TECHNISCHE
UNIVERSITÄT
WIEN
Vienna University of Technology

DISSERTATION

Visual Steering to Support Decision Making in Visdom

ausgeführt zum Zwecke der Erlangung des akademischen Grades eines Doktors der technischen Wissenschaften unter der Leitung von

Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Eduard Gröller
Institut für Computergraphik und Algorithmen
Abteilung für Computergraphik

eingereicht an der Technischen Universität Wien
bei der Fakultät für Informatik

von

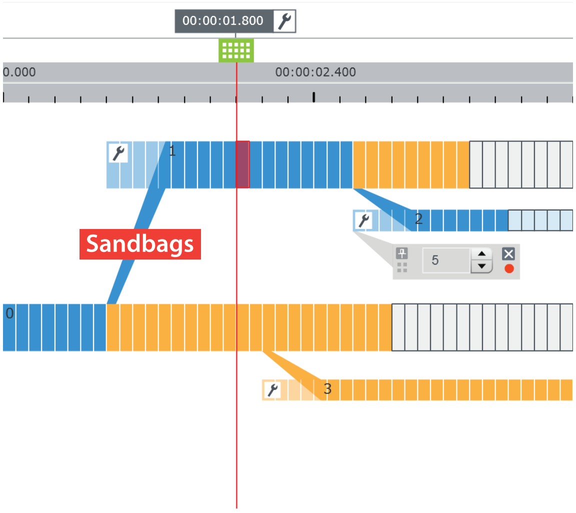
Jürgen Waser
Matrikelnummer 0026417
Nußdorferstraße 62/21
1090 Wien

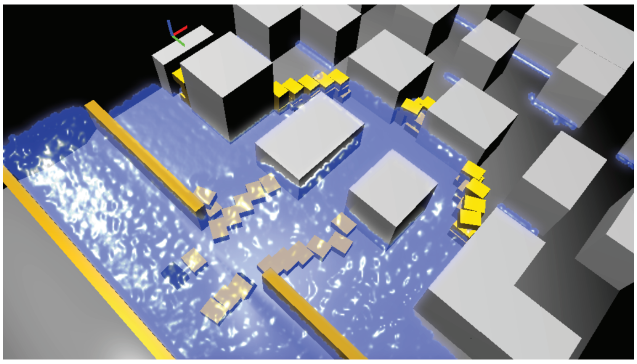
Wien, am 19. Mai 2011

Jürgen Waser

Visual Steering to Support Decision Making in Visdom

Dissertation





visdom
http://visdom.at

Kurzfassung

Computer-Simulationen werden häufig zur Untersuchung von natürlichen Prozessen herangezogen. Durch gezielte Änderung von Simulationsparametern ist es möglich, ein Verständnis komplexer Zusammenhänge zu gewinnen. Eine Software, mit der sich verschiedene Szenarien durchspielen und vergleichen lassen, kann sinnvoll als Entscheidungshilfe eingesetzt werden. Die Entwicklung leistungsstarker Simulationskomponenten ist eng mit einer steigenden Komplexität verbunden, welche durch die Präsenz einer Vielzahl heterogener Parameter gekenngezeichnet ist. Die Steuerung einer Simulation kann durch Verwendung visueller Interaktion stark vereinfacht werden, allerdings gibt es nur wenige Ansätze, die eine gezielte Verküpfung von Simulation und Visualisierung in einer Software anstreben. In der vorliegenden Arbeit werden die grundlegenden Komponenten der Visdom-Applikation vorgestellt, welche Simulation, Steuerung und interaktive, visuelle Analyse der Ergebnisse in einer einzigen Anwendung verbindet. Dadurch können die Auswirkungen alternativer Entscheidungen untersucht werden, ohne Details über die zugrundelegende Simulationstechnik wissen zu müssen. Visdom bietet zu diesem Zweck die interaktive World-Lines-Darstellung, die zur Erzeugung und Verwaltung mehrerer Simulationsläufe herangezogen wird. Die Methode verwendet bekannte Metaphern aus der multimedialen Welt zur Visualisierung von Simulationsläufen als sogenannte Tracks. Interaktiv können Benutzer neue Entscheidungen einfliessen lassen, welche in der World-Lines-Ansicht als Verzweigungen dargestellt werden. Darüberhinaus können Parameterstudien erstellt werden, die sich zur Berücksichtigung von statistischen Unsicherheiten eignen. Mithilfe mehrerer Positionszeiger werden gekoppelte Visualisierungen sowohl in der Zeit als auch zwischen alternativen Lösungen navigiert. Auf diese Weise ermöglicht das System die vergleichende Analyse mehrerer Simulationsläufe. Da der Simulationsprozess eine Vielzahl heterogener Daten erzeugt, stellen wir einen genetischen Algorithmus zur Verfügung, der die AnwenderInnen bei der Suche nach Erklärungen unterstützt. Im Kern der Applikation steht ein Datenflussmodel, welches einen hohen Grad an Modularität gewährleistet. Über einen flexiblen Kontrollfluss, welcher intern und visuell vom eigentlichen Datenfluss getrennt ist, werden die Parameter von World Lines zu den relevanten Modulen übertragen. Dabei verwenden wir dynamische Visualisierung innerhalb des Flussdiagrams, um auf relevante Steuerungsvorgänge aufmerksam zu machen. Die Anwendbarkeit des Systems wird in Fallstudien aus dem Hochwassermanagement überprüft. In einem virtuellen Dammbruchszenario besteht die Aufgabe darin, mit einer möglichst geringen Anzahl von Sandsäcken Schutzdämme zu entwerfen, die dem steigenden Wasserspiegel standhalten um eine Siedlung zu schützen.

Abstract

Computer simulation has become an ubiquitous tool to investigate the nature of systems. When steering a simulation, users modify parameters to study their impact on the simulation outcome. The ability to test alternative options provides the basis for interactive decision making. Increasingly complex simulations are characterized by an intricate interplay of many heterogeneous input and output parameters. A steering concept that combines simulation and visualization within a single, comprehensive system is largely missing. This thesis targets the basic components of a novel integrated steering system called Visdom to support the user in the decision making process. The proposed techniques enable users to examine alternative scenarios without the need for special simulation expertise. To accomplish this, we propose World Lines as a management strategy for multiple, related simulation runs. In a dedicated view, users create and navigate through many simulation runs. New decisions are included through the concept of branching. To account for uncertain knowledge about the input parameters, we provide the ability to cover full parameter distributions. Via multiple cursors, users navigate a system of multiple linked views through time and alternative scenarios. In this way, the system supports comparative visual analysis of many simulation runs. Since the steering process generates a huge amount of information, we employ the machine to support the user in the search for explanations inside the computed data. Visdom is built on top of a data-flow network to provide a high level of modularity. A decoupled meta-flow is in charge of transmitting parameter changes from World Lines to the affected data-flow nodes. To direct the user attention to the most relevant parts, we provide dynamic visualization inside the flow diagram. The usefulness of the presented approach is substantiated through case studies in the field of flood management. The Visdom application enables the design of a breach closure by dropping sandbags in a virtual environment.

Contents

Preface xi

1 Introduction 1
- 1.1 Flood Management . . . 1
- 1.2 Flood Simulation . . . 5
- 1.3 Interactive Decision Making . . . 7
- 1.4 Scope of this Thesis . . . 8

2 World Lines 13
- 2.1 Introduction . . . 13
- 2.2 Related Work . . . 14
- 2.3 Overview on World Lines . . . 17
- 2.4 Visual Representation of Related Simulation Runs . . . 18
- 2.5 Navigating the Multi-View System . . . 20
- 2.6 Steering Mode of World Lines . . . 22
- 2.7 Interactive Visual Analysis . . . 27
- 2.8 Evaluation . . . 32
- 2.9 Implementation . . . 40
- 2.10 Discussion . . . 40

3 Multiverse Data-Flow Navigation 45
- 3.1 Introduction . . . 45
- 3.2 Problem Description . . . 46
- 3.3 Proposed Solution . . . 49
- 3.4 Related Work . . . 51
- 3.5 The Multiverse Data-Flow Navigation Cycle . . . 52
- 3.6 Visualization of Capabilities . . . 54
- 3.7 Interactive Job Assignment . . . 56
- 3.8 Scope-based Settings . . . 61
- 3.9 Discussion . . . 63

4 Nodes on Ropes 67

A picture is a fact.
— *Ludwig Wittgenstein*

4.1 Introduction . . . 67
4.2 Problem Description . . . 69
4.3 Proposed Solution . . . 70
4.4 Related Work . . . 72
4.5 Parameter Studies with World Lines . . . 74
4.6 Meta-flow . . . 75
4.7 Dynamic Visualization with the Flow Diagram . . . 78
4.8 Evaluation . . . 88
4.9 Implementation . . . 94
4.10 Discussion . . . 95

5 Visual Human+Machine Learning **99**
5.1 Introduction . . . 99
5.2 Related Work . . . 101
5.3 Visual Learning . . . 102
5.4 Hypotheses . . . 108
5.5 Machine Learning Algorithm . . . 112
5.6 Evaluation . . . 114
5.7 Implementation . . . 117
5.8 Discussion . . . 118

6 Summary and Conclusions **121**

Bibliography **125**

Curriculum Vitae **136**

I was born not knowing and have had only
a little time to change that here and there.
— *Richard P. Feynman*

Preface

THIS thesis is a summary of work carried out from 2008 to 2011 at the VRVis Vienna. I would like to thank my supervisors, Meister Eduard Gröller and Raphael Fuchs, for their valuable input, inspiration and kind guidance. Meister is both a solution-oriented, professional researcher and a man with extraordinary, enjoyable ideas, e.g., making me play songs from Igor Presnyakov during my conference talks and asking for pictures of myself with crazy looks. Raphael is an innovative scientist who has laid the foundation for this work. He is co-founder of the Visdom [2] system which is jointly being developed at the VRVis Vienna and the ETH Zürich. Without the Visdom team, this work would not have been possible. Benjamin Schindler constantly applies his profound software-engineering skills to improve the tool. Robert Carnecky is contributing indispensable rendering technology to the system. Special thanks to Hrvoje Ribičić, my co-worker at the VRVis, who is not only an exceptional programmer and scientist, but also an ambitious colleague who keeps helping out even if he should already be sleeping. Deadlines may keep you awake, it is *nice* and *productive* to share this experience. I very much appreciate the collaboration with Günter Blöschl who is our expert in flood forecasting and management systems. I would also like to pay tribute to a theoretical physicist and friend, Christoph Mayrhofer, who helps me refresh my knowledge in physics through stimulating discussions. I owe the pleasing design of the Visdom logo and the Visdom website to the creative mind of Anneliese Heinzl. I thank Stefan Bruckner for providing me with his aesthetic latex template of his PhD thesis. Finally, I want to thank my beloved Angela. She is there whenever I need her, making this life sweet.

This work was supported in part by a grant from the Austrian Science Fund (FWF):P 22542-N23 (Semantic Steering).

Jürgen Waser

Vienna, May 2011

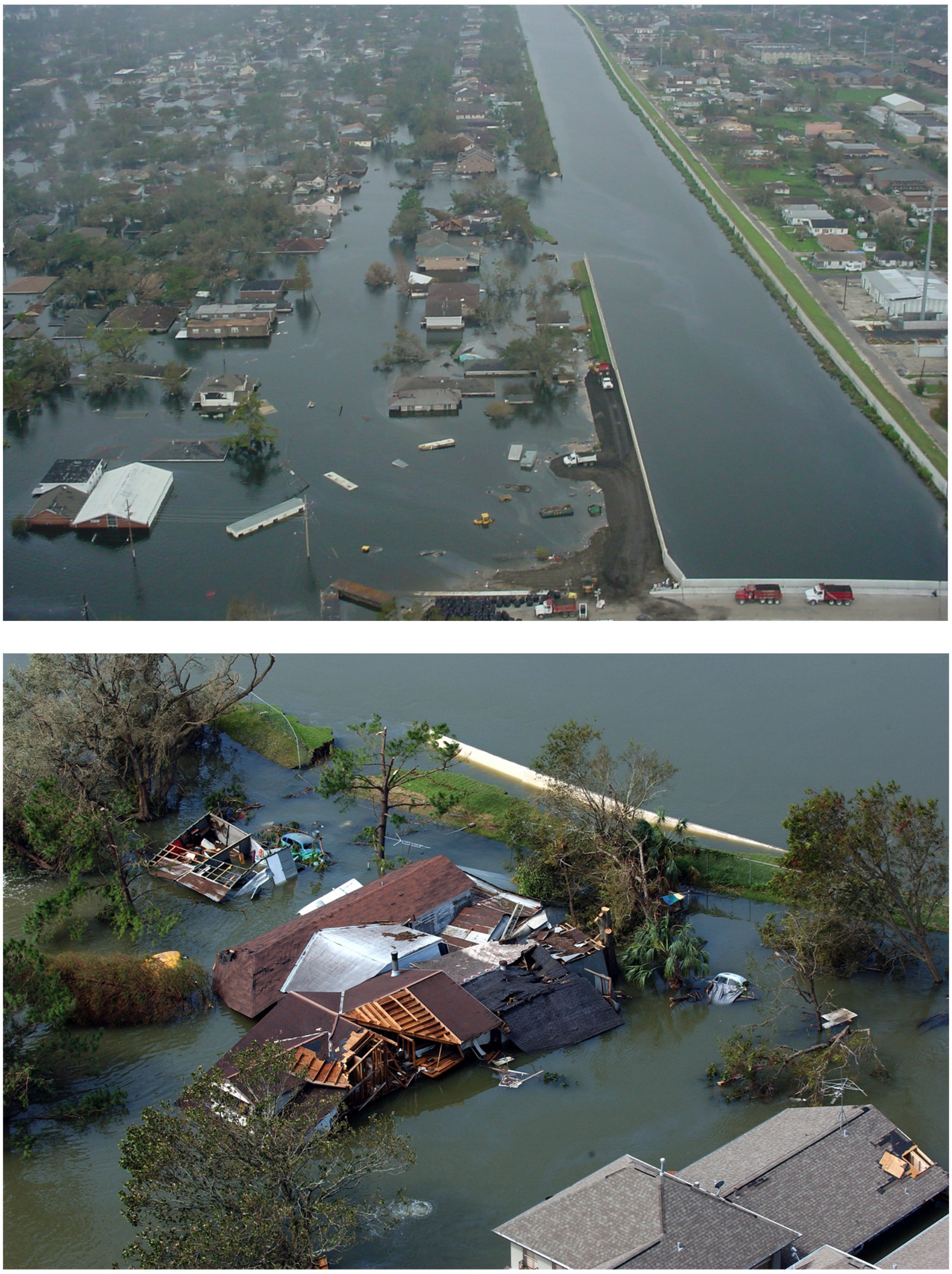


Figure 1.1 – New Orleans 2005: In the aftermath of hurricane Katrina, a levee-breach causes city flooding [132]. The effects on the neighborhood are disastrous.

Floods are 'acts of God,' but flood losses
are largely acts of man.
— *Gilbert F. White*

CHAPTER 1
Introduction

Flood disasters are the most common natural risk and tremendous efforts are spent to improve their simulation and management. However, simulation-based investigation of actions that can be taken in case of flood emergencies is rarely done. This is in part due to the lack of a comprehensive framework which integrates and facilitates these efforts. This chapter introduces the levee-breach scenario that is investigated in several parts of the thesis. We provide an overview on flood-simulation technologies and the requirements on interactive steering to make a feasible decision support system.

1.1 Flood Management

THE Center for Research on the Epidemiology of Disasters (CRED) has pointed out floods as the most common natural disaster in 2010 [26]. The problem is that most of the populated areas in the world are vulnerable to flood disasters. World-wide, floods are likely to become increasingly severe and more frequent due to climate change [101], population growth, deforestation or change of land-use. We cannot prevent natural hazards but we can prepare to keep damage as low as possible. Simulation technology can help to get a better understanding of these natural phenomena.

Our flooding scenario is based on the events in New Orleans that occured in the aftermath of hurricane Katrina in August 2005. New Orleans is situated below the sea level and highly vulnerable to flood disasters. For this reason, the city is protected by a series of flood walls and levees channeling the flow of water. After the passing of hurricane Katrina, multiple breaches in the flood protection system took place and more than 80 percent of the city was submerged in water [132]. One of the major breaches occured on the 17th Street Canal and was responsible for most of the flooding (Figure 1.1). The U.S. Army Corps of Engineers (USACE) tried to close the breach by dropping heavy sandbags with a helicopter (Figure 1.2). However, in their initial attempt, the bags were dumped too close to the breach and consequently were washed away. Multiple attempts were needed before the army was able to finally close the breach.

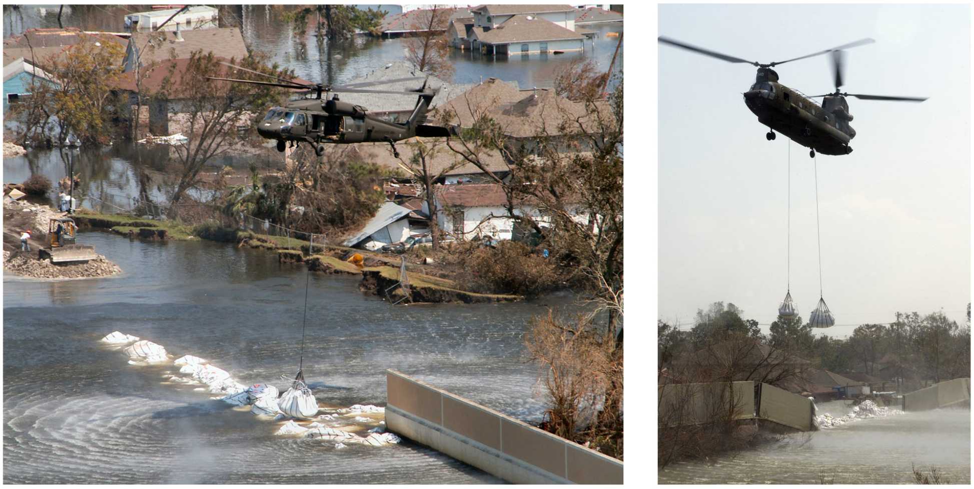


Figure 1.2 – Helicopters drop sandbags to seal the levee-breach in a trial-and-error approach. Initially, the sandbags are dumped too close to the breach and are washed away [132].

Such trial-and-error procedures are commonly applied in case of emergencies due to the lack of systematic research in the field of actions that can be taken during a flood event. For this reason, Sattar et al. [112] have recreated the levee-breach situation of New Orleans using a laboratory model on a 1:50 scale (Figure 1.3). In addition, the researchers have investigated various possible methods for breach closure utilizing different procedures such as the construction of multi-barrier embankments. In this manner, they were able to find strategies that would have worked in New Orleans. Even though such a real-world simulation is quite accurate and gives reasonable results, it is tedious and time-consuming to set up and to test alternative options. Even if changes are only small, it is hard to explore how various choices taken at a certain time affect the state of the system. This is due to the difficulty of returning the setup to a previous state.

For these reasons, we utilize the presented Visdom system [2] for the investigation of breach-closure procedures in a virtual environment. Figure 1.4 shows screenshots of our scenario that is loosely based on the events in New Orleans. The scene comprises a neighborhood that is protected by levees. The fluid behavior is simulated with a particle-based technology that is amendable for parellalization on the graphics processing unit (GPU). Users can interactively drop sandbags into the simulation domain in order to test alternative arrangements. The sandbags are modeled as simple bricks to achieve a real-time experience with respect to simulation steering. The virtual environment features the advantage of saving and restoring system states. As a consequence, the user can go back to any point in time and alter the barrier arrangements as desired. Compared to the real-world simulations, the computational approach enables the exploration of alternative choices in a fast and efficient manner.

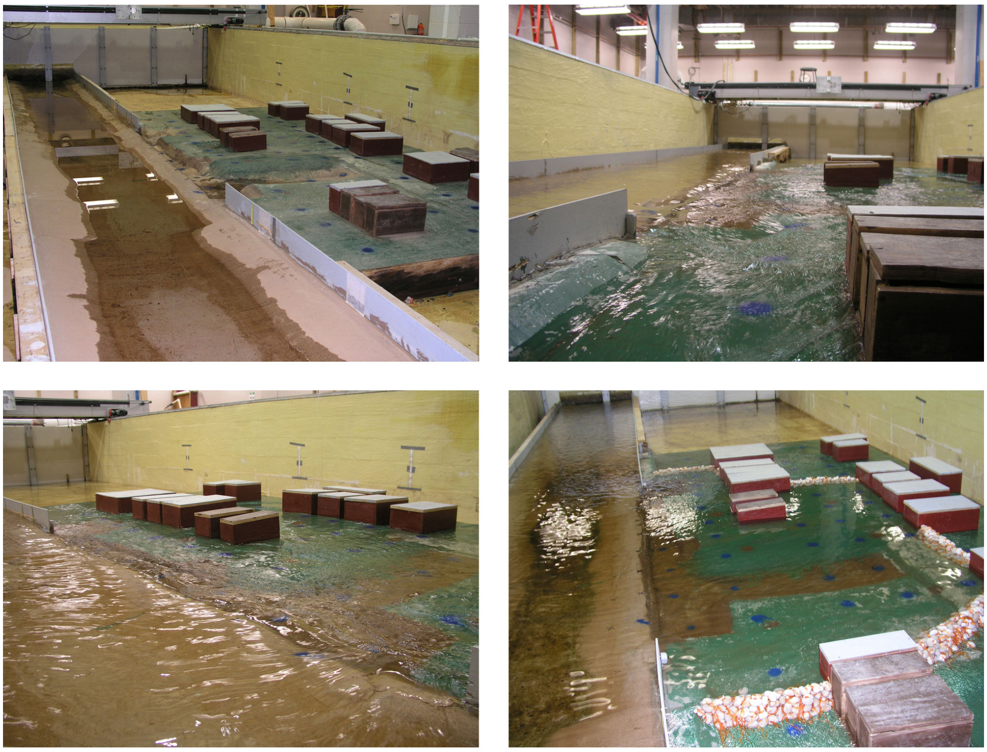


Figure 1.3 – Recreation of the levee-breach scenario in a hydraulic laboratory. Researchers study alternative barrier arrangements to accomplish a successful breach closure [112].

The key to a successful decision support is the ability to test and explore alternative options.

In collaboration with domain experts we have identified the following potential practical applications of the presented system: The usage as an *offline*-tool for planning or training as well as an *online*-tool which is applicable in case of emergencies [138].

Offline planning The system can be used to study different actions that can be taken during a flood event prior to the actual occurence. The gained knowledge supports the creation of flood management plans as required by the new EU flood management directive [39]. In this setting, the goal of decision making is not to find the optimum solution but to rapidly exclude solutions that are not robust. In our levee-breach scenario, a well-performing solution is given if the respective response measures minimize the adverse effects on the neighborhood and its inhabitants. Such a solution can be characterized as robust if the solution still leads to reasonable effects even if we modify specific input parameters of the simulation. The investigation of such modifications is necessary since we need to account for the uncertainty in the forecasts and management implications. Uncertainty can come from uncertain

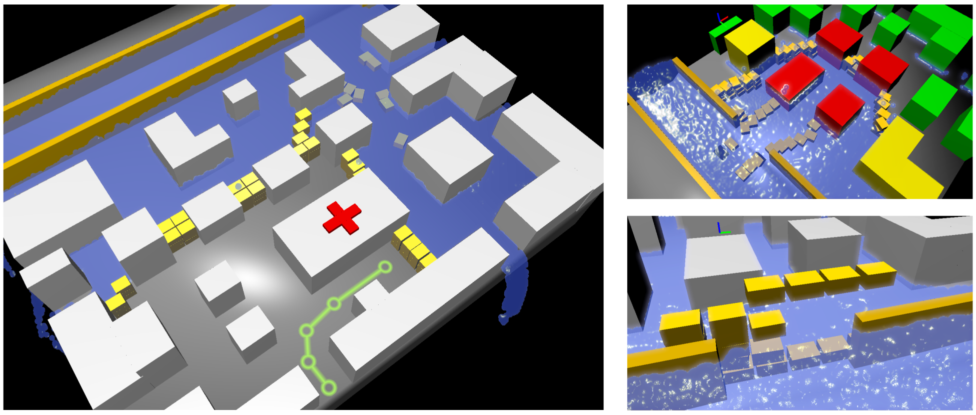


Figure 1.4 – Investigation of breach-closure methods in a virtual environment as proposed in this thesis. Users can drop sandbags to test alternative arrangements. A particle-based simulation component is utilized to evaluate the effects of decisions in real-time.

weather predictions and uncertain locations of levee breaches, among other factors. This approach is relatively new in flood management [74] and has been made possible by meteorological ensemble forecasts that became available for operational purposes at the beginning of this century. To customize the system for the problem at hand, flood managers require the ability to modify the simulation setup, because real-world applications differ case-by-case. Domain experts need to account for different input parameters and varying criteria for cost-benefit and risk analysis.

Offline training The application can be used for the training of flood-emergency personnel in an offline mode. Staff members can practice alternative response actions without the need for special simulation expertise. This way, they become better prepared to deal with the consequences of flooding. The training of knowledge and skills is also the goal of serious computer games. The Levee Inspection Simulator [32], for example, is a serious game in which levee patrollers are required to inspect the condition and stability of levees in a 3D environment. However, serious games do not allow the trainee to investigate and compare alternative decisions in an interactive visualization environment.

Online decision making The system shows potential for real-time usage during a flood event. Our future vision is that, even under time-critical circumstances, emergency personnel on-site will be able to analyze the imminent situation quickly to choose the best response strategy (Figure 1.5). Suppose we have a severe weather warning and know the dams of the city could break. Using a hand-held device, first responders are capable of loading the local geometrical and hydrological data into the remote system and start flood simulation from scratch. The imminent flood

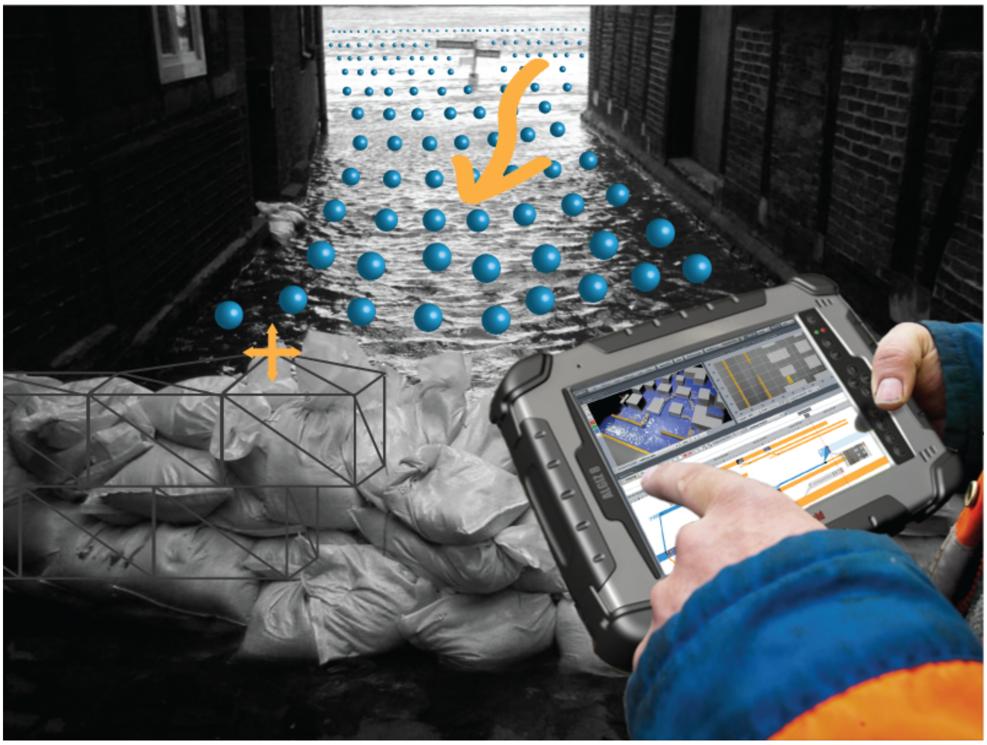


Figure 1.5 – Future application that envisions the presented system in an online mode during a flood event (mock-up). A first responder on-site consults the mobile client of the Visdom application for designing a temporary flood protection. The remote server is in charge of evaluating the compute-intensive parts of simulation and visualization.

is computed quickly using a fast simulation method like the smoothed particles hydrodynamics technique (SPH). In an interactive cycle, the analyst places barriers to evaluate possible measures and performs additional simulations on the updated city geometry. Later the weather conditions change unexpectedly and quick responses are necessary. The analyst modifies the respective boundary conditions, triggers a new simulation and concurrently, the visualization shows where the flooding risk increases most.

1.2 Flood Simulation

The simulation of fluids has come a long way in the last decades. In this work, we require a flexible simulation tool that provides a good tradeoff between speed and accuracy. In this section, we describe the state of the art in flood simulation from an application perspective. We will give a short description of the advantages and shortcomings of the relevant methods to substantiate our choice of SPH.

One- or two-dimensional hydraulic models can be considered as the standard for flood simulation. To calculate the temporal evolution of flooding waves, including routing and arriving time, and their interaction with geometric structures we have to solve the Navier Stokers equations (NS) for incompressible flows. The area of computational fluid dynamics (CFD) is concerned with the development of sophisticated numerical schemes to solve the NS-equations. Traditionally, engineers apply techniques like the finite volumes method, finite elements (FEM) or the finite difference method to solve the involved partial differential equations. Even though these techniques provide good results for the simulation of floods [47, 129], they have one major drawback: they are too slow for time-critical applications. Computations take between days and months in order to get results, even more so if multiple scenarios are involved. To tackle the problem of very long computation times in traditional CFD, we need an alternative approach. Mueller et al. [94] state that less accurate methods which allow for the simulation of fluid effects in real-time open up a variety of new applications. During the design phase, real-time methods help to test whether a certain decision is promising. Blöschl et al. [20] have developed a fast flash-flood forecasting model which is in operational use. Urban inundation simulations close to real-time performance have been realized with the two-dimensional shallow water equations [42, 145]. Kass and Miller [66] were the first to use the shallow water equations in computer graphics. As of today, this simplified model, which assumes depth averaged fluid properties, has become an important tool for large-scale inundation simulation. The method and its extensions have been largely adopted for flood and dam-break simulations [81, 15].

