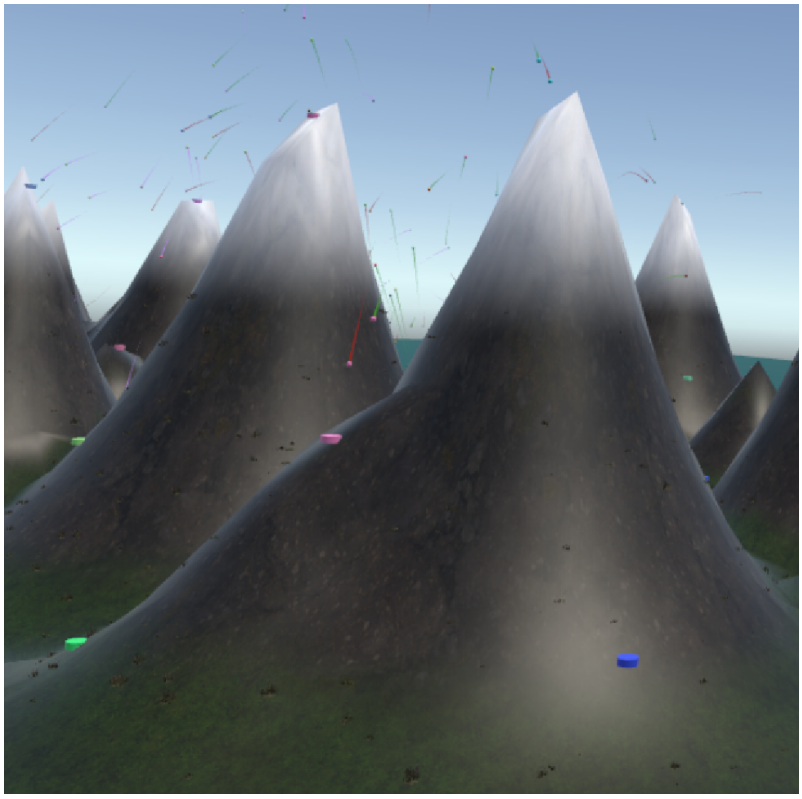


Figure 4.4: Altitude-based biomes on the relief: grass at the base, rock at mid altitudes and snow at the summit, with smooth transitions. The bands are measured against the tallest peak currently present, so the full spectrum is always distributed across the visible relief.

Flat colors read as artificial in VR, so each band carries two chromatic variants mixed by per-pixel procedural noise at multiple scales. Optionally, each band can also receive a real texture image (grass, rock, snow), projected in world space and tiled at a physically meaningful rate. The texture multiplies the band color, so an unlinked slot degrades gracefully to the flat-color behavior. Swapping a landscape's look is thus a configuration operation of linking image files, never a shader edit.[1]

Water. The peaks emerge from a nearly opaque animated water plane slightly above ground level. Since the inter-node ground is built flat at height zero, most of the world submerges and only the active peaks emerge, as islands rising from the sea, with the waterline as a natural metric reference. The water is kept nearly opaque on purpose. A translucent first attempt let the submerged plain show through and read as two superimposed layers. The wave motion is purely a shader effect (analytic normal perturbation from two scrolling sine waves) over a single static quad. Water is disabled together with the terrain in the ablation condition, because it indirectly encodes the presence of relief and leaving it on would contaminate the comparison. It also disables itself automatically when the formula mode preserves negative values, whose valleys the plane would intersect.

Vegetation. Foliage elements (grass tufts, ferns, shrubs) are scattered on the lower band of each hill, the same band the biome shader paints as grass, depicted in Figure 4.5. The placement satisfies several constraints at once. It sits above the waterline. It occupies the band that moves least as peaks grow and shrink. And it coexists with sharp peaks through a profile that widens the skirt while keeping the summit pointed. Each element is re-seated on the surface at every terrain update and hidden when it leaves the valid band. The amount of foliage visible at each node is driven by its own formula, authored in the same grammar and with the same options as the relief (a chosen column aggregated over incoming or outgoing flows, with a temporal anchor), and evaluated at the cursor each tick so it follows the data over time rather than being fixed. It is thus not bound to the relief. Given the same formula it echoes the terrain, given a different one it shows a complementary quantity. The evaluation configuration uses the latter, the terrain carrying each node's incoming flow and the vegetation its outgoing flow, so arrivals and departures are read side by side. A candidate pool is sized once per node from the formula's timeline maximum, and the per-tick count reveals a subset of it, so the foliage animates without any runtime instantiation. Distribution is anchored to each peak rather than to a global disc. An earlier global-scatter attempt produced a visible ring at the disc boundary and bald circles around nodes, immediately read as machine-generated, and was rejected.

[1] The grass, rock and snow base-color textures, and the foliage models (grass tufts, ferns, shrubs) used by the vegetation layer, are taken from Unity's Terrain Sample Asset Pack: https://assetstore.unity.com/packages/3d/environments/landscapes/terrain-sample-asset-pack-145808.



Figure 4.5: Vegetation scattered on the lower, grass band of a hill, placed relative to each peak rather than over a global disc. Anchoring the scatter to the peak avoids the boundary ring and the bald circles around nodes that a flat disc-scatter produced and that read as machine-generated.

Integrating third-party foliage assets surfaced instructive hidden assumptions. The asset materials embedded a distance fade tuned for human-scale scenes, which made vegetation vanish everywhere except near the viewer in a kilometer-scale world, neutralised by pushing the fade threshold beyond the world extent, and the materials target a different render pipeline, which currently renders acceptably but is flagged as fragile for standalone builds. The general lesson, that external assets embed assumptions about scale and pipeline which only become visible on collision, is recorded as part of the cost of procedural integration.

4.5.4 Ablation Support

A single controller swaps the terrain (and its dependent water and vegetation) for a flat neutral plane, atomically, leaving all other channels untouched. Every height query in the system goes through one sampling point that returns zero when the terrain is off, so no subsystem needs to know which condition is active. Designing the ablation in from the start, instead of retrofitting it, keeps the experimental conditions clean (Chapter 5).

4.6 Flow Visualization

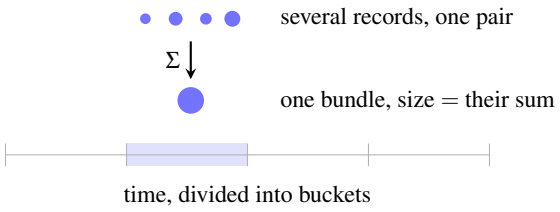
Unlike the terrain and the nodes, which persist for the whole session, a flow is transient. It exists only for the span of its event, traveling from origin to destination and then vanishing. Flows are also the most numerous moving objects in the scene and the most prone to clutter, which makes them the part of the system where legibility and runtime cost press hardest. Two constraints run through their implementation. First, nothing may be created or destroyed during playback, so flows are drawn from a pool and their entire state, including position along the curve, is reconstructed from the timeline cursor, not accumulated, which is what lets the user seek freely. Second, when many are in flight at once, each must still answer at a glance how much is moving, between which two nodes, and in which direction, without merging into an indistinct mass.

4.6.1 Size from a Per-Record Formula

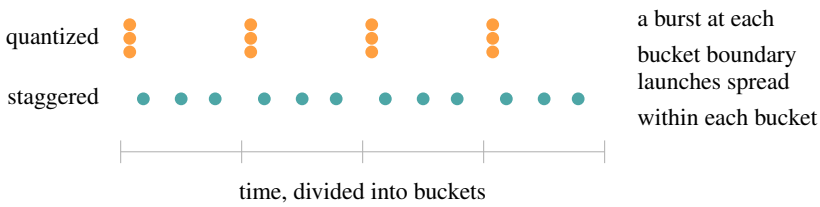
Each flow's sphere and trail width are scaled by the analyst's flow formula evaluated over the record's attributes. At load, one pass over the dataset evaluates the formula everywhere, applies the configured post-processing, and records the global minimum and maximum. At spawn, the record's value is mapped through those precomputed bounds into a configured output range. Precomputing the bounds, not adapting them to what is currently on screen, keeps sphere size comparable across the whole session. The same size always means the same magnitude. Linear and logarithmic normalization modes are offered. The logarithmic mode exists because long-tailed datasets otherwise render the typical record invisibly small next to the extremes.

4.6.2 Visual Aggregation

Records sharing a directed pair and a temporal window are merged into a single bundle, visualized as one larger sphere, with the formula's column inputs summed over the constituents. The window count is configurable, trading temporal detail against visual density, and setting it fine enough recovers per-record behavior, so aggregation degrades gracefully rather than being a separate mode. Bundle storage is sparse, holding only the populated pair-and-window combinations, and within each window the visual launch of each pair is staggered deterministically as a function of the pair itself, dissolving the synchronized-burst artefact of quantised timestamps without breaking seek reproducibility. A random stagger would make the scene irreproducible across seeks. Figure 4.6 shows both steps, the merge into one bundle and the staggered launch within a bucket.



(a) records on one pair within a bucket merge into a single bundle



(b) within a bucket, launches are staggered instead of fired all at once

Figure 4.6: Visual aggregation. (a) Records that share a directed pair and fall in the same bucket are merged into one bundle, whose size is the sum of their magnitudes. (b) To avoid a synchronized burst at each bucket boundary, the launch of each pair's bundle is offset deterministically within the bucket.

A consequence accepted by design is that a sphere's visual landing does not coincide to the millisecond with the growth of its destination hill. Both derive from the same aggregated data, but at different granularities, the hill at window level, interpolated smoothly between windows, and the sphere at the level of the staggered event. Making the relief react to each individual arrival would be both prohibitively expensive and incorrect, as it would reintroduce the bursts and break the pure-function-of-time requirement.

4.6.3 Color, Trails and Reuse

Each flow carries its endpoints' identities. The sphere takes the destination's color, constant for its whole life, and the trail takes the origin's color, fading out behind, shown in Figure 4.7. This replaced an earlier scheme that interpolated between the two colors along the path, which routinely produced unidentifiable gray mid-tones, exactly when the sphere was most visible. The destination goes on the sphere on purpose. "Where is this going?" is the analytically dominant question, while the origin remains available redundantly through the trail's direction and color.

Two presentation details required dedicated work. The stock URP material renders trail vertex colors black, because it interprets them as lighting data, not albedo. A minimal custom shader (vertex color times base color, unlit, double-sided) restored the encoding at negligible cost. And because flow spheres are pooled rather than instantiated, a reused sphere previously dragged a straight trail segment from its old position to its new spawn point, producing dozens of spurious straight lines converging on busy nodes, the scene's single worst clutter source. The fix is temporal. Trail emission is suspended before the repositioning and resumed after, with the trail history cleared, so the sphere moves with the "pen lifted".

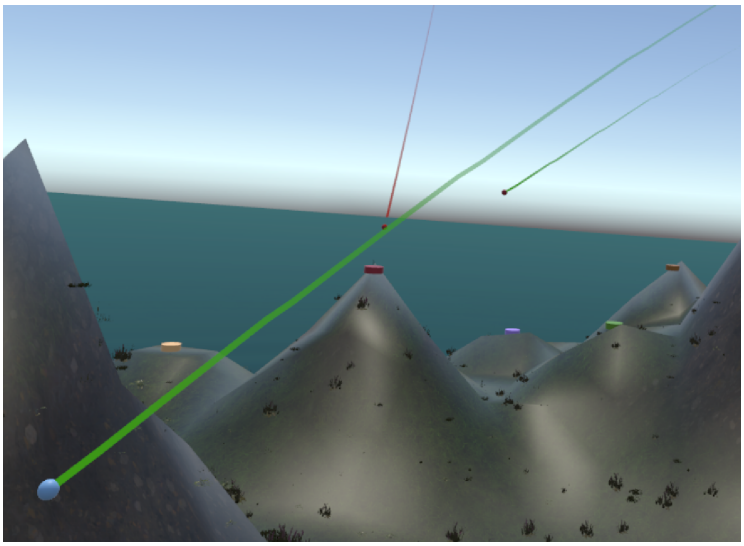


Figure 4.7: A flow in transit: a sphere in the destination's color trailed by a ribbon in the origin's color, fading out behind it, traveling along its curve between the two nodes. Sphere and trail width both scale with the flow's magnitude.

4.6.4 Trajectories over Animated Relief

Flow curves must clear any relief between their endpoints, since a curve that pierces a mountain reads as the flow passing through the terrain, and must keep clearing it while the relief animates. Each trajectory is a single cubic curve defined by its endpoints and two independent tangents. Tangent directions follow the terrain horizon seen from each endpoint. One pass over the height samples along the path finds the slope that forces the highest launch angle, and clearing that obstacle clears all the others. Tangent magnitudes are then grown until the sampled curve maintains the configured clearance, falling back to a gentle minimal arc over flat ground, as illustrated in Figure 4.8. The check samples the terrain at the current animation instant, so trajectories follow the relief as it rises and falls, and the recomputation is cheap enough to run on every terrain update for all active pairs.

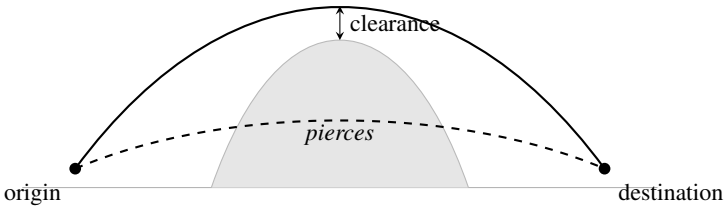


Figure 4.8: Routing a flow curve over the relief. A low arc (dashed) between two stations can pass through an intervening hill; instead, the trajectory's tangents are grown until the cubic clears the terrain horizon by a configured margin (solid). The check re-runs as the terrain animates, so the curve keeps clearing the hill while it grows and shrinks.

Two limitations are accepted and documented. With several tall scattered peaks along one path, two tangents cannot hug the relief's upper contour and the whole arc must rise; intermediate control points would be the principled fix, left out to keep curves cheap to recompute every tick. And because curves are recomputed while spheres are in flight, a peak growing under an active path momentarily bends the trajectory into a slight swerve; this is a conscious trade between always-current clearance and per-flight stability, resolved in favor of clearance, with alternatives (freezing curves at launch, continuous-function reshaping, lower recompute cadence) identified as future work.

4.7 Timeline and Interaction

The interaction design follows one position. The environment is the information, and the interface only controls system state. There are no node lists, rankings or floating numbers. This is partly because any textual readout would answer the study's tasks before the participant looked at the world, the reason an earlier panel version with a ranked node list was removed, and more basically because the thesis claims the environment itself communicates the data, a claim only testable if the environment is the sole source of information.