As with any approximation, the shallow water equations are limited and thus not applicable to every problem: the equations describe one vertical level only, so they cannot directly encompass any factor that varies with height. Thus, we can apply the shallow water equations in order to get an overview of the situation (large-scale, where 3D is not important) but not for details. For more accurate calculations, we have to simulate in three spatial dimensions. The lattice Boltzmann method (LBM) is a fast cell-based technique for viscous fluid dynamics. The advantages of LBM over other CFD techniques is that complex boundaries are relatively easy to implement [123], however, the technique is not suitable if large Reynolds numbers are involved. Particle based fluid simulation methods are commonly preferred to Eulerian fluid presentations when it comes to interactive applications. The reason is that these methods do not require the generation of grids which are costly in terms of modelling, memory and computation. Also, geometric boundaries do not need to be voxelized. For this work, we propose the widely-used smoothed particles hydrodynamics (SPH) technique [92, 77]. SPH delivers interactive timings for small particle numbers [94] and, as Kipfer and Westermann [71] show, SPH can provide a realistic appearance for environmental flood simulation. Ghazali and Kamsin [46] illustrate that SPH can be used to model flash-flood behavior with adequate realism. Chatelain [27] shows a large-scale simulation using remeshed smoothed particle hydrodynamics. At present, there are GPU accelerated fluid simulation modules available for free on the web: Hoetzlein [58] offers an open source GPU implementation of SPH. The open source

GPU-SPHysics [55] code has been used to model water waves and dam breaks [56] as well as water flooding with various types of levee failures [31]. NVidia provides a free and efficient GPU implementation of SPH in their PhysX package [100]. For this work, we have integrated PhysX as the simulation component. This package implements all functionality that is necessary to handle the dynamic boundary conditions as induced by the sandbags. We point out, that the Visdom system is highly modular, so SPH can easily be replaced by alternative hydrodynamic models. This is important for practical applications because most institutions have a record of their favourite methods they are familiar with.

1.3 Interactive Decision Making

In the previous section, we have discussed simulation techniques which are used to investigate the dynamics of fluids. In this section, we describe related work about utilizing simulations to support the process of decision making. Computational steering is a powerful concept that enables domain experts to interact with a simulation during its execution. Today, the work flow of computational simulations is increasingly demanding to the user since simulations become more and more complex, comprising many different input parameters and large amounts of heterogeneous data results. This is especially true for computational fluid dynamics (CFD) where the traditional work flow is to prepare input, to execute a simulation, and to visualize the results in a post-processing step. However, more insight and a higher productivity can be achieved if these activities are done simultaneously. This is the underlying idea of simulation steering: researchers change parameters of their simulation on the fly and immediately receive feedback on the effect [116, 102, 125, 135]. In this work we try to take this approach one step further: Researchers change parameters of their simulation on the fly and can then analyze both the original outcome and the alternative interactively. The key to a successful decision support is the ability to compare multiple simulation runs in an integrated steering environment.

The absence of integration of different methods and systems and their incompatibility make the development of integrated solutions the issue of the day. Until now there is little focussed research in this direction and the same is true for interactive simulation tools. There are some efforts to combine web technologies and visualization with a simulation system. The DHI-Group distributes a web-based decision support system with a scenario editor and a simulation editor [34]. The scenario editor helps in setting up the simulation while the simulation editor provides examination of simulation details. However, there is no visual interactive environment to perform these tasks. Other web-based tools provide 2D visualizations of the results. Yamaguchi incorporates a geographical information system (GIS) to determine the geometric boundaries of the simulation [145]. Jo has implemented a web-based flood-management system [62] for coastal regions. Liu [82] describes a web-based system where design decisions are facilitated by a collaborative, web-based tool including visualization using the VTK toolkit. Kim et al. [70] have presented a fire evacuation system based on mobile devices.

In general, these tools lack the ability to setup and compare alternative scenarios in an intuitive way. Users need to be able to pose ’What if’ questions to the system. In the context of our scenario, these questions include: What happens if the levee-breach occurs three meters further down the river? Are the sandbags going to be washed away? Is the evacuation path safe even if the river velocity increases? The underlying simulation system should then compute an answer to each of these questions, generating a potentially large number of solutions that we term *parallel worlds*. A filter mechanism is then needed, that helps the user to pick the best solution out of these parallel worlds. Interactive visual analysis (IVA) is a concept that supports the user in this process. The goal of IVA is to help users understand what the simulation data means. If we employ this technique in a comparative way, we can support the user in making the right decisions.

1.4 Scope of this Thesis

In this work, we present an integrated, modular decision-support system that allows the user to steer and learn from simulations in an intuitive way. Our objective is to establish an integration of methods from visualization, computational steering and simulation. The most important components that have been developed in the course of this work are summarized in Figure 1.6. Chapter 2 introduces a visual management strategy for simulation runs which we term *World Lines* (Figure 1.6a). This concept provides a concise overview of multiple related simulation runs to let users create, navigate and compare alternative scenarios. The World Lines view is part of a system of multiple coordinated views. These views include interactive and comparative 3D renderings of the scene (Figure 1.6b-c) as well as steering monitors for entering input parameters such as sandbag positions (Figure 1.6d). This steering environment is built atop of a modular data-flow network (Figure 1.6e). In Chapter 3 we explain how to use World Lines to navigate the data-flow nodes in time and across multiple simulation runs. Chapter 4 shows how parameter changes can be transmitted from World Lines to the affected nodes. For this purpose, we extend a standard data-flow with a *meta-flow* of steering information which has a visual representation in an augmented flow diagram. To substantiate the usability of this approach in a practical context, we show how to create parameter studies with World Lines in order to account for uncertain knowledge with respect to simulation input parameters. Chapter 5 illustrates how the reasoning process of the user can be supported by the machine. The suggested approach utilizes machine learning algorithms to speed up the visual search for explanations in a large set of heterogeneous simulation results (Figure 1.6f).

One of the most ingenious moments in 1980’s cinema is the interpretation of time travel in *Back to the Future Part 2*. The film makes use of the space-time continuum to explain the theory and pitfalls of time travel. This is based on a real physics theory

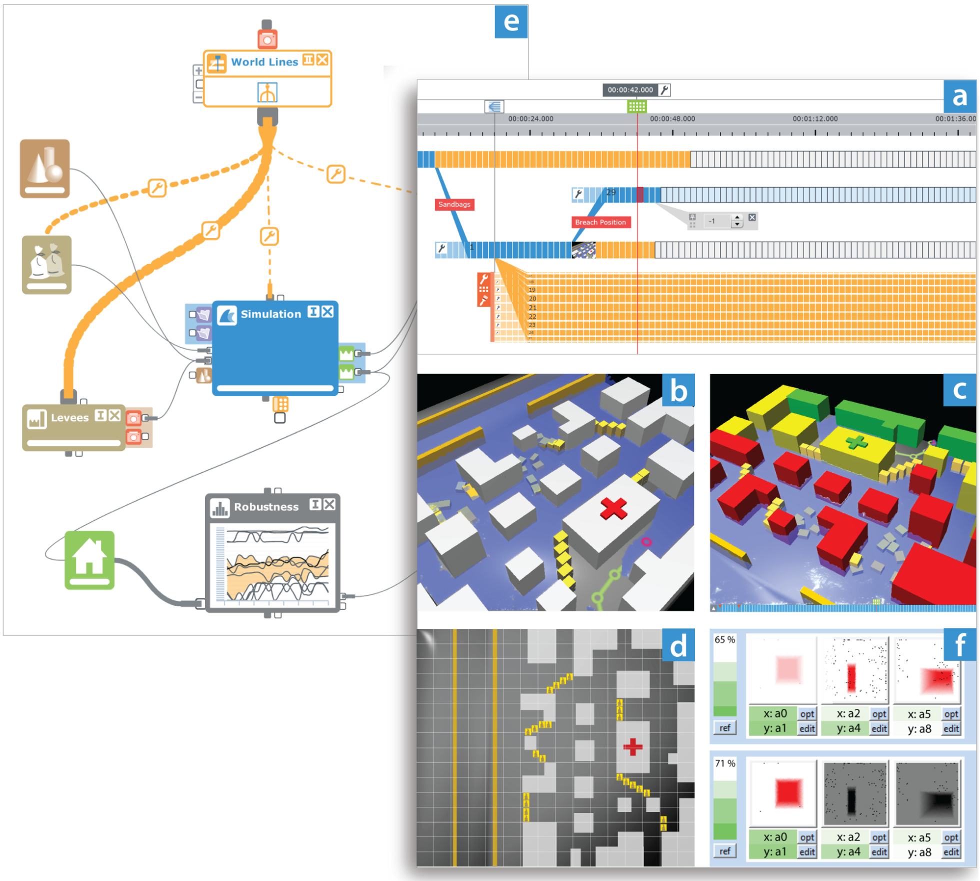


Figure 1.6 – Overview on the contributions of this thesis. (a-d, f) The system comprises multiple linked views for entering simulation input parameters and for analyzing simulation results. (a) At its center stands a management strategy called World Lines which represents multi-runs as tracks. (f) A machine learning algorithm generates hypotheses to support the search for explanations in simulation results. (e) The underlying data-flow network is extended by a meta-flow of steering information.

where space-time is a mathematical model that describes time and space as a single continuum. The brilliance of its use in the movie shines when Doc Brown depicts one of his hypotheses on a blackboard in the deserted Hill Valley library (Figure 1.7). He points out how the antagonist Biff disrupted the space-time continuum when he stole the time machine to bring back the sports almanac to the year 1955, thereby creating an alternate version of 1985. To illustrate his theory, Doc Brown sketches time as a line, labelling relevant points as the past, the future and the year 1985, which is the present in the movie. ”Somwhere in the past, the timeline skewed into this tangent creating an alternate 1985”, says Doc Brown while he draws a second time line that visually originates from the first line at the point in time, when Biff gave the almanac

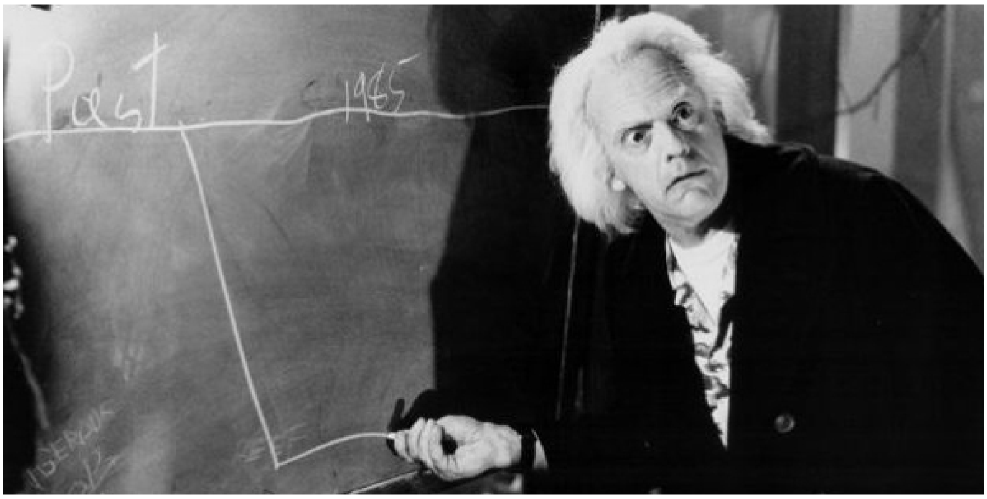


Figure 1.7 – Doc Brown, the inventor of the time machine in the movie *Back to the Future Part 2*, sketches history timelines to illustrate the creation of an alternative world.

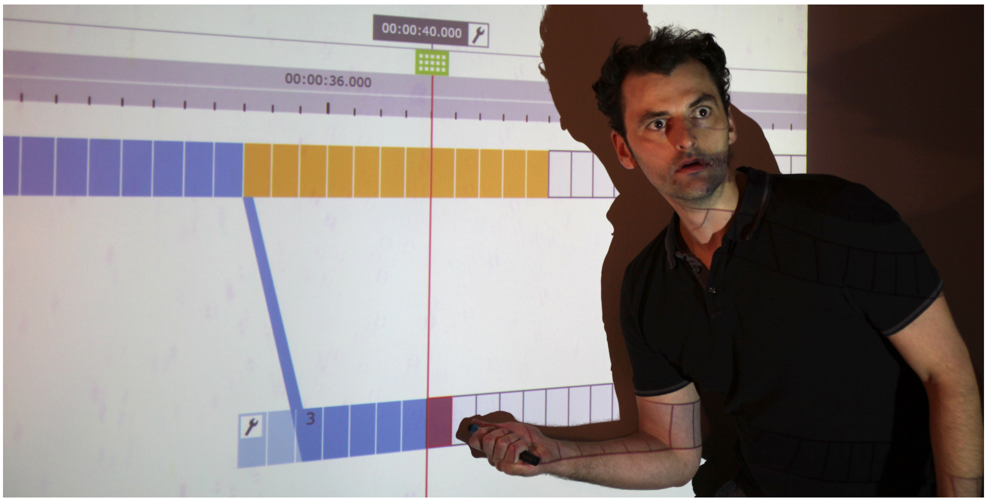
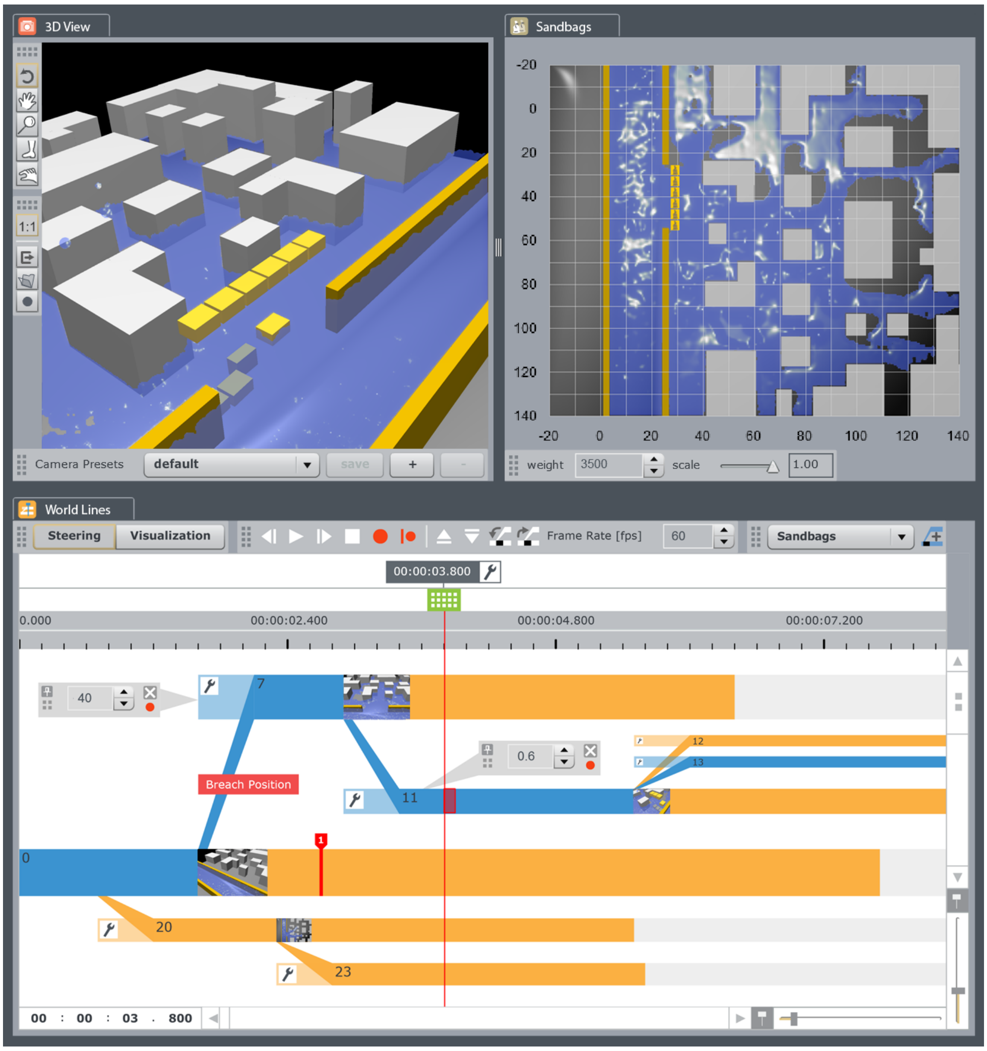


Figure 1.8 – Doc Waser, the author of this thesis, utilizes World Lines to investigate alternative futures.

to his younger self. The design of the World Lines visualization is based on this simple 2D representation of time and parallel worlds (Figure 1.8). In the next chapter, we show how to use this visualization to control multiple heterogeneous simulation runs.



Camera Presets default

weight 3500 scale 1.00

Frame Rate [fps] 60

Sandbags

00:00:03.800

00 : 00 : 03 . 800

Figure 2.1 – Screenshot of the Visdom application. World Lines is the central component for the management of multi-runs in a horizontal tree-like visualization. In combination with linked monitors, the tool enables the user to solve complex problems such as the design of a breach closure.

> Nothing is more difficult, and therefore more precious, than to be able to decide.
>
> — *Napoleon Bonaparte*

CHAPTER

2 World Lines

In many application areas, decisions can only be made by exploring alternative scenarios. The goal of World Lines is to support users in this decision making process. In this setting, the data domain is extended to a set of alternative worlds where only one outcome will actually happen. World Lines integrates simulation, visualization, and computational steering into a single unified system that is capable of dealing with the extended solution space. World Lines represents simulation runs as causally connected tracks that share a common time axis. This setup enables users to interfere and add new information quickly. A World Line is introduced as a visual combination of user events and their effects in order to present a possible future. To quickly find the most attractive outcome, we suggest World Lines as the governing component in a system of multiple linked views and a simulation component.

2.1 Introduction

In the last decade, computational simulation has experienced a tremendous progress. Computer hardware and simulation techniques have developed beyond what has been considered possible in many respects. Today, computational simulation is an ubiquitous tool in industry and research. But already new modes of application are in demand. Instead of performing a single simulation, users want to study multiple related simulations at once. They want to change input parameters in order to understand their impact. To study the influence of the relevant parameters, users need to be able to go back to any point in time to alter or refine their choices, to modify the simulation setup and trigger additional simulations.

In many cases the exact development of the situation cannot be predicted, instead, multiple scenarios must be considered (Figure 2.2). In such cases valid solutions can be found only by comparing a set of different simulation runs and analyzing the alternative scenarios they represent. This introduces an extended space of possibilities: instead of a single simulation run, users are confronted with a whole range of related, parallel worlds. Such an environment, where the user is able to pose 'what if?' questions to a simulation framework, are one step in the direction of the problem-solving environment which Johnson [63] has identified as one of the most important research problems in scientific visualization.

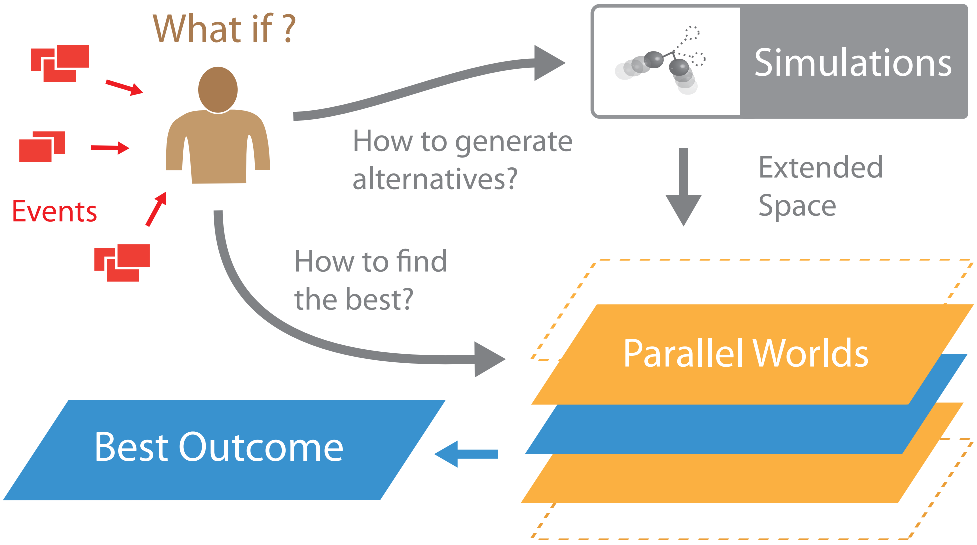


Figure 2.2 – Problem Description: The investigation of a time-dependent problem comes down to a series of 'what if?' questions. Real-world events (red boxes), often unpredictable, further require to study alternative scenarios in order to be able to make decisions. The user needs a concept to effectively steer a simulation system to produce a set of required parallel worlds. Moreover, this concept should allow the user to quickly filter the multitude of alternative solutions to find the best outcome.

The combination of steering with visualization has been a common goal of the visualization research community for twenty years, but it is rarely ever realized in practice [87]. This is in part due to a missing concept to abstract the management for generation, storage, and visualization of data describing multiple alternative scenarios. The World Lines approach (Figure 2.3) which we present in this chapter integrates simulation, visualization, and interactive analysis into a unified system. Users interact with the World Lines view to create and navigate through multiple simulation runs. The user is not required any special simulation expertise since the complexity of the underlying simulation system is hidden. World Lines employs linking and brushing to enable comparative visual analysis of multiple simulations in linked views. Analysis results can be mapped to various visual variables that World Lines provides in order to highlight the most compelling solutions.

2.2 Related Work

In this chapter we discuss a novel visualization approach to steer, visualize and solve problems based on the simulation of many possible worlds. Here, we present some of the related work in these fields.

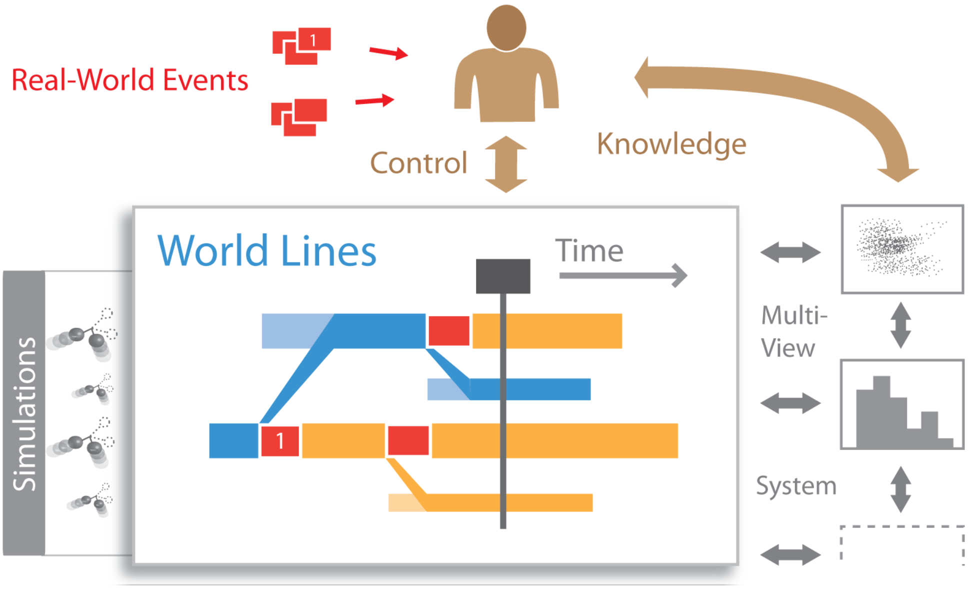


Figure 2.3 – Proposed Solution: We suggest a novel view called World Lines that enables control over multiple simulation runs, hides complexity and is capable to deal with the extended space. World Lines is part of a system of multiple linked views that enables interactive comparative analysis across alternative worlds.

Simulation Steering Mulder et al. [93] give a survey of simulation-steering environments. They stress that the user interface is a critical component of a computational steering environment. Johnson et al. [64] point out major topics when building a simulation-based problem solving environment: control structures, data distribution, data presentation, and user interfaces. Treeck et al. [136] present a steering system that enables modification of geometry via basic transformations. Matkovic et al. [87] suggest to combine CFD simulation and visualization by writing out multiple simulation runs as ensemble data and comparing these runs using the COMVis System. Kreylos et al. [78] state some basic ideas for interactive visualization and steering of CFD simulations. During a long simulation run, this system enables the user to specify the region where grid refinement is needed. They were able to speed up the prototyping process by allowing the designer to early see the first results from a multidimensional simulation space and to quickly go back into the simulation and request more runs in particular parameter regions of interest.

History, Provenance and Processes GRASPARC [21] and Hyperscribe [144] identify the ability to preserve states as an important feature for iterative problem solving. A history tree records information as the simulation progresses so that the calculation can be stopped and rolled back to previous points in time. A modified set of input parameters can be specified in order to restart the simulation and create a branch

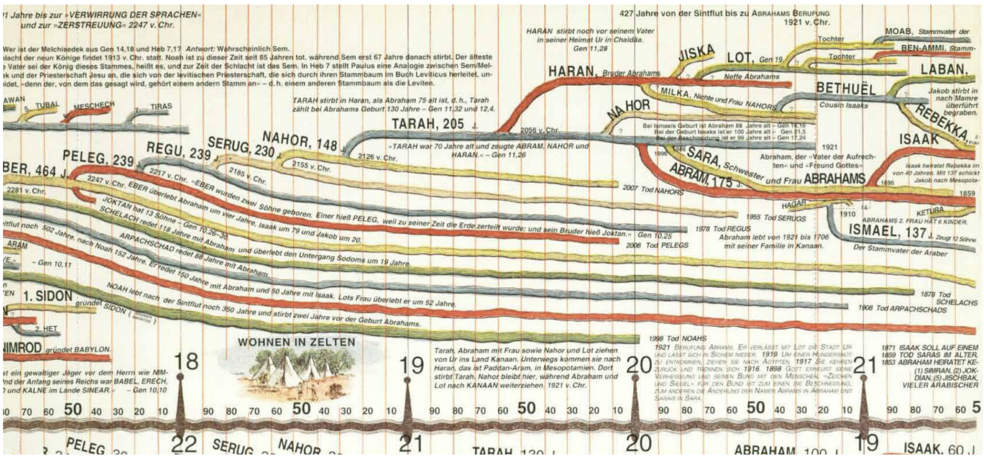


Figure 2.4 – Excerpt from the Victorian wall atlas that utilizes history timelines to depict a genealogical tree [4].

point in the tree. This history tree is visualized as a set of colored spheres connected via cylinders. Various project management solutions use line-based visualizations to display processes [7, 45]. AsbruView [75, 76] provides a temporal view that presents a plan hierarchy in a tree view similar to those used in file managers. In an additional topological view, each plan is displayed as a track. The Victorian wall atlas [4] illustrates a genealogical tree as thick history lines in a horizontal layout (Figure 2.4). VisTrails [120] adapts a history tree for capturing and reusing provenance in a visual exploration system. Graph-based layouts have been adopted for data exploration [84] and process visualization [88, 107]. Business process visualization deals with complex events which have to be monitored, steered and optimized [83, 97, 124]. In the existing graph-based or history-tree approaches, a rather sparse representation is used. This is useful if a broad spectrum of different processes has to be visualized that can consist of many different activities [50]. Multichronia relies on such a graph-based visualization to represent multiple simulation runs as nodes in a tree [108]. In the setting of this work, the basic component is given by a time step in one simulation process. There are no different activities within this process, but many continuous time steps are automatically generated by the simulation. Therefore a dense representation is required.

Interactive Analysis of Simulation Data The goal of interactive visual analysis (IVA) is to help users understand what the data means. IVA copes with complex multivariate and multidimensional data sets, by including the power of the human cognition [146]. The analyst needs to search for complex, often hidden correlations and interplays between data items. The visual information-seeking mantra - overview first, filter, zoom in, details on demand - as defined by Shneiderman summarizes the main idea [118]. Using multiple, interactively linked views of the same data set allows the

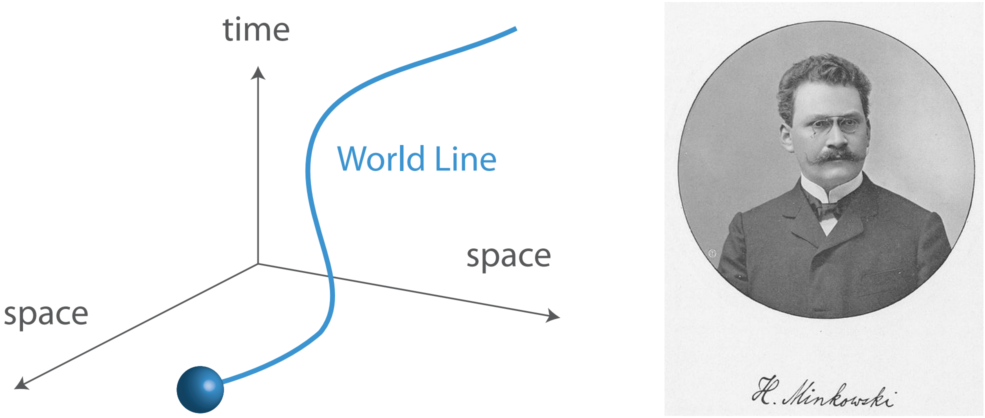


Figure 2.5 – In his famous talk ”Raum und Zeit” in 1908, Hermann Minkowski introduced time and space as intermingled entities in a four dimensional space-time continuum which is known as Minkowski space-time since then [91]. He was the first to coin the term World Line as the path of an object through this continuum.

user to productively combine the information gathered from different views [54, 110]. Weaver [140, 141] shows that, as the number of linked views increases, it may be necessary to visualize the structure and operation of the visualization. Linking and brushing allows the user to select an area or parameter range of interest by interactively placing selections on a rendering. Other views and interactions are linked to the selections and focus on information related to the selected subset. Hauser [51] states that as soon as a notion of interest in some subset of the data is established, we can visualize the selection in full detail while reducing the amount of visual information about the remaining data. Doleisch et al. [36] apply multiple linked views to the analysis of CFD data. For additional information we refer to the related state of the art report on the visualization of multi-variate scientific data [43].

2.3 Overview on World Lines

The concept of World Lines originates in physics [91]. It is a general way to represent the course of events. In their original setting, World Lines describe the movements of objects through space-time (Figure 2.5). In a more general setting, we can consider a World Line as a description of changes to system states over time. From this perspective, the entries of a journal, a sensor-log or the course of a simulation all have a corresponding representation as a World Line. Later, the concept of World Lines was extended to include the uncertainties of quantum-mechanics events, allowing a set of World Lines to describe a multitude of possible alternative worlds that can emerge in a probabilistic setting [41]. This extended version of World Lines now allows us to represent concepts such as alternative choices, uncertain outcome of events and even equivalent results obtained by different chains of events and decisions.

We start with some definitions. A *state-space* is the space spanned by all variables of a system including positions, time values or internal object states. A *frame* is a representative of one point in state-space, for example a simulation result at a given point in time. A *track* is a consecutive set of frames. For example, a single simulation run that comprises a specific set of simulation parameters is represented as such a track. An *event* is a modification of the system parameters (e.g., a user intervention or an external data update) that results in a change of the system behavior and is recorded as a *branch*. A set of causally related tracks is called a *World Line* and presents a possible outcome. The *parallel worlds* at a given point in time are defined as the frames of all tracks that temporally overlap at this time value.

We present the World Lines view as the driving component in a system of multiple linked views that deals with the extended space of parallel worlds. The view will operate in two modi. A steering mode for generating and controlling simulation runs and a visualization mode for comparing them. In the steering mode, when starting an additional simulation by changing parameters, the system creates a new branch that originates from the parent track at the current position in time. A single simulation run is visualized as an animated track and a cursor evolving in time. Each track stores the system configuration that has lead to this track. For navigation and frame selection, a movable World Lines cursor is provided. Depending on the current operational mode, this cursor can assume different shapes. The currently selected frame is shown in linked monitors where users set and edit events that influence the behavior of the simulation track. In addition, inline widgets are provided to edit the track-characteristic properties in place. Since simulation tracks are connected in a tree-like fashion, any changes to a track affect all child tracks and the system can automatically re-simulate their evolution. The visulization mode can be used to comparatively analyze a quantity of interest in various interactive visualization styles. This way, the user is quickly informed about the best outcomes.

2.4 Visual Representation of Related Simulation Runs

We visualize World Lines as a tree of tracks that are connected by branches. Each of these components has regions the user can interact with (Figure 2.6). A track has the shape of a colored ribbon that spans the duration of a simulation run. In the tree canvas, a set of parallel tracks is vertically arranged such that they share a common time axis indicated by a timescale on the upper parts of the World Lines view. Tracks are identified by a numerical label placed onto the track.

User intervention events are given special attention by the concept of *branching*. Branching occurs when the user modifies parameters of an existing track at a specific point in time. The newly created track visually originates from the parent track. This parent-child relation between simulation runs is visualized as a skewed quadrangle which we call the incoming branch of the child track. Tracks are designed to visualize their full time range but emphasize their origin (parent track) at the same time. For

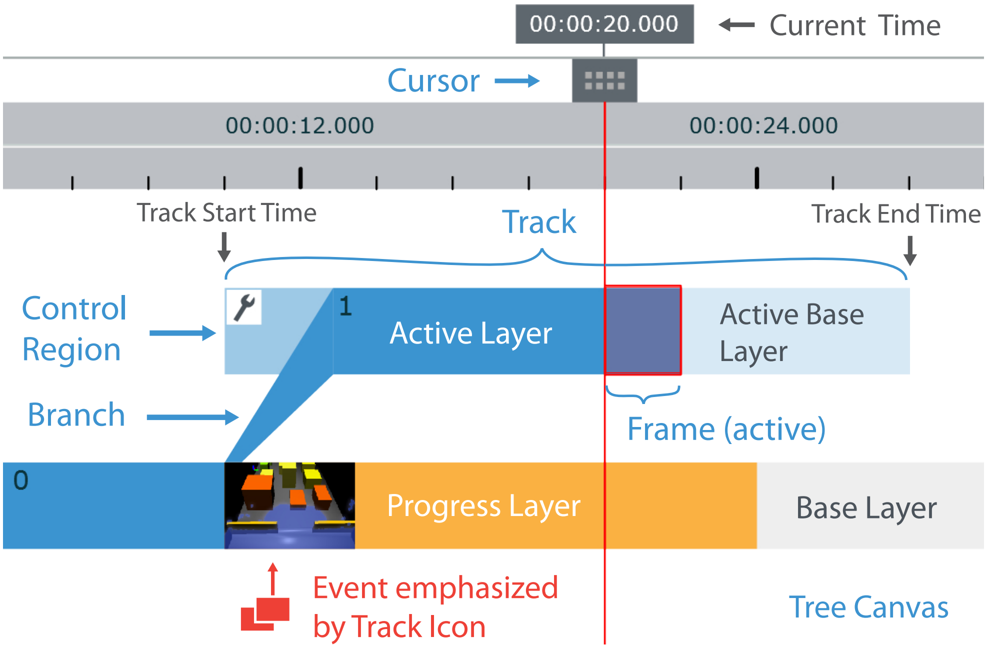


Figure 2.6 – Basic visual elements of World Lines. A single state is presented as a frame. Simulation runs are visualized as tracks that share a common time axis. Causal relations are depicted through branches. The World Lines cursor defines the active frame. Different layers indicate the status of the system.

this purpose, the track comprises a separate control region at its front. It receives a lower opacity value to better accentuate the incoming branch.

The creation of simulation runs through steering is represented as a sequence of branching actions. The resulting tracks form a horizontal tree-like visualization. The tree thus directly visualizes the causal relation between simulation results and shows which events influence which other events. In this approach to computational steering, the user is an essential part in the loop of simulation and visualization. The search for the optimal solution becomes the search for the best World Line. This World Line can be interpreted as a sequence of events and decisions necessary to obtain the planned outcome in practice.

2.4.1 Layers to visualize simulation states

We compose tracks and branches as different visual layers (Figure 2.6) to present the various stages of the exploration process. All colors and opacity values can be set by the user.

- The *base layer* represents the part that has not yet been simulated. The length of a track's base layer determines the time span of the related simulation run.

- The *progress layer* is placed on top of the base layer and indicates the current simulation progress.
- The *active layer* highlights the active World Line (Figure 2.7) as one path through the tree (one course of events) that is currently in focus. This layer has full contrast on top of the progress layer but has a lower opacity above frames without progress (active base layer) to show the separation between simulated and not-yet simulated frames.

2.4.2 z-index to arrange visual elements

The *z-index* is a crucial parameter when arranging visual elements in the tree canvas. This parameter determines whether a component is placed on top (highest z-Index) or below other elements, potentially hiding other tracks or branches of the visualization. We define the following basic rules for the specification of a component's z-index.

- The incoming branch of a track is placed on top of the track.
- The most recently created track receives a higher z-index than all available tracks.
- Tracks that are part of the active World Line are located on top of all other tracks. Within the active World Line, each track is placed on top of it's parent track.

2.5 Navigating the Multi-View System

World Lines can be regarded as a novel component in a system of multiple coordinated views that has to deal with the extended space of alternate simulation runs. The role of components that are linked to World Lines can be manifold. They might act as steering components that allow for the configuration of the input parameters which are associated with the active track (Section 2.6). They can visualize simulation and analysis results as given by the active frame or a set of selected, parallel frames (Section 2.7).

2.5.1 Track Activation

World Lines offers convenient ways to navigate the system through time and parallel worlds. The multi-view environment is synchronized with the state that is associated with the active frame which is defined by the active track and the current time (Figure 2.7). We have developed interactive concepts to ease the specification of the active frame. A track can be activated by mouse-click interaction. Consequently, all direct ancestors of the new active track are found and combined into a linear data structure to form the updated active World Line (Figure 2.7). Each track and branch that is part of this causal combination is covered with the active layer to highlight the

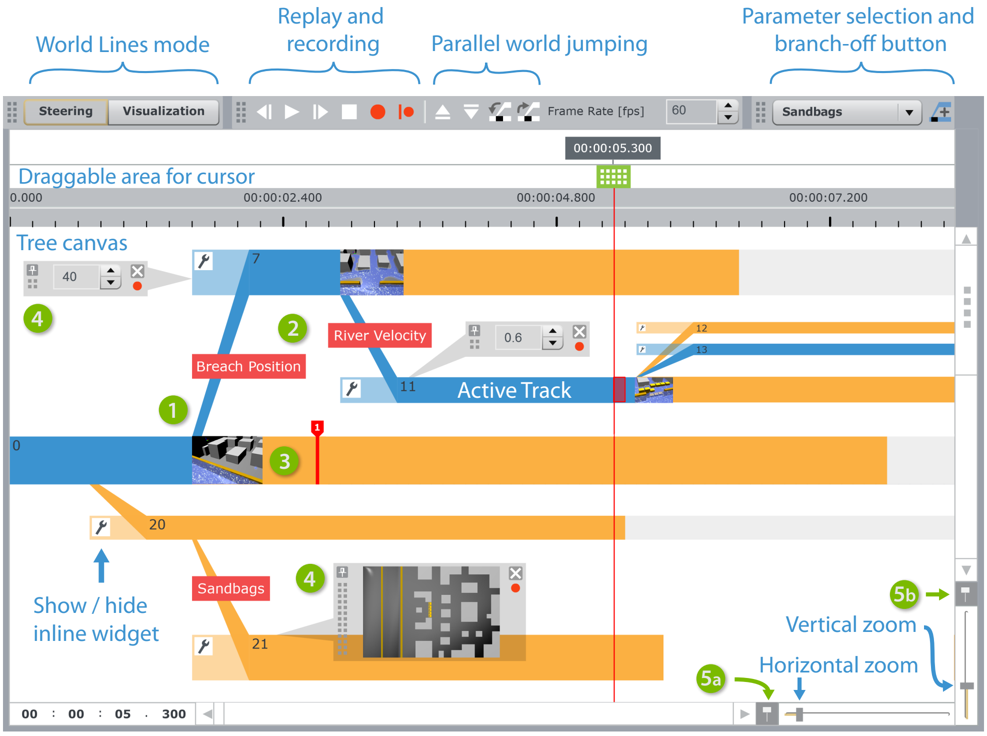


Figure 2.7 – Screenshot of World Lines in the steering mode. (1) New tracks are created by branching. (2) The relevant parameter is indicated by a label placed onto the incoming branch. (3) At the branch-off location, track events are emphasized by track icons. (4) Toggable inline widgets can be used to edit parameters. The active World Line (blue) represents the current preferable course of events and determines timer-based playback and recording. When navigating and zooming, the active frame remains in the horizontal and/or vertical center if (5a) the horizontal focus button and/or (5b) the vertical focus button is pressed respectively.

user's choice. As mentioned earlier, all components of the new active World Line receive a higher z-index. Direct track activation is one method to navigate the system through parallel frames.

2.5.2 World Lines Cursor

The *World Lines cursor* is designed to indicate the current time and to highlight the active frame. The cursor consists of a draggable box which is placed above the timescale as well as a vertical line that spans the entire tree canvas. The current system time is shown in a label that is attached to the top of the draggable box. To accentuate the active frame, the vertical line of the cursor is augmented with a rectangular focus element that surrounds the active frame. If the frame size is below a certain limit, this focus element is reduced to a shape that looks similar to a cursor in a text editor. Inspired by modern audio and video editing software, we have added cursor functionality to enable direct navigation in time. The time can be set by dragging the cursor in the horizontal direction.

2.5.3 Jumping

As an alternative to dragging the cursor, we can jump from one point in time to another one by clicking into the corresponding horizontal position above the timescale. This process is supported by a timescale indicator which appears when the mouse cursor is moved over the draggable area. As with the cursor, this indicator uses a text label to display the target time of the jump action. We realize a set of advanced navigation buttons to enable further types of jumping (Figure 2.7). These buttons can be used to jump from one track to another one, across parallel tracks or along the active World Line to navigate from one user intervention to the next. In addition, the World Lines view has an editable time label to enter jump-target times directly.

2.5.4 Timer-based Simulation and Replay

The World Lines view has a timer that is running at a user-defined sampling rate to enable simulation recording and replay. In this case the cursor follows the active World Line. When the cursor encounters a branch that is part of the active World Line, it automatically activates the child track connected to this branch. The user can choose to keep the active frame in focus at the center of the view. This gives the impression that the view is moving behind a static cursor in either horizontal or vertical direction or both. This mode is convenient for monitoring an ongoing simulation as we can follow the progress even if the generated visualization exceeds the window bounds.

2.5.5 Zooming

We have adapted standard methods that allow for zooming the view in both horizontal and vertical direction. The zoom is designed to keep the active frame in focus at the horizontal and vertical center of the view. When zooming horizontally, only the tracks are stretched, branches and the control region keep their size. Standard scroll bars are employed to navigate into areas of the tree canvas that are currently not visible. It is convenient to press the horizontal and/or vertical focus buttons next to the scroll bars in order to quickly center the view around the active frame.

2.6 Steering Mode of World Lines

In this section we describe different aspects of World Lines for the generation and management of multiple, related simulation runs. The visual representation of alternative scenarios with World Lines offers multiple ways for user interaction. The user can manipulate initial and boundary conditions as well as inherent parameters

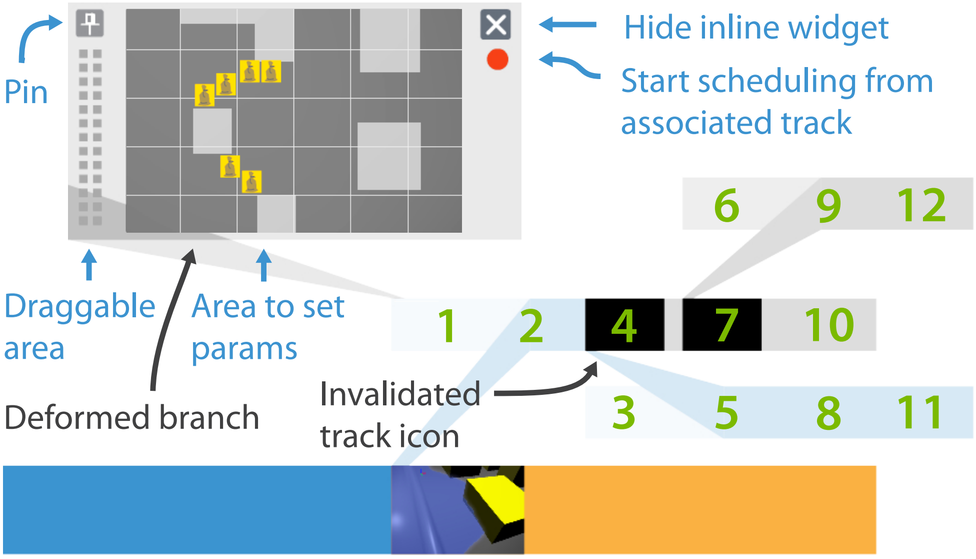


Figure 2.8 – Inline widgets offer a fast way to customize the input parameters of simulation runs. When modifiying parameters, the track and its descendants are invalidated. Hitting the record button of an inline widget, notifies the scheduler to re-simulate all affected tracks. The numbers on the tracks outline the ordering in which frames are being simulated if the user opts for per-time step scheduling and prioritization of the active World Line.

of the simulation. These interventions reflect the user's choices, for example, the modification of inflow conditions or a change in the shape of the simulation geometry.

2.6.1 Branching

At the position of the World Lines cursor, the user can create a new track by modifying parameters of the active track. When creating a new track in this way, the parent track is not required to have simulated frames. Branching can also occur on track sections that do not show progress at the moment. This way, users can plan a set of related simulation runs in advance. A simple annotation centered above the branches identifies the type of parameter that has been changed. The visibility of these *branch labels* can be toggled. If there is not enough space, labels are hidden by default.

Inline Widgets The parameters that are relevant for a specific track are either steered by its associated *inline widget* or in a linked view. We will first concentrate on inline widgets which comprise an interactive area with different controls to enable quick adjustments of the setup (Figure 2.8). As part of the tree canvas, inline widgets have the highest possible z-index to keep them above tracks and branches. In the control region of the affected track, a configuration button is provided that enables the user to open or close the inline widget (Figure 2.7). Each inline widget is associated with an incoming branch that directly relates the widget to the affected track. While users change the position of the inline widget interactively, the branch is deformed to maintain this visual association as good as possible (see Figure 2.8). Several World Lines features (like zooming) change the position of tracks within the tree canvas. The user can decide whether an inline widget is pinned to a desired position or if the widget has to follow the movement of the corresponding track. The background color and transparency of inline widgets can be customized by the user. By default, inline widgets and their incoming branches are black at a very low transparency value so that all tree elements below shine through.

Track Icons To further emphasize the real-world event that led to the new track, we show an interactive *track icon* at the branch-off location of the parent track. These track icons show a snapshot of the current simulation state as generated by one of the views that are linked to World Lines. Internally, a track icon stores a snapshot of each view available at the branch-off location. By clicking onto the icon, the user can switch through the stored snapshots.

2.6.2 Steering via linked views

When steering the parameters of a specific track, we can take advantage of the multi-view framework. When selecting a track, all views and steering controls are updated to show the parameters of the selected track. The user can edit the track-specific parameters in linked steering views. These views can be of any type that is applicable to the domain-specific problem. For example, we have implemented a linked 2D view that allows the user to place geometric primitives into the scene. This view is part of our case study and is demonstrated in Section 2.8.

2.6.3 Scheduling

The *World Lines scheduler* is a system entity that uses automatic timer-based recording in order to simulate the predefined base layers according to a user-defined ordering. The scheduling can be adjusted by the user in several ways. The basic tool is interactive track resizing. The user can elongate the base layer of the track in order to predetermine the length of the related simulation run. Alternatively the user can edit the end time of a track using the track's context-menu entry. In combination with branching the user can set up a set of related runs that can be automatically simulated.

The processing order of tracks is determined by the *visual priority queue* of World Lines. This structure orders all available tracks according to their vertical position in the tree canvas. The top-most track has the highest priority. Two mechanisms have been developed to enable interactive modifications of the vertical track layout. The first technique is based on direct movement of tracks. Here, the user drags one or several tracks that have been selected with a rubberband tool. The second method is based on automatic layout. Using a track's context menu, the user can re-layout the tree canvas such that the track moves up, down, to the center or to the top of the visual priority queue. In Section 2.7.2, we describe animated transitions to support the user in perceiving automatic layout changes.

Having the priority queue set up, the user can choose between two scheduling methods:

- *Per-track scheduling.* The system completely simulates one track at a time, starting from the tracks's progress time until the end time as indicated by the track's base layer. If the currently handled track has no progress at all, this necessitates a recursive search up to the root of the World Line that this track belongs to. Causality demands that the system handles all predecessing frames prior to simulating the actual track of interest.

- *Per-time step scheduling.* The scheduler treats one time step at a time, simulating all parallel frames as ordered from top to bottom. This mode is useful if the user wants to compare alternative simulation runs as soon as possible. This can give early insight in order to be able to remove undesired outcomes quickly.

In addition, each of the suggested types provides the option to process the active World Line prior to the rest of the tracks that are present in the visual priority queue (Figure 2.8). The presented scheduling approach has been realized in a way that the ordering of tracks as well as the scheduling type can be changed during simulation runtime.

2.6.4 State modification

Each track represents a simulation run and is assigned a set of input parameters that constitute this particular run. As an alternative to branching, users may decide to change the parameters of an existing track directly. This can be useful, if we quickly need to account for unexpected changes that are to be propagated to a whole subtree. When the simulation parameters change, the corresponding tracks and their descendants become invalid. This means that the progress of each affected component is reset to zero and all track icons are replaced by a black rectangular shape. The changed setting is automatically propagated to all of the track's descendants which do not specifically steer the same type of setting. For example, if we change the flow velocity in the parent run but the child changes geometry, it makes sense to propagate the new flow-velocity setting to the child. After the new settings have been propagated through the tree, we can re-simulate the invalidated components. Inline widgets are equipped with their own record button (Figure 2.8) to quickly start priority-based scheduling from their associated track.

2.6.5 Simplification and Collapsing World Lines

To simplify the visualization and to reduce potential clutter in the user interface, tracks can be collapsed into their parent track by clicking onto the incoming branch.

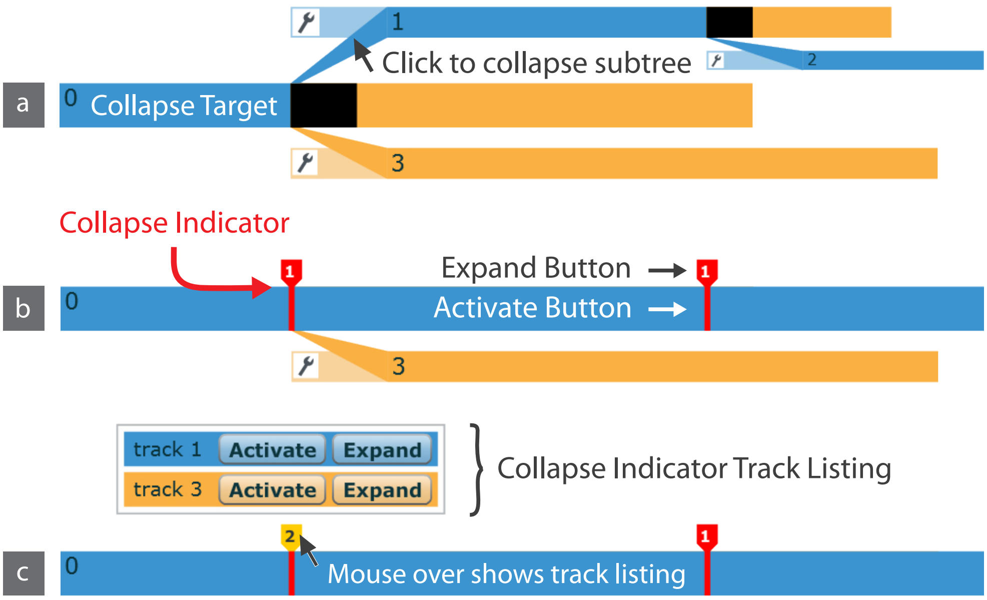


Figure 2.9 – Collapsing World Lines. (a) Clicking the incoming branch of a track triggers an animated collapse of a subtree into a collapse target. (b) Collapsed tracks are visually identified by a collapse indicator that is positioned at the start position of a collapsed track. (c) The track listing handles the case if several tracks share the same indicator.

In the collapsed status, all visual and interactive components are removed from the track. At the track's start time, a *collapse indicator* is drawn for each collapsed track (Figure 2.9). This element serves as a visual surrogate and maintains a visual separation to other tracks. This indicator consists of two interactive elements that enable expansion and activation of the collapsed track. Expansion is accomplished by clicking onto a marker-shaped button that is placed above the collapse target. The second interaction point is given by a vertical line button that spans the thickness of the collapse target. When clicked, the collapsed track is activated and elevated with respect to the z-index. Our previously defined z-indexing rule guarantees that the active layer of collapsed tracks is visually transfered to the collapse targets.

Special care has to be taken if several tracks have the same start time and are collapsed into a common target. The label of the expand button provides information on how many tracks are managed by the respective collapse indicator. We propose a table-like listing of collapse targets to let users expand or activate particular tracks of choice. This table appears above the collapse indicator as soon as the user moves the mouse cursor over the expand button (Figure 2.9c).

We have identified the visual priority queue as a method to rank tracks according to their vertical position in the tree canvas. The introduction of collapsed states requires an extension to this rule. Even though we believe that most of the time a

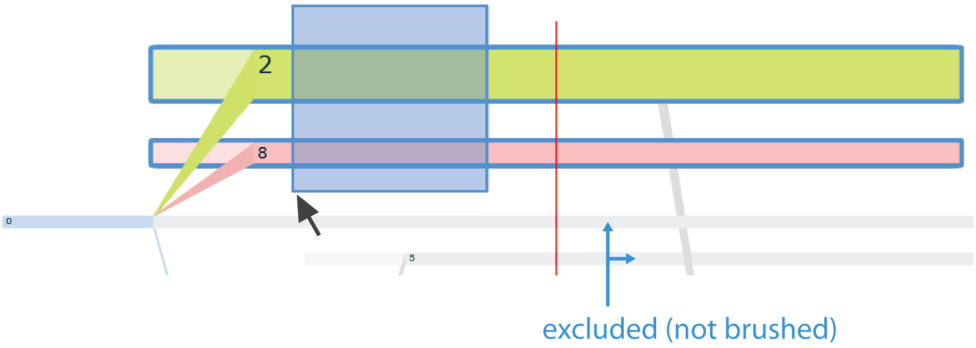


Figure 2.10 – The user brushes tracks with a rubberband tool to select the related simulation data for analysis and visualization.

collapsed track which is not part of the active World Line will be ignored and kept from re-simulation, it can be prioritized by its z-index. The table listing mentioned above sorts the collapsed tracks according to the z-index. The user can change this ordering by dragging rows of the table.

To simplify interaction the user can *flatten* a whole World Line. Starting from the leaf, all tracks of a World Line are recursively collapsed. This simplification results in a straight line that can be further moved to the center of the view to receive full focus attention.

2.7 Interactive Visual Analysis

The idea of interactive visual analysis in our context is to depict various attributes using multiple views and to allow the user to interactively select (brush) a subset of the data in these views. All corresponding data items in linked renderings are highlighted as well, providing the analyst with information about the interplay of the attributes involved [44]. World Lines currently supports brushing parallel worlds by selecting a set of tracks with a rubber-band tool. All brushed tracks are highlighted with a colored border (Figure 2.10). In this way, users are able to select a subset of parallel worlds to visualize and analyze their outcome. World Lines enables a system that supports different types of analyses on simulation results. In the standard case of *per frame analysis*, the system is synchronized with the active frame, showing parameter setup and results (data values) for one track at the current time step. More advanced is the World Lines' capability for *per time step analysis* or *parallel analysis*. In this case a set of parallel frames is compared at a given time step. The multi-view system can be augmented with linked views to visualize analysis results. In the following subsections we explain how per frame and parallel analysis can also be directly displayed with World Lines.

2.7.1 Visualization Mode

The following basic properties of frames, tracks and branches are used to visualize data dimensions: color, opacity, thickness and position. When the user switches to the visualization mode, all interactive steering components are hidden and user-controlled layout is disabled. The arrangement and appearance of tracks is solely determined by the analysis values. Tracks without progress are not displayed in the visualization mode. We propose three different visualization methods for World Lines. Each of the suggested techniques is configurable through a transfer function (TF) that is linked to the World Lines view (Figure 2.11). In the visualization mode, the World Lines cursor shows a label to numerically present the analysis result at the current time step for each track. Using the transfer function, an analysis result is mapped to a user-defined subset of the visual variables. For example, an analysis value can be mapped to the color and opacity of a single frame or a complete track.

Current time-step visualization (Figure 2.11) To visualize information about the current time step, the tracks are rearranged, recolored and resized to visually present the quantity of interest for all selected frames. The transfer function value on the vertical axis is used to sort and rank the tracks. The best simulation outcome is given by the highest evaluated value. Using automatic layout we can visualize this ordering of the tracks. If mapping to position is set, the visualization arranges the tracks according to their rank from top to bottom. The vertical axis value can also be mapped to track thickness. In this way, the user is quickly informed about the best simulation setup at the current time step. Alternatively, the tracks can be arranged starting from the canvas center outwards in order to mimic a non-linear zoom effect. There are two reasons for a track not being part of this visual mapping. Either the track has no simulation result at the current time or it has not been brushed by the user. Excluded tracks receive a grey color at a low opacity value and are shown in the marginal area of the evaluated layout. The track layout as generated by the current time-step visualization can be accepted as the layout for the steering mode. Consequently, the ordering in the visual priority queue for scheduling can be adapted from the results obtained by a comparative analysis.

Frame-wise visualization (Figure 2.12) This mode has the purpose to show the evolution of the analysis result over time. Every frame stores the analysis value as obtained at the frame's time step and parallel world. Each frame of each involved track is colored according to the transfer function mapping of the analysis result. The granularity of this visualization mode determines how many frames are combined for the display. Changing the granularity is not only useful because of efficiency but also to visualize temporal averages or other statistical quantities across many frames.