The default VR interaction stack was pared down. The analyst flies freely, head-oriented, with no gravity or collision and a speed tuned to cross the world in tens of seconds, because the useful analytical viewpoints, from oblique overviews to close node inspection and rapid repositioning, do not benefit from a walking metaphor. Rotation is snap-based at 15° with a short debounce, closer to perceptual continuity than the conservative 45° default, but still discrete per step, avoiding the drift sensitivity of continuous turn. On load, the viewer is placed automatically at a scale-proportional overview facing the data. The arc-teleport ray, the left controller's interaction ray, and per-hover haptic pulses were all removed. Each is reasonable in generic VR applications, but in an analytical scene with a hundred-plus interactive nodes they become persistent visual or sensorial noise.

The remaining interface is asymmetric by design. The right controller acts on the world. Its ray reveals a node's label on hover, with labels otherwise hidden to keep the scene free of text clutter, and, with the trigger, teleports the viewer to a position facing that node, so the analyst can identify, decide and jump in one continuous gesture. The left controller controls the system. One button summons a minimal floating panel (current normalized time, a scrub slider, play/pause) positioned in front of the camera, as can be seen in Figure 4.9. While it is open, locomotion is suspended, turning timeline manipulation into a deliberate pause. Pausing freezes the whole scene, including in-flight spheres, so the analyst can examine an instant in place. This asymmetric scheme currently assumes a right-handed user. The roles assigned to the two controllers are fixed, and a left-handed participant in the expert assessment found it awkward to point at nodes with the non-dominant hand, so making the assignment swappable is a small improvement left to future work.



Figure 4.9: The minimal floating control panel, summoned by the left controller: current normalized time, a scrub slider and a play/pause button, and nothing else. While it is open, locomotion is suspended and the scene is frozen, in-flight spheres included, so the analyst can examine an instant in place.

4.8 Performance Engineering

VR imposes high, stable frame rates. Combined with free temporal navigation and potentially large datasets, this is viable only through a consistent set of techniques, each already introduced above and gathered here:

- **Object pooling.** Flow spheres are drawn from and returned to a pool, and nothing is instantiated or destroyed during playback.
- **Time-indexed precomputation.** Per-node activity is baked into lookup tables at load, and per-tick terrain updates cost a fixed small number of reads and arithmetic operations per node, independent of record count. Flow-formula bounds and aggregation bundles are likewise baked once.
- **Partial geometry updates.** The terrain's triangle topology and UVs are built once per dataset load, and animation rewrites only vertex heights and normals.
- **Visual aggregation.** The number of in-flight visual objects scales with active pairs and windows, not with records.

4.9 Verification

Correctness of the system was established along two lines: properties that hold by design, and targeted runtime checks of the parts where a silent error would corrupt the analysis.

The strongest guarantee is structural and falls out of the study apparatus itself. The experiment needs a ground-truth answer for each task (which node receives or sends the most flow in a window, when a node peaks, which directed pair is busiest), and that oracle computes its answer directly from the dataset, with the same per-node, per-time-bucket aggregation of arrivals and departures that feeds the terrain table. Because the decision each task reduces to is an *argmax*, which node or pair is largest, it is invariant to the monotonic post-processing the relief applies for presentation (the optional logarithm, the normalization, the minimum-visible-height compression of Section 4.5.1). Those transformations change every hill's height but never the ranking of the nodes by magnitude. The tallest hill the participant sees is therefore the node the oracle reports. The visualization and its own correctness oracle cannot disagree, which both validates the data-to-relief mapping and supplies the experiment's ground truth from a single source. The same oracle also reports the margin between the top value and the runner-up, which flags the instants where the answer is genuinely ambiguous and operationalises the *almost correct* marking band of the user study (Section 5.3.4).

The time architecture was checked against its defining property, that scene state is a pure function of normalized time. Seeking to an instant and arriving at it by forward playback must produce the same set of in-flight flows at the same progress, since both are reconstructed from the cursor alone. This equality was verified by comparison across the two paths. Visual aggregation was checked to be total-preserving. A bundle sums its constituents, so no record is dropped, and per-pair counts reconstruct from the bundles. Finally, formula misconfiguration, the most likely operator error, is caught at load rather than failing silently. The expression parser rejects any identifier that does not name a declared term, and each term whose column never appears in the data raises a single warning, turning the otherwise mute failure modes (flat terrain, uniformly sized flows) into an explicit diagnostic.

4.10 Datasets and Observed Limits

Beyond the study datasets (Section 5.3.3), three real datasets of different domains were prepared, each adapted purely through its settings file: domestic flights (the validated reference),[2] urban taxi trips,[3] and network traffic flows.[4] The latter two were chosen as stress tests, valuable for gauging the limits of the approach because they show exactly where it stops working.

Table 4.1 characterizes each dataset by the structural metrics that decide whether it remains tractable, together with a dynamic one. The *peak flows* column reports, for the busiest time window, the raw record count and, after it, the number of flows actually drawn once aggregation collapses each directed pair into a single sphere. The gap between the two is the clutter reduction the aggregation buys: large where the data is sparse and asymmetric (flights, 2004 → 194) and absent where almost every record is its own pair (network, 551 → 551), exactly the case aggregation cannot help. The window means are lighter than these peaks. The study sets average three and twenty-six drawn flows, and the flights set one hundred and eighty.

Table 4.1: Each dataset by its structural metrics and a dynamic one. Density is directed pairs over the maximum $N(N-1)$. *Peak flows* is the busiest window's raw record count and the distinct directed pairs actually drawn after aggregation (window count 150 for the Divvy study sets, 200 for the others). The taxi figures are measured over its dominant month, January 2020; the file's sparse pre-2020 timestamps otherwise stretch the timeline. *Limit* names the scalability axis each dataset stresses, or — inside the regime the system targets.

Dataset	Nodes	Pairs (dens.)	Records	Size	Peak (raw → drawn)	Limit
Study S	16	28 (12%)	6 357	1.1 MB	198 → 9	—
Study M	59	345 (10%)	18 788	3.3 MB	527 → 75	—
Flights	107	224 (2%)	336 776	41 MB	2004 → 194	—
Taxi	262	31 027 (45%)	6.4 M	566 MB	59647 → 7265	volume, density
Network	10 000	99 928 (0.1%)	100 000	12 MB	551 → 551	node count

These metrics show that scalability is not a single axis but three independent ones:

- **Record volume** (taxis: 6.4M records, 566 MB). The pipeline loads the full file into memory, putting this at the edge of feasibility. A resource limit, and the easiest to mitigate through subsetting, plus the existing visual aggregation on the rendering side.
- **Graph density** (taxis: 262 zones, 31 027 directed pairs, 45% origin-destination density). A near-complete graph has no spatial structure for force-directed placement to reveal. Attraction forces balance in all directions and the layout collapses into a central clump, depicted in Figure 4.10. This is not an implementation defect but a mismatch between dense origin-destination matrices and the metaphor itself, which reads well on sparse or asymmetric

[2] Public dataset, available at https://www.kaggle.com/datasets/mahoora00135/flights.
[3] Public dataset, available at https://www.kaggle.com/datasets/diishasiing/revenue-for-cab-drivers.
[4] Public dataset, available at https://www.kaggle.com/datasets/saidasalloum/network-flows.

flows. Density of this kind breaks the layout only when the node count is high as well, since a graph of a few nodes stays readable however densely it is connected. A legibility limit, unaffected by subsetting.

- **Node count** (network: 100k records but 10 000 distinct nodes, ∼100k directed pairs). The scene fails to open. Layout repulsion is quadratic in nodes, terrain evaluation is linear in nodes per vertex per tick, and one curve per directed pair approaches one curve per record. A performance limit. The design assumes hundreds of nodes, consistent with the intended analytical use, since a scene with thousands of nodes would stop being readable in VR long before it stopped running.

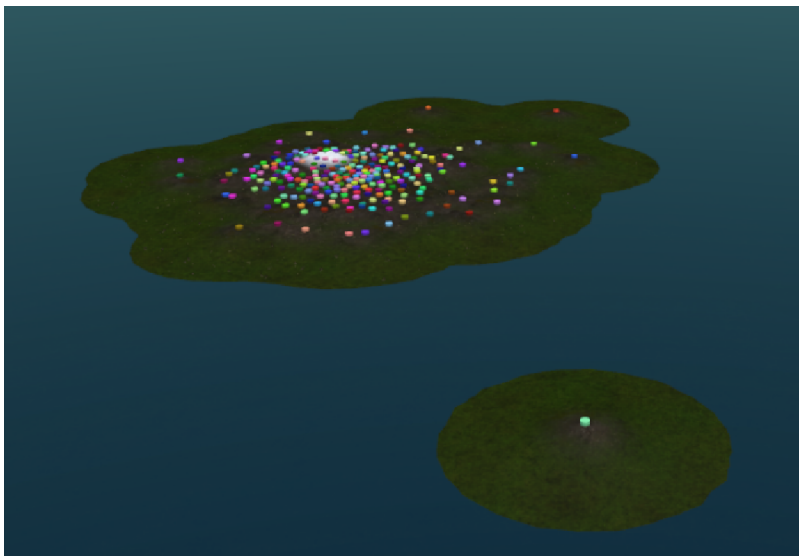


Figure 4.10: The graph-density limit on the taxi stress dataset. A near-complete origin-destination matrix offers no spatial structure for the force-directed layout to reveal, so the attraction forces balance in all directions and the nodes collapse into a central clump. This is a mismatch between dense matrices and the metaphor, not an implementation defect, and is unaffected by subsetting.

This three-way distinction matters for the evaluation. "Dataset complexity" is not a single quantity. The user study of Chapter 5 varies it within the regime where the metaphor holds, while these limits mark where that regime ends. The system is therefore not a flow visualizer that works everywhere, but one that reads convincingly on the sparse, asymmetric flows of human-scale networks.

Chapter 5

Expert Assessment and Ablation Study

The central claim of this thesis is that a procedurally generated environmental context, derived from the data itself, contributes to the analytical understanding of flow data when compared with an equivalent but abstract 3D representation. The evidence for it comes in three strands, the first establishing that the system runs comfortably in VR at all, the other two measuring the analytical value of what it generates. The user study follows an ablation logic. Instead of benchmarking the system against a different external tool, which would confound many variables at once, it removes a single channel, the procedural environment, from an otherwise identical scene, which is the comparative methodology prevalent in immersive-visualization research [10].

The three strands map onto the research questions of Section 1.3:

- **(RQ1), the framework.** How procedural generation principles can be adapted to structure flow data into meaningful immersive environments. It is answered constructively, by the system and the design rationale of Chapters 3 and 4, while the VR-performance premise on which the rest of the evaluation depends is established separately in Section 5.1.
- **(RQ2), diegesis.** Whether mapping flow attributes to procedural geometry and environmental context yields a representation of higher *diegesis*, with the environment an integral part of the data presentation and not a container for it, than an equivalent abstract scene. It is assessed by expert evaluation with the SCORE framework (Section 5.2).
- **(RQ3), analytical utility.** How the procedurally generated environmental context affects task performance and perceived cognitive load as dataset complexity increases. It is the object of the controlled user study (Section 5.3).

The hypothesis under test is graded, not binary. It holds not just that the environment helps in absolute terms, but that its contribution grows with the visual complexity of the scene. In a sparse scene every channel communicates clearly and the marginal contribution of an additional context channel is small. In a dense scene the channels compete, and the environment, being a context channel and not a point-information channel, is hypothesised to degrade less than the others. Complexity is therefore varied across dataset sizes, so that the interaction between environment and complexity can be measured alongside the main effect. The initial design used three

levels, which would additionally reveal the shape of the trend. A pilot session showed that the resulting six conditions made sessions infeasibly long, and complexity was reduced to two levels (Section 5.3.1).

5.1 System Performance

Everything the rest of this chapter measures presupposes that the system sustains the frame rates immersive comfort requires. An unstable or low frame rate is itself a confound, since it induces discomfort and degrades reading, as the pilot session made clear (Section 5.3.2). This section reports the runtime behavior of the system, substantiating the feasibility premise that the architecture of Chapter 4 was engineered to meet (Section 4.8).

Measurements are taken on a single hardware configuration, a laptop driving a Meta Quest 3 over Quest Link,[1] with the scene running in the Unity Editor in Play mode and rendered in Single-Pass Instanced stereo to the headset at its 72 Hz refresh rate. The Editor adds a fixed overhead, so the figures are a conservative lower bound on a standalone build. Each prepared dataset (Section 4.10) is loaded through its own settings file, with no other change. A lightweight probe then records four quantities over a sampling window in the procedural condition during active playback, with flows in flight and the terrain animating, the most demanding regime. These are the frame rate as the mean and the 1% low, the one-off build time from scene start to the first interactive frame, the peak engine-tracked memory, and the managed allocation per frame. The 1% low is reported because it captures the brief stutters that dominate the perception of smoothness, and the peak memory, dominated by the engine baseline, is chiefly a relative figure. The basic condition removes the terrain, water, vegetation and their per-tick updates, performing a strict subset of the procedural condition's work, and is therefore unconditionally cheaper.

Table 5.1: System performance per dataset, measured in the procedural condition during active playback. Memory is the engine-tracked peak, dominated by the baseline. The taxi and network stress sets never reach an interactive frame rate.

Dataset	Nodes	Pairs	Mean FPS	1% low FPS	Build (s)	Peak mem. (MB)
Study S (small)	16	28	71.8	67.8	2.6	1574
Study M (medium)	59	345	63.2	30.7	2.8	1421
Flights (ref.)	107	224	45.4	17.4	5.6	1393
Taxi (stress)	262	31 027	0.4	0.2	69	1315
Network (stress)	∼10 000	∼100k	0.04	0.04	193	1495

Across the study regime the system holds comfortable rates (Table 5.1). The small subset sits at the headset's 72 Hz ceiling, the medium subset averages 63 FPS, and the denser flights reference falls to 45 FPS. The frame time is almost entirely on the CPU, with the GPU near-idle at a few milliseconds, so the binding cost is the per-tick terrain and trajectory work, which grows

[1] An AMD Ryzen 7 7840HS Central Processing Unit (CPU), an NVIDIA GeForce RTX 4060 Laptop GPU (8 GB) and 32 GB of RAM.

with node and pair count, not the rendering. That the cost falls on the CPU is a property of this implementation and not of the work itself. The per-vertex and per-pair computation is data-parallel, each result independent of the others, so it could run on the GPU instead, which would be the clearest way to raise these limits and is set out as future work (Chapter 6). This is the user-visible face of the node-count limit of Section 4.10, and it matches the pilot, where comfort degraded once the scene reached the order of a hundred nodes (Section 5.3.2). The two stress datasets confirm the other two axes. The taxi set (6.4 M records) and the network set (ten thousand nodes) never reach an interactive frame rate, with build times of roughly one and three minutes and playback effectively frozen. These are the record-volume and node-count limits characterized analytically in Section 4.10.