Inline Function Graphs (Figure 2.13) As an alternative to the frame-wise visualization, we can use inline function graphs to inspect the temporal evolution of a result. A function graph is drawn into each track. At each frame, the vertical position

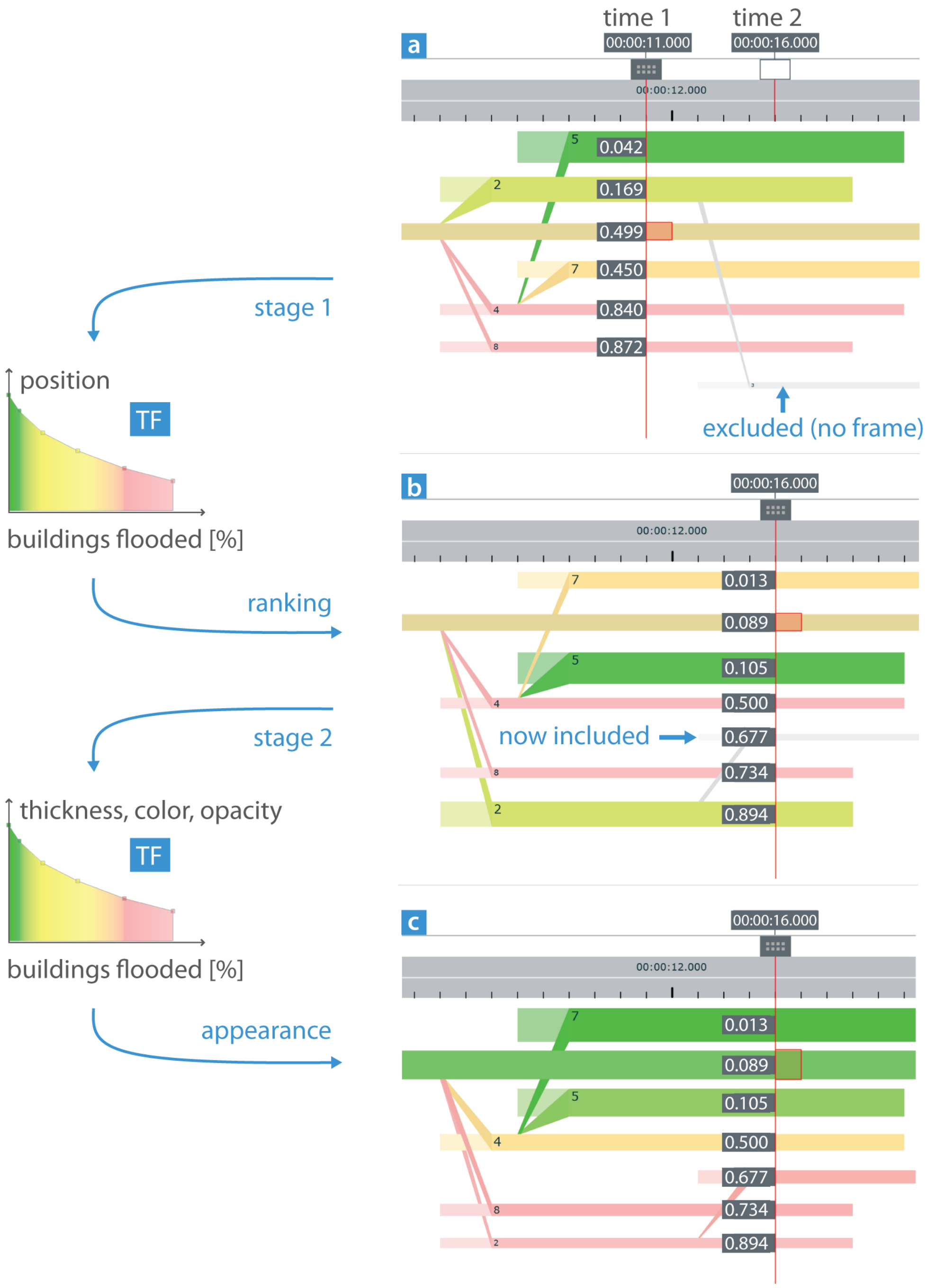


Figure 2.11 – Current time-step visualization. By default, all tracks that have a simulated frame at the current time step are included, i.e., ordered and colored according to the visual mapping of analysis values with a user-defined transfer function (TF). The cursor has labels to annotate the analysis results (here the percentage of flooded buildings). World Lines undergo a two-stage animated transition when jumping from time 1 to time 2. The position is changed first, then the appearance is updated.

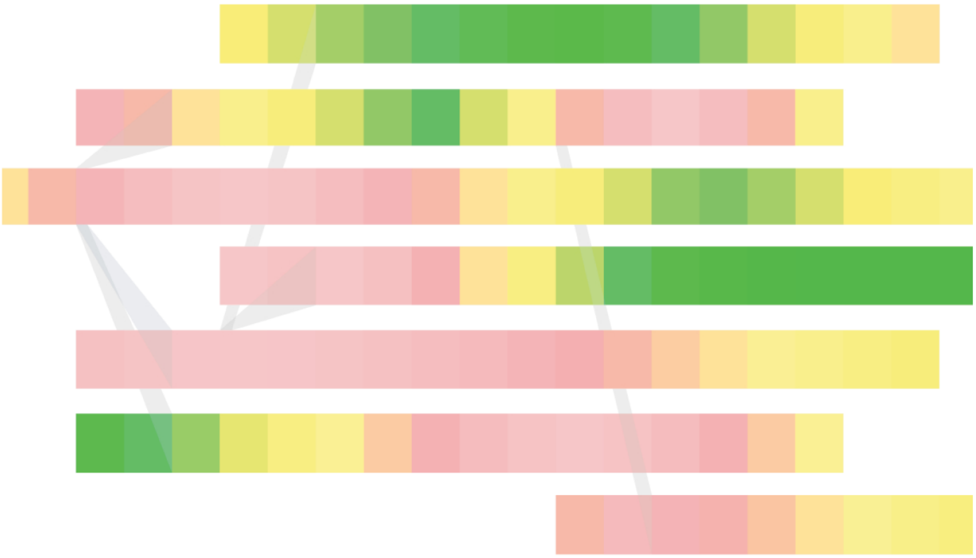


Figure 2.12 – Frame-wise visualization.

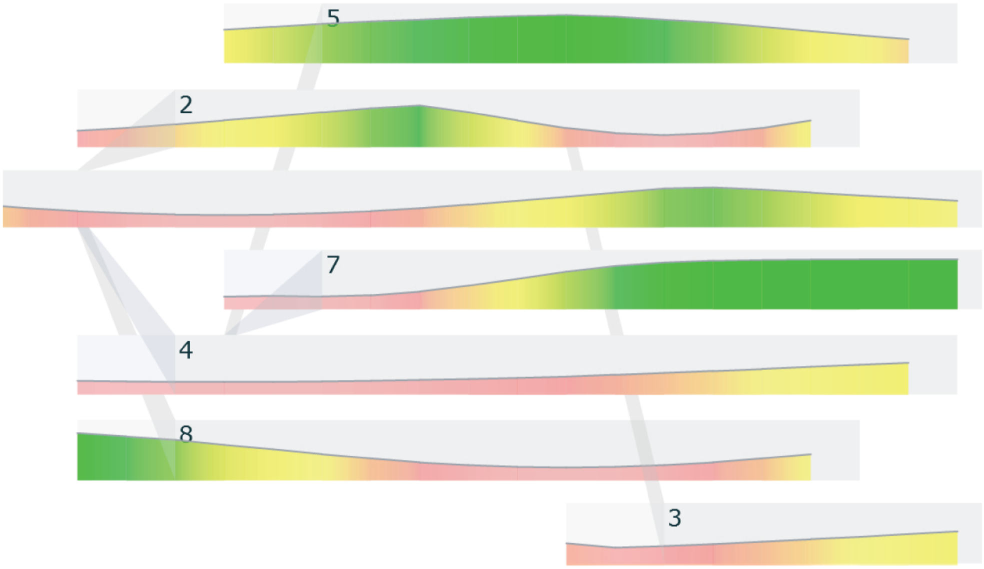


Figure 2.13 – Inline function graphs.

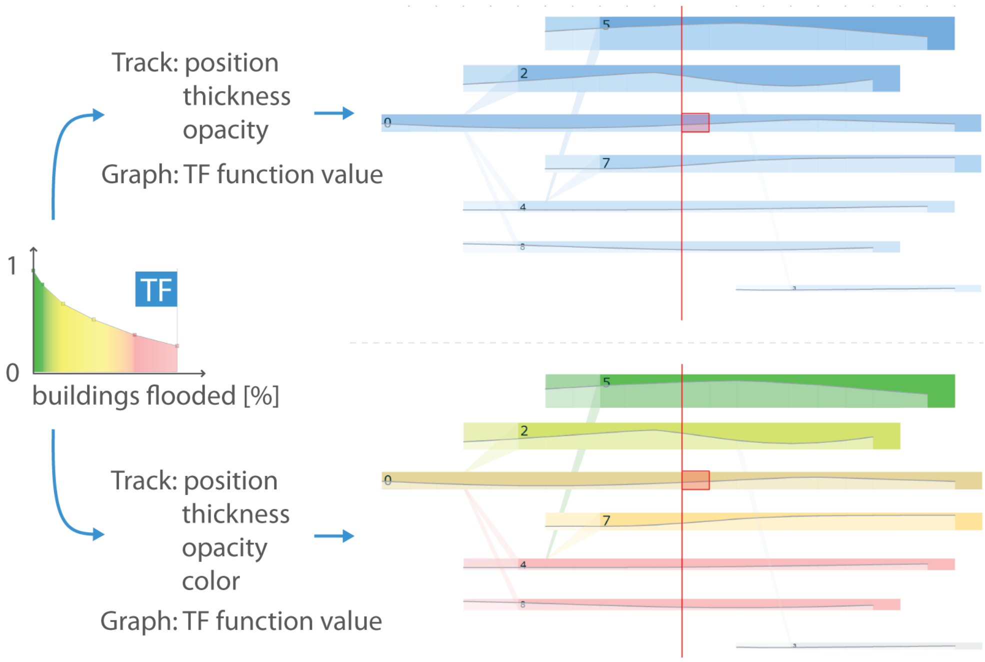


Figure 2.14 – Combined visualizations. The visual variables of the tracks depict analysis results at the current time step. The inline function graphs show the temporal evolution of these results.

of the graph is given by the transfer function mapping. Optionally, we can fill the areas below the graph to account for the user-defined color mapping. The frame-wise visualization or the inline function graphs can be combined with the current time step visualization (Figure 2.14). For example, the visualization can be configured to reflect the results at the current time step via track arrangement and track opacity, combined with inline function graphs to give an overview on the temporal evolution.

2.7.2 Animated Transitions

Heer and Robertson [53] demonstrate that staged animated transitions can significantly improve graphical perception. World Lines is ideally suited to exploit this effect. We have implemented animated transitions in various stages of the World Lines view. In the steering mode, we animate the collapse-expand transitions. When a World Line is flattened recursively, we wait until the animated collapsing of one track is finished before continuing with the next step in the recursion. Each implemented layout algorithm uses animated transitions to guide the user through the change. The current time-step visualization applies smooth transformations to handle layout changes from one time step to the next (Figure 2.11). Here, we use two stages. First, the position is animated to let users perceive changes in track ranking. Secondly, we interpolate the analysis results and apply the transfer function at each iteration. This results in

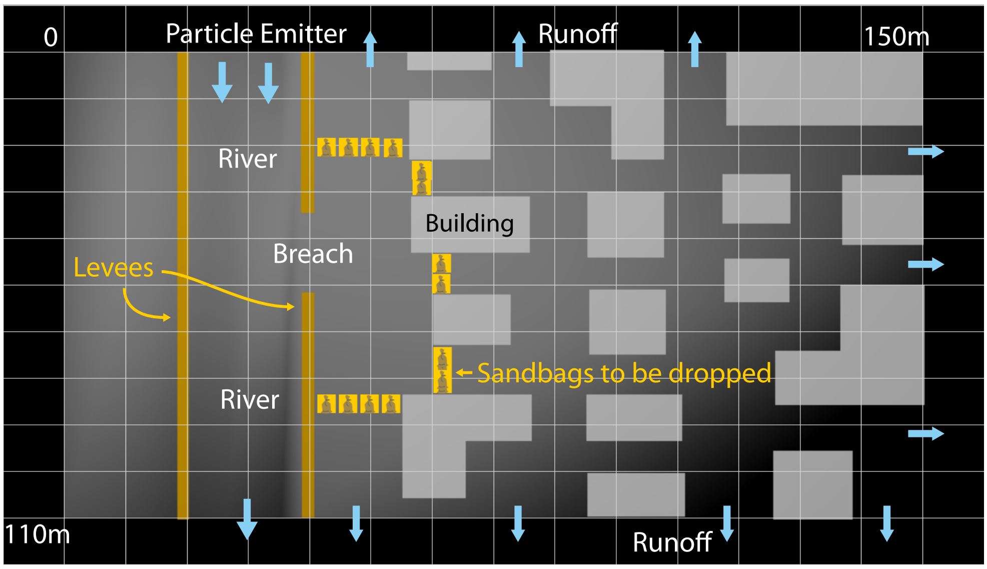


Figure 2.15 – Scenario setup explained with a screenshot of the bag designer. The bag designer is a domain-specific linked steering monitor that allows the user to dump sandbags at specific locations.

a smooth transition of track color, opacity and thickness according to the transfer function.

2.8 Evaluation

In the following subsection we discuss an application of World Lines on a small flooding scenario.

2.8.1 Case Study

Our setup is based on a real-world levee-breach scenario as described by Sattar et al. [112]. Using a laboratory model, the authors have investigated various possible methods for breach closure, utilizing procedures such as single- and multi-barrier embankments with different ways of positioning sandbags. Our simplified site (Figure 2.15) comprises a river that is located next to a neighborhood of 19 buildings. Levees have been built to protect the houses from flooding. We assume that a severe weather condition results in a breach of the levee and floods the city. The following case study shows how World Lines allows the user to analyze the situation and design a breach closure. We utilize SPH to simulate fluid behavior. The flowing river has been modeled with a particle emitter. The emission rate of the particle emitter and the initial particle speed are used to control the water-flow velocity. Outside the scene particles are discarded to model the runoff of the water. In this situation the neighborhood can

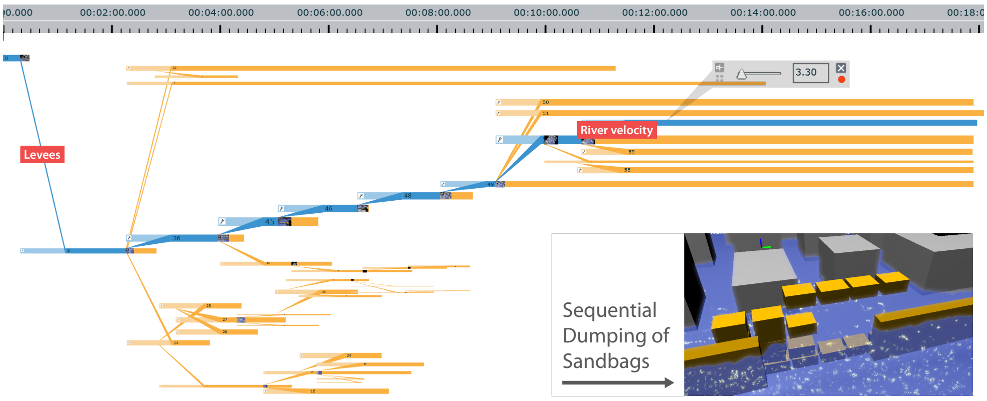


Figure 2.16 – Final structure of World Lines as a result of an exploration process that involved 59 simulation runs. We have identified 7 stages that led to the final solution. These phases are explained in more detail through Figures 2.17 until 2.24. The lower right part shows how sandbags are dropped one at a time.

be secured from flooding by construction of a breach enclosure. The neighborhood has to be protected by dumping sandbags of two different weights, heavy bags (5000kg) and light bags (3500kg), which can be carried by a helicopter. Barrier construction is restricted to sequential dumping of sandbags (Figure 2.16), since a helicopter dumps one bag at a time. The goal of this case study is not accuracy in fluid simulation but to show that our approach enables a fast and intuitive exploration process. For this purpose, we configure the simulation to operate in real-time. All components in the simulation scene are non-erodible and non-porous. We model sandbags as bricks that cannot be deformed but assemble well. We use approximately 22000 particles for simulating the flooding. This means, we are able to test the stability of the barriers with a simulation of at most 10 minutes of particle-propagation time.

Using World Lines, we attempt to find a multi-barrier system that is capable to protect the city with respect to the following criteria: The number of flooded buildings has to be reduced as fast as possible. The number of dumped sandbags is to be kept as low as possible. Barriers that consist of lighter bags are easier and faster to assemble, we therefore look for a solution that utilizes as few heavy bags as possible. The multi-barrier system needs to be stable even if the river velocity increases.

In this case study, World Lines is linked to two views. One view shows an orthographic birds-eye perspective on the scene to act as a steering monitor that enables user-defined positioning of sandbags (Figure 2.15). Context-menu interaction and mouse-drag operations are employed to determine the exact location of dumping. Sequential dumping reflects the ordering of bag placement. A second view shows a 3D rendering of the scene. Particles are rendered in real-time using a fast GPU surface extraction technique [134]. In addition, the view is capable to render comparative information obtained from multiple simulation runs. Buildings can be colored in case they are flooded beyond a user-defined warning level, in any of the brushed tracks of the World Lines view.

The given problem turned out to be quite challenging. To our knowledge, there is no alternative framework that allows for a complex exploration as was needed. In total, 59 simulation runs had to be performed and inspected. Figure 2.16 shows the final appearance of World Lines when zoomed out. We have identified 7 major phases that led to the final solution. Figures 2.17-2.24 display details and important results obtained from the 3D view at different stages of the process. In stage S1 the city is flooded in about 2.5 minutes when a part of the levees breaks down (Figure 2.17). In stage S2, a brute force approach is tested (Figure 2.18). Heavy bags are placed directly at the gap. This embankment turned out to be unstable and caused a re-flooding of the buildings. As a consequence, a multitude of different dumping techniques have been investigated in stage S3 which we refer to as the exploration stage (Figure 2.19). A couple of methods as proposed by Sattar et al. [112] have been studied where buildings are used as part of the multi-barrier systems. We have found that the ordering and timing of the construction is crucial, making World Lines ideally suited to test these procedures. In all cases of stage S3, the water pressure on the embankments becomes too high as soon as more than four embankments are present. Bags were washed away and barriers collapsed. A major breaktrough (S4) was possible by going back in time and placing a spur of heavy bags from within the river towards the front row of the buildings (left barrier in the 3D view of Figure 2.20). This barrier deflects the flow and reduces flow velocity through the breach. After another set of trials, it was found that a second spur further improves the flow situation. In stage S5, using the knowledge gained until this point, it was possible to quickly build a stable solution (track 50 in Figure 2.21) that utilizes barriers of light bags in the second row of the buildings. The next stage S6 was concerned with the optimization of this solution. Track 51 (Figure 2.22) replaces one embankment by a barrier of light bags closer to the breach in order to protect an additional building. Track 53 utilizes heavy bags to construct the same barrier. Track 55 represents the attempt to fully enclose the water by placing another embankment at the downstream side of the river. To find out which of these tracks performs best, we have elongated their base layers and simulated a longer time span using per-time step scheduling. This approach has revealed that the barrier systems are unstable except for those given by track 51 and track 53, pointing to track 53 as the optimal outcome.

In the final stage S7, we had to prove whether the found solution remains stable even if we increase the velocity of the river (Figure 2.23). We have branched off twice with different river velocities to create track 58 and track 59. Another scheduled simulation has generated the frames needed to make a prediction. To quickly visualize whether all the cases are stable, we have brushed track 53, 58 and 59 in order to analyze the percentage of flooded buildings. The linked 3D view gives comparative information on the brushed tracks. A building is colored in red if it is flooded, or in yellow if it is in danger, in **any** of the brushed simulation runs. A final combined frame-wise visualization (Figure 2.24) clearly indicates that the multi-barrier system

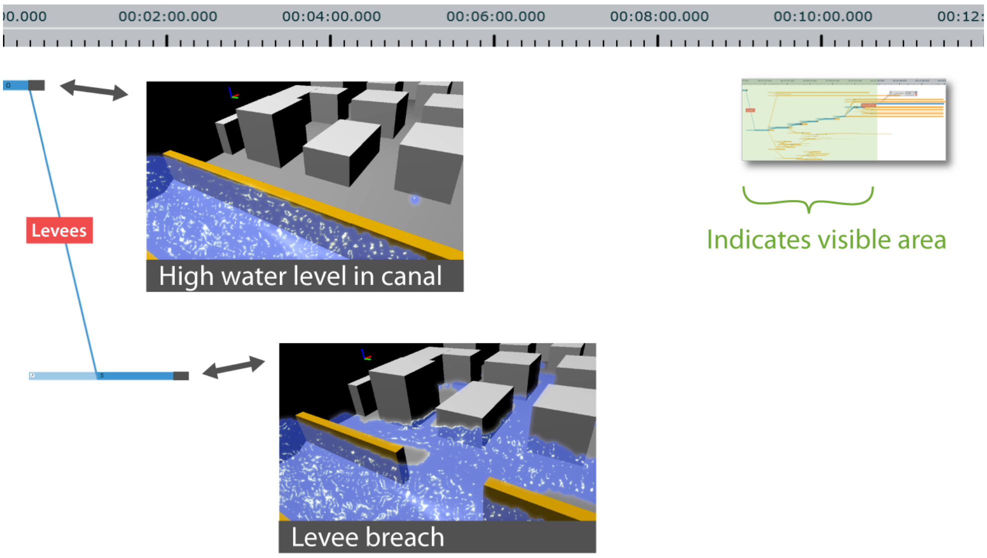


Figure 2.17 – Stage S1: Flooding. Due to a high water level in the canal, we induce a levee breach through branching. A navigation legend indicates the clipping with respect to the fully zoomed out World Lines in Figure 2.16.

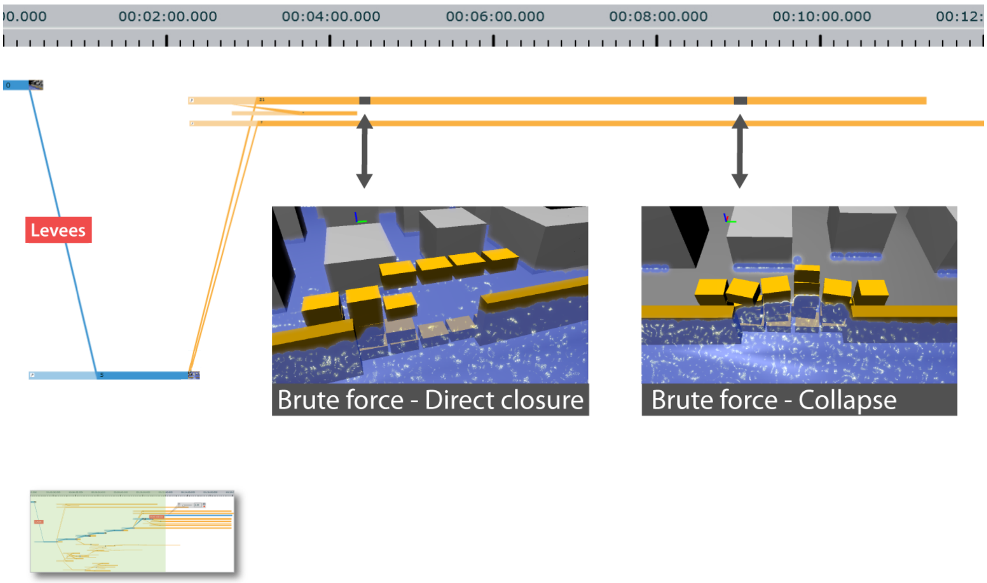


Figure 2.18 – Stage S2: Brute force closure. The initial attempt to directly close the breach fails. The barrier collapses.

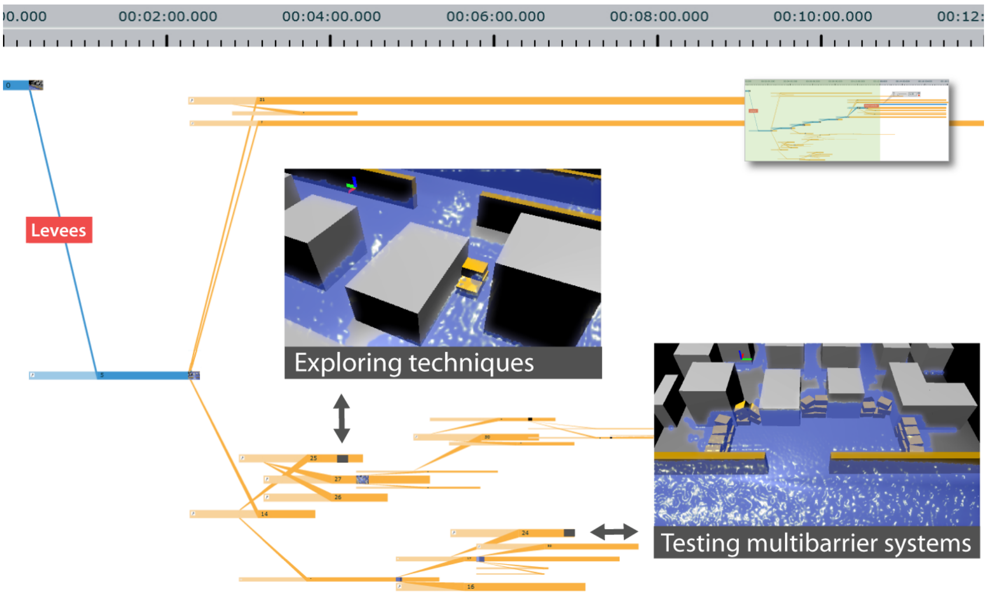


Figure 2.19 – Stage S3: Exploration of different multi-barrier systems and dumping techniques as known from the literature [112]. Every test fails, bags are washed away.

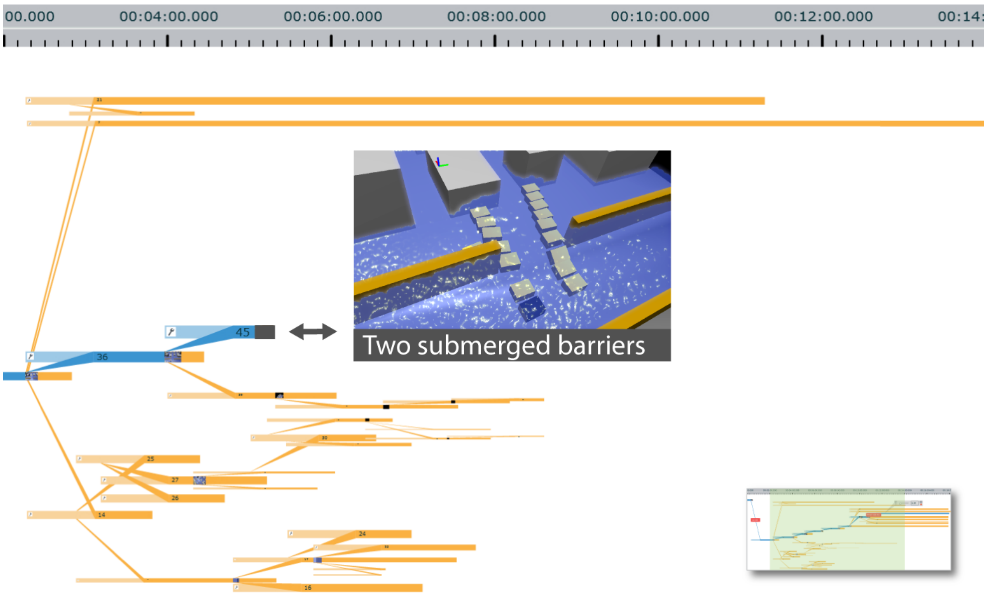


Figure 2.20 – Stage S4: Breakthrough with heavy, submerged bags that deflect the flow through the breach.

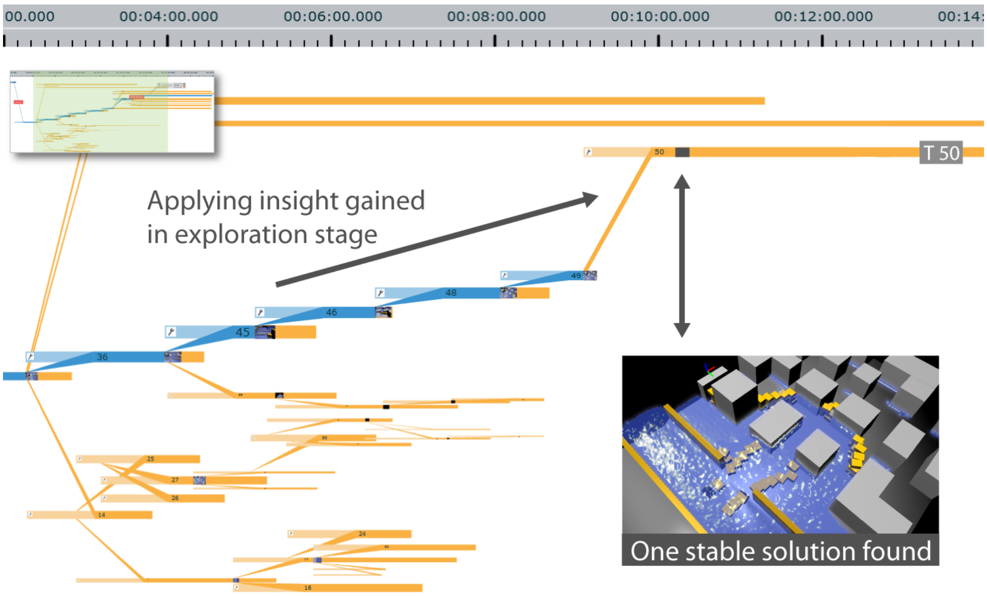


Figure 2.21 – Stage S5. Quick construction of a stable solution (Track 50) by applying the knowledge that has been obtained during the exploration phase.

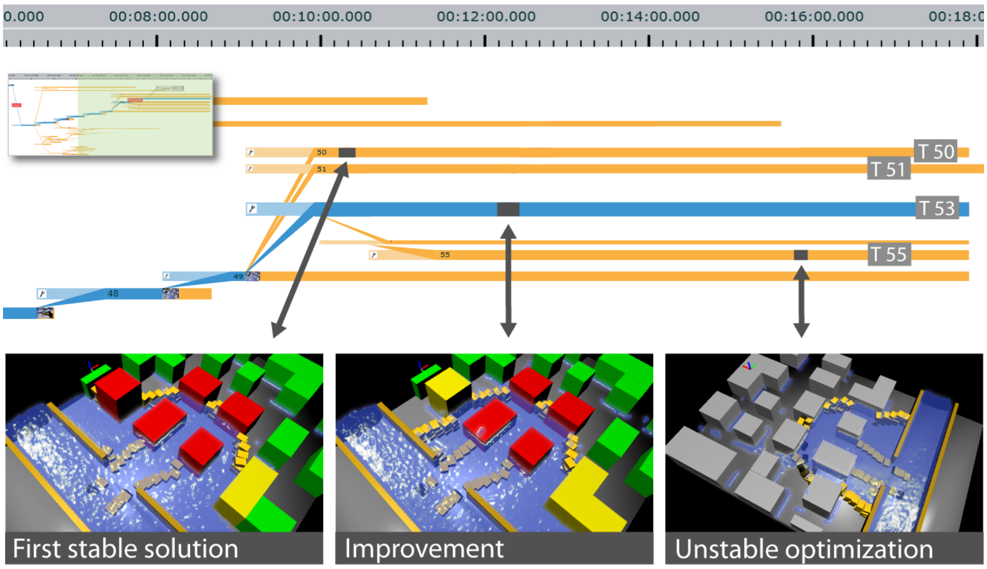


Figure 2.22 – Stage S6. Refinement through minor alterations of the first stable solution (Track 50). The final solution saves more buildings (Track 53). There is also an unstable attempt to further close the downstream parts (Track 55).

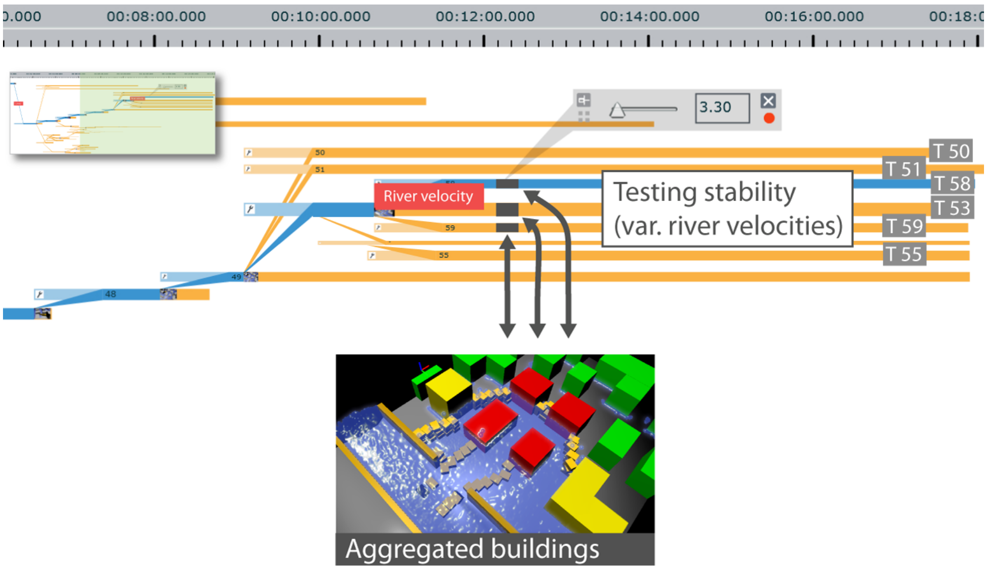


Figure 2.23 – Stage S7. Stability tests. Investigation of various river velocities to verify the robustness of the found solution. The comparative information through the coloring of the buildings in the 3D view shows that the solution is robust even if the river velocity increases. A building is green if it is safe, yellow if in danger or red if flooded in any of the compared tracks.

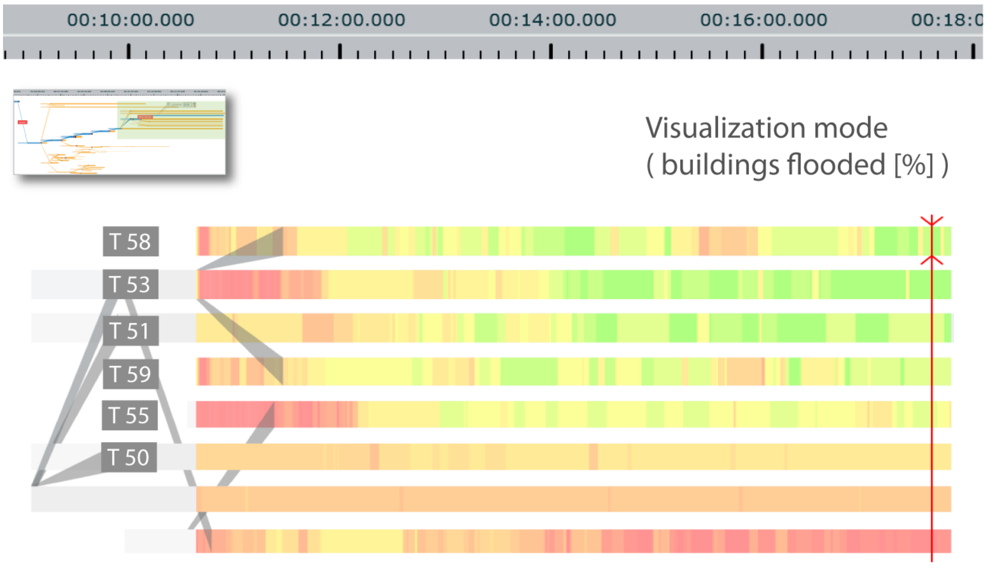


Figure 2.24 – Visualization mode that analyzes the percentage of flooded buildings using a combined current time-step and frame-wise visualization of all tracks that were investigated during the refinement and stability-test stages.

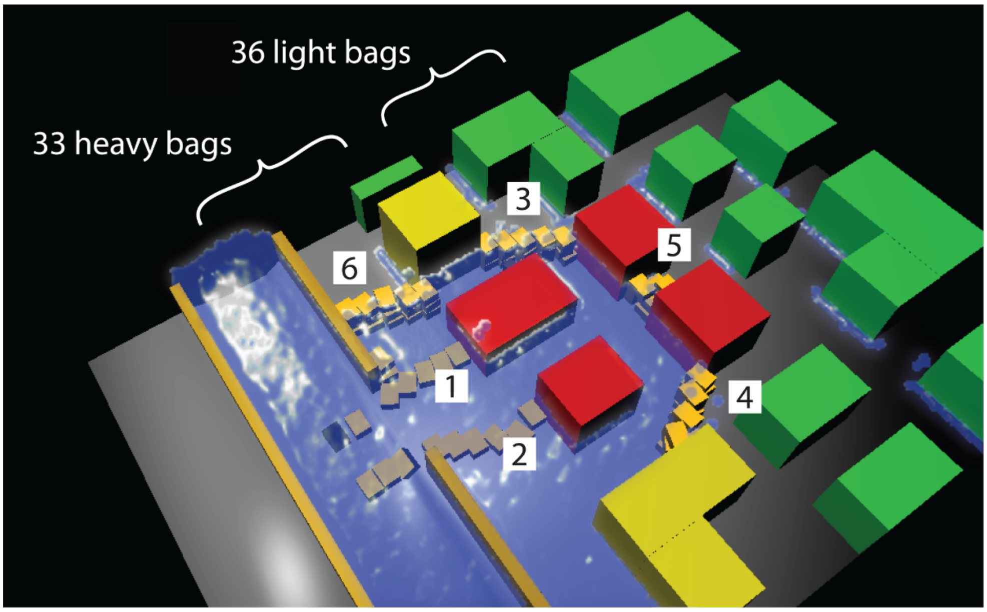


Figure 2.25 – Final solution. 70% of the buildings are protected by a multi-barrier system that consists of 6 embankments. The ordering in which the barriers are to be constructed is depicted through the numbers.

of track 53 is the best and most stable solution found in the exploration process. The specifics of the successful protective measures are explained in Figure 2.25.

2.8.2 Domain expert feedback

The World Lines case study was assessed by an expert with experience in flood forecasting and management systems [19]. He emphasized the importance of flood management plans, in particular as required by the new EU flood management directive [39]. Currently, comparative analyses of placing barrier systems are rarely done. Placement is usually based on the experience of the flood management staff although hydrodynamic simulation models and laboratory models could be used for this purpose. The expert was impressed by the ability of the system to concurrently simulate several scenarios as this would greatly enhance the ease with which alternative management strategies can be compared. He considered both the navigation and the steering to be intuitive, in particular the embedded widgets. Additional visualization options for comparing the scenarios would be useful, such as difference maps of water levels and sand bag locations. In a practical context, World Lines could be used in two ways. First, and more importantly, it could be an efficient tool for training flood management staff in an off-line mode. The most important issue in these types of management decisions is not to find an optimum solution but to rapidly exclude poorly performing solutions. The goal is to identify robust placing schemes that will work for a range of boundary conditions most of which are not well known. Second, the system shows potential for real-time application during a flood event.

2.9 Implementation

We have implemented World Lines as a module of our pluggable steering and visualization system *Visdom*. This framework comprises a client-server architecture to enable control over the web. The server is written in C++ and uses the GPU (CUDA) to handle the compute intensive parts of simulation and rendering. The client is an Adobe Air application that is built upon the Flex [5] framework. Interaction and steering is accomplished via interfaces on the client. The World Lines view is completely implemented as a module of the client to take advantage of the scripting capabilities in Flex. We make use of features such as XML handling, style sheets and animated state transitions. Client and server are connected via a permanent real-time socket. Information is exchanged in XML (settings) and binary format (renderings).

Each track has a specific simulation setup. Simple input parameters such as the position of the barriers are stored in an XML format on the client. For efficiency reasons we do not store the entire system setup within a track but only the changes with respect to the parent track. We consider more complex modifications such as deformations of simulation boundaries as a change in the system state rather than a change in the input parameter setup. These states require the storage of larger data structures and are kept along with the simulation results in a data management facility on the server. In this data container, each state is uniquely defined by a track identifier and a time value. Often it is not necessary to store the status at every iteration of the underlying simulation. Therefore the frame size in the World Lines view need not be identical to the time-step size of the simulation. In our case study it is sufficient to have a frame size (1s) that is 60 times larger than the internal time-step size. State saving and retrieval accumulates to 26 ms on an Intel Core 2 Quad processor with 2.4GHz and 4GB RAM.

World Lines operates on interfaces that can be implemented to steer external simulation modules. The server can also be installed on a high-performance computer. A module that implements the simulation interface would handle the distribution of simulation work among the clusters. For this work, we have written a plugin to adapt the PhysX simulation engine [100] which simulates one frame (1s) of our case study in 666ms on a GeForce 8800GTX graphics card with 768MB memory. The 3D view renders one image in 22ms. The parallel analysis module has been written in CUDA to provide fast analysis results (50 ms for 20000 particles). The client uses a caching mechanism for the results to enable smooth frame-wise visualization.

2.10 Discussion

In this chapter we have described the concept of World Lines that enables the user to steer, analyze, and compare multiple related simulation runs. In the presented implementation, simulation runs are created interactively through branching. In this manner, we were able to design a breach closure for a synthesized flooding scenario. We have found that the exploration would have greatly benefitted from the ability to clone subtrees from one track to another, giving the ability to reuse barrier arrangments. The stability tests in the final stage of the case study have revealed another shortcoming of the system. Users need to be able to create several tracks at once to cover full parameter ranges. To test the robustness of the found barrier, we had to manually create tracks to cover different river-velocity values. In chapter 4, we will explain an approach to automate this process. For now, we have presented a steering method that is fully controlled by the user. When automatic track creation, as in the case of parameter studies, comes into play, scalability can be an issue. Automatic track layout, advanced track folding, as well as navigation will be an important research topic as soon as a very large number of World Lines has to be managed. If necessary, layout algorithms have to be ported to run on the graphics processing unit (GPU). To reduce visual clutter in the World Lines view, tracks that lead to equivalent simulation outcomes could be collected into clusters similar to the approach of Bruckner et. al [22]. In our case, an advanced form of track collapsing could be used to create the clusters.

For practical applicability, we have to consider issues that are related to speed and memory efficiency. It is important to be able to steer simulations generated on a large number of GPUs in parallel. Currently all GPUs have to reside on a single machine. One shortcoming of the current implementation is that simulation and analysis cannot run concurrently, sharing the available CPUs and GPUs. In a problem-solving environment it is important to detect wrong decisions or designs quickly and to terminate corresponding simulation runs as soon as possible. We plan to improve the current implementation in a way that new simulation steps are computed in background processes and to allow for interactive visual analysis as soon as simulation data is available. It is also an open question how to automatically decide which time steps are stored and which can be discarded to save memory. The current system works with a simple user-defined stride such that only every n-th step of the simulation is actually stored for analysis.

In our current project, we envision the real-time support for the planning of actions during a flooding event. Even though the fidelity of the inundation simulation is not yet sufficient, we believe that the current system demonstrates the usefulness of our approach. The method is independent of the underyling simulation technique. It is easy to think of other application areas: industrial prototyping or surgery planning could also benefit from a system that allows the user to compare design decisions interactively. The linked steering monitors are currently tailored to the flooding scenario. When steering simulations in other areas, we will have to provide new input metaphors through domain-specific steering monitors. One route for future research is the investigation of generic steering monitors that are commonly applicable to different problems. The World Lines view itself can be regarded as a generic approach for entering simulation parameters. Finally, World Lines open up a variety of interesting topics for future research that are related to the analysis of parallel worlds. We need innovative comparative views to display the interplay of simulation parameters and simulation outcomes.

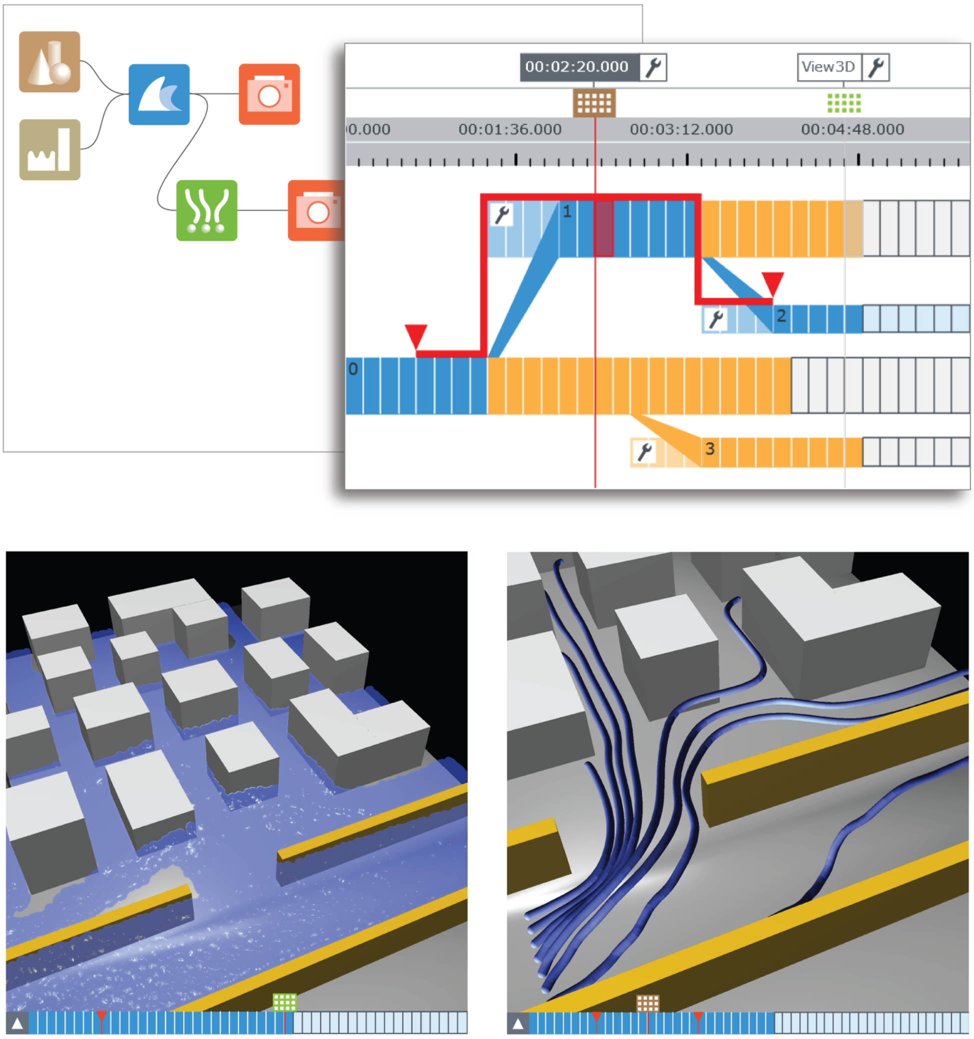


Figure 3.1 – Investigation of material transport through water in a levee-breach scenario. World Lines navigate the underlying data-flow nodes across time and tracks to calculate pathlines that describe the transport phenomena.

In the beginning the Universe was created.
This has made a lot of people very angry
and been widely regarded as a bad move.
— *Douglas Adams*

CHAPTER 3

Multiverse Data-Flow Navigation

Modularity plays an important role in a simulation-steering environment. A data-flow network provides the ability to build a simulation setup according to the task at hand. In this chapter, we show how to utilize World Lines to navigate a data-flow system across time and parallel worlds. Based on multiple cursors, users simply select what they want to see. The usage of internal data-flow algorithms guarantees that each node generates the required data automatically.

3.1 Introduction

REAL-WORLD applications differ case-by-case. In the setting of flood management, the decision-making process strongly depends on local conditions. For one scenario, the placement of sandbags might be the right choice, for another scenario, alternative measures are more suitable. In an urban area, it can be more effective to preinstall the foundations of mobile protective walls. Small villages could be enclosed by polder dikes to protect them from major flooding. In all cases, a thorough cost-benefit analysis of protective measures is required which is tailored to the task at hand. No matter what changes have to be made to tackle new problems, considerable efforts are needed to modify the steering environment, if the underlying setup is hardcoded. Modern data-flow systems offer a modular architecture that allows the user to add or change simulation and visualization components through an interactive flow diagram [116, 133]. In such a graphical interface, the modules (nodes) are represented as boxes that have input and output ports (see top, left image in Figure 3.1). The connections depict the data-flow between the nodes which all run as separate processes. In this chapter, we show how to combine World Lines with a data-flow system to enable navigation of modules through time and parallel worlds. We exploit the data-flow modularity to extend the levee-breach scenario with analysis and visualization nodes to investigate the tranportation of material through the water. This can be driftwood or oil that leaks out of a local industrial facility. An understanding of these transport phenomena is crucial when designing protective measures, since driftwood or oil can have adverse effects on infrastructure and people.

According to our domain expert in hydrological modelling and flood processes [19], pathlines are well suited to describe the transportation of material [138]. Figure 3.1

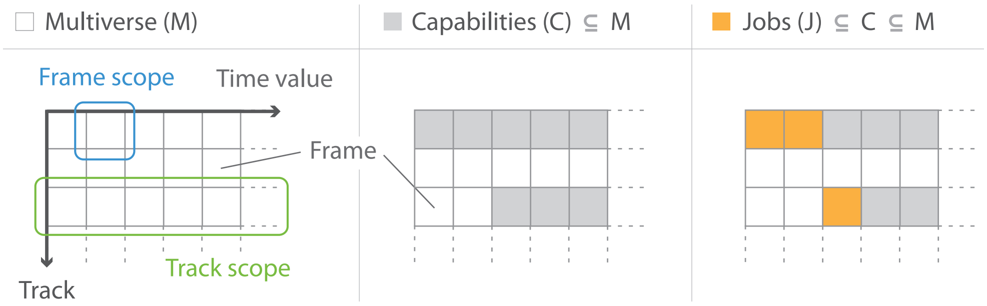


Figure 3.2 – Definitions. A frame is defined by a time value and a track. The multiverse is the space of all frames. The capabilities of a node is a subset of the multiverse and consists of all frames the node can produce data for. The jobs of a node are defined by a subset of the capabilities the node has to generate data for. The scope of a settings object defines for what frames the settings object is valid.

shows a screenshot of pathlines that have been generated for the levee-breach scenario. The data-flow diagram depicts the nodes that are required to perform the computations. The data-flow consists of geometric boundaries (brown node), levees (ochre), the simulation component (blue), the time-dependent pathline integration (green) and two view nodes (red). This data-flow setup is referenced throughout the chapter in order to explain the presented concepts. As we are interested in many, different time-dependent features, the data-flow execution requires a sophisticated management of data that varies with time and across parallel worlds. To our knowledge, there is no generic approach that enables intuitive data-flow navigation while handling data items automatically.

3.2 Problem Description

In this section, we introduce the most important terms and formulate the problems involved with navigating a complex data-flow network. During execution of a data-flow system, each node calculates one or more data items which are made available for further processing through output ports. A data item is characterized through a *frame*, comprising a time value and a track. The space of all frames is a *multiverse* and can be represented by a two-dimensional grid (Figure 3.2, column one). We use this representation of the multiverse in the inlays of the nodes in Figure 3.3 to summarize the navigation problems. The data-flow network corresponds to the setup in Figure 3.1 which has the function to generate pathlines.

Depending on its type, configuration and position within the network, a node is capable of calculating a particular subset of the multiverse. We refer to this subset as the *capabilities* of a node (Figure 3.2, column two). This quantity describes the set of frames the node can produce data items for. For example, the terrain in Figure 3.3 generates only one data item which represents the static, geometric

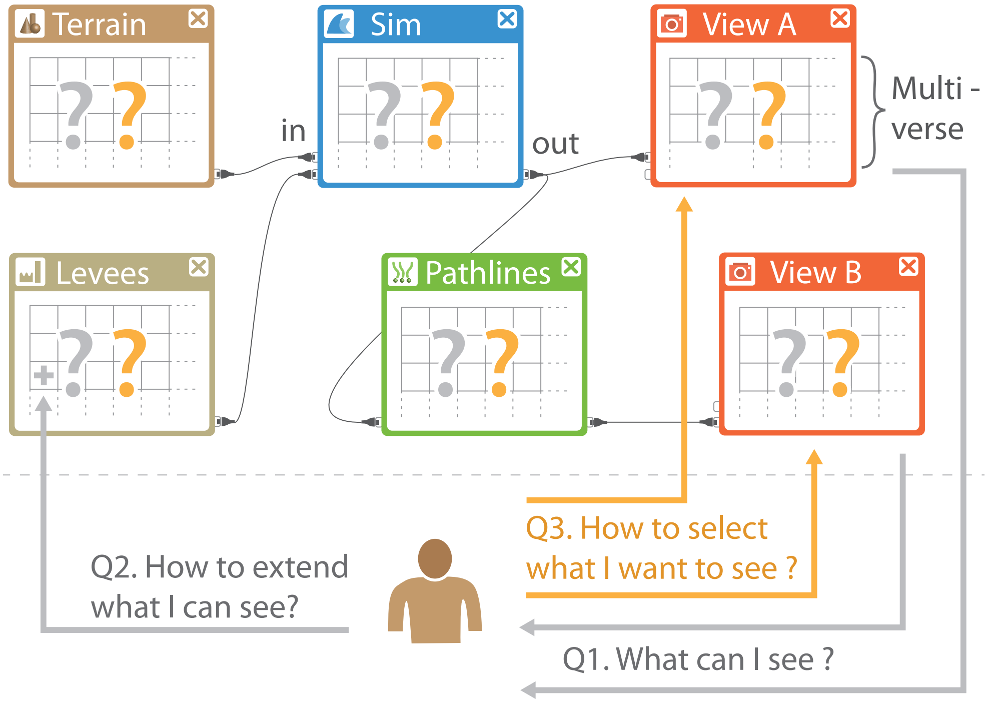


Figure 3.3 – Problems when navigating a complex data-flow system. (Q1) The system lacks a visualization to report what the view nodes can generate. (Q2) The user needs control mechanisms to increase the number of data items a node can produce, e.g., to generate an additional data item in the levees node that models a levee-breach. (Q3) Intuitive navigation concepts are missing to let users select what they want to see.

boundary conditions of the simulation. The time-dependent simulation and pathlines node are capable of generating data for several frames. The capabilities depend on the configuration of the nodes. The most relevant parameter in case of the simulation node is the simulation end time of each track. The pathlines node is an integration node which performs a temporal integration over several frames to compute the trajectories that individual fluid particles follow. The most important parameter in this case is the integration range, determining the time span of the integration. The exact evaluation of a node's capabilities depends on its position within the network. For example, the capabilities of the pathlines node depend on the number of frames the simulation node can produce. A more complex computation is required, if a node has several input nodes. The problems involved with the internal evaluation of node capabilities have been addressed by Schindler et al. [114]. In this chapter, we are interested in the visual representation of node capabilities. More specifically, we require a visualization that represents the capabilities of all view nodes, so that users are able to see what the view nodes can produce (Figure 3.3, Q1).

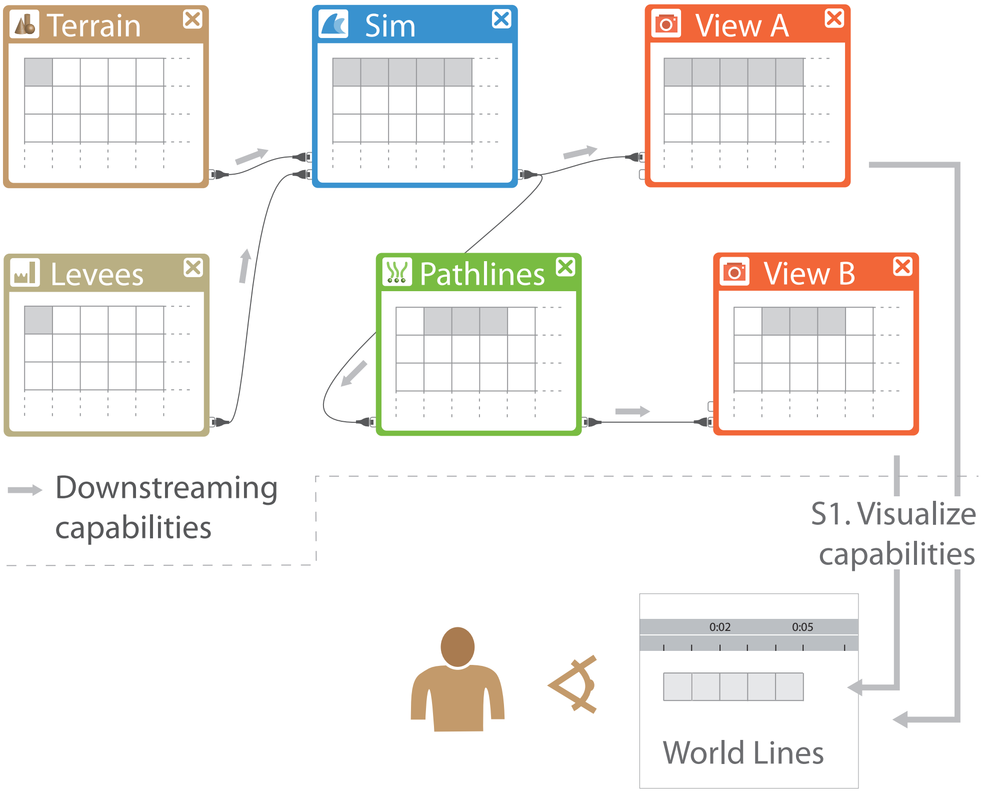


Figure 3.4 – Proposed solution, part one. A data-flow downstream process (grey arrows on connections) is in charge of computing node capabilities [114]. (S1) World Lines visualize the unified capabilies of the view nodes, i.e., all frames the view nodes can show.

The investigation of alternative scenarios requires the ability to manually extend the capabilities of a node (Figure 3.3, Q2). By default, the levees node in Figure 3.3 calculates one geometric data item that models the intact levees of the city. To manually enforce a levee-breach, the node needs to be able to calculate an additional data item. The capabilities need further extension, if we want to study alternative levee-breach locations.

With the visualization of capabilities at hand, we require interactive navigation concepts that let users select what they want to see (Figure 3.3, Q3). For this purpose, a mechanism is needed that automatically assigns *jobs* to the nodes. A job is defined by a subset of the capabilities which a node has to compute (Figure 3.2, column three). Currently, the user is responsible for distributing the work load to each of the involved nodes manually.

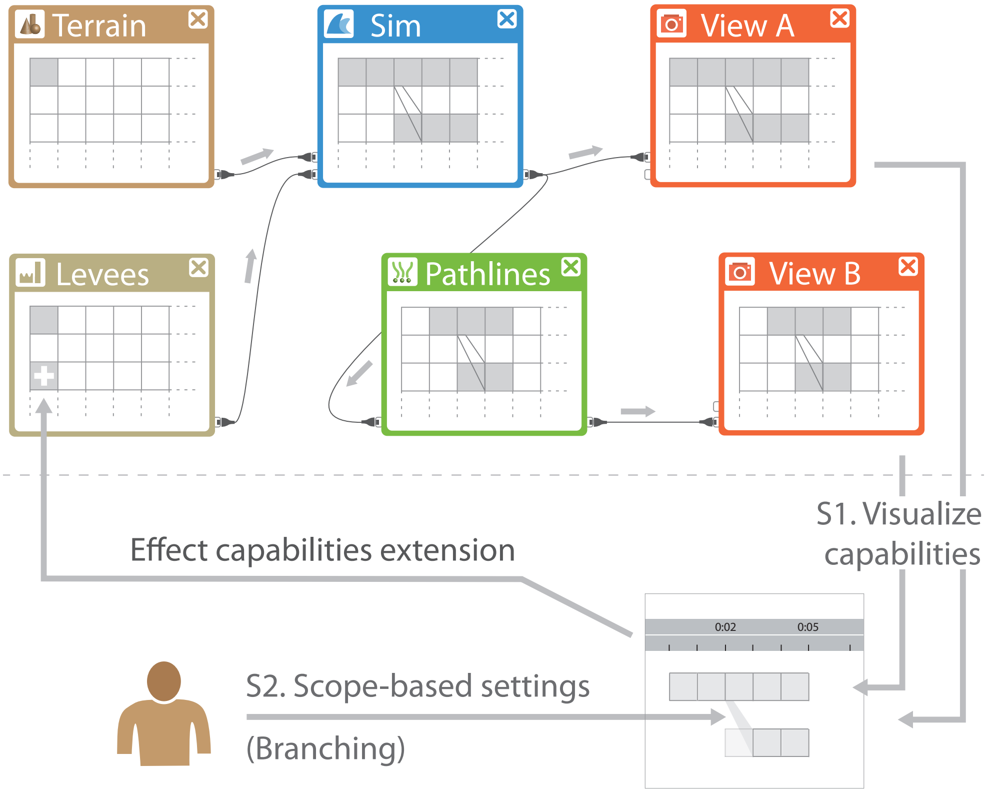


Figure 3.5 – Proposed solution, part two. (S2) The capabilities of a node can be extended through the specification of settings at track scope. Interactively, this is accomplished via branching in World Lines. Here, the user branches with a parameter of the levees node to model a breach. The extended capabilities are downstreamed the data-flow to update the World Lines view with the newly created track.