The same pass surfaced one finding that revises a design claim. The per-frame managed allocation is not zero. It measures about 15 KB on the small subset, 80 KB on flights and 107 KB on the medium subset, tracking the number of directed pairs, not the node count. The cause is the per-tick re-routing of the flow trajectories. Every terrain update rewrites one spline per directed pair, and the knot-mutation calls of the third-party splines library allocate internally on each rewrite, while the project's own trajectory mathematics stays allocation-free. The allocation is bounded and produced no perceptible stutter at the study scale, but it means the system is low-allocation, not strictly allocation-free as the design intended. The problem was found during this final measurement pass, too late in the project to re-engineer safely, so the design was kept and the cost is documented here; removing it is set out as future work (Chapter 6).

5.2 Expert Assessment with SCORE

Diegesis, a term this thesis borrows together with the SCORE framework from narrative theory, separates elements that belong to the world being presented from elements layered on top of it. In a film, the orchestral score the characters cannot hear is non-diegetic, whereas a song coming from a radio inside the scene is diegetic. Transposed to situated and immersive visualization, a data representation is diegetic to the degree that it belongs to the environment the analyst inhabits instead of floating over it as an overlay. The data-world concept targets this property for the environment itself, the relief, biomes, water and vegetation that the procedural condition adds (Section 3.2). It does not claim full diegesis for every element. The nodes, the flow spheres and their trails stay deliberately non-diegetic markers, distinct from the world so that the representation remains abstract enough for any dataset (Section 4.4). What the assessment compares is the diegesis of the environment, between a procedural scene in which the surroundings are the visualization and an abstract scene in which the same data hangs in empty space.

SCORE [78] makes diegesis assessable by decomposing it into five dimensions, scored independently so that an environment yields a profile, not a single number. In broad terms, *spatial proximity* captures how close a representation sits to its referent; *concreteness*, how tangible and object-like rather than symbolic it is; *coherence*, how consistently it obeys the internal logic of

the surrounding world; *referential context*, the cues the environment supplies to anchor a representation to what it stands for; and *environmental context*, the degree to which the surroundings themselves participate in conveying the data. The precise operational definitions and rating procedure are those of the framework's authors and are given, as administered, in Appendix C; what matters for this evaluation is that the five-dimension profile lets the procedural and basic conditions be compared dimension by dimension, not only in aggregate.

Independently of the user sessions, the exact system configurations used in the study are assessed by a small panel of immersive-visualization experts. The assessment covers the central contrast between the procedural environment and its absence, at each of the complexity levels used in the study, verifying both whether the procedural condition scores higher on diegesis and whether the advantage persists as the scene densifies.

The assessment plays two roles in the evaluation. It answers (RQ2) directly and dimension by dimension, locating where any diegesis advantage originates. A gain concentrated in environmental context and concreteness would point to the terrain and its dressing doing the work, whereas a gain in referential context would implicate the color and labeling that the basic condition shares. And it provides an interpretive frame for the user study. If conditions with markedly different SCORE profiles produce indistinguishable task performance, that dissociation is itself informative. If the profiles track the performance differences, the two instruments agree with each other. Because the panel scores the very configurations the participants use, the two evaluations measure the same stimuli and are directly comparable.

Two methodological choices shaped the assessment. SCORE is applied by experts, not by participants. It was designed as a specialist characterization instrument, and non-specialist participants lack the technical vocabulary to judge dimensions such as coherence or referential context as the framework requires. And the application is itself a small contribution to the framework. Its authors identify extension to fully virtual environments as future work, the original being situated in Augmented Reality, and this thesis inscribes itself in that extension.

5.3 User Study Design

Turning the ablation into a runnable experiment means keeping the procedural environment as the only thing that varies, while holding off everything else that could be mistaken for its effect. Four such threats shape the design: the order in which a participant meets the conditions, the learning that carries from one condition to the next, the differences in ability and prior experience between participants, and the domain of the data itself. The design answers each in turn, with a within-subjects structure so that every participant serves as their own control, a counterbalanced presentation order, disjoint but equivalent dataset subsets at each size so that no answer can be reused between a participant's two visits, and a single dataset sampled at two scales. All of this sits under a constraint that a pilot made concrete. A session has to stay short enough to remain comfortable in a headset.

5.3.1 Factorial Structure

The study is a within-subjects 2×2 factorial design, shown in Figure 5.1. Each participant experiences four conditions, crossing:

- **Environmental context.** *Basic* (B): flat neutral plane, no relief, water or vegetation; versus *Procedural* (P): the full generated terrain, with hills, biomes, water and vegetation driven by the data. All other channels (node layout, colors, flows, aggregation) are identical between the two.
- **Dataset complexity.** *Small* and *medium*: subsets of one dataset at two scales, differing in node count and consequent visual density.

	Small (S)	**Medium (M)**
Basic (B)	Basic × Small (BS)	Basic × Medium (BM)
Procedural (P)	Procedural × Small (PS)	Procedural × Medium (PM)

Figure 5.1: The within-subjects 2×2 design: environmental context (Basic versus Procedural) crossed with dataset complexity (Small versus Medium). Each participant visits all four conditions; order is counterbalanced with a Latin square in which same-size datasets are never adjacent, so that familiarity acquired at one size does not transfer immediately to its second visit.

The design initially crossed the environment with three complexity levels, adding a large subset of over a hundred stations, for six conditions in total. A pilot session led to dropping the largest level and reducing the design to the four conditions above. The pilot and the three changes it produced are described in Section 5.3.2. With two levels the interaction between environment and complexity remains measurable; what is lost is the shape of the trend (Section 5.7).

The within-subjects choice, with each participant as their own control, is dictated by statistical power at the achievable sample size (Section 5.3.7). Condition order is counterbalanced with a Latin square in which each condition appears equally often in each position and same-size datasets are never adjacent within a participant's sequence, so that familiarity acquired at one size does not transfer immediately to its second visit. The participant count is accordingly a multiple of four.

5.3.2 Pilot Study and Design Refinement

The design was exercised end to end in a pilot session with a single participant, who ran the original six-condition version in full, the environment crossed with three complexity levels (small, medium, and a large subset of over a hundred stations). The pilot was not meant to produce performance data but to check session length, task clarity and comfort before committing to recruitment, and it changed the design in three ways.

First, the session ran to about forty-five minutes, beyond a comfortable single sitting in a head-mounted display, with the time concentrated almost entirely in the two large conditions. The individual questions were answered quickly. What was slow was reading a scene of that density to locate the answer. Dropping the large level, by joint decision with the supervisor, removed most of the duration while keeping the small-versus-medium contrast the hypothesis needs, and brought the projected session to roughly half an hour (Section 5.3.7).

Second, the large conditions exposed a performance ceiling. With over a hundred stations the frame rate fell low enough to be felt. The terrain animation and the many simultaneous trajectories at that node count exceeded the comfort budget (Section 5.1), the scene became visibly less smooth, and the participant reported the onset of nausea, the only point in the session where discomfort was noted. This both reinforced dropping the large level and confirmed, from the user side, the node-count limit characterized analytically in Section 4.10. A scene with that many nodes becomes uncomfortable in VR before it stops running.

Third, the pilot showed that in the denser scenes the correct answer was often barely distinguishable from its closest competitor, so a participant who read the visualization correctly could still name an adjacent node. A binary correct/incorrect score would have penalised sound analytical reading as if it were failure. Correctness was therefore re-cast on the three-level scale of Section 5.3.4 (correct, almost correct, incorrect), separating an imprecise answer reached by the right procedure from a genuinely wrong one.

The pilot's small and medium conditions coincide, in the same order, with the first counterbalancing sequence of the reduced design, so its task correctness and response time for those four conditions are pooled with the main study, giving twenty participants for those measures. Its binary marking maps onto the correct and incorrect levels, with no almost-correct. The questionnaires it answered under the six-condition protocol are not comparable and are excluded, leaving nineteen for the cognitive-load and sickness instruments.

5.3.3 Datasets: One Domain at Two Scales

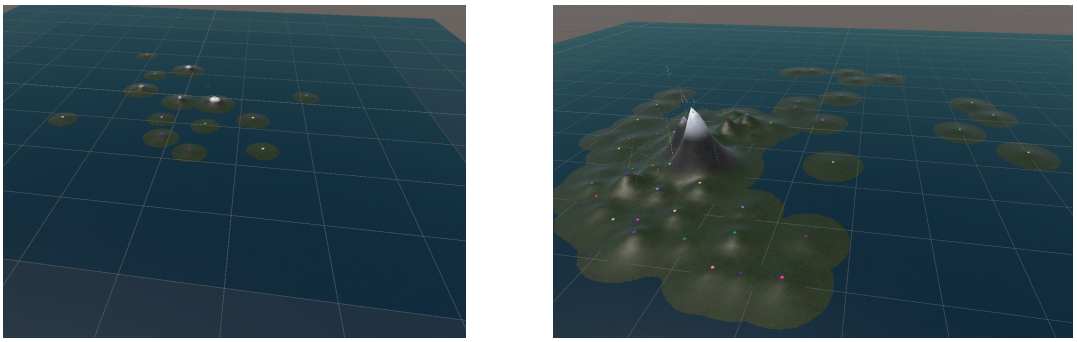
The complexity levels are two scales of a single dataset, trips from a large city's public bike-sharing system,[2] where stations are nodes and each trip is a flow, illustrated in Figure 5.2. Each scale is drawn independently by the same stratified procedure, so the smaller is not a strict subset of the larger; what is held constant is the domain, not the particular stations. Using one subsampled domain, instead of different datasets per level, removes the principal confound of the alternative design. With different domains, any observed difference would be inseparably confounded between "more complex" and "different subject". Here domain, task vocabulary and topology stay constant, and density is the only variable. The generality of the system across domains is a claim of (RQ1), demonstrated separately by its application to several datasets (Section 4.10); the performance study gains rigor by staying mono-domain.

[2]The Divvy bike-sharing system of Chicago, July 2021; public trip data available at `https://www.kaggle.com/datasets/evangower/cyclistic-bike-share`.

Station selection for each subset is spatially stratified, taking the busiest station of each cell of a grid over the city map, not simply the globally busiest stations. The distinction is consequential. The busiest stations concentrate downtown and connect almost all-to-all, a near-complete graph that collapses force-directed layout into an unreadable clump, the same degeneration documented with the taxi stress test in Section 4.10. The stratified sample preserves the neighborhood structure the layout needs, keeps both scales legible, and still guarantees each selected station enough activity for the tasks to have clear answers. This keeps the study inside the regime the system was designed for. Table 5.2 reports the two resulting subsets.

Table 5.2: The two study subsets, sampled from one month of the bike-sharing dataset by spatial stratification. Density is the number of directed station pairs over the maximum possible $N(N-1)$; round trips returning to the same station are not counted as pairs.

Subset	Stations	Directed pairs	Density	Trips
Small (S)	16	28	0.117	6 357
Medium (M)	59	345	0.101	18 788



(a) Small (S): 16 stations.

(b) Medium (M): 59 stations.

Figure 5.2: The two complexity levels, sampled from the same bike-sharing dataset: (a) Small and (b) Medium. Only the number of stations and the resulting visual density change; domain, task vocabulary and topology are held constant.

5.3.4 Tasks

In each condition the participant performs four short analytical tasks over the flow network:

- **Most incoming flow:** navigate to a specified time window and identify the node receiving the most flow within it.
- **Peak moment of a node:** focus on a specified node and identify the instant at which its activity peaks, within an adequate tolerance.
- **Busiest directed pair:** for a specified time window, identify the origin-to-destination pair carrying the most flow.

- **Most outgoing flow:** for a specified time window, identify the node sending the most flow.

These four tasks cover categories that recur in graph-visualization and spatio-temporal task taxonomies: lookups over the network's nodes (the incoming and outgoing variants), an elementary *when* query over the temporal axis, and a lookup over the network's edges [55, 79], which keeps them representative of genuine analytical operations, not idiosyncratic to this system. They also span the range over which the environment's benefit is expected to vary. Three of them bear on per-node quantities the generated environment carries, incoming flow by the terrain relief, outgoing flow by the vegetation density, and the peak moment by the relief's animation over time, while the busiest-pair task depends on the flows alone, which are identical in both conditions, and so approaches a control on which little or no environment effect is predicted. This contrast is by design, and methodologically central. If the procedural condition helps on the three channel-aligned tasks but not on the pair task, the benefit is attributable to the information the generated features carry, not to a generic novelty, engagement or spatial-reference effect that a prettier 3D scene might produce regardless of what it encodes.

Responses are scored for correctness and response time. Correctness is marked on a three-level scale: *correct*, *almost correct*, and *incorrect*, where *almost correct* marks an answer reached by the right procedure but imprecise in its specifics. The pilot showed that in denser scenes the correct answer can be nearly indistinguishable from the runner-up, and a binary criterion would penalise participants whose reading of the visualization, the very ability under measurement, is sound. The analysis may treat correctness as graded or collapse the first two levels into a binary measure, with the criterion stated explicitly. The tasks are kept simple on purpose. With a modest sample, complex tasks inflate variance and let individual characteristics (VR familiarity, visuospatial ability) dominate the experimental effect. Each task is answerable in both conditions, through hill height and animation with the terrain and through flow density and rhythm without it, possibly with different effectiveness, and that difference is exactly what is being measured.

Because each participant visits each dataset size twice (once per environment condition), reusing the same stations would let the second visit recall the first answer. Each size therefore exists as two disjoint but equivalent subsets of stations (A and B), and a participant never sees the same subset twice at a given size. The four task types are asked in every condition, but over different stations. Which subset is paired with which environment is itself counterbalanced across the Latin square, so that at sample level each subset appears equally often under each environment and any residual difficulty difference between subsets cancels in the group means. The analysis is correspondingly conducted at group level. The subsets are generated by one neutral procedure, not tuned to equal difficulty, and the counterbalancing, not any engineered parity, is what makes the comparison fair (Appendix D.2).

5.3.5 Measurement Instruments

Pre-questionnaire. Before the first condition, participants report prior familiarity with graphs and network diagrams, with data visualization, and with VR. These serve as covariates. The benefit of

the environment may plausibly differ with visual literacy, and recording it allows the results to be interpreted accordingly.

Cognitive load. The raw variant of NASA Task Load Index (NASA-TLX) [80, 81], comprising the six subscales without the pairwise weighting phase, is administered after each of the four conditions. Each subscale (mental, physical and temporal demand, performance, effort and frustration) is rated on a 0–10 scale, and the raw index is their unweighted arithmetic mean. Omitting the pairwise weighting is exactly what distinguishes the raw variant from the original. The original instrument uses a 0–100 range. It was administered here on the 0–10 scale that is the widest a Google Forms linear-scale question permits, and the values rescale to the conventional range by multiplying by ten. Per-condition measurement is essential for (RQ3), which concerns the variation of load with complexity. A single end-of-session measure could not attribute load to a level. The raw variant is validated, correlates strongly with the weighted version, and takes about a minute, making repeated administrations feasible where the full weighting procedure would not be.