3.3 Proposed Solution

Our solution is based on World Lines that operate on top of a data-flow which implements algorithms for node-capabilities evaluation and job assignment [114]. As illustrated in Figure 3.4, the multiverse can be mapped onto the visual entities that World Lines provide. As such, we can visualize node capabilities through a frame-wise representation of tracks (Figure 3.4-S1). By default, we show the unification of the capabilities of all view nodes that are present in the underlying network. The usage of the capabilities computation of Schindler et al. [114] guarantees that the view nodes correctly report what they can show, even if more complex time-dependent nodes such as pathlines are involved. In this computation, the data-flow is traversed in topological order from the source nodes to the view nodes. This downstreaming of capabilities is illustrated through grey arrows on the data-flow connections in Figure 3.4.

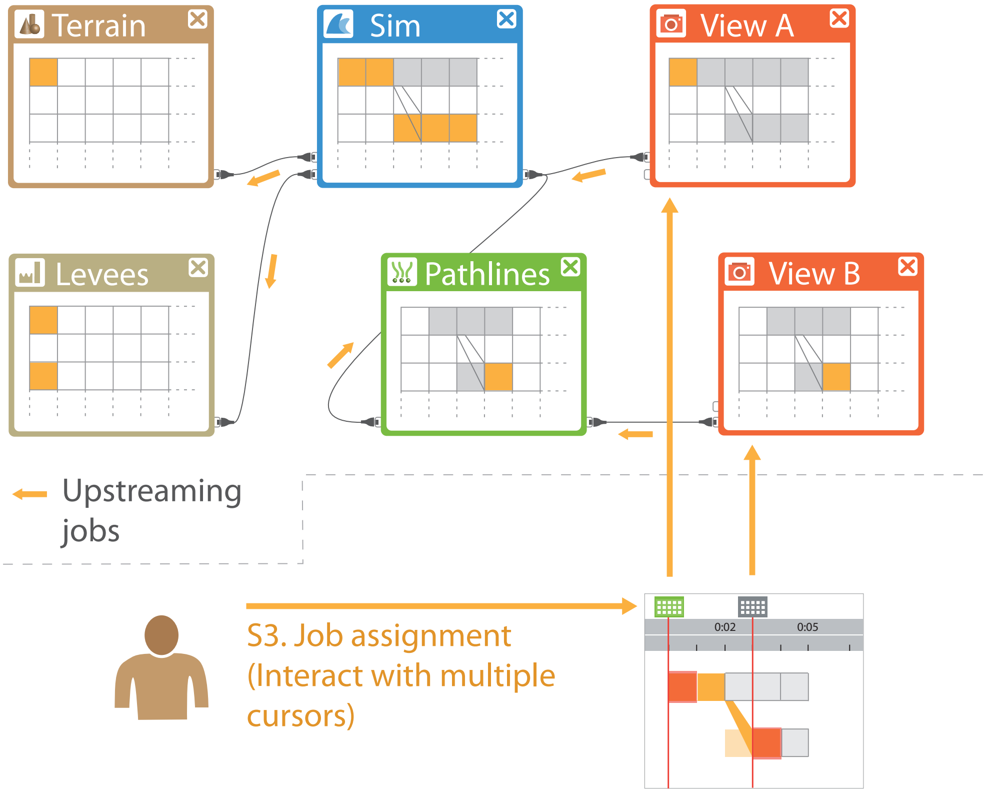


Figure 3.6 – Proposed solution, part three. (S3) The user tells the view nodes what to calculate through interaction with multiple cursors. We utilize a data-flow upstream process (yellow arrows on connections) which assigns jobs to other nodes automatically [114].

The extension of node capabilities is established through the specification of node settings at a particular scope. The scope of a settings object defines for what frames the object is valid within the multiverse. A settings object defined at *frame scope* is valid for this frame. A settings object assigned to *track scope* is valid for all frames within the track (Figure 3.2, column one). The terrain in Figure 3.5 comprises one settings object that is valid across the whole multiverse. As such, the terrain-node capabilities consist of only one data item. The same is true for the levees node before the user intervenes. The extension of the levees-node capabilities is accomplished via the concept of branching in World Lines (Figure 3.5, S2). When branching with a parameter that the levees node provides, the user can specify settings that are valid across the newly created track. The levees node is then capable to evaluate two data items, one for the root track and one for the created track which models a breach. Figure 3.5 illustrates how the extended capabilities are streamed down to the view nodes in order to update the visualization in the World Lines view. In addition, we show how World Lines allow for specification of node settings at frame scope, e.g., to change a transfer function or camera perspective over time. The hierarchical representation of World Lines guarantees that the correct settings are fetched when a frame is calculated.

Interactive job assignment is accomplished via multiple cursors (Figure 3.6, S3). For each view node, the World Lines view provides an individual cursor. The introduction of an active cursor allows the user to apply World Lines navigation techniques for sending requests to view nodes selectively. Each request contains a subset of the capabilities that the view nodes have to calculate during the next data-flow execution. The user need not worry about job assignment to nodes other than views, the data-flow upstream process of Schindler et al. [114] guarantees that each node in the network generates the required information. In this process, the data-flow is traversed in topological order from the view nodes to the source nodes. This upstreaming of jobs is illustrated through yellow arrows on the data-flow connections in Figure 3.6.

Extended navigation concepts through World Lines are the subject of this chapter. We show how to group multiple cursors to allow for flexible linkage of views with respect to navigation in time and across tracks. Each view receives a *view navigator* to enable quick association of images with cursors. In addition, we introduce special-purpose cursors for frame brushing and for time-dependent integration nodes to facilitate the creation of pathlines. The contributions of this chapter can be summarized as follows:

- Frame-wise representation of node capabilities in the World Lines view.
- Multiple cursors to let users assign jobs to view nodes selectively.
- Cursor grouping and view navigators for flexible navigation linkage.
- Special-purpose cursors to facilitate the configuration of node parameters which influence multiverse navigation.
- The ability to specify node settings at track scope and frame scope.
- The feature to use World Lines without a simulation component, i.e., for flexible analysis of time-dependent data which is loaded from files.

3.4 Related Work

In 1989, Upson et al. [133] published a seminal paper on the data-flow based visualization system AVS [6]. In the following years, this topic received a surge of attention and several large systems besides AVS, such as IRIS Explorer [98], the Visualization Data Explorer [59] or VTK [115] have been developed. The data-flow model is based on a topologically sorted graph. To generate visualizations, the nodes are *executed* in a downstream process, where data flows from the source nodes to the sink nodes. Each visited node is in charge of an independent sub-process [48]. From its input ports, a node fetches all data that is required to compute the results of the sub-process. The resulting data items are then passed on to the connected nodes.

In its original design, the data-flow model cannot encompass complex algorithms that do any temporal processing such as particle tracing or statistical analysis over time. Biddiscombe et al. [17] extend the VTK data-flow to support nodes which require access to multiple data items of different time steps. Their solution is based on a demand-driven pipeline as opposed to a data-driven model where the data only flows downstream. The demand-driven approach is more powerful, since nodes can request - and consequently access - multiple time steps from nodes that are further upstream. To realize this, the authors suggest a combination of downstream and upstream processing. Schindler et al. [114] provide a generic formulation of these data-flow passes to enable a demand-driven pipeline in the World Lines steering environment. In this extended version, nodes not only have access to multiple time steps but also to parallel worlds. This way, a single node is able to generate comparative information on multiple simulation runs. We have shown the results of such a parallel analysis in Section 2.7.1.

The time slider [73] is the predominant interaction element for time navigation in media players and audio or video editing tools. This component has been largely adopted for time navigation in scientific visualization systems. These systems often provide multiple coordinated views of the same data set to let users productively combine the information gathered from different views [121]. There are different approaches to accomplish a coordinated time navigation among the linked views, if the underlying logic is a data-flow model. To achieve time synchronization, the time parameter can be global or can be converted to a node connector and streamed down the nodes [6]. Voreen [90] relies on a property-linking mechanism to achieve time synchronization between views. In general, there is no visual interface to achieve a more sophisticated, synchronized navigation behavior between different views, e.g., to enforce a temporal offset between two views to study the evolution of a tsunami wave. The comparative analysis of multiple simulation runs could benefit from a time-value synchronization, if each view is associated with one particular simulation run, enabling a direct comparison of alternative outcomes through navigation in time.

3.5 The Multiverse Data-Flow Navigation Cycle

In this section, we provide an overview on the most important components in the multiverse navigation cycle. The chart in Figure 3.7 depicts an operational step-by-step work-flow of the data-flow tasks (green boxes) and World Lines tasks (blue boxes).

Downstreaming capabilities This is the data-flow process to evaluate the computable frames of all view nodes. The downstreaming is illustrated through grey arrows on the data-flow connections in Figure 3.5. Each node displays its computed capabilities (grey) in a multiverse grid. In the standard case (e.g. red view nodes), the capabilities of a node are given by the intersection of the capabilities of all nodes connected to the input of the node. A node which

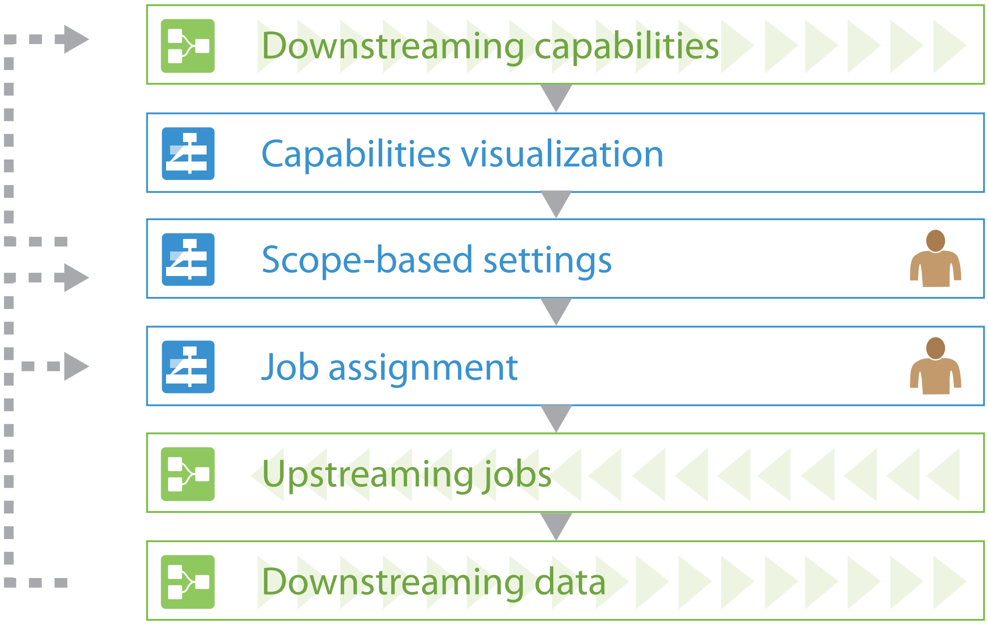


Figure 3.7 – Flow chart of the basic steps in the navigation cycle. Arrows with dashed lines depict conditional transitions (see text). Image production involves three data-flow traversals (green) and three steps in the World Lines view (blue). The only tasks of the user are concerned with cursor movement (job assignment) and modification of settings through World Lines.

performs temporal computations, further modifies the capabilities after the intersection. For example, the simulation node (blue) appends several frames to each track to match the simulation-end time of the track.

Capabilities visualization The capabilities of all view nodes are transmitted to the World Lines view. World Lines employ a frame-wise representation to visualize the received capabilities.

Scope-based settings In this step, users configure nodes. If a setting has influence on a node's capabilities, the capabilities-downstream process is triggered again (see dashed line in Figure 3.7). The same action occurs if the user creates a new track, since branching effectuates an extension of the related node's capabilities.

Job assignment Each view node is associated with a cursor in the World Lines view. Through cursor navigation, the user requests the computation of one or more frames from the related view node.

Upstreaming jobs The user request is communicated from the view node upstream to all connected nodes in the data-flow. The upstreaming is illustrated through yellow arrows in Figure 3.6. Each node displays its jobs (yellow) in a multiverse grid. In the standard case, the jobs of a node are defined as the union of all the jobs of all nodes connected to the output of the node. Nodes which perform temporal computations require further processing. For example, the pathlines node requests more frames from its input nodes to be able to perform the temporal integration across several frames. A fluid simulation is an iterative process that respects causality, generating one time step after the other. This is why the simulation node has to generate all frames which lie ahead to the requested frames and which have not yet been simulated. The algorithm guarantees that all frames are simulated in the correct order, even if multiple branches are involved.

Downstreaming data The final step concerns the standard data-flow execution where each node performs its allocated jobs to compute all requested data. The navigation cycle starts anew if the user assigns additional jobs or modifies node parameters.

In the following sections, we amplify the interactive aspects of the navigation cycle. For details on the internal data-flow algorithms, we refer to the work of Schindler et al. [114].

3.6 Visualization of Capabilities

In the visualization mode of World Lines, we utilize a frame-wise representation of simulation steps to illustrate the evolution of analysis results in time (see Section 2.7.1). An augmented design of tracks is necessary, if we want to display node capabilities in the steering mode. Figure 3.8 shows screenshots of World Lines in the steering mode. Track layers draw borders to visually separate frames, i.e., white borders between processed frames and grey borders between frames that have not been processed yet. The width of a frame is determined by the step width of the simulation. This representation has several advantages. For reasons of performance and efficiency, simulations often use adaptive time steps. The presented system supports the loading of external simulation data through file-loader nodes. With the frame-wise visualization of capabilities, we are able to display and navigate frames that comprise different durations (Figure 3.8d). Moreover, the display offers brushing on a per-frame level to support flexible, interactive visual analysis.

If more than one view node exists in the underlying data-flow network, the unification of their capabilities is displayed (Figure 3.8c). Optionally, the user can visualize individual capabilities of specific nodes. Figure 3.8b shows the capabilities of view node 'View B' which is connected to the pathlines node. Compared to the capabilites of 'View A' (Figure 3.8a) , which is directly connected to the simulation node, we can see that 'View B' cannot generate images for as many frames. This is due to the connected pathlines node, which cuts-off frames that come from the simulation node. The number of cut-off frames depends on the integration range, i.e., the forward and backward integration times. In Figure 3.8b, the backward integration

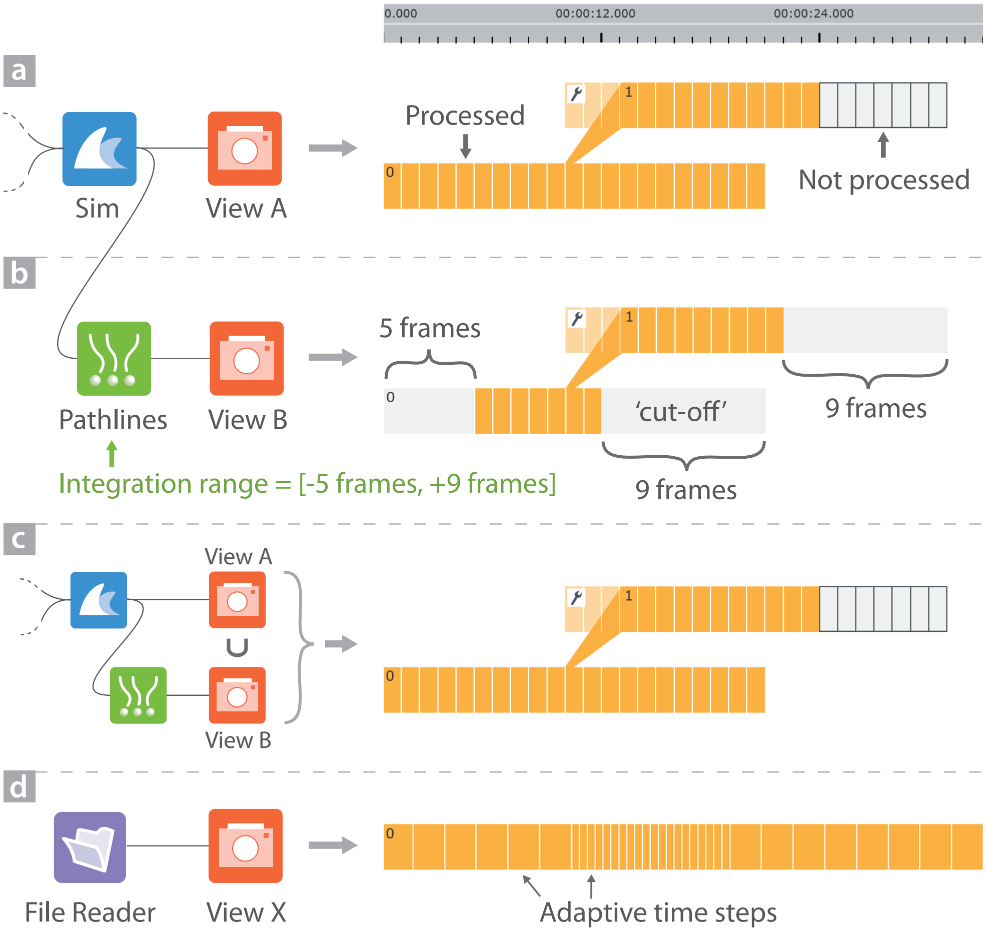


Figure 3.8 – Frame-wise representation of tracks to visualize capabilities. (b) Capabilities of node 'View B', which is connected to pathlines, comprise less frames than the capabilities of (a) 'View A'. (c) Unified capabilities of 'View A' and 'View B'. (d) Visualization of adaptive time steps loaded from external files.

time is set to the duration of five frames, the forward value corresponds to nine frames. For this reason, we cannot compute pathlines for the last nine frames of each track. The first five frames of the root track cannot show pathlines either, but the start frames of the branched-off track (id=1) can do so, since it is possible to temporally integrate across causally connected tracks.

The rendering of tracks as a set of individual frames becomes expensive as soon as a large number of frames is involved. For this reason, frames outside of the viewport are culled. For large zoom levels, the frame-wise representation of tracks is turned off, if individual frames cannot be resolved.

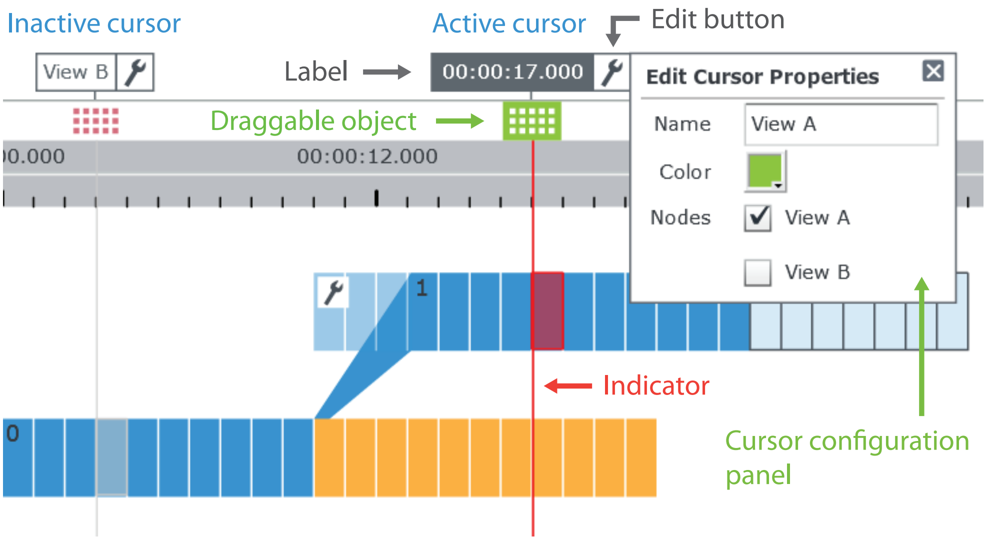


Figure 3.9 – Basic components of a cursor. The World Line which belongs to the active cursor is colored in blue.

3.7 Interactive Job Assignment

By navigating multiple World Lines cursors, users specify what frames they want to see. The frame selected by a cursor determines a request that is sent to the underlying data-flow network, assigning jobs to the controlled view nodes respectively. The users need not worry about the internals, a correct flow execution is guaranteed. The most important components of a cursor are summarized in Figure 3.9. In Section 2.5.1, we have introduced the notion of an active frame. The entire system of linked views is synchronized with the state of the active frame. However, in this definition, it is not possible to navigate multiple views to multiple different frames. This hinders the synchronous comparison of alternate frames in different views. To overcome this limitation, we provide a cursor for every linked view. Each of these cursors has the following properties:

Time value This property is controlled by the horizontal position of the cursor.

World Line As opposed to one global World Line which we termed the active World Line in Section 2.5.1, each cursor now has its own. A World Line is a specific path through the tree of tracks.

Associated nodes This is the list of nodes the cursor navigates.

Name and color These properties are used for identification in linked views.

A cursor can only access frames which are found in the capabilities of each associated node. All navigation techniques presented in Section 2.5 need to account for this fact.

For example, the jump action to a particular time value snaps the cursor position to the closest valid frame. All cursor properties can be edited inline. As with inline widgets for tracks, we follow the principle of keeping property editors close to the visual entity. Therefore, an edit-button next to the cursor label toggles the visibility of the cursor configuration panel (Figure 3.9).

3.7.1 Cursor Activation

Only one cursor can be active at a time. A cursor is activated by clicking onto its draggable object. The World Lines view immediately highlights the associated World Line by coloring. Active and non-active cursors differ with respect to appearance as shown in Figure 3.9. Non-active cursors receive indicators with a grey tone while active cursors are emphasized with colored indicators.

Even though individual views can be navigated with their own cursors, the notion of an active frame is still important. This is due to the optional presence of steering monitors that have been introduced with the World Lines system in Section 2.6.2. These monitors are linked to the World Lines view to enable modification of track settings in a semantic way. The state of these monitors is synchronized with the track containing the active frame.

3.7.2 Special Purpose Cursors

The presented World Lines cursor is used to select a single frame. In the following we refer to this type of cursor as a standard cursor. To manipulate navigation-related node parameters conveniently, we introduce special purpose cursors. Figure 3.10 lists the available cursors along with example visualizations.

Integration Cursor This type of cursor can be used to visualize and control the integration range of an integration node such as the pathlines node. To achieve this, the cursor is attached with a colored ribbon that follows the World Line (Figure 3.10b1). The ribbon forms a stair-case pattern if spanning multiple tracks of the World Line. This indicator component is interactive in a way similar to a dual-slider widget. The users can drag the knobs of the component to manipulate the integration range in forward and backward time. In Figure 3.10b2, the user increases the forward time. The image to the right shows the effects on the pathline rendering. When moving the cursor, the user can opt for one of two integration behaviors. The first option retains the relative distance from the knobs to the cursor position, resulting in a constant time span for the integration. To animate the creation of pathlines, the start knob can be pinned to its location while the cursor navigates. The latter increases or decreases the integration range as the end knob moves relative to the cursor.

Brush Cursor The frame-wise representation of node capabilities provides the ability to brush individual frames. Brushed frames receive a thick, green border on their top and bottom sides (Figure 3.10d). The brushing mechanism supports a comparative visual analysis of multiple frames. The buildings in Figure 3.10d are colored in yellow,

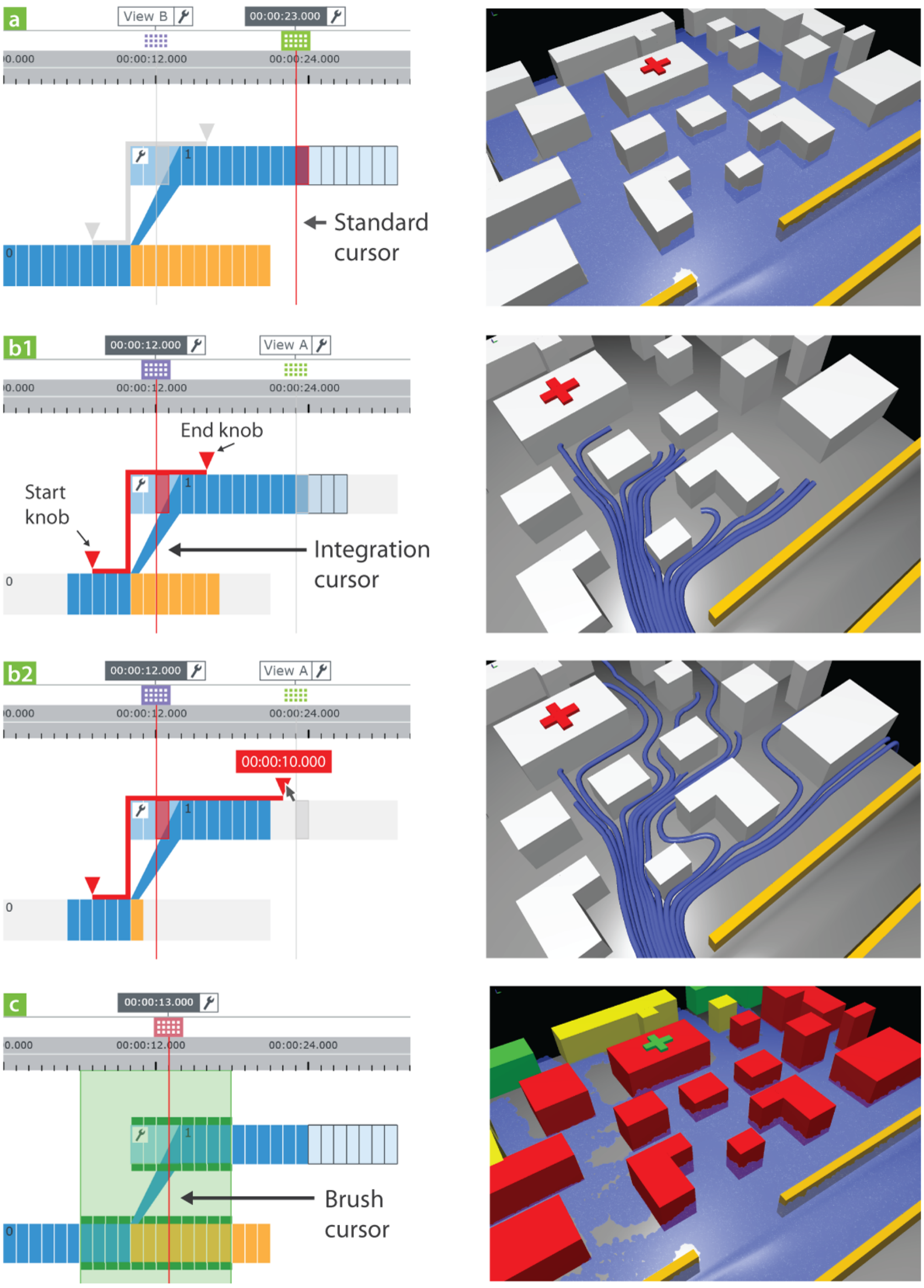


Figure 3.10 – Special purpose cursors. The right column displays renderings created with the active cursors of the left column. (a) Standard cursor for navigation to a single frame. (b1) Integration cursor to manipulate the integration range of the pathlines node. World Lines show the capabilities of 'View B' only. (b2) The user increases the forward integration time, consequently less frames can be produced for 'View B'. The rendered lines become longer. (d) Brush cursor to select multiple frames for comparative analysis.

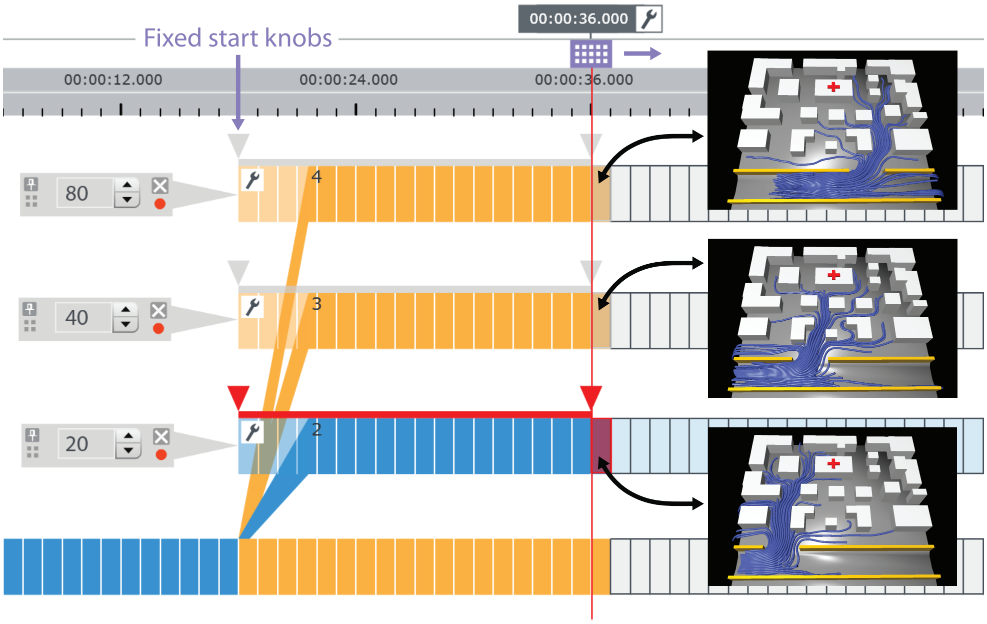


Figure 3.11 – Cursor grouping supports the synchronous comparison of material transport in three different levee-breach scenarios. The cursor group consists of three integration cursors, each of which has its own World Line. When moving the cursor group, the linked views display the temporal evolution of the pathlines synchronously.

if they are in danger, and colored in red, if they are flooded in **any** of the brushed frames. The brush cursor can be used to retain a selection of multiple frames relative to the cursor during navigation. In Section 2.7.1, the current time-step visualization mode for World Lines was presented. In this mode, all frames that are parallel at the current time are automatically compared according to user-defined analysis criteria. The brush cursor now enables a generic extension to this mode, allowing for the inclusion of additional frames. This is useful for analyzing statistical quantities such as average values across several frames instead of inspecting the frames at a single time step alone.