Simulator sickness. Two complementary instruments are combined. The Simulator Sickness Questionnaire (SSQ) [82] scores sixteen symptoms on a four-point scale (none, slight, moderate, severe; 0–3) and combines them, with the standard weights, into three subscales, nausea, oculomotor and disorientation, and a total severity score.[3] It is taken before and after the session, so that any change in discomfort over the whole session can be measured. The single-item Fast Motion Sickness Scale (FMS) [83] is a verbal rating from 0 (no sickness) to 20 (severe), designed for fast repeated sampling, taken after each condition. Its trajectory across the conditions, not any single value, is what is read. The combination retains the richer instrument's detail without overloading the transitions, and serves both the secondary question of whether discomfort differs between conditions and well-being monitoring, with a session free to stop at any point without invalidating the data already collected. Figure 5.3 shows the full session and where each instrument is administered.

[3] With S_N, S_O, S_D the summed item scores of the three (overlapping) symptom groups, the scores are $N = 9.54\,S_N$, $O = 7.58\,S_O$, $D = 13.92\,S_D$ and the total TS $= 3.74\,(S_N + S_O + S_D)$ [82].

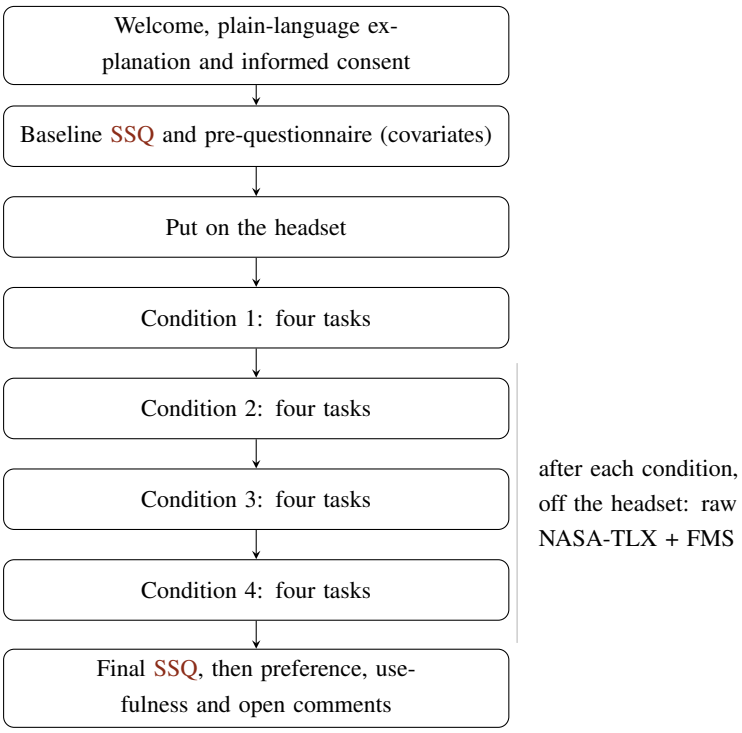


Figure 5.3: The session a participant goes through. The four conditions are ordered by the participant's Latin-square sequence (Section 5.3.1). After each condition the participant removes the headset for a short break, answers the raw NASA-TLX and a single FMS rating, and puts it back on for the next when ready. The SSQ is taken only before and after the whole session.

5.3.6 Procedure

A session runs as follows. The participant is welcomed and given a plain-language explanation of the study: its purpose, that it involves wearing a VR headset and answering short questions about animated data, that the session may be stopped at any moment and for any reason, discomfort in particular, and that the recorded data are anonymous and used only for this research. The session proceeds once the participant agrees to take part. No personal identifying data are collected beyond the covariates of the pre-questionnaire, there is no deception, and the only foreseeable risk is the ordinary discomfort VR can cause, which the simulator-sickness monitoring already tracks. The consent step is correspondingly brief.

After a baseline SSQ and the pre-questionnaire, the participant puts on the headset and completes the four conditions in the order fixed by their Latin-square sequence (Section 5.3.1). In each condition they perform the four tasks (Section 5.3.4). After each condition they take off the headset for a short break and answer the raw NASA-TLX and a single FMS rating for that condition, putting it back on for the next when they feel ready. The SSQ is administered once more at the end of the session. These breaks reset attention and limit cumulative discomfort. Answers are marked live by the tester, the researcher running the session, on a three-key input (correct, almost correct, incorrect) and logged with their response time by the session-management tool, which also presents the conditions in the correct order and holds the ground-truth answer key, consulted

only after the participant has committed to an answer, never during the search. The full protocol, the questionnaire items and the tester's answer sheet are reproduced in Appendix D.

5.3.7 Participants

The study ran with twenty participants, a multiple of four so that each Latin-square sequence was traversed equally. The size is justified by the within-subjects design, which gives higher power at small samples, by session duration, and by the prevailing practice in comparable immersive-visualization studies. Duration matters here because the pilot's six conditions took about forty-five minutes, whereas four conditions with questionnaires and breaks bring a session nearer half an hour. The statistical analysis favors non-parametric methods, is explicit about confidence intervals and about the kinds of conclusion a sample of this order can support, and treats the per-size and per-task breakdowns as exploratory.

The sample skewed young and male: nineteen participants were aged 18 to 24 and one 35 to 44, and seventeen were men and three women. On the pre-session covariates, prior familiarity with graph and network diagrams (mean 4.4 on a 1–5 scale), with data visualization in general (4.3) and with 3D games (4.5) was uniformly high, while prior experience with VR was low (2.3, median 2). The participants were therefore fluent readers of abstract graphs but, for the most part, new to VR, a profile worth bearing in mind when reading their preference for the immersive condition. No participant reported a colour-vision deficiency, the third covariate, recorded because node identity is carried by colour. Had one done so, that participant's data would have been examined separately for a colour-dependent effect. The sample is too small to test any of these covariates as a formal moderator, so subgroup patterns, such as whether the environment's benefit varies with prior VR experience, are treated as exploratory, not analysed inferentially.

5.4 What the Comparison Can and Cannot Show

A natural objection precedes any result. If the procedural condition proves better, is that because of the particular channels chosen (incoming flow on the relief, outgoing flow on the vegetation), or because of procedural generation in general? The evaluation is built so that this question is answered with evidence, not assumed, and so that the limit of what it can claim is stated plainly.

What the ablation isolates. The comparison holds every other channel constant and removes only the generated environment, so any measured difference is attributable to that environment, as configured for the study, and to nothing else in the scene. The study therefore measures one concrete realization of the data-world concept, not procedural generation in the abstract. This is an intentional scoping, since a controlled comparison requires a fixed configuration. The generality of the approach is argued separately, by its application across domains and by the configurability of the channels themselves (RQ1).

Separating channel information from a generic effect. The four tasks were chosen to make exactly this separation (Section 5.3.4). Three of them read quantities that the environment carries. The busiest-pair task reads a property of the flows alone, identical in both conditions, which the

environment does not encode. If the procedural condition helps on the three channel-aligned tasks but not on the pair task, the benefit is attributable to the information the generated features carry, and not to a generic effect that any richer 3D scene might produce, whether greater engagement, novelty, or simply a better spatial frame of reference. If instead the advantage appears uniformly, the pair task included, that points to such a generic effect, which is itself a finding and not a failure. The SCORE assessment (Section 5.2) adds an independent check. A performance advantage that tracks the diegesis dimensions the channels drive corroborates the channel explanation, while one that does not flags a generic cause.

What the study cannot attribute. Three claims are explicitly out of reach, and are not made. The study does not decompose the benefit among the individual features. It cannot say how much comes from the relief, how much from the vegetation, and how much from the spatial structure the environment provides, because all are added and removed together. It does not establish that these channels are optimal. A different mapping could perform differently. And it does not guarantee that the effect size transfers to other channel choices. Separating the contribution of each feature would require a factorial design that ablates the relief, the vegetation, the color and the motion one at a time, which is left as future work (Chapter 6).

What counts as a confound here. Under the framing of this thesis the analytical channel is the procedural generation as a whole, not any single feature. Part of what a strict reading would call a confound, the spatial reference and the sense of place that an inhabited generated world provides, is therefore part of the claim and not a nuisance to be removed. The value of the data-world is jointly the information its features carry and the navigable structure it gives that information. The ablation measures their sum by design, because that sum is precisely what the thesis means by an environment that is the visualization, not a container for it.

5.5 Results

The results are reported in the order of the questions they answer. The expert assessment of diegesis, addressing (RQ2), comes first, then the user study of task performance and perceived cognitive load, addressing (RQ3).

5.5.1 SCORE Expert Assessment

Two immersive-visualization experts independently rated the four study configurations on the five SCORE dimensions, following the procedure of Section 5.2. The two raters agreed closely, differing on a single cell of the twenty, so the profiles are reported as their means (Table 5.3).

The procedural environment scored higher on diegesis than the basic condition, and by the same margin at both complexity levels. The composite SCORE rose from 0.92 at the small size and 1.06 at the medium one under the basic condition to 1.69 at both procedural conditions, a gain of about 0.70 on the instrument's 0–5 scale. The advantage is concentrated in exactly the dimensions the generated environment is meant to drive. Concreteness rose from 1.0 to 3.0 and environmental context from 1.0 to 2.0, while coherence barely moved (1.75 to 2.0). Weighted into the composite,

Table 5.3: Expert SCORE profile of the two conditions, averaged over the two raters and the two complexity levels, since the procedural advantage was identical at both sizes. Each dimension is rated 0–5; the composite weights them by the framework's coefficients, given in parentheses. Spatial proximity and referential context are structurally zero in a fully virtual scene.

SCORE dimension (weight)	Basic	Procedural
Spatial proximity (0.09)	0.0	0.0
Concreteness (0.15)	1.0	3.0
Coherence (0.29)	1.75	2.0
Referential context (0.14)	0.0	0.0
Environmental context (0.33)	1.0	2.0
Composite SCORE (0–5)	0.99	1.69

the environmental-context dimension, at a third of the total weight, and concreteness account for almost the whole difference.

Spatial proximity and referential context were rated zero in every condition, basic and procedural alike. A fully virtual scene offers no physical object for a representation to sit beside or to point back to, so these two dimensions, designed for the augmented-reality setting the framework originates in, are structurally unavailable here and cap the achievable total. This is why the absolute values are low. The assessment establishes the contrast between the conditions, not the absolute level, and that contrast favors the procedural environment on every dimension a virtual scene can express.

The profile matches the design intent and, as the rest of this section will show, the user-study results. The diegesis advantage lives in the environment itself becoming part of the data presentation, the environmental-context dimension, and in the relief reading as a tangible, object-like surface, the concreteness dimension, not in the color and labeling the basic condition already shares. That the gain is flat across the two sizes, while the user study's load benefit grows with density (Section 5.6), is coherent: diegesis is a property of how the representation is built, judged on a static configuration, whereas the load benefit is a property of reading it under task pressure. The panel's two members make these figures descriptive, not inferential, a limitation their close agreement only partly offsets (Section 5.7).

A two-member panel measuring a contrast between conditions supports little beyond a narrow reading, and that reading is the answer to (RQ2). Mapping the flows onto the generated environment produces a more diegetic visualization than an abstract 3D scene, which is independent, qualitative support for the thesis's central move, that the environment can be the visualization and not a container for it. Whether that higher diegesis also brings a measurable analytical benefit is the question the user study takes up next.

5.5.2 Task Correctness

The user study ran with twenty participants. Task correctness and response time are analysed over all twenty: nineteen completed the four-condition protocol, and the pilot contributes its small and medium conditions, which coincide, in the same order, with the first counterbalancing sequence (Section 5.3.2). The questionnaire instruments, which the pilot answered under the earlier six-condition protocol, are analysed over the nineteen. With a sample of this size the per-size and per-task breakdowns remain exploratory. All comparisons are within-subjects and tested with the exact Wilcoxon signed-rank test, summarized by the median difference, the matched-pairs rank-biserial effect size r_{rb} (where $+1$ means every participant moved the same way), and a bootstrap 95% confidence interval. Correctness uses the three-level scale, scoring *almost correct* at 0.8, a near miss that named the runner-up rather than a half-failure. The exact weight does not drive any conclusion: re-running the analysis with the weight set anywhere from 0 to 1 leaves every effect and its significance essentially unchanged (the relief tasks stay at $\Delta \approx 0.46$–0.51, $p < 0.001$; the vegetation and control tasks stay null).

Both procedural conditions outscored both basic conditions (Table 5.4). Aggregated over the three channel-aligned tasks, the procedural environment improved correctness by $\Delta = 0.31$ on the 0–1 scale ($p < 0.001$, $r_{rb} = 1.0$, Confidence Interval (CI) $[0.23, 0.39]$); no participant scored higher without the environment than with it.

Table 5.4: Correctness per condition and task, scored on the three-level scale (correct = 1, almost correct = 0.8, incorrect = 0), averaged over the twenty participants. The last two columns average each task over the two sizes. The relief tasks (most incoming, peak moment) separate the procedural conditions sharply from the basic ones; the vegetation and control tasks do not.

Task	BS	BM	PS	PM	B avg	P avg
Most incoming (relief)	0.32	0.88	0.97	0.99	0.60	**0.98**
Peak moment (relief animation)	0.35	0.38	0.92	0.94	0.37	**0.93**
Most outgoing (vegetation)	0.59	0.45	0.46	0.53	0.52	0.50
Busiest pair (control)	0.80	0.74	0.85	0.68	0.77	0.77
All four tasks	0.52	0.61	0.80	0.79	0.56	0.79

The decomposition by task is the central result. The gain is carried entirely by the two relief tasks: most incoming improved by $\Delta = 0.38$ ($p < 0.001$, $r_{rb} = 0.97$) and peak moment by $\Delta = 0.57$ ($p < 0.001$, $r_{rb} = 1.0$). The vegetation task, most outgoing, showed no effect ($\Delta = -0.03$, $p = 0.69$), and the busiest-pair control task showed none either ($\Delta = -0.01$, $p = 0.76$), as predicted for a property the environment does not encode. Taken together the two relief tasks improved by $\Delta = 0.47$ ($p < 0.001$, $r_{rb} = 1.0$). Table 5.5 collects the paired comparisons.

Table 5.5: Paired comparisons (Procedural minus Basic), exact Wilcoxon signed-rank. Correctness is over twenty participants, cognitive load over nineteen (the pilot answered the questionnaires under the earlier protocol). Correctness is on the 0–1 scale, load on the form's 0–10 scale; r_{rb} is the matched-pairs rank-biserial effect size.

Comparison	Δ (P−B)	p	r_{rb}	95% CI
Correctness, channel-aligned	+0.31	< 0.001	1.00	[0.23, 0.39]
relief tasks only	+0.47	< 0.001	1.00	[0.36, 0.58]
most incoming	+0.38	< 0.001	0.97	[0.25, 0.51]
peak moment	+0.57	< 0.001	1.00	[0.40, 0.73]
most outgoing (vegetation)	−0.03	0.69	−0.14	[−0.18, 0.13]
busiest pair (control)	−0.01	0.76	0.08	[−0.22, 0.20]
NASA-TLX load, overall	−2.10	< 0.001	−1.00	[−2.65, −1.54]

5.5.3 Response Time

Response times point the same way without reaching significance in aggregate ($\Delta = -3.2$ s, $p = 0.50$). The pattern is again task-specific. The relief tasks were both faster under the procedural environment (most incoming about 11 to 13 versus 22 to 32 seconds in median, peak moment about 22 to 30 versus 35 to 55), the control task was unchanged, and the vegetation task was slower under the environment (about 30 versus 16 seconds). The relief gains are therefore not a speed-accuracy trade-off. The procedural condition was both more accurate and faster there. There is a non-significant tendency for the environment to help response time more at the medium size than the small ($\Delta = -5$ s in the size interaction, $p = 0.41$).

5.5.4 Perceived Cognitive Load

The raw NASA-TLX index (mean of the six subscales on the form's 0–10 scale) was lower under the procedural environment in every participant ($\Delta = -2.10$, $p < 0.001$, $r_{rb} = -1.0$; Table 5.5). The per-condition means show why this matters for the graded hypothesis (Table 5.6). Under the basic condition, load climbed from the small to the medium dataset ($4.2 \rightarrow 5.6$), whereas under the procedural environment it stayed low and barely moved ($2.7 \rightarrow 2.8$). The load reduction was accordingly larger at the medium size ($\Delta = -2.7$, $p < 0.001$) than at the small ($\Delta = -1.5$, $p = 0.004$), though both are significant. On the cognitive-load axis, then, the environment's benefit grows with scene density.

Table 5.6: Raw NASA-TLX load per condition (mean of the six subscales, form scale 0–10; lower is better), with standard deviation over nineteen participants. Load rises with density in the basic condition but not in the procedural one.

	BS	BM	PS	PM
Mean load	4.19	5.59	2.74	2.84
SD	2.08	1.72	1.31	1.36

5.5.5 Simulator Sickness

Discomfort stayed low and no session was stopped. The single-item FMS averaged 2.5 on its 0–20 scale, with a single peak of 10, and did not accumulate over a session (means of 3.0, 2.3, 2.2 and 2.6 at the four successive positions); it did not differ between the basic and procedural conditions (2.3 versus 2.7), so the relief animation and the denser visuals of the procedural condition did not add measurable sickness. The SSQ total severity, weighted by the standard subscale formula [82], rose only slightly from a mean of 14.6 at baseline to 16.1 at the end of the session ($\Delta = +1.6$, $p = 0.49$, not significant), the increase falling within the between-participant spread of the baseline itself, since several participants arrived already reporting some fatigue. The nausea noted in the pilot was confined to the large conditions that were subsequently dropped (Section 5.3.2); within the two retained sizes the comfort budget held.

5.5.6 Subjective Preference and Open Feedback

Asked which condition let them understand the data better, eighteen of the nineteen participants chose the procedural environment; the one exception qualified the choice by noting that the terrain helps when flow is heavy but adds clutter when it is sparse. Rated usefulness of the generated environment averaged 4.32 out of 5 (median 4.0). The open comments are consistent with the quantitative pattern and sharpen it. The relief was repeatedly singled out as making incoming-flow questions "trivial"; several participants reported that the terrain helped more as the flow grew dense, exactly the regime where the basic condition's spheres became hard to follow. The vegetation drew the opposite reaction. Multiple participants found the shrubs too low in contrast against the terrain and hard to read at a distance, and one questioned whether count is the right thing to map to vegetation at all. The busiest-pair question was described as the hardest and least supported by the environment, as its control role intends. A few comments concerned the flow rendering rather than the generated environment: several participants found the per-flow sphere and node sizes hard to judge, and one found the colour at a sphere's leading tip difficult to read against its trail. Recurring feature requests, playback-speed control, smoothing of the per-tick terrain animation, and an on-demand per-node history readout, are recorded as future work (Chapter 6).

5.6 Discussion

The data give a clear and coherent picture. The procedural environment improved analytical correctness and lowered perceived load, and both effects are specific to the channel the environment actually encodes.

The graded hypothesis holds on the load axis, not on the accuracy axis. The hypothesis was that the environment's contribution grows with scene density. In correctness, the procedural advantage is already large at the small size and does not grow toward the medium one. If anything it is slightly smaller at medium, because the basic-small most incoming question proved unexpectedly hard, with many participants defaulting to the visually prominent tourist station instead of the one busiest in the queried window. In perceived load, by contrast, the prediction is met. The basic condition's load climbs with density while the procedural condition's does not, so the load reduction is concentrated at the medium size. This is what one expects of a context channel that offloads working memory, and the open comments converge on it, with participants reporting that the terrain mattered most when the flow was heaviest. The level that would have shown the effect most strongly in accuracy, the large dataset, is precisely the one the pilot forced us to drop for comfort (Section 5.3.2), which is the cost of that decision made visible.

The benefit is attributable to the relief channel, not to a generic effect. Section 5.4 set out the test in advance. If the procedural condition helped uniformly, including on the busiest-pair control, the cause would be a generic novelty, engagement or spatial-reference effect and not the information the environment carries, a concern sharpened by a sample largely new to VR (Section 5.3.7). That is not what happened. The control task showed no advantage, and neither did the vegetation task, while the two relief tasks improved sharply. The benefit tracks the channel, which is the outcome that supports the channel-information account over a generic "prettier scene" account.

The vegetation channel did not work, and that is a finding. The relief carries a clear, readily read quantity. The vegetation density, as currently realized, does not. The open feedback explains why, pointing to weak contrast against the terrain and poor legibility at distance, and the response times suggest the channel may even have cost effort without paying it back. This was a first realization of a second generated channel, and the verdict is that it was the wrong encoding, not that a second channel is impossible. It motivates the per-feature factorial ablation and the search for stronger secondary encodings already set out as future work (Chapter 6). A participant's own suggestion, a per-node heat readout or a width-based encoding, is one candidate.

Relation to the mixed evidence on 3D. The visualization literature is divided on whether three-dimensional and immersive presentations help. Some studies find advantages for graph reading in 3D and in VR [50, 51], while others caution against three dimensions adopted without a task that justifies them [9]. These results sit naturally between the two positions. The environment helped where its channel encoded the queried quantity (the relief tasks) and did not help where it encoded nothing relevant (the control) or encoded it poorly (the vegetation). The vegetation is the informative case, since it carried a relevant quantity, the outgoing flow, yet did not help, because

it was not legible. What helps is therefore not the presence of an immersive environment but a generated feature that carries the answer in a form the eye reads easily, which is the justified-3D position applied to a generated world.

Research questions. (RQ2), on diegesis, is answered by the expert assessment (Section 5.5.1): the procedural condition scored higher on diegesis at both sizes, with the gain concentrated in the environmental-context and concreteness dimensions the relief drives, the profile the user-study results had led us to predict, and one that lines up with the relief's performance advantage, not with the inert vegetation. (RQ3), on how the environmental context affects performance and load as complexity grows, receives an answer: the environment improves correctness, by way of the relief channel, and lowers perceived load, with the load benefit growing as the scene densifies. These conclusions rest on twenty participants (nineteen for the load and sickness instruments). A larger sample would still sharpen the per-size and per-task estimates and settle whether the accuracy advantage carries any dependence on density at the two sizes retained.

5.7 Limitations of the Study

Several structural limitations are acknowledged in advance. The mono-domain choice that removes the confound between complexity and domain limits the generality of the measured effect to the tested domain (urban bike mobility). The framework's cross-domain generality is argued by other means, but the performance effect is not directly transferable. The stratified sampling keeps the study inside the metaphor's legible regime, so the study measures the environment's utility where the system is applicable, not how it behaves where the metaphor degenerates (dense origin-destination matrices, node counts orders of magnitude higher). Those regimes are mapped separately in Section 4.10. The reduction to two complexity levels, imposed by the pilot's session duration, restricts what the study can say about how the environment's benefit scales with density. Two points measure the interaction but not the shape of the trend, and the densest regime, exactly where the hypothesis predicts the largest benefit, falls outside the tested range. The simplicity of the tasks guarantees interpretability but underrepresents real analytical work, which chains questions, forms hypotheses and iterates.

The sample size detects medium average effects but not subtle interactions or subpopulation differences, and it is too small to analyse the recorded covariates (prior visualization and VR experience) as formal moderators. Node identity is carried by colour, so a colour-vision deficiency would matter for the comparison. None was reported, but a sample of this size cannot exclude an undetected case.

The evaluation also cannot attribute any effect to individual environmental features, only to the generated environment as a whole. That boundary, and the control built into the task set to address it, are set out in Section 5.4.

The SCORE assessment carries limitations of its own. Beyond the small panel, whose variance the raters' close agreement only partly offsets, the raters were not blind to the conditions, since the presence or absence of the generated terrain is immediately obvious, and the rate-discuss-rate

protocol leaves the revised ratings non-independent across the panel. Two of its five dimensions, spatial proximity and referential context, are structurally inapplicable to a fully virtual scene and were zero throughout, so the instrument measures the contrast between conditions, not an absolute level.

Some participants also reported friction with the locomotion itself, needing head movement to change altitude and preferring to stand, a prototype usability factor that the seated free-flight scheme does not fully address and that may have added a roughly constant effort across conditions. Finally, the study runs on a single headset family and runtime, and device-specific characteristics (resolution, field of view, refresh rate) may influence results.

Chapter 6

Conclusions and Future Work

This thesis set out to demonstrate that Procedural Content Generation can serve Immersive Analytics not as decoration but as the visualization itself, showing that an environment generated from a flow dataset, with its terrain, water, vegetation and animated flows, can carry analytical meaning and be navigated as a world, not read as a chart.

The work began from a concrete difficulty. Event-based flow data, common across finance, transport and communications, packs network topology, time, direction and magnitude into a single structure, and conventional 2D representations of it suffer from clutter and temporal occlusion. IA argues that an inhabited 3D space can ease reading of this kind, and PCG is the mature technology for generating such spaces, yet the two had mostly met with the generated world serving as a passive container for data laid on top of it.

The state of the art left three gaps that framed the thesis. Existing immersive systems depended on data that already carried spatial coordinates, treated the generated space as a backdrop kept separate from the data it held, or were single-purpose tools that could not be repurposed. The data-world proposal, an environment generated from and bound to the dataset, was the response to all three, and the chapters above built and evaluated one realization of it for flow data.

This chapter summarizes the contributions, revisits the research questions, and identifies the limitations and the work that naturally follows.

6.1 Summary of Contributions

The main contributions of this work are:

- **A functional procedural framework for immersive flow visualization.** A working system that translates arbitrary event-based flow datasets into navigable VR environments, configured entirely through portable data (a JSON file accompanying the dataset) with no code changes, demonstrated across domains as different as air traffic, international finance, urban mobility and network traffic.

- **The procedurally generated environment as an analytical channel.** A configurable mapping in which features of the generated world carry analyst-defined, time-varying per-node quantities: the terrain relief and, independently, the vegetation density, typically mapping a node's incoming and outgoing flow. The supporting design vocabulary includes formula-based semantics separated from visual normalization, finite-support peaks combined by maximum for per-node legibility, and biome and water layers that amplify rather than encode. The relief and vegetation are the two channels realized here, and the same machinery admits further generated channels.
- **An architecture for freely navigable time in VR.** The combination of a single timeline owner, scene state as a pure function of normalized time, and time-indexed precomputation, which together deliver deterministic bidirectional seeking at VR frame rates and are reusable beyond this prototype.
- **An ablation-ready experimental design and its instrumentation.** The environmental context as a separable channel, a 2×2 within-subjects study crossing it with dataset complexity over two scales of one domain, and the accompanying application of the SCORE framework to fully virtual environments, an extension its authors identified as future work.
- **An empirical delineation of the approach's limits.** Stress tests showing that scalability decomposes into three independent axes (record volume, graph density and node count), with distinct causes and mitigations, delimiting the regime in which the metaphor is applicable.

6.2 Research Questions Revisited

With the system built and the study run, the three research questions of Section 1.3 can now be answered in turn.

(RQ1), Framework. The question of how procedural generation principles can be adapted to structure multidimensional flow data into meaningful immersive environments is answered constructively by the system. The key adaptations proved to be: reducing domain specificity to a declarative schema plus arithmetic formulas over canonical fields, making every generated feature a function of data and time and not of authored placement, and separating analytical semantics from visual presentation so that the same generative machinery serves different datasets and questions.

(RQ2), Diegesis. The expert assessment with SCORE answers this question in the affirmative. A two-member expert panel rated the procedural environments higher on diegesis than their abstract counterparts, by the same margin at both complexity levels, with the advantage concentrated in the environmental-context and concreteness dimensions the relief drives, and the spatial-proximity and referential-context dimensions structurally inapplicable to a fully virtual scene (Section 5.5.1). The profile aligns with the relief's advantage in the user study, the two instruments agreeing with each other. The small panel makes the result descriptive, to be read alongside the objective task data, not on its own.

(RQ3), Analytical utility. The user study answers this question over twenty participants. The procedurally generated environment improved analytical correctness and lowered perceived cognitive load relative to an otherwise identical abstract scene, and both effects were specific to the channel the environment actually encodes. The gain was carried by the relief, which maps incoming flow, while the vegetation channel and the unencoded control task showed no effect. That specificity is evidence that the benefit comes from the information the generated features carry, not from a generic effect of a richer scene. The graded hypothesis, that the contribution grows with scene density, was met on the cognitive-load axis, where the environment's advantage grew from the small to the medium dataset, but not on the accuracy axis, where the advantage was already large at the small size. The sample remains modest, so the per-size and per-task estimates are best read as indicative.

6.3 Limitations

The principal limitations, gathered from the preceding chapters:

- **Scope.** The framework is validated on event-based flow data only; the generalisation to other data topologies is architectural (pure-data configuration, mediated subsystems) but not demonstrated.
- **Regime of applicability.** The metaphor degrades on dense origin-destination matrices (layout collapse) and the design assumes node counts in the hundreds; both limits are characterized but not overcome.
- **Presentation choices not perceptually validated.** The color palette, biome thresholds and related presentation decisions rest on informed judgment and visual iteration, not on formal perception studies. Color-vision deficiency in particular is only partially mitigated.
- **Trajectory simplifications.** Two-tangent arcs cannot hug multiple scattered peaks, and curves recomputed under in-flight spheres produce momentary swerves. Both are accepted trade-offs with identified fixes.
- **Platform dependence.** The prototype runs tethered to a desktop. A standalone headset build remains to be produced. Third-party vegetation materials target a different render pipeline and are fragile outside the editor.
- **Evaluation scope.** The performance evidence, once collected, is bounded by the study's own limitations (mono-domain, simple tasks, modest sample, single device family) as discussed in Section 5.7.