Per default, a cursor for each view node is generated. In addition, a node such as pathlines, can request the creation of associated special purpose cursors. Users can also create additional cursors and assign multiple nodes to them. This can be useful for bookmarking or to create cursors for the synchronized navigation of several selected views while keeping individual navigation cursors at the same time.

3.7.3 Cursor Grouping

A cursor can be associated with a list of views in order to navigate their nodes synchronously. In this case, the affected nodes receive job assignments for the computation of exactly the same frames. There are cases where the analyst is interested in inspecting the temporal evolution of nodes that are offset to each other with respect to time or parallel worlds. This is useful, for example, to monitor the evolution of several simulation scenarios in several views side by side (Figure 3.11). For this purpose, the concept of nested cursor grouping has been developed. A cursor group is formed through selection with a rubberband tool. Every action that is applied to one cursor of the group modifies all other members too. In a context menu, the user can specify whether the contained cursors should share a common World Line or a common time value or both. Figure 3.11 shows a group of three integration cursors to generate pathlines. All cursors share a common time value, but each of them has its own World Line to enable animated, comparative inspection of parallel worlds in three views. By moving the cursor group forward in time, we can compare the evolution of pathlines for three different levee-breach locations. Cursor groups can also be further nested, i.e., included into other groups in order to define complex view-linkage behavior. It is possible to analyze the data-flow network to group cursors automatically. An example is the combination of an integration cursor and a standard cursor if the underlying nodes are connected. It is important to note, that the ability to request several frames from a single node at once, is only possible with the data-flow processing of Schindler et al. [114].

3.7.4 View Navigators

In many visualization systems, monitors are equipped with a slider-based component directly below the views to enable navigation in time. This is an intuitive and fast way to change the time value of a view without interacting with other components. Based on this approach, we propose the direct navigation of a World Line in view navigators (Figure 3.12). A view navigator consists of three main components:

Cursor selector This is a list of cursors which are capable of navigating the view. To be included, a cursor must contain the related view node in its list of associated nodes. Through the cursor selector, the user opts for a World Lines cursor to be linked to the view navigator.

Surrogate cursor A view navigator is linked to a particular World Lines cursor. The surrogate cursor is a miniature version thereof. When navigating the surrogate cursor, the World Lines cursor reflects the changes and vice versa. We use color coding to track the visual linkage between surrogate and World Lines cursor.

Flattened World Line This is a horizontal navigation bar representing the World Line associated with the linked cursor.

The flattened World Line is able to transport basic information obtained from its counterpart in the World Lines view. This includes the visualization of progress and the indication of branch locations. We can now modify the list of nodes associated with cursors or use the aforementioned grouping methods to link the multiverse navigation between several view navigators in a highly customizable way.

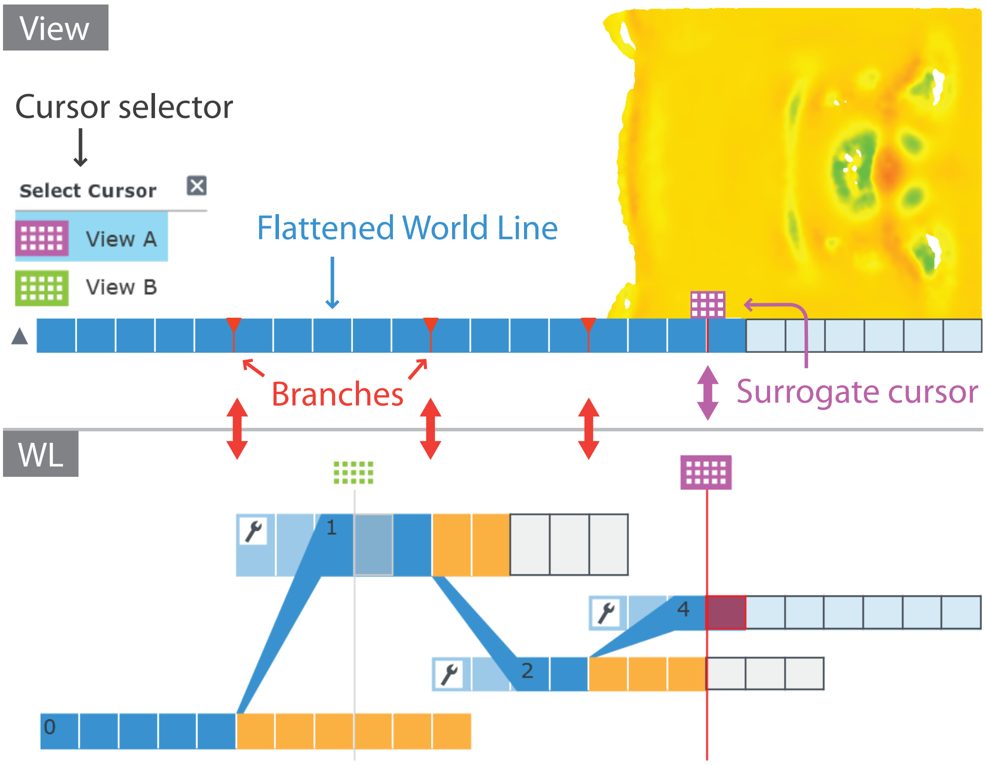


Figure 3.12 – View navigator below a view for convenient navigation. The view navigator displays a flattened version of the World Line that is associated with the selected cursor in the World Lines view (WL).

3.8 Scope-based Settings

To study alternative scenarios with World Lines, we require the ability to extend the capabilites of a node. In the use case of Figure 3.5, we want to model different types of levee breaches. For each structure of interest, we create a track through branching. Consequently, additional settings objects are constructed and assigned to the related levees node. Each of these objects has a *track scope*, since it is valid for all frames in the associated track. In this section, we generalize the idea of settings validity in a multiverse data-flow. At first, the system is extended to allow for settings at a *frame scope*. This means that users can assign settings to individual frames. This feature enables interesting routes for time-dependent parameter modification. As an example, we consider the analysis of fluid velocities in the levee-breach scenario. A transfer function controls the coloring of the fluid in a 3D view according to velocity magnitudes. At times, the transfer-function range, which is defined by the spatial minimum and maximum velocity, is almost constant across several frames. Occasionally however, the velocity field comprises peaks due to breaking dams or dropped sandbags. In this case, the analysis could benefit from a transfer function that

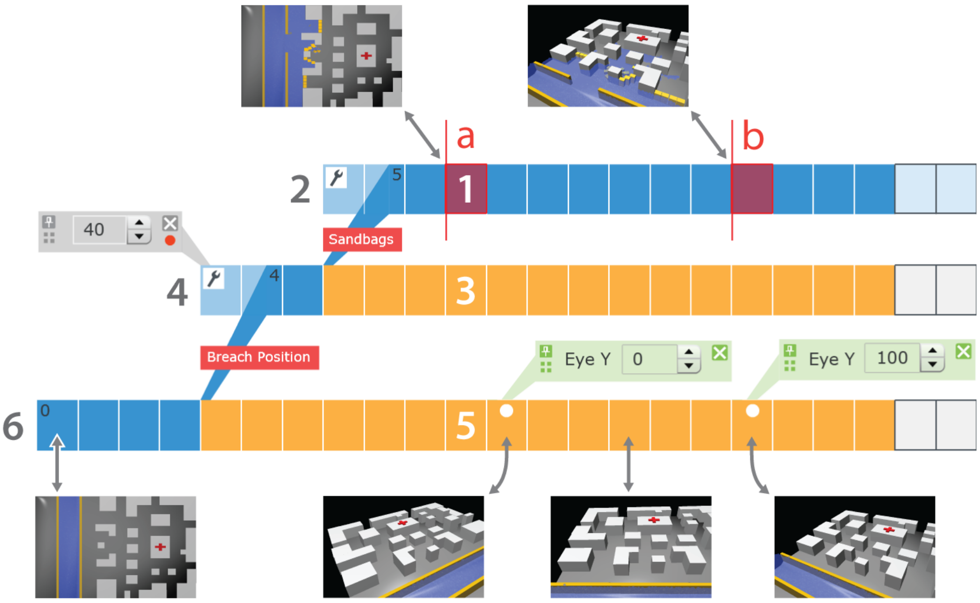


Figure 3.13 – Hierarchical lookup of scope-based settings. The root track has keyframes (white dots) that comprise individual settings for the camera position. The numbers depict the search path for settings when calculating the frame indicated by cursor a. The camera setting of the root track is found. At cursor position b, the system fetches the camera location defined at the parallel frame of the root track.

dynamically adjusts during navigation from frame to frame. To achieve this, World Lines provide the ability to assign settings to individual frames. Frames are furnished with a dot if they comprise individual settings (Figure 3.13). Similar to modifications at track scope (Section 2.6.1), inline widgets offer a comfortable way to alter the parameters.

Schindler et al. [114] demonstrate a hierarchical representation of settings objects inside a node. The proposed model is based on a mapping of the track hierarchy into internal data structures. A node automatically fetches the correct settings object during the computation of data for a frame. In the following, we explain this lookup from an application perspective. Figure 3.13 uses numbers to illustrate the lookup chain when requesting a frame at the position of cursor a. In the following text, we refer to these numbers in parentheses. At the beginning, the lookup algorithm searches for settings in the scope of the requested frame, i.e., the frame to compute data for (1). If nothing is found, the lookup continues in the scope of the track which contains the requested frame (2). From there, the algorithm searches in the parent track, starting at the frame that is parallel to the requested one (3). If this frame has no settings either, the scope of the parent track is inspected (4). This recursive search goes up the track hierarchy until a settings object is found or the root-track is encountered. In the scope of the root track (6), a settings object must be present.

The settings hierarchy can be exploited to achieve interesting effects. In Figure 3.13, we experiment with time-varying camera positions. Two frames of the root track define explicit camera locations along the y-axis which is parallel to the direction of the river flow. These settings have a time-global effect. This means, that all parallel frames of descendant tracks use the same camera perspective, if not overriden in terms of the settings hierarchy. When navigating to the frame at the position of cursor b, the scene is rendered with the same perspective as defined in the parallel frame of the root track.

To achieve smooth transitions of settings between several frames, we adapt an interpolation technique based on key-frames [33]. The frames that explicitely define settings become key-frames, the parameter values of frames inbetween are interpolated. This way, the user can animate parameters over time. In Figure 3.13, we apply this functionality to simulate a moving camera along the y-axis. Because of the hierarchical lookup, this camera movement applies to all tracks that do not explicitely override the camera positions.

3.9 Discussion

In this chapter, we have shown how to combine the modularity of a data-flow system with the multiverse exploration power of World Lines. As a motivation, we introduced a data-flow setup for the computation of complex, time-dependent features. The goal is to visualize pathlines which describe the transport of material through the water. To achieve this, users interact with multiple World Lines cursors. There is no need to deal with multiverse node configurations, the processing algorithms of Schindler et al. [114] take care of correct data-flow execution.

Multiple cursors provide flexible navigation of view nodes across time and alternative scenarios. The concept of cursor grouping can be exploited for comparative analysis in multiple views. A useful extension would be the auto-creation of view monitors when users create new cursors via cloning. This requires the ability to change data-flow nodes and connections through interaction with World Lines. Through the notion of scope-based settings, we have introduced the ability to specify settings at the frame scope. Currently, it is not possible to define key-frame parameters via interaction with linked steering monitors. The setup of the animated camera, for example, would be greatly simplified, if World Lines would provide the option to automatically record the current monitor perspective as a scope-based setting of the active frame. This approach is taken by modern 3D authoring tools, where users simply interact with views to modify key-frame values.

The presented approach taps new areas of application. The presence of a simulation node is not a requirement. The system can handle time-varying data that originates from external files. A file-loader node is in charge of reading several time steps. The capabilites of the file node contain several entries which are downstreamed to the view nodes. This way, we can use World Lines to analyze the loaded data. Moreover, we can use the branching functionality to record the user work-flow during visual analysis. In this respect, the contribution is similar to the VisTrails approach [120] which utilizes a graph-based representation of the analysis steps. The basic purpose of the VisTrails history-tree is to return the system state to previous states which can be regarded as a sophisticated version of undo/redo functionality. There are no flexible, time-dependent navigation tools, nor is it possible to synchronously compare the results obtained at different steps of the recorded work-flow. These limitations are not given, if using World Lines for the analysis of time-dependent data. As an example for improved comparative reasoning, we consider the investigation of vortices in a fluid-simulation data-set. We create two tracks that comprise different transfer-function settings. One transfer function is designed to highlight small eddies in the data, the other one to put the focus on large vortices. The evolution of the vortices can be inspected in two views side-by-side. In addition, branching can be used for optimized playback of different system states, because the application caches computed frames.

According to Thomas and Cook [128], presentation and production are often the most time-consuming parts of visual analysis. The discussed concepts can be exploited to ease the creation of production-ready videos. Monitors are already equipped with record buttons to export a sequence of images during World Lines playback. With the adaption of key-frames, the user can easily animate settings over time. Branching already provides a way of 'editing' a video. A World Line defines the route of a cursor when traversing the frames. More advanced scheduling methods could offer ways to create a visualization story. According to Wohlfart et al. [142], the viewer of a presentation can more easily understand the analysis implications if being able to interact and re-investigate the visualizations. The basics for this approach are naturally given by World Lines. When playing back an exploration process, the user can stop at any point in time and change parameters on-the-fly. An interesting direction for future research is the extension of World Lines to implement various kinds of story consumption, from total passiv viewing to total separation from the story. In this manner, the Visdom application is a system that targets a complete integration of all steps in the decision making. This involves simulation, computational steering, visualization and presentation.

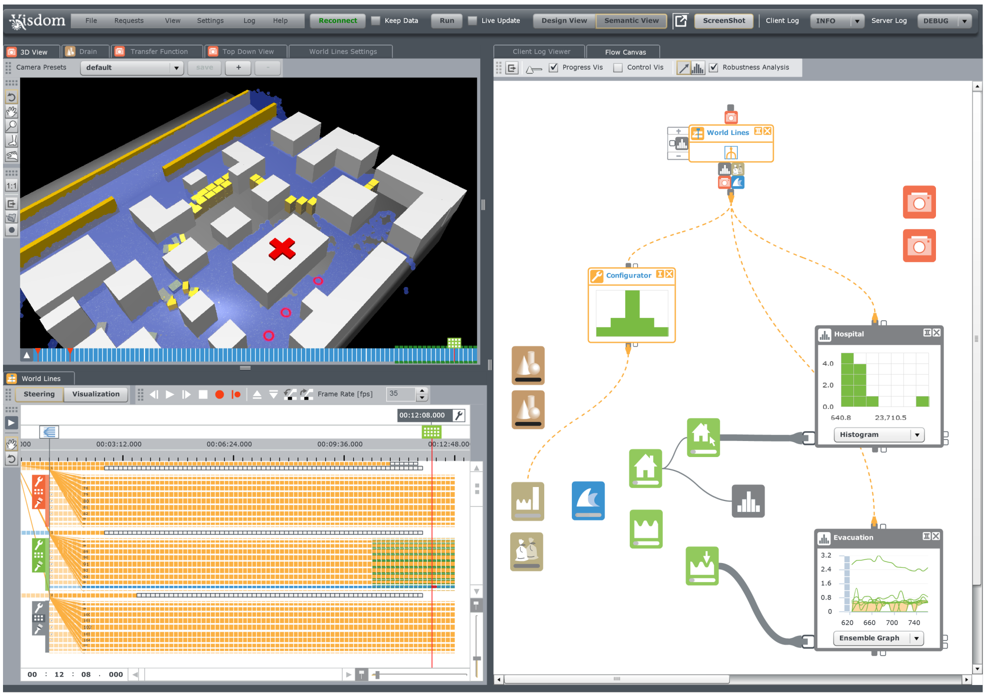


Figure 4.1 – Screenshot of the Visdom application utilizing Nodes on Ropes. The user investigates uncertainties with respect to levee-breach locations.

> Doubt is not a pleasant condition, but certainty is absurd.
>
> — *Voltaire*

Chapter 4

Nodes on Ropes

A successful decision support depends on a thorough analysis of uncertainties. We need to account for uncertain knowledge about the environment, such as levee-breach locations. The steering process has to reveal how these uncertainties in the boundary conditions affect the confidence in the simulation outcome. In this chapter, users steer parameter studies through the World Lines interface to account for input uncertainties. The transport of steering information to the underlying data-flow components is handled by a novel meta-flow. The meta-flow is an extension to a standard data-flow network, comprising additional nodes and ropes to abstract parameter control. The meta-flow has a visual representation to inform the user about which control operations happen. We expose system internals and show that simulation steering can be comprehensible at the same time. This is important because the domain expert needs to be able to modify the simulation setup in order to include local knowledge and experience. Finally, we present the idea to use the data-flow diagram itself for visualizing steering information and simulation results.

4.1 Introduction

World Lines provide the ability to test and compare alternative options in an integrated visualization environment. In chapter 2, we utilize World Lines to devise a robust emergency strategy for protecting a neighborhood from flooding. The starting point of the conducted study is a breach in the levees that causes city flooding. The task of the user is to design a breach closure by dropping sandbags in order to protect as many buildings as possible.

The case study in this chapter is based on different assumptions, inducing a modified set of requirements for the steering task. Figure 4.2 shows an overview of the simulation domain and the case-study tasks. We assume that a severe weather condition is steadily raising the water level in the river bed. A domain expert, who has experience and indispensable knowledge about local conditions [19], draws the conclusion that the levees are likely to fail within the next couple of days. The local expert proposes a small number of eligible precautions that can be realized in the given time frame. It is the task of the user to evaluate these measures in the steering

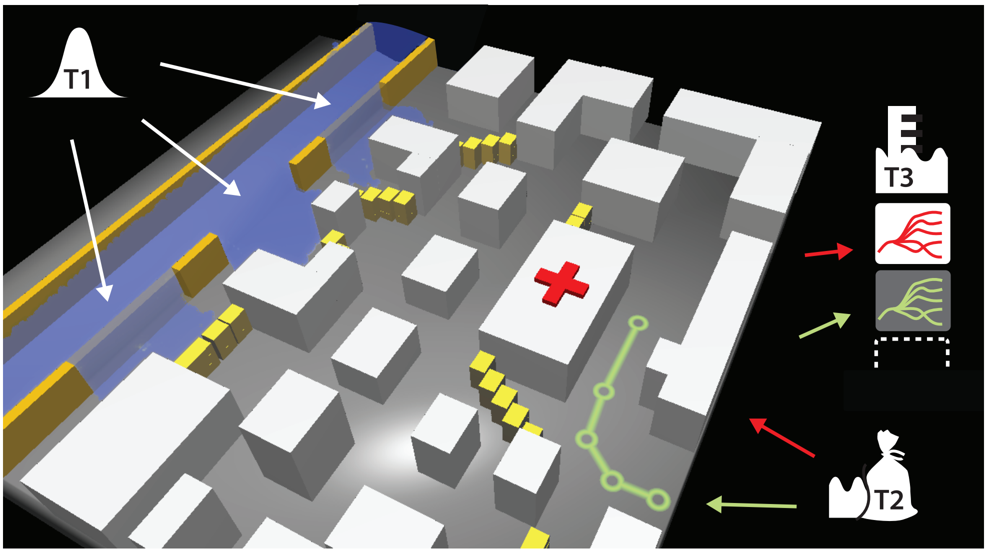


Figure 4.2 – Scenario and task description: (T1) The flooding expert is uncertain about the location of the imminent levee breach. (T2) Hospital (red cross) and evacuation path (green) have high priority, the damage to buildings has to be kept to a minimum. (T3) Expert needs to analyze many risk curves with respect to important results such as water levels to determine the best response strategy.

environment and pick the best response strategy. The focus is thus not on designing a breach closure on an already flooded city, but to choose a predefined response strategy that can be accomplished prior to the flooding. The decision making is guided by the following rules:

Task 1: The domain expert is uncertain about the exact location of the levee breach, but is able to define a confidence interval with respect to the breach location [11, 12]. According to Stewart and Melchers [122], a risk analysis should take into account a thorough investigation of the uncertainties. For this reason, the user has to perform an ensemble simulation to investigate a distribution of possible breach locations (see Figure 4.2, T1).

Task 2: A successful response strategy needs to fulfill certain requirements (Figure 4.2, T2). For example, the people in the hospital should be kept safe under any circumstances. An evacuation path from the hospital should be present at any time. Furthermore, the damage to buildings has to be kept as low as possible. The system has to allow that all relevant parameters can be entered and that all the necessary information is generated to verify whether these requirements are fulfilled. To accomplish this, the user needs a deeper understanding about what is going on in the underlying simulation setup.

Task 3: The steering process should yield risk visualizations [30, 74] for all generated information (see Figure 4.2, T3). In a scheduled simulation, these visualizations are updated each time new simulation data is available to allow for reasoning as soon as

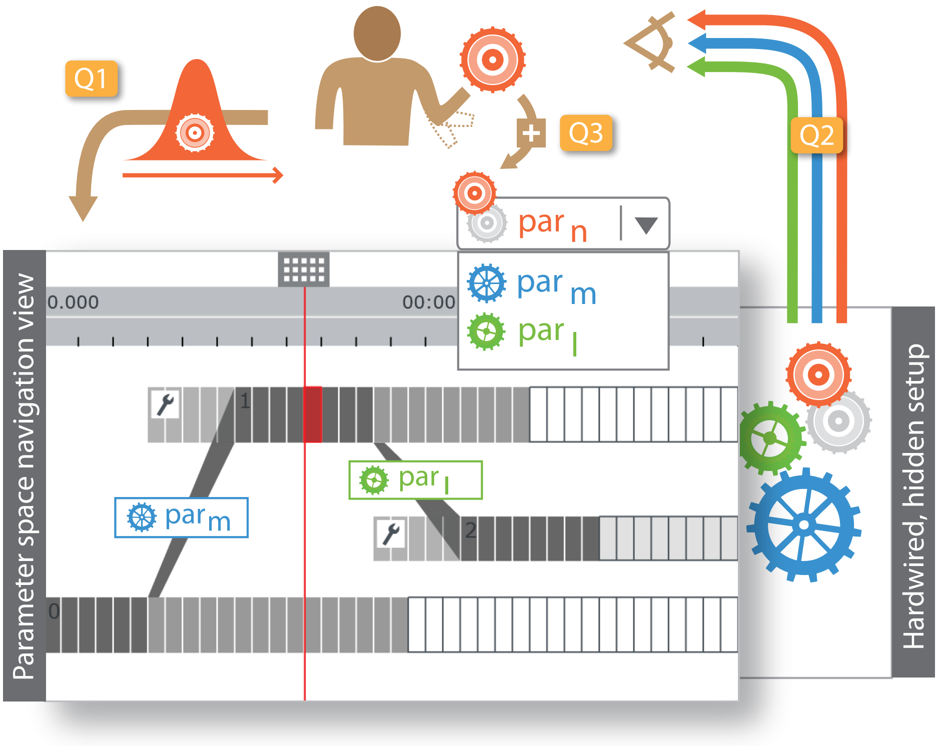


Figure 4.3 – Problem description: (Q1) How to create parameter studies to test uncertainties with respect to input parameters? (Q2) How to visualize what is going on in the underlying simulation setup? (Q3) How to modify simulation setup and extend parameter space to allow exploration with respect to input parameter par_n (red)?

possible. Each visualization shows the temporal distribution function of a computed result, ideally accompanied by uncertainty bounds. To simplify fast reasoning, the generated visualizations should be automatically organized in a concise overview. To our knowledge, there is no system to accomplish all user tasks in a single steering environment.

4.2 Problem Description

In this section we generalize the requirements for the presented system. Figure 4.3 summarizes the problems involved with state-of-the-art applications which we tackle in this chapter. The steering process that has to be perfomed is equivalent to interactive navigation in a multi-dimensional parameter space with the goal to generate insight. Modern simulation-steering environments provide interactive navigation views to ease this exploration in parameter space. The World Lines view [139, 3] is an example for such a view. In this view, simulation runs are represented as tracks that evolve in time. With World Lines, parameter space navigation is accomplished via the concept of branching: when a parameter changes, a new track is created which covers an additional point in parameter space. This navigation approach is not sufficient if we have to take uncertainties into account. A logical next step is to introduce the ability to simulate parameter studies according to statistical distributions (Figure 4.3, Q1). Analogous to interval arithmetic, simulation steering is not done for one or a few values of a specific parameter but for entire intervals or distributions thereof.

Navigation views usually operate on a hardwired simulation setup that is opaque to the user (Figure 4.3, Q2). Hiding internals in such a black box can be the right choice for the unexperienced user. In many cases however, true understanding of a phenomenon is only gained if the user has access to information contained inside the black box. Better insight is obtained if the user learns about how the system parts change during navigation. State-of-the-art applications lack a concise visualization that shows which system parts are affected by navigation, and that automatically updates the presentation after every relevant change. Moreover, it can be essential to be able to quickly identify important properties of intermediate results such as fluid pressure, barrier movements or water levels, that are generated by the various components of the system. Ideally, a visualization of internal system parts is also capable of presenting an overview of relevant information computed by each part.

The user requires the ability to modify the simulation setup itself (Figure 4.3, Q3). During interactive exploration, it is possible that the parameter space has to be extended to cover additional input dimensions not considered in the first place. This option also provides the ability to investigate different, related systems. Moreover, users should be able to alter the system dynamically to generate additional information needed without loosing any results that have been computed before the modification.

4.3 Proposed Solution

Our solution to the problems is based on a standard data-flow system. Each node of a data-flow diagram can contribute dimensions to the parameter space. Even though modular and extensible, a standard data-flow is not sufficient to solve the presented problems. Steering a simulation via dynamic modifiation of a data-flow is not feasible. We suggest a smart data-flow diagram which is extended by a meta-flow of steering information. The data-flow determines the transport of data that is generated during the execution of the simulation algorithm, including simulation results, derived data and the visualization. The purpose of the meta-flow is to provide an advanced control of the parameters required throughout the data-flow. This way, the communication of parameters between nodes is decoupled from the standard data-flow, enabling flexible information exchange and control exchange between nodes.

Inspired by the coordination graph in Improvise [140], we visualize the meta-flow as a an additional network of ropes that is directly embedded into the flow diagram (Figure 4.4). The meta-flow is built with ropes (dashed lines) to top and bottom connectors of nodes while the data-flow propagates trough wires from left to right.

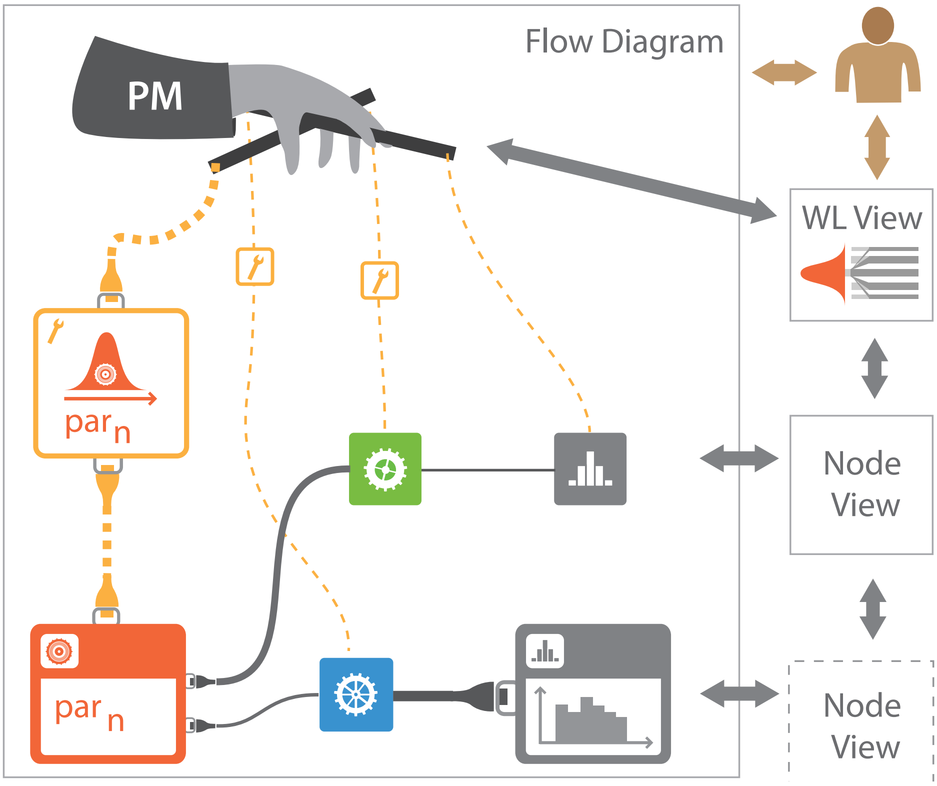


Figure 4.4 – Proposed solution: The simulation setup is presented in a data-flow diagram that is extended by a meta-flow (yellow, dashed lines) to decouple parameter control from data propagation. World Lines (WL) navigates via the puppet master (PM), a special meta-node which employs meta-communicators (yellow nodes) to control other nodes. The meta-flow highlights the control path to the red node, since in the WL view, the user studies uncertainties with respect to par_n (red). The thickness of data-flow wires (grey lines) reflects scalar values of computed results on a per-wire basis. The robustness-analyzer nodes (grey) show more detailed information.

The interactive visualization of the data-flow and the meta-flow enables straightforward manipulation of the computations performed during simulation. The idea is to add novel types of meta-nodes which do not perform computations but which control other nodes. Such a node operates on a complex flow network in a way comparable to a puppet master pulling the strings. Therefore we call these control nodes puppet masters (PM). A puppet master utilizes special helper nodes (meta-communicators) to gain fine-grained control over nodes, enabling interaction, visualization and steering inside large parameter spaces. World Lines employ a puppet master to steer a user-defined subset of nodes. In the flow diagram, users can modify the explorable parameter space by configuring the meta-flow.

The meta-flow enables modifications of control paths within the flow diagram. Still, the behavior of the underlying flow network and of individual nodes remains difficult to grasp. We believe that the behavior of individual system parts is best understood in the place where users create them, namely in the flow diagram directly. We suggest the extension of the flow diagram with emphasis techniques from visualization such as levels-of-detail (LOD). The goal is to dynamically highlight important per-node information about the meta-flow and computational results. The flow diagram in Figure 4.4 modifies the thickness of ropes and the LOD of nodes to identify the most relevant parts in the current steering process. With this technique, it can be seen that the user mainly modifies parameter par_n, provided by the red node. In this manner, the meta-flow visualization can be regarded as a concrete example for automatically storing knowledge gathered during interactive exploration [29].