6.4 Future Work

Several directions follow naturally from this work:

- **Beyond flow data.** Extending the framework to other abstract data topologies, such as hierarchies, collections of time series, or high-dimensional tabular data, reusing the configuration model and the time architecture, and building toward the general procedural engine that motivated the thesis.
- **Richer relief semantics.** True valleys for signed quantities (negative balances as depressions below the waterline), and intermediate control points in flow trajectories so that arcs follow the upper contour of rugged relief.
- **Allocation-free trajectories.** The per-tick re-routing rewrites one spline per directed pair through a third-party library whose knot mutation allocates managed memory proportional to the pair count (Section 5.1). Replacing that library with a custom cubic evaluated directly from cached control points, or gating the refresh on actual height change so untouched ticks allocate nothing, would restore the intended allocation-free playback.
- **More generated channels.** The relief and vegetation already show that the generated environment can carry more than one per-node quantity. Further features (rock strata, water level, atmospheric density, the density of generated settlements) could each become an additional configurable channel, turning the world into a multi-variable display whose channels stay perceptually separable, alongside a study of how many such channels a viewer can read at once before they interfere.
- **More diegetic data markers.** The node pillars and the flow spheres are kept abstract so that no dataset is given a meaning it does not carry, but that choice leaves part of the scene outside the world it sits in. Giving the markers forms that belong to the generated environment, where this does not mislead, and measuring whether a more diegetic representation helps or distracts the reader, is a question this thesis leaves open.
- **Scaling the metaphor.** Community-based node aggregation, focus interactions that restrict visible flows to the observed node, and GPU-particle flow rendering: the identified responses to the density and node-count limits. The last of these is already scaffolded: the flow renderer sits behind an `IFlowRenderer` interface, and Unity's Visual Effect Graph is present in the project for a future implementation that feeds the cached splines into a particle system in place of the pooled objects.
- **Moving the per-tick computation to the GPU.** The final performance measurements located the bottleneck not in the rendering but in the computation that runs every tick, which runs on the CPU while the GPU stays almost idle (Section 5.1). The work is data-parallel: each terrain vertex and each flow trajectory is computed on its own, from the same per-node table, without depending on the others. The GPU can carry out these calculations easily, through a compute shader that handles one vertex, and one directed pair, per thread and reads the node weights from a shared buffer. The vertex heights and normals, together with the spline control points, could all be produced this way. What needs care is the few heights

the CPU still reads for non-visual use, placing the node pillars on the surface and deciding where a teleport lands, which would either be read back from the GPU at the cost of a frame of delay or computed on the GPU as well. This matters beyond raw speed, since the cost grows with the number of nodes and pairs, so moving it onto hardware that is otherwise doing nothing is the most direct way past the size limits the stress datasets reached (Section 4.10). It was realised late in the development, and is possibly the most important change for letting the system handle larger data.

- **In-scene configuration and controls.** Two extensions belong in the framework's own interface rather than its configuration file. The first moves the simpler parameters into in-scene controls (peak sharpness, world size, texture sets), editing the same pure-data models that the JSON file defines, and eventually offers a guided authoring interface for the formulas themselves. The second adds the runtime controls that study participants asked for (Section 5.5): a playback-speed control on the timeline, which the timeline owner already carries internally and which mainly needs surfacing in the panel and keeping the flows synchronized to the cursor at non-default speeds; smoother per-tick terrain animation; and an on-demand per-node history readout. Both keep configuration as portable data, the interface editing the model rather than replacing it.
- **Standalone deployment.** An Android OpenXR build for untethered use, including replacing pipeline-incompatible third-party materials.
- **Broader evaluation.** Replication on other domains and devices, more ecologically valid task sets, and, if the graded hypothesis is supported, a focused investigation of which properties of the environmental context (relief, color bands, water reference line, motion) carry the benefit.

6.5 Closing Remarks

The work delivered a complete, configurable and performant realization of the data-world concept for flow data, together with the experimental apparatus to measure its value. The evaluation is encouraging. Where a generated feature carried the answer in a form the eye reads easily, the relief, participants were both more accurate and under less cognitive load than in the equivalent abstract scene, and they overwhelmingly preferred the generated environment. Where a feature encoded its quantity poorly, the vegetation, the environment did not help, a reminder that the value lies in the mapping and not in the spectacle. Beyond any single effect size, the framework shows that the gap between procedural world generation and immersive analytics can be closed. The world can, in a precise and reproducible sense, be the data.

References

[1] Tom Chandler et al. “Immersive analytics”. In: *2015 Big Data Visual Analytics (BDVA): Hobart, Australia, 22 - 25 September 2015*. IEEE, Institute of Electrical and Electronics Engineers, 2015, p. 7314296. DOI: 10.1109/BDVA.2015.7314296.

[2] Karsten Klein, Michael Sedlmair, and Falk Schreiber. “Immersive analytics: An overview”. In: *it - Information Technology* 64.4-5 (Aug. 2022), pp. 155–168. ISSN: 2196-7032, 1611-2776. DOI: 10.1515/itit-2022-0037.

[3] Richard Skarbez et al. “Immersive Analytics: Theory and Research Agenda”. In: *Frontiers in Robotics and AI* 6 (Sept. 2019), p. 82. ISSN: 2296-9144. DOI: 10.3389/frobt.2019.00082.

[4] Julian Togelius et al. “Procedural Content Generation: Goals, Challenges and Actionable Steps”. In: *Artificial and Computational Intelligence in Games*. Ed. by Simon M. Lucas et al. Vol. 6. Dagstuhl Follow-Ups. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2013, pp. 61–75. ISBN: 978-3-939897-62-0. DOI: 10.4230/DFU.Vol6.12191.61.

[5] Noor Shaker, Julian Togelius, and Mark J. Nelson. *Procedural Content Generation in Games*. Computational Synthesis and Creative Systems. Cham: Springer International Publishing, 2016. ISBN: 978-3-319-42714-0. DOI: 10.1007/978-3-319-42716-4.

[6] Zhicheng Liu, Chen Chen, and John Hooker. “Manipulable Semantic Components: A Computational Representation of Data Visualization Scenes”. In: *IEEE Transactions on Visualization and Computer Graphics* 31.1 (Jan. 2025), pp. 732–742. ISSN: 1941-0506. DOI: 10.1109/TVCG.2024.3456296.

[7] Chenglong Wang, John Thompson, and Bongshin Lee. “Data Formulator: AI-Powered Concept-Driven Visualization Authoring”. In: *IEEE Transactions on Visualization and Computer Graphics* 30.1 (Jan. 2024), pp. 1128–1138. ISSN: 1941-0506. DOI: 10.1109/TVCG.2023.3326585.

[8] Barrett Ens et al. “Grand Challenges in Immersive Analytics”. In: *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. CHI ’21. New York, NY, USA: Association for Computing Machinery, 2021, pp. 1–17. ISBN: 978-1-4503-8096-6. DOI: 10.1145/3411764.3446866.

[9] Tamara Munzner. *Visualization Analysis and Design*. New York: A. K. Peters/CRC Press, 2014. ISBN: 978-0-429-08890-2.

[10] Matthias Kraus et al. “Immersive Analytics with Abstract 3D Visualizations: A Survey”. In: *Computer Graphics Forum* 41.1 (2022), pp. 201–229. ISSN: 1467-8659. DOI: 10.1111/cgf.14430.

[11] Julian Togelius et al. “What is procedural content generation? Mario on the borderline”. In: *Proceedings of the 2nd International Workshop on Procedural Content Generation in Games*. PCGames ’11. New York, NY, USA: Association for Computing Machinery, June 2011, pp. 1–6. ISBN: 978-1-4503-0872-4. DOI: 10.1145/2000919.2000922.

[12] Julian Togelius et al. “Search-Based Procedural Content Generation: A Taxonomy and Survey”. In: *IEEE Transactions on Computational Intelligence and AI in Games* 3.3 (Sept. 2011), pp. 172–186. ISSN: 1943-0698. DOI: 10.1109/TCIAIG.2011.2148116.

[13] Georgios N. Yannakakis and Julian Togelius. “Experience-Driven Procedural Content Generation”. In: *IEEE Transactions on Affective Computing* 2.3 (July 2011), pp. 147–161. ISSN: 1949-3045. DOI: 10.1109/T-AFFC.2011.6.

[14] T.J. Rose and A.G. Bakaoukas. “Algorithms and approaches for procedural terrain generation”. In: 2016. DOI: 10.1109/VS-GAMES.2016.7590336.

[15] H. Schott et al. “Large-scale Terrain Authoring through Interactive Erosion Simulation”. In: *ACM Transactions on Graphics* 42.5 (2023). DOI: 10.1145/3592787.

[16] L. Brůha and J. Kolář. “A procedural footprint enhancement of global topographic surface with multiple levels of detail”. In: *International Journal of Digital Earth* 13.4 (2020), pp. 527–545. DOI: 10.1080/17538947.2018.1543362.

[17] D.E. Evans. “Archipelago”. In: 2022, pp. 652–654. DOI: 10.1145/3527927.3531197.

[18] M.R. Lewis et al. “Balance trucks: Using crowd-sourced data to procedurally-generate gameplay within mobile games”. In: *2018 10th International Conference on Virtual Worlds and Games for Serious Applications (VS-Games)*. 2018, pp. 1–4. DOI: 10.1109/VS-Games.2018.8493412.

[19] R. Spick and J.A. Walker. “Realistic and textured terrain generation using GANs”. In: 2019. DOI: 10.1145/3359998.3369407.

[20] W. Li. “Intelligent Interface for Synthesizing Procedural Stone Forest Landscape”. In: 2023, pp. 565–568. DOI: 10.1109/AIHCIR61661.2023.00100.

[21] W. Li. “Synthesizing Procedural Landscape for Volcanic Eruption Simulation in Virtual Reality”. In: *2023 7th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*. Oct. 2023, pp. 1–5. DOI: 10.1109/ISMSIT58785.2023.10304961.

[22] D. Kutzias and S. von Mammen. “Recent Advances in Procedural Generation of Buildings: From Diversity to Integration”. In: *IEEE Transactions on Games* 16.1 (2024), pp. 16–35. DOI: 10.1109/TG.2023.3262507.

[23] M. Janovský, P. Tobias, and V. Cehák. “3D Visualisation of the Historic Pre-Dam Vltava River Valley—Procedural and CAD Modelling, Online Publishing and Virtual Reality”. In: *ISPRS International Journal of Geo-Information* 11.7 (2022). DOI: 10.3390/ijgi11070376.

[24] Gustavo Alomia et al. “Automatic 3D Urban Installation Generation in Virtual Cities”. In: *2019 International Conference on Virtual Reality and Visualization (ICVRV)*. Nov. 2019, pp. 217–222. DOI: 10.1109/ICVRV47840.2019.00052.

[25] I. Bejleri et al. “Using 3d rule-based modeling to interactively visualize “complete streets” design scenarios”. In: *Transportation Research Record* 2675.10 (2021), pp. 14–30. ISSN: 0361-1981. DOI: 10.1177/0361198121999051.

[26] Benjamin Williams and Christopher J. Headleand. "A Time-Line Approach for the Generation of Simulated Settlements". In: *2017 International Conference on Cyberworlds (CW)*. Sept. 2017, pp. 134–141. DOI: 10.1109/CW.2017.32.

[27] M.A. AlFadalat, W. Al-Azhari, and L. Dabbour. "Procedural Modeling Based Shape Grammar as a Key to Generating Digital Architectural Heritage". In: *Journal on Computing and Cultural Heritage* 16.4 (2023). DOI: 10.1145/3606701.

[28] P. Ren et al. "Data-driven modeling for Chinese ancient architectures". In: vol. 2. 2016, pp. 59–62. DOI: 10.1145/3014027.3014029.

[29] Y. Raies and S. von Mammen. "A Swarm Grammar-Based Approach to Virtual World Generation". In: vol. 12693 LNCS. 2021, pp. 459–474. DOI: 10.1007/978-3-030-72914-1_30.

[30] Z. Xi et al. "TerraCraft: City-scale generative procedural modeling with natural languages". In: *Graphical Models* 141 (2025). DOI: 10.1016/j.gmod.2025.101285.

[31] S. Alaka and R. Bidarra. "Mixed-initiative generation of virtual worlds - a comparative study on the cognitive load of WFC and HSWFC". In: 2024. DOI: 10.1145/3649921.3659852.

[32] A.G. Kravets et al. "New Level-Generation Method for a Single-Player 3D Game". In: *Studies in Systems, Decision and Control*. Vol. 639. 2026, pp. 213–227. DOI: 10.1007/978-3-032-06787-6_17.

[33] Puru Ojha and Y. Raghu Reddy. "ACMGVR: Architecturally Consistent Mazes for Games in Virtual Reality". In: *Companion Proceedings of the 2024 Annual Symposium on Computer-Human Interaction in Play*. CHI PLAY Companion '24. New York, NY, USA: Association for Computing Machinery, 2024, pp. 221–226. ISBN: 979-8-4007-0692-9. DOI: 10.1145/3665463.3678818.

[34] P. A. Ferreira et al. "Adaptable and Customizable Walkable Virtual Environments in Small Spaces: Generating a Virtual Reality Museum Experience". In: *2025 International Conference on Graphics and Interaction (ICGI)*. Nov. 2025, pp. 1–8. DOI: 10.1109/ICGI68463.2025.11302519.

[35] A. Rita Rebelo, P.A. Ferreira, and R. Nóbrega. "Adaptive Virtual Environments in Small Physical Spaces: Navigation Design for Customized User Experiences in Virtual Reality". In: 2024. DOI: 10.1145/3613905.3650942.

[36] F. Rupp and K. Eckert. "G-PCGRL: Procedural Graph Data Generation via Reinforcement Learning". In: *2024 IEEE Conference on Games (CoG)*. Aug. 2024, pp. 1–8. DOI: 10.1109/CoG60054.2024.10645633.

[37] B. von Rymon-Lipinski et al. "Level graph - Incremental procedural generation of indoor levels using minimum spanning trees". In: vol. 2019-August. 2019. DOI: 10.1109/CIG.2019.8847956.

[38] Zhiqi Xiao et al. "Large Scale Farm Scene Modeling from Remote Sensing Imagery". In: *ACM Trans. Graph.* 43.6 (Nov. 2024), 260:1–260:12. ISSN: 0730-0301. DOI: 10.1145/3687918.

[39] S. Yang et al. "Procedural Generation of 3D Scenes for Urban Landscape Based on Remote Sensing Images". In: *2024 IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. 2024. July 2024, pp. 1–7. DOI: 10.1109/AVSS61716.2024.10672567.

[40] T. Niese et al. “Procedural Urban Forestry”. In: *ACM Transactions on Graphics* 41.2 (2022). DOI: 10.1145/3502220.

[41] J. Huang et al. “Visualizing Ecological Data in Virtual Reality”. In: *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. Mar. 2019, pp. 1311–1312. DOI: 10.1109/VR.2019.8797771.

[42] J. Huang et al. “Walking through the forests of the future: using data-driven virtual reality to visualize forests under climate change”. In: *International Journal of Geographical Information Science* 35.6 (2021), pp. 1155–1178. ISSN: 1365-8816. DOI: 10.1080/13658816.2020.1830997.

[43] M. Brehmer, R. Kosara, and C. Hull. “Generative Design Inspiration for Glyphs with Diatoms”. In: *IEEE Transactions on Visualization and Computer Graphics* 28.1 (Jan. 2022), pp. 389–399. ISSN: 1941-0506. DOI: 10.1109/TVCG.2021.3114792.

[44] Rilla Khaled, Mark J. Nelson, and Pippin Barr. “Design metaphors for procedural content generation in games”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’13. New York, NY, USA: Association for Computing Machinery, 2013, pp. 1509–1518. ISBN: 978-1-4503-1899-0. DOI: 10.1145/2470654.2466201.

[45] A. S. Gill et al. “A Pilot Study Investigating Student Interaction Preferences in Immersive Virtual Reality Environments”. In: *2022 IEEE International Conference on Teaching, Assessment and Learning for Engineering (TALE)*. Dec. 2022, pp. 133–140. DOI: 10.1109/TALE54877.2022.00029.

[46] A. Lugilde-López et al. “Biarc Spline Interpolation with Procedural Grid and Segments with Armature to Support Colonosopy Simulators in Unity”. In: 2022. DOI: 10.1109/SIIE56031.2022.9982356.

[47] E.H. Korkut and E. Sürer. “Developing a framework for heterotopias as discursive playgrounds: a comparative analysis of non-immersive and immersive technologies”. In: *Virtual Reality* 28.1 (2024). DOI: 10.1007/s10055-023-00905-w.

[48] Y. -P. Ma and H. -H. Lien. “Using Game Technology to Enhance the Interaction and Visualization Ability of 3DGIS Historic Site Modeling”. In: *2018 1st IEEE International Conference on Knowledge Innovation and Invention (ICKII)*. July 2018, pp. 66–69. DOI: 10.1109/ICKII.2018.8569094.

[49] Paulo Jorge Palhau Moutinho. “Immersive 2D Scatterplots and Parallel Coordinates for Information Visualization in a Virtual Reality Environment”. Master’s Dissertation, Mestrado em Engenharia Informática e Computação. Faculdade de Engenharia da Universidade do Porto, 2022.

[50] Colin Ware and Glenn Franck. “Evaluating stereo and motion cues for visualizing information nets in three dimensions”. In: *ACM Transactions on Graphics (TOG)* 15.2 (1996), pp. 121–140. ISSN: 0730-0301. DOI: 10.1145/234972.234975.

[51] Oh-Hyun Kwon et al. “A Study of Layout, Rendering, and Interaction Methods for Immersive Graph Visualization”. In: *IEEE Transactions on Visualization and Computer Graphics* 22.7 (July 2016), pp. 1802–1815. ISSN: 1941-0506. DOI: 10.1109/TVCG.2016.2520921.

[52] Maxime Cordeil et al. “Immersive Collaborative Analysis of Network Connectivity: CAVE-style or Head-Mounted Display?” In: *IEEE Transactions on Visualization and Computer Graphics* 23.1 (Jan. 2017), pp. 441–450. ISSN: 1941-0506. DOI: 10.1109/TVCG.2016.2599107.

[53] Anete Medina Pereira. “Immersive Visualization of CI/CD Pipelines”. Master’s Dissertation, Mestrado em Engenharia Informática e Computação. Faculdade de Engenharia da Universidade do Porto, 2025.

[54] Yalong Yang et al. “Origin-Destination Flow Maps in Immersive Environments”. In: *IEEE Transactions on Visualization and Computer Graphics* 25.1 (Jan. 2019), pp. 693–703. ISSN: 1941-0506. DOI: 10.1109/TVCG.2018.2865192.

[55] Natalia Andrienko and Gennady Andrienko. *Exploratory Analysis of Spatial and Temporal Data: A Systematic Approach*. Springer, 2006. DOI: 10.1007/3-540-31190-4.

[56] O. Rogla, N. Pelechano, and G. Patow. “Procedural Semantic Cities”. In: 2017, pp. 113–120. DOI: 10.2312/ceig.20171217.

[57] Wanwan Li. “Simulating Virtual Construction Scenes on OpenStreetMap”. In: *Proceedings of the 2022 6th International Conference on Virtual and Augmented Reality Simulations*. ICVARS ’22. New York, NY, USA: Association for Computing Machinery, 2022, pp. 14–20. ISBN: 978-1-4503-8733-0. DOI: 10.1145/3546607.3546610.

[58] P. Ding et al. “A generation method and verification of virtual dataset”. In: vol. 645. 2020, pp. 469–475. DOI: 10.1007/978-981-15-6978-4_55.

[59] J.W. Park and S.H. Oh. “A Study on Creation and Usability of Real Time City Generator via Procedural Content Generation: - Focus on virtual reality contents for senior”. In: 2019. DOI: 10.1109/ISMAC.2019.8836162.

[60] A. Gambi, M. Mueller, and G. Fraser. “AsFault: Testing self-driving car software using search-based procedural content generation”. In: 2019, pp. 27–30. DOI: 10.1109/ICSE-Companion.2019.00030.

[61] M. Mallia et al. “Automatic Creation of a Virtual/Augmented Gallery Based on User Defined Queries on Online Public Repositories”. In: *VR Technologies in Cultural Heritage*. Vol. 904. Cham: Springer International Publishing, 2019, pp. 135–147. ISBN: 978-3-030-05819-7. DOI: 10.1007/978-3-030-05819-7_11.

[62] M. Hayashi, S. Bachelder, and M. Nakajima. “Automatic generation of art exhibitions from the internet resources”. In: *2020 IEEE 9th Global Conference on Consumer Electronics (GCCE)*. 2020, pp. 642–643. DOI: 10.1109/GCCE50665.2020.9291722.

[63] H. Cecotti. “Great Paintings in Fully Immersive Virtual Reality”. In: *2021 7th International Conference of the Immersive Learning Research Network (iLRN)*. June 2021, pp. 1–8. DOI: 10.23919/iLRN52045.2021.9459403.

[64] M. Sra, S. Garrido-Jurado, and P. Maes. “Oasis: Procedurally Generated Social Virtual Spaces from 3D Scanned Real Spaces”. In: *IEEE Transactions on Visualization and Computer Graphics* 24.12 (2018), pp. 3174–3187. DOI: 10.1109/TVCG.2017.2762691.

[65] M. Keshavarzi, A. Bidgoli, and H. Kellner. “V-Dream: Immersive Exploration of Generative Design Solution Space”. In: *HCI International 2020 - Late Breaking Papers: Multimodality and Intelligence*. Vol. 12424 LNCS. Cham: Springer International Publishing, 2020, pp. 477–494. ISBN: 978-3-030-60117-1. DOI: 10.1007/978-3-030-60117-1_35.

[66] Josh Urban Davis et al. “Designing Co-Creative AI for Virtual Environments”. In: *Proceedings of the 13th Conference on Creativity and Cognition*. C&C ’21. New York, NY, USA: Association for Computing Machinery, 2021, pp. 1–11. ISBN: 978-1-4503-8376-9. DOI: 10.1145/3450741.3465260.

[67] Y. Mao et al. “A Hybrid Intelligence Approach to Training Generative Design Assistants: Partnership Between Human Experts and AI Enhanced Co-Creative Tools”. In: vol. 368. 2023, pp. 108–123. DOI: 10.3233/FAIA230078.

[68] A. Wieser et al. “A DIGITAL FRAMEWORK FOR GENERATING AND EVALUATING DIGITAL CIRCULAR TWINS”. In: vol. 2024. 2024, pp. 974–981. DOI: 10.35490/EC3.2024.318.

[69] A.O.R. Franco et al. “Generating Rooms using Generative Grammars and Genetic Algorithms”. In: vol. 2022-October. 2022. DOI: 10.1109/SBGAMES56371.2022.9961112.

[70] I.F. Capasso-Ballesteros and F. De La Rosa R. “Generating automated rules-based game design prototypes with marugen”. In: 2019, pp. 146–149. DOI: 10.1109/ICVRV47840.2019.00034.

[71] B. Onrust et al. “Ecologically Sound Procedural Generation of Natural Environments”. In: *International Journal of Computer Games Technology* 2017.1 (2017), p. 7057141. ISSN: 1687-7055. DOI: 10.1155/2017/7057141.

[72] T. Klein et al. “Multi-Scale Procedural Animations of Microtubule Dynamics Based on Measured Data”. In: *IEEE Transactions on Visualization and Computer Graphics* 26.1 (Jan. 2020), pp. 622–632. ISSN: 1941-0506. DOI: 10.1109/TVCG.2019.2934612.

[73] J. Zhang et al. “Procedural modeling of rivers from single image toward natural scene production”. In: *Visual Computer* 35.2 (2019), pp. 223–237. DOI: 10.1007/s00371-017-1465-7.

[74] N. Mikulicic and Z. Mihajlović. “Procedural generation of mediterranean environments”. In: 2016, pp. 261–266. DOI: 10.1109/MIPRO.2016.7522149.

[75] M. H. Raha et al. “FIRETWIN: Digital Twin Advancing Multi-Modal Sensing, Interactive Analytics for Tactical Wildfire Response”. In: *2025 IEEE 30th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*. Oct. 2025, pp. 1–8. DOI: 10.1109/CAMAD67323.2025.11229918.

[76] Danny Holten. “Hierarchical Edge Bundles: Visualization of Adjacency Relations in Hierarchical Data”. In: *IEEE Transactions on Visualization and Computer Graphics* 12.5 (Sept. 2006), pp. 741–748. ISSN: 1941-0506. DOI: 10.1109/TVCG.2006.147.

[77] Thomas M. J. Fruchterman and Edward M. Reingold. “Graph drawing by force-directed placement”. In: *Software: Practice and Experience* 21.11 (1991), pp. 1129–1164. DOI: 10.1002/spe.4380211102.

[78] Tarik Hasan, Khalad Hasan, and Barrett Ens. “SCORE: A Framework for Quantifying Diegesis in Situated Visualization for Augmented Reality”. In: *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems*. CHI ’26. New York, NY, USA: Association for Computing Machinery, 2026, pp. 1–19. ISBN: 979-8-4007-2278-3. DOI: 10.1145/3772318.3791911.

[79] Bongshin Lee et al. “Task taxonomy for graph visualization”. In: *Proceedings of the 2006 AVI workshop on BEyond time and errors: novel evaluation methods for information visualization*. BELIV ’06. New York, NY, USA: Association for Computing Machinery, 2006, pp. 1–5. ISBN: 978-1-59593-562-5. DOI: 10.1145/1168149.1168168.

[80] Alex Cao et al. “NASA TLX: Software for assessing subjective mental workload”. In: *Behavior Research Methods* 41.1 (Feb. 2009), pp. 113–117. ISSN: 1554-3528. DOI: 10.3758/BRM.41.1.113.

[81] Sandra G. Hart. “NASA-Task Load Index (NASA-TLX); 20 Years Later”. In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. Vol. 50. 9. 2006, pp. 904–908. DOI: 10.1177/154193120605000909.

[82] Robert S. Kennedy et al. “Simulator Sickness Questionnaire: An Enhanced Method for Quantifying Simulator Sickness”. In: *The International Journal of Aviation Psychology* 3.3 (1993), pp. 203–220. DOI: 10.1207/s15327108ijap0303_3.

[83] Behrang Keshavarz and Heiko Hecht. “Validating an Efficient Method to Quantify Motion Sickness”. In: *Human Factors* 53.4 (2011), pp. 415–426. DOI: 10.1177/0018720811403736.

Appendix A

Study Dataset Configuration File

This appendix reproduces, in full, the configuration file (Listing A.1) used for the medium-complexity condition of the user study (Section 5.3.3), the larger of the two complexity levels retained for the study. It is included as the operational instance of two requirements of the approach, dataset independence and non-programmer, portable configuration (Section 3.5). Every dataset-specific aspect of the system is expressed here as plain data, with no references to scene objects or engine types. Switching to another dataset, or to the small complexity level, is a matter of editing this single file and never the scene or the code. The small condition differs from the file below only in the `csvFile` entry and in the world-scale parameters.

The file is organized in blocks that mirror the pipeline (Section 4.1.4). The `dataset` block names the source file and declares the schema that maps the dataset's real headers (`start_station_name`, `started_at`, and so on) onto the canonical fields. The `terrain` block carries the two analytical formulas, for relief and for vegetation density, in the syntax described in Section 3.4.1 and illustrated in Listing 3.1. The `flows` block carries the per-flow size formula and the aggregation parameters (Section 4.6). The remaining entries (`playback`, `layout`, `nodes`, and the `appearance`, `animator`, `splines`, `trail` sub-blocks) tune the visual presentation and the performance budget, and have no effect on the analytical meaning of the scene.

Listing A.1: Complete configuration file for the medium-complexity study condition (`settings.divvy.M.json`).

```
{
  "dataset": {
    "csvFile": "divvy_M.csv",
    "delimiter": ",",
    "schema": {
      "origin": "start_station_name",
      "destination": "end_station_name",
      "departure": "started_at",
      "arrival": "ended_at",
      "type": "member_casual",
```

```
      "timestampFormat": "yyyy-MM-dd HH:mm:ss",
      "passThroughColumns": []
    }
  },

  "playback": {
    "durationSeconds": 180,
    "speed": 1,
    "autoPlayOnLoad": true,
    "autoResumeOnScrub": false
  },

  "layout": {
    "worldRadius": 400,
    "nodeElevation": 0.25,
    "reprojectDuration": 0.5,
    "forceDirected": {
      "iterations": 250,
      "temperature": 0.2,
      "cooling": 0.95,
      "randomSeed": 42
    }
  },

  "nodes": {
    "baseScale": 0.6
  },

  "terrain": {
    "appearance": {
      "resolution": 128,
      "worldSize": 1500,
      "maxHeightFraction": 0.12,
      "baseRadiusFraction": 0.07,
      "radiusGrowth": 0.4,
      "peakSharpness": 4,
      "minRenderHeight": 0.02,
      "minPeakHeightFraction": 0.04
    },
    "formula": {
      "terms": [
        {
          "name": "arrivals",
          "field": "",
          "direction": "Incoming",
          "timeAnchor": "Arrival",
          "windowSize": 0.05
        }
      ],
      "expression": "arrivals",
      "postProcessing": "ClampToZero",
      "logScale": false,
      "normalise": true,
      "timeBuckets": 512
    },
    "animator": {
      "minUpdateInterval": 0.1,
```

```
      "forceUpdateDelta": 0.005
    },
    "vegetation": {
      "enabled": true,
      "densityPerSquareMetre": 0.004,
      "maxPerNode": 800,
      "maxTotalInstances": 30000,
      "minHeightFraction": 0,
      "maxHeightFraction": 0.6,
      "waterMargin": 0.2,
      "scaleMin": 1,
      "scaleMax": 3,
      "randomSeed": 1337,
      "minInstancesPerNode": 15,
      "maxInstancesPerNode": 800,
      "formula": {
        "terms": [
          {
            "name": "departures",
            "field": "",
            "direction": "Outgoing",
            "timeAnchor": "Departure",
            "windowSize": 0.05
          }
        ],
        "expression": "departures",
        "postProcessing": "ClampToZero",
        "logScale": false,
        "normalise": true,
        "timeBuckets": 512
      }
    }
  },

  "flows": {
    "splines": {
      "arcHeightRatio": 0.25,
      "maxArcHeightFraction": 0.0533,
      "minArcHeightFraction": 0.01,
      "clearanceFraction": 0.015,
      "sampleSpacingFraction": 0.012
    },
    "duration": {
      "defaultSeconds": 3,
      "multiplier": 1
    },
    "trail": {
      "time": 1.2,
      "width": 0.1
    },
    "aggregation": {
      "buckets": 150,
      "durationFraction": 3.0
    },
    "visualFormula": {
      "terms": [
        { "name": "trips", "field": "" }
```

```
      ],
      "expression": "trips",
      "postProcessing": "ClampToZero",
      "normalisation": "Log",
      "outputMin": 0.4,
      "outputMax": 5.0
    }
  }
}
```

Appendix B

The Configuration Formula Language

This appendix specifies the small language through which the analyst defines what the generated environment encodes, without editing code. It is the operational basis of the non-programmer configuration requirement (Section 3.5) and of the mapping strategies of Section 3.4. Two formula kinds share one arithmetic grammar: the per-node formula that drives terrain relief and vegetation density (Section 3.4.1), and the per-record formula that sizes each flow (Section 4.6.1). Listing A.1 of Appendix A shows both in use.

B.1 Expression Grammar

An expression combines named terms with arithmetic. Identifiers must match a declared term name; numeric literals and parentheses are allowed. The grammar, in Extended Backus-Naur Form (EBNF), is:

```
expression ::= term   { ("+" | "-") term }
term       ::= factor { ("*" | "/") factor }
factor     ::= "-" factor | primary
primary    ::= number | identifier | "(" expression ")"
number     ::= digit {digit} ["." digit {digit}] | "." digit {digit}
identifier ::= (letter | "_") { letter | digit | "_" }
```

Operators follow standard precedence. Multiplication and division bind tighter than addition and subtraction, unary minus binds tightest, and parentheses override both. Division by a zero or undefined denominator yields zero instead of failing, so that transient zero denominators during a timeline scrub cannot break the visualization. An empty expression denotes a flat, unencoded channel. The expression is parsed once at load, and an unmatched identifier or a syntax error halts the pipeline with a concrete message.

B.2 Per-Node Terms

A per-node formula, for terrain or for vegetation, is a list of named terms together with the expression that combines them. Each term aggregates one quantity per node over a window of the timeline. Table B.1 lists the term fields and Table B.2 the formula-level options.

Table B.1: Fields of a per-node term. The term yields one value per node and per timeline bucket.

Field	Meaning
`name`	Identifier used inside the expression.
`field`	CSV column to aggregate. Empty means count semantics (one per matching record).
`direction`	`Incoming` (the record arrives at this node) or `Outgoing` (it leaves this node).
`timeAnchor`	`Departure`, `Arrival`, or `InFlight` (the record's own interval between departure and arrival, which ignores `windowSize`).
`windowSize`	Contribution width in normalized time for the Departure and Arrival anchors. Zero is a single-bucket pulse.

Table B.2: Formula-level options applied after the expression is evaluated per node.

Option	Meaning
`expression`	Arithmetic combination of the term names (Section B.1).
`postProcessing`	`ClampToZero` (negatives become zero), `AbsoluteValue` (magnitude only), or `Identity` (pass-through).
`logScale`	Apply $\log(1+\max(0,x))$ before normalization, for heavy-tailed data.
`normalise`	Rescale so the global maximum is one, the range the terrain expects.
`timeBuckets`	Number of discrete timeline buckets the table is precomputed over, from 16 to 4096 (default 512).

B.3 Per-Record Flow Formula

The per-flow formula sizes each flow event, its sphere and trail, from the record's own columns. It reuses the same expression grammar, but its terms carry no direction and no time anchor, only a `name` and a `field`, since they are evaluated once per record, not aggregated per node. After post-processing and an optional logarithm, the value is linearly remapped from the dataset's observed range onto a configured output interval, so that the smallest event maps to `outputMin` and the largest to `outputMax`. An empty expression leaves every flow at unit size.

B.4 Worked Examples

The configuration of the user-study dataset (Appendix A, Listing A.1) instantiates all three channels:

- **Terrain relief.** One term `arrivals` with empty `field` (count), direction `Incoming`, anchor `Arrival`, window 0.05; expression `arrivals`. Each node's hill tracks how many trips arrive at it in the current window.
- **Vegetation density.** One term `departures` (count, `Outgoing`, `Departure`, 0.05); expression `departures`. The complementary outgoing channel.
- **Flow size.** One term `trips` (count); expression `trips`; logarithmic normalization to $[0.4, 5.0]$. Busier bundles read as larger spheres with thicker trails.

A net-flow relief, for instance, would declare two terms `in` and `out` and write the expression `in - out` with `Identity` post-processing, raising hills where arrivals dominate. The other datasets the framework was applied to (Section 4.10) use field-based terms such as `bytes`, `amount` and `dist` in place of counts, with no other change to the pipeline.

Appendix C

SCORE Expert-Assessment Materials

This appendix reproduces the instrument used for the expert assessment of the environment's diegesis (Section 5.2, answering (RQ2)), the counterpart, for the SCORE strand, of what Appendix D is for the user study. The realized panel and its results are reported in Section 5.5.1.

C.1 Dimensions and Rubric

SCORE rates a visualization on five ordinal dimensions, each from 0 to 5, after Hasan et al. [78]. Table C.1 gives the anchors shown to the raters.

Table C.1: The five SCORE dimensions and the 0–5 anchors shown to the raters [78].

Dimension	Anchors, from 0 to 5
S Spatial proximity	Non-situated, In-situ, Linked, Situated, Overlaid, Embedded (1:1). How close the representation sits to what it represents.
C Concreteness	Symbolic, Geometric primitive, Schematic, Simplified replica, Detailed replica, Realistic facsimile. How object-like, not symbolic, the form is.
O cOherence	Incohesive, Weak, Partial, Moderate, Strong, Cohesive. Consistency with the logic of a plausible world.
R Referential context	No alignment, Ambiguous, Indirect, Clear, Meaningful, Inseparable. Strength of the link to the referent (a node or value).
E Environmental context	No alignment, Ambiguous, Indirect, Clear, Meaningful, Inseparable. How far the surrounding environment conveys the datum.

C.2 Procedure

A small panel of domain experts rates each of the four study conditions (BS, BM, PS, PM) live in the headset, with the headset view mirrored to a screen so the rest of the panel follows each pass. The session uses the framework's rate-discuss-rate protocol [78]. After a shared calibration on the paper's published example figure, each condition runs as follows: every expert explores the scene and rates the five dimensions privately, the panel then discusses its reasoning aloud without

forcing consensus, and each expert submits a revised individual rating. Individual ratings are kept, not collapsed to a consensus, which preserves the inter-rater spread. Conditions are presented in a counterbalanced order, with same-size conditions never adjacent.

C.3 Form Items

Each expert submits one response per condition. The form records an identifier, the condition (BS, BM, PS or PM), the five dimension ratings on a 0–5 linear scale with the anchors of Table C.1, and an optional free-text justification. The export carries one row per expert and condition, with columns for the rater, the condition, the five dimension scores, and the comment.

C.4 Scoring

Each row's composite is the weighted sum the framework derives by principal-component analysis [78]:

$$\text{SCORE} = 0.09\,S + 0.15\,C + 0.29\,O + 0.14\,R + 0.33\,E, \qquad \text{SCORE} \in [0,5].$$

cOherence and the two context dimensions carry most of the weight, with O and E together accounting for 62% of it. Per condition, the dimension means and the composite are averaged over the panel. The environment effect is read as the basic-to-procedural contrast at each size, PS $-$ BS and PM $-$ BM, and its persistence from the small to the medium scale. With a small panel the agreement is reported descriptively, as the per-condition range across raters, not with a reliability coefficient. The limitations of this assessment (its non-blind nature, the small panel, the non-independence of group rating, and the structural inapplicability of two dimensions to a fully virtual scene) are discussed with the rest of the study's limitations in Section 5.7.

Appendix D

User Study Protocol and Instruments

This appendix reproduces the materials used to run the user study of Chapter 5: the participant briefing and consent step, the counterbalancing scheme, the questionnaire items, and the tester's answer sheet and automatic log format. Together they make the procedure of Section 5.3.6 reproducible.

D.1 Briefing and Consent

Before the headset is fitted, the tester reads a short briefing, in the participant's language, conveying the following. The session is part of a master's dissertation on immersive data visualization. It involves wearing a virtual-reality headset and answering a few short questions about animated data shown in four scenes. It takes about half an hour, including questionnaires and breaks. The participant may stop at any moment and for any reason, discomfort in particular, and stopping does not invalidate the data already collected. No personal identifying data are recorded beyond the background questions below; responses are anonymous and used only for this research. The participant then states whether they agree to take part, and the session proceeds only on agreement.

There is no deception, no sensitive data, and no physical risk beyond the ordinary discomfort virtual reality can cause, which the simulator-sickness instruments (Section D.4) monitor throughout.

D.2 Counterbalancing

Each participant is assigned to one of four sequences by their identifier, $s = ((id - 1) \bmod 4) + 1$, so that consecutive participants traverse different sequences and the four are filled equally when the sample is a multiple of four. Within a sequence each of the four conditions appears once, no two same-size conditions are adjacent, and first-order carryover is near-balanced (Table D.1).

The no-reuse requirement of Section 5.3.4 is met with two disjoint subsets, A and B, of each dataset size, drawn so that the questions asked over one cannot be answered from memory of the other. The subset-to-condition mapping alternates between sequences (columns M1/M2 of

Table D.1): under M1 the basic conditions use subset A and the procedural conditions subset B, and under M2 the assignment is swapped. With two sequences on each mapping, every subset occurs equally often in each environment condition, so any residual difficulty difference between subsets cancels at group level and does not confound the environment effect.

Table D.1: The four counterbalancing sequences. Condition codes combine environment (B = basic, P = procedural) with size (S = small, M = medium). The M1/M2 column gives the subset-to-condition mapping (Section D.2).

Sequence	Assigned to (*id*)	Position 1	Position 2	Position 3	Position 4	Subset map
1	$id \equiv 1 \pmod 4$	BS	PM	PS	BM	M1
2	$id \equiv 2 \pmod 4$	PS	BM	BS	PM	M2
3	$id \equiv 3 \pmod 4$	BM	PS	PM	BS	M1
4	$id \equiv 0 \pmod 4$	PM	BS	BM	PS	M2

D.3 Pre-Questionnaire

Administered once, before the first condition. Items 1 to 3 are rated on a five-point scale (1 = none, 5 = very high). They serve as covariates for interpreting the environment's benefit by visual literacy and prior immersion.

1. Familiarity with graphs and network (node-link) diagrams.
2. Familiarity with data visualization in general (charts, dashboards).
3. Prior experience with virtual-reality head-mounted displays.
4. Self-reported color-vision deficiency (yes / no / unsure), recorded because node identity is encoded by color.

D.4 Per-Condition and Inter-Condition Questionnaires

Raw NASA-TLX, after each condition. The six subscales below are each rated on a 0–10 scale. The raw index is their unweighted mean (Section 5.3.5).

1. **Mental demand:** how mentally demanding was the task?
2. **Physical demand:** how physically demanding was the task?
3. **Temporal demand:** how hurried or rushed was the pace?
4. **Performance:** how successful were you in accomplishing it?
5. **Effort:** how hard did you have to work?

6. **Frustration:** how insecure, irritated or annoyed were you?

SSQ, at baseline, mid-session and end. Each of the sixteen symptoms is rated 0 (none), 1 (slight), 2 (moderate) or 3 (severe): general discomfort, fatigue, headache, eye strain, difficulty focusing, increased salivation, sweating, nausea, difficulty concentrating, fullness of head, blurred vision, dizziness with eyes open, dizziness with eyes closed, vertigo, stomach awareness, and burping. Scoring follows Kennedy et al. [82] (Section 5.3.5).

FMS, between the remaining conditions. A single verbal rating from 0 (no sickness at all) to 20 (severe sickness) [83].

D.5 Answer Sheet and Log Format

For each task the tester records the participant's answer, compares it with the ground-truth key, available to the tester and consulted only after the participant commits, and marks correctness on the three-level scale (Table D.2). Response time is measured from the moment the task is posed to the moment the participant commits. The task set and ground-truth definitions are those of Section 5.3.4.

The session-management tool logs one row per task to a CSV file with the following columns: `participant_id`, `sequence_id`, `ab_mapping` (M1/M2), `condition_index`, `representation` (B/P), `size` (S/M), `subset` (A/B), `task_index`, `task_type`, `ground_truth`, `mark` (correct / almost / incorrect), `duration_s`, and the start and answer timestamps. The marking is entered live on a three-key control (one key per correctness level), and the tool also drives the condition order and holds the ground-truth key, so the recorded order always matches the assigned sequence.

Table D.2: Answer-sheet template (one group of rows per condition, in the order of Table D.1). "Mark" is C (correct), Q (almost correct, right procedure but imprecise) or E (incorrect).

Task	Type	Participant answer	Ground truth	Mark (C/Q/E)	Time (s)
1	most incoming				
2	peak instant				
3	busiest pair				
4	most outgoing				