In addition, we create direct visualizations of computed data inside the flow diagram. Figure 4.4 gives examples for such visualizations. Statistical quantities are displayed on data-flow wires and inside robustness-analyzer nodes. The thickness of the wires encodes scalar values like mean or standard deviation. The diagram itself becomes a dynamic view that is an integral part of a system of multiple linked views associated with certain nodes. This constellation improves the linkage between nodes in the data-flow and the resulting data that is shown in the associated views.

The contributions of this chapter can be summarized as follows:

- Configurable meta-flow as an abstraction for information and control exchange between nodes.
- Puppet masters for centralized navigation in parameter space spanned by nodes.
- Dynamic visualization inside the flow diagram. The ropes and nodes of the meta-flow depict the control-flow between nodes, the work-flow and inherent knowledge. The wires and nodes of the data-flow dynamically indicate the computed data per node and connection.
- Parameter studies in the World Lines navigation view to evaluate uncertainties through calculating with intervals and distributions.

4.4 Related Work

Popular modular visualization environments (MVE) such as AVS [6, 133] or IRIS Explorer [98] are based on the visualization pipeline model [48]. They provide modules to accomplish the constituent tasks in the visualization process [24]. In a graphical interface, the different modules (nodes) are presented as boxes. They have input and output ports and are connected via simple mouse clicks. The connections represent the data-flow between the modules which all run as separate processes. A key feature of MVEs is their extensibility [143]. The simulation process can be incorporated into an MVE providing the opportunity to steer while using visualization components [116, 102, 125]. Hyperscribe [144] employs a history node that is part of the data-flow to record provenance during the steering process. The recorded data can be navigated and reused by sending it upstream from the history node to other nodes.

To enable the transport of node parameters from node to node in the data-flow of AVS [6], parameters can be converted to node connectors. An alternative solution is to connect node parameters via a special-purpose connection [18, 90]. If connected this way, the values of parameters are synchronized. Felger and Schröder [40] suggest to extend the data-flow with a backwards pipeline of inverse mappings to enable picking and other interactions with visualizations. In Improvise [141], the coordination between views is configured using relational expressions and can then be visualized as a coordination graph.

Further advancements in data-flow systems are related to the process of building visualizations. Macros are reusable groups of nodes that can be treated effectively as one module [1, 6]. SmartLink [126] suggests assistants which take the history of user decisions into account to propose connections between nodes. Scheidegger et al. [113] use provenance information to guide the creation of data-flow nodes and provide automatic suggestions. VisTrails [120] adapts a history tree for capturing and reusing provenance while modifying the data-flow setup. VisMashup [111] simplifies the creation of custom applications by leveraging an existing collection of visualization pipelines and their provenance. MeVisLab [1] uses halos to improve the visibility of connections between nodes in a complex network. To reduce clutter, the user can toggle the visibility of connectors.

Level-of-detail Context sensitive level-of-detail (LOD) is a common technique to improve the layout of interfaces. Matkovic et al. [88] discuss virtual instruments which are capable of displaying the current value and the value from the near past. The instruments can be anchored to 3D models and assume different levels of details at different zoom levels. Continuous Zoom [14] is a fisheye-based method for navigating large networks. The technique divides the display into rectangular regions and enlarges individual sections based on scale factors and degree-of-interest.

Visual Programmig Visual programming approaches focus on creating executable data-flows at a very low level, similar to a text-based programming environment [65]. Therefore the resulting networks tend to be more difficult to understand than those which are created in MVEs. LabVIEW [96] is the most prominent example. Because of the complexity of the created data-flows, there are several proposals to visually augment the flow diagram. For example, wires are colored according to the data type they transport [37]. The data-flow allows for loop and switch statements that are shown as boxes which can contain sub-data-flows [86]. Impure is a visual programming language written in ActionScript to visualize data from the internet [16].

Navigation in parameter spaces Parameters are core components of any computation. By modifying parameters, the user navigates from point to point in a multi-dimensional parameter space. There are a number of different interaction techniques to ease this navigation which are usually tailored to the task at hand. The most common way of navigation in parameter space is through selection from parameter

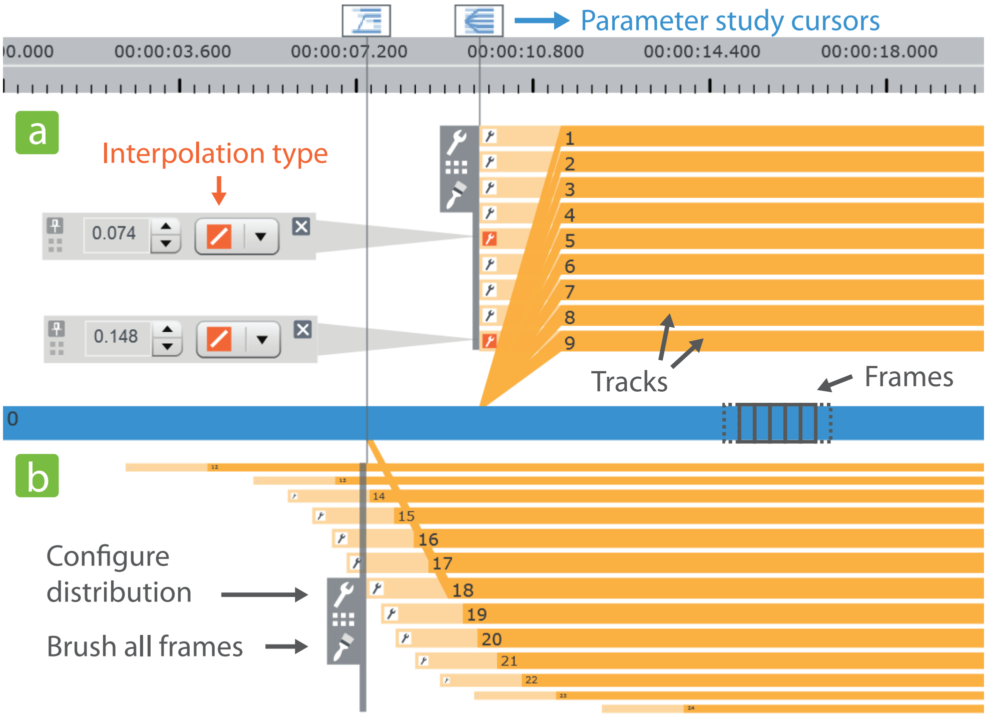


Figure 4.5 – Parameter studies in World Lines. (a) Parallel parameter study using a custom distribution function. (b) Temporal parameter study using a Gaussian distribution of track start times.

studies. Often this is done by entering numbers into a regular grid [10, 72]. The AVS [6] animator module is a data-flow node for generating keyframe animations. The node has access to all available node parameters in order to create snapshots of points in parameter space. Bruckner and Möller [22] sample the entire input parameter space to perform many simulation runs offline. The results are sorted according to similarity and presented for exploration in a cluster-timeline view. Amirkhanov et al. [8] present a parameter-exploration system to test the stability of specimen placements in computed tomography. The World Lines approach [3, 139] follows the concept of steering where simulation data is generated on-the-fly. Here, parameter space traversal is accomplished via the creation of new simulation runs. The World Lines view is an interactive visualization that represents simulation runs as tracks.

4.5 Parameter Studies with World Lines

In this section we propose an approach to account for uncertainties with respect to simulation input parameters using World Lines. In Section 2.6.1, we have shown how the user can explore the parameter space by interactive branching: any modification of a parameter creates a new track that visually originates from the parent track. However, this way only one parameter can be modified at a time and it can be time-consuming to test many alternative choices. To speed up the exploration process, we suggest the introduction of parameter studies. Instead of single branches, the investigator opts for one of the following parameter-study types:

Parallel parameter study This type creates an ensemble of tracks that originate from a common frame and that share a common start time (Figure 4.5a). This can be used for example to test different levee-breach positions given by a statistical distribution.

Temporal parameter study This type investigates different branching times for the same parameter change, resulting in an ensemble of tracks that have different start times. (Figure 4.5b). This can be used for example to test alternative times of levee-breach occurrence again given by a statistical distribution.

In the World Lines view, a *parameter-study cursor* serves as a component to visually group all tracks that are contained in the associated parameter study (Figure 4.5). Users can choose between sampling according to a predefined statistical distribution or according to a custom function. Figure 4.5b shows a temporal parameter study that is based on a Gaussian distribution of the track start times. Here, the cursor's temporal position is given by the mean value of the distribution. Dragging the cursor in horizontal direction adjusts the mean value and shifts all tracks in time. An embedded configuration panel can be used to further configure the distribution properties. In addition, the parameter-study cursor provides convenient interaction techniques to apply operations to all associated tracks at once. This includes dragging, to customize the layout, brushing, to select all frames contained in the parameter study, and collapsing, to reduce visual clutter by showing only the track that is associated with the mean value of the distribution. Custom parameter-study functions are evaluated by interpolation. The user selects two or more tracks of the parameter study that should contain parameter values to be interpolated. Value and interpolation type are specified through inline widgets (Figure 4.5a).

4.6 Meta-flow

The previous section discusses a novel way to setup parameter studies using the World Lines view. This section describes how this information is transmitted to the underlying data-flow network to perform the required computations. A standard data-flow is a static graph that defines a specific application. In such a hardwired setup, it is not easy to define and understand how nodes communicate. We propose the management of node parameters and node relations in a separated meta-flow network which is directly embedded into the data-flow diagram. Via ropes, nodes are able to control other nodes. To provide the user with more precise control over this steering mechanism, we employ special meta-flow nodes called *meta-communicators* that

meta-node category	creation	requires rope (from - to)	purpose and example
puppet master (PM)	manual	meta - meta	controlling many nodes (WL)
meta-communicator	automatic (part of rope)	meta (of PM) - meta	fine-tuning of control behavior per rope (configurator)
		settings/results - settings	parameter linking, interaction (linker)
parameter provider	manual	settings - settings	provide item to list-based parameters (particle emitter)

Table 4.1 – Overview of components in the meta-flow.

are treated as part of a rope. These meta-communicators are automatically created as soon as a rope is established. The meta-flow allows the user to configure and perform complex steering tasks without the need for extensions to the linear data-flow model, such as looping or upstreaming [103]. The main advantage is that the control operations are now decoupled from data-flow execution. In the flow diagram, this separation is visualized as a vertical meta-flow through ropes and a horizontal data-flow through wires. Users can now cleary see and modify how the nodes control each other. Table 4.1 provides an overview of all meta-flow nodes and their functionality in the presented system. In the following sub-sections we give a more detailed explanation of these components.

4.6.1 Puppet masters

Puppet masters are a generic concept to steer many nodes. A puppet master is realized as a meta-node that is able to gain full control over all meta-connected nodes. Puppet masters are able to automatically navigate a data-flow system in parameter space. The connection between a puppet master and a controlled node needs to be established via the meta-flow connector named *meta* located both at the top and at the bottom of each node. An unlimited number of ropes is allowed to be attached to this connector. Users can influence the steering behavior with respect to individual nodes via the special meta-communicator named *configurator*. A configurator is automatically placed onto a rope as soon as the rope is created. In addition, a puppet master can be associated with an optional view to simplify parameter-space navigation.

Figure 4.6 shows a screenshot of a data-flow network for simulation steering that utilizes World Lines as a puppet master to modify parameters across several nodes. The World Lines view is coupled with this puppet master which has the required knowledge and power to transmit the user-defined steering information down to the meta-connected nodes. The user can modify the available parameter space by adding and removing ropes to specific nodes. The puppet master makes use of the configurator nodes to give the user detailed control over which parameters should be steered. In such a configurator node, the user selects steerable parameters for modification in World Lines. Furthermore, the configurator node can be used to directly initiate branch events and parameter studies in the World Lines view. The computed parameter distributions of parameter studies can be visualized as histograms or line graphs within the configurator node. The key advantage of employing World Lines as a puppet master is to achieve a generic, centralized control strategy, separated from the data-flow and presented in a concise visualization. Since parameters are no longer passed through the data-flow, we do not have to overload the data-flow with all the connectors and wires that are required to transport time values, track information, or the various parameter values. The World Lines node has input connectors for data-wires as well. These connectors are present, because the World Lines view has the ability to directly visualize data-flow results through coloring of tracks and frames [3, 139]. In the presented framework, it is simple to implement additional puppet masters. For example, we employ an animation puppet master for changing parameters when creating animations. We are currently working on a unit-tester puppet master which supplies meta-connected nodes with fixed input parameters and checks their output for correctness. A perturbation puppet master adds small random changes to the parameters of all meta-connected nodes. Furthermore, we are working on transfering the visual human+machine learning [44] functionality into a puppet master which controls brushes in information visualization views.

4.6.2 Parameter linking

Voreen [90] uses connections that are separated from the data-flow, to enable parameter linking between nodes, such as coupling the camera perspective between views. We utilize the meta-flow concept to accomplish this task. Nodes are equipped with two optional meta-flow connectors, namely the *settings* connector placed on top and bottom of a node and the *results* connector, located at the bottom only since node results can only be read. The results connector allows linkage to information that has been generated by a node during data-flow execution. The properties of the linking mechanism are customized in a special meta-communicator, the *linker*. The rope visually reflects the direction property of the linkage (directed or bi-directional) through arrow glyphs.

The introduction of *parameter providers* offers a flexible way for the specification and sharing of list-based parameters. A typical list-based parameter is the set of particle emitters of a simulation node or the set of light sources of a 3D view. A parameter provider is a meta-node that supplies its value (e.g., a single light source) as an item to the list-based parameter of meta-connected nodes. Figure 4.6 shows two emitter-parameter providers that are meta-connected in order to set the particle emitters of a fluid-simulation node. The advantage of this approach over existing systems is that we can supply an arbitrary number of items to a list-based parameter while sharing individual items among nodes. A standard data-flow system would require a variable number of connectors to accomplish this task.

Parameter linking of results to settings can be used to implement user interactions. In this work, we use this mechanism to enable spatial selection in views (e.g., to define an evacuation path or to select a building). When clicking into a 3D view, the underlying view node is executed and transforms the mouse coordinates into a point in 3D space. This point is available as a node result to other nodes and can be used to

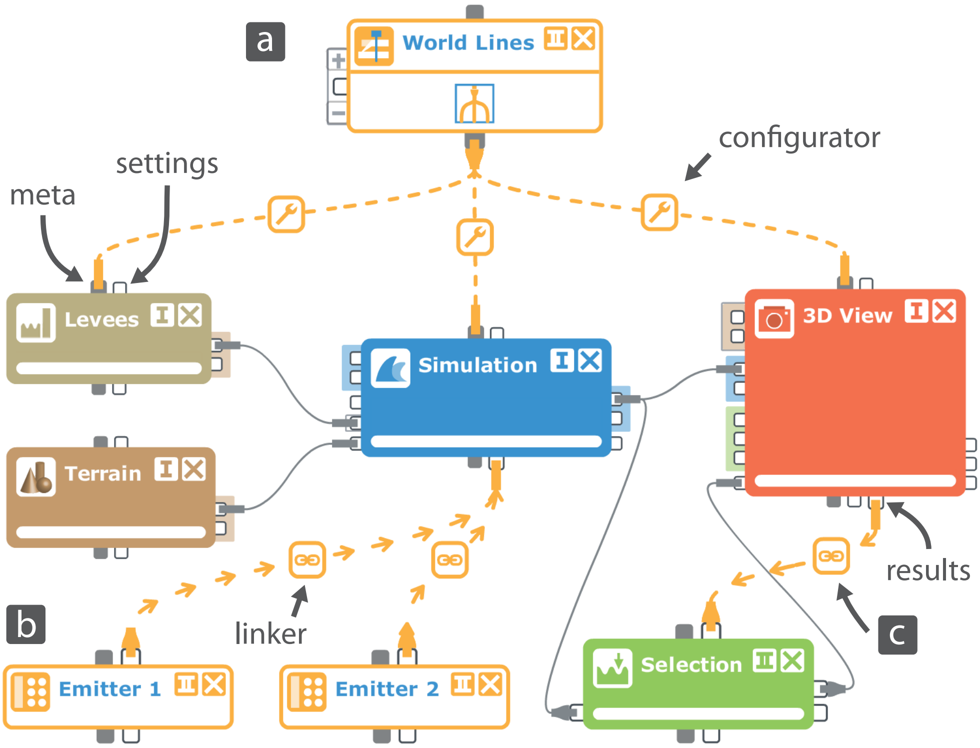


Figure 4.6 – The role of the meta-flow for simulation steering with World Lines (WL). (a) WL puppet master controls nodes via configurators that are part of ropes. (b) Parameter providers supply emitter settings to the fluid simulation node (blue) through linker nodes. (c) Interaction (spatial selection in fluid) is accomplished through linking from the results meta-connector of a 3D view (red) to the settings connector of an upstream node (green).)

determine selected items in space. This approach is comparable to the upstreaming-technique in standard data-flow systems. With our approach, we avoid such changes to the data-flow model and we do not have to introduce new data-flow connectors for each parameter that the user wants to link. Moreover, for our spatial selection nodes, we provide the necessary linking specifications by default and the user only has to establish a rope from a 3D view to the node. The 3D view then automatically supplies the necessary tools (such as a picker tool) to let users accomplish the spatial selections.

4.7 Dynamic Visualization with the Flow Diagram

Until now, the presented meta-flow improves the user situation in terms of building a network for parameter-space navigation. Puppet masters utilize the meta-flow to transport steering information to the meta-connected nodes in a fully automated manner. To this point, the details of this communication remain transparent to the user. However, in many cases it is desirable to understand what is happening in the network at runtime while navigating in parameter space. In this section, we explain how to use the meta-flow components to visualize the control-flow between nodes, the work-flow of the user and his/her inherent knowledge. In addition, we show how to use the data-flow to present a concise overview of relevant computed data inside the flow diagram. Before we go into details about the dynamic visualization, we explain basic design decisions.

4.7.1 Basic design

Figure 4.7 shows the basic design of a node in the flow diagram. The most important widgets for parameter modification are displayed inline. A node may contain an optional image to show a generated rendering or visualization inline. Meta-flow connectors are simply arranged on top and at the bottom of a node. The connector structure with respect to the horizontal data-flow tends to be more complicated and demanding. For this reason, we organize data-flow connectors internally and visually into *sequential and associative connector groups*.

Associative connector groups contain a set of connectors that are related to each other. For example, the associative connector group *geometry* comprises a mesh connector and a color connector. Another example is the connector group *particles* which contains a velocity connector and particle positions. This way it is possible to access the whole particle field and to provide quick access to the important velocity attribute. The 3D-view node in Figure 4.7 comprises the three dominant associative groups that are required for the flooding scenario: The *particles* group, representing water, the *geometry* group, to handle terrain, buildings and levees, and the *sandbags* group, representing the sandbags.

A sequential connector group contains a dynamically alterable number of the same connector or associative connector group. Using such dynamic input often leads to a reduced number of required nodes to accomplish a task. As an example, consider a view node like in Figure 4.7 that is capable to display geometry. With the sequential connector group, it is possible to connect several geometry objects to this view node without the need for a series of nodes that merge the geometry into one data structure beforehand. To our knowledge, there is currently no data-flow system that allows the user to dynamically change the number of input connectors of a node.

The visual appearance of nodes (e.g., color and border) is determined by the node category (e.g., view, simulation, geometry). Figure 4.8 provides an overview on all nodes that are used to study the levee-breach scenarios of this thesis. Connectors and connector groups are styled and shaped according to the nature of the transported information (e.g., geometry). Connection styles such as color and line style depend on connection type (meta-flow rope, data-flow wire, ropes that represent parameter links). Connections are further equipped with plugs on either side. These plugs can be dragged interactively to change connections.

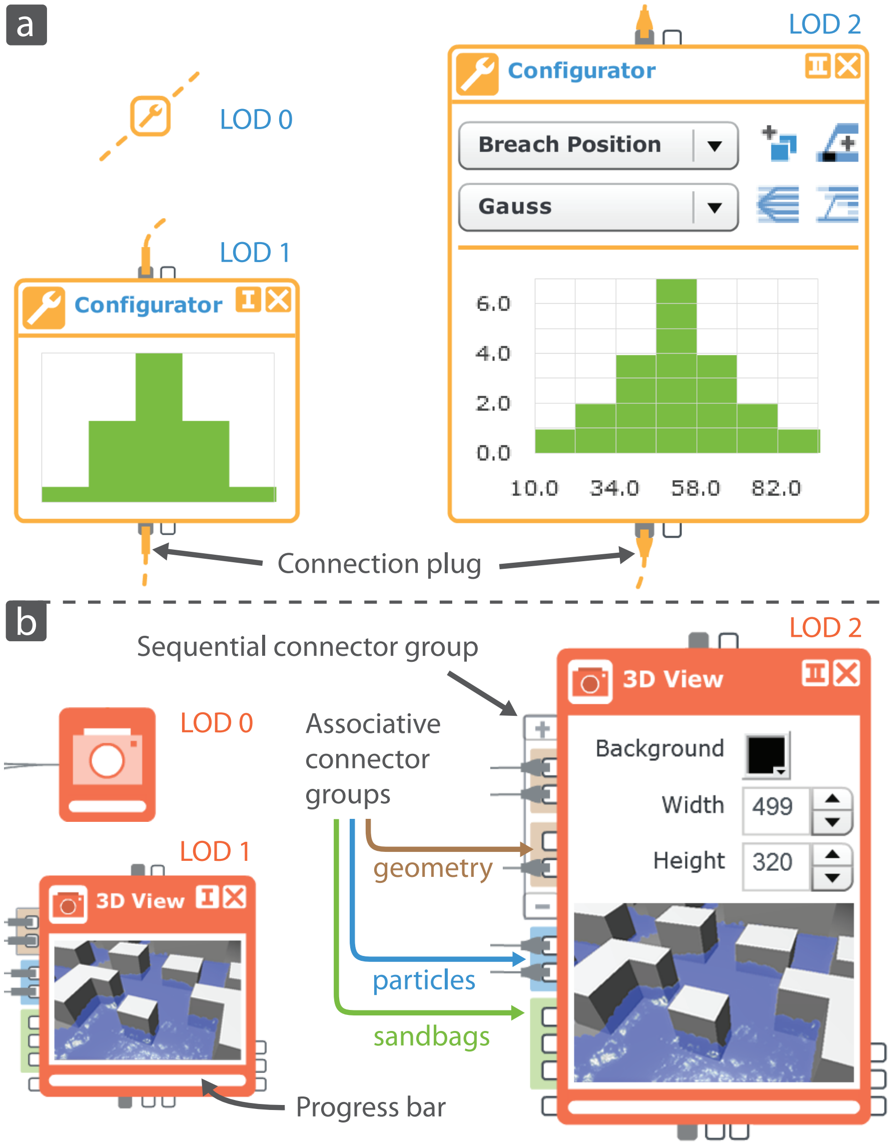


Figure 4.7 – Nodes at different levels of detail (LOD). (a) At higher LODs, the meta-communicator node (configurator) shows the input distribution of a parameter study (green bars) associated with the data-flow node the configurator is responsible of. (b) Data-flow node (View 3D) with connector groups and inline components.

4.7.2 Flow diagram simplification

To create concise, dynamic visualizations in the flow diagram, it is vital to reduce the visual clutter introduced by a potentially large number of meta-flow and data-flow components. In this section, we adapt a number of simplification strategies to reduce the information amount displayed. Other systems let users combine nodes

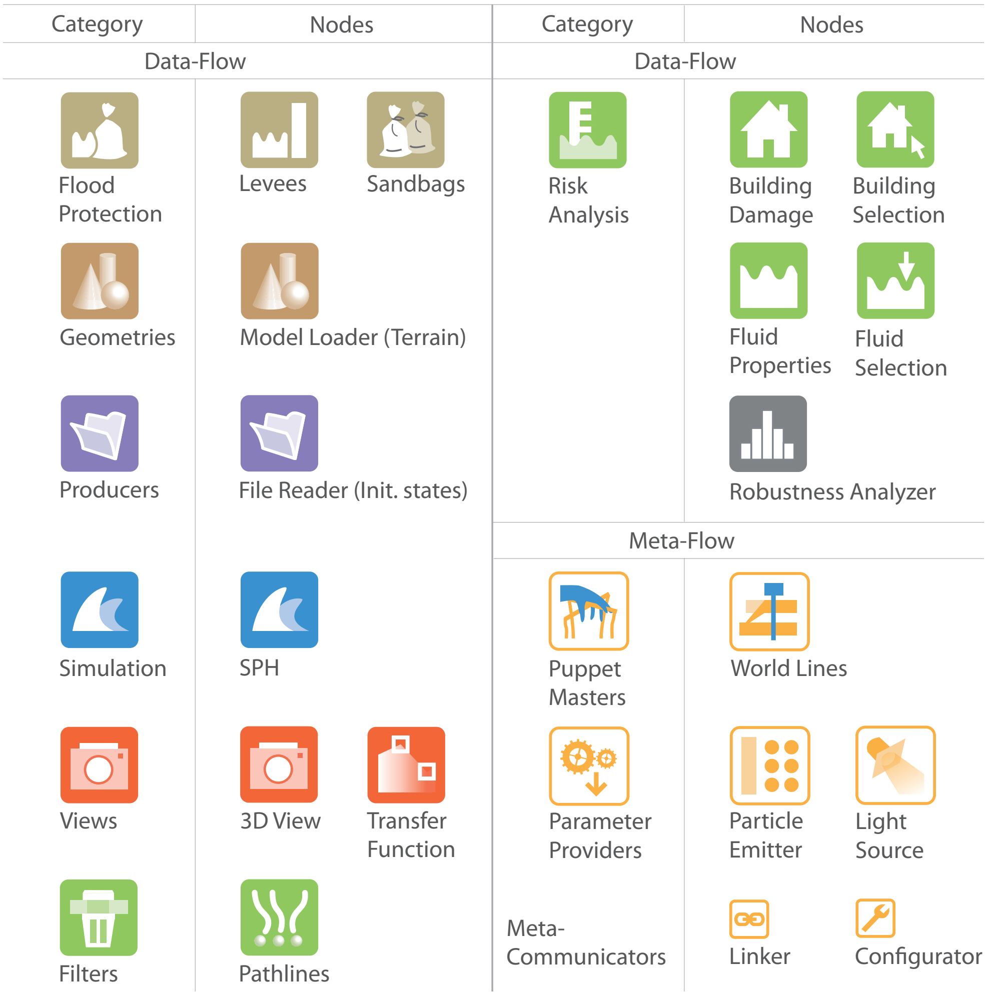


Figure 4.8 – Categorization of all nodes that are used to investigate the flooding scenarios in this thesis.

into a sub-network and place them into an aggregated node [111]. In this section, we investigate other simplification strategies to reduce the information amount displayed. Since the goal is a dynamic visualization, it should be possible to perform these simplifications automatically while preserving the mental model for the user. The discussed simplifications are level of detail and connection reduction.

Level-of-detail (LOD) When networks become larger, it is increasingly more difficult to get an overview of the complete setup. In case of an interactive flow diagram, it is not feasible to apply a standard zooming technique where elements are simply scaled. Relevant information has to stay readily perceivable. The most important interaction points have to remain large enough for user interaction. This is why we

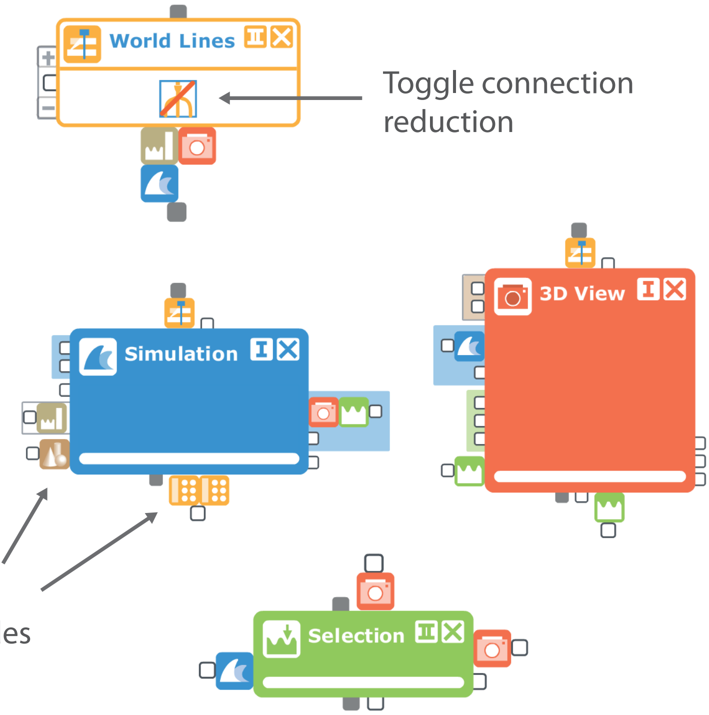


Figure 4.9 – Screenshot of the flow diagram in Figure 4.6 after applying connection reduction to all ropes and wires. If a node has only one connection, it is not visible.

propose a level-of-detail zooming approach. In this work, we are able to reduce the visual clutter of our flow diagrams significantly by introducing three different levels of detail. The appearance of a data-flow node and a meta-communicator for each LOD is depicted in Figure 4.7. At LOD 2, a node displays all inline elements and shows all connectors and connector groups at their full extent. At LOD 1, sequence connector groups are reduced to display only one item. All connectors are scaled down, connection plugs receive a smaller shape and some inline components are hidden. If inline space remains, the node tries to fit in an optional image or other inline components. At LOD 0, the node is completely reduced to a small icon and all connections to a node lead to the same point. Meta-communicators have a smaller size than other nodes at LOD 0 since they are part of ropes and occur quite frequently.

To give an adequate overview of larger networks, we apply scaling to node positions at lower levels of details. A simple coordination of node space takes care of avoiding node overlap when increasing the level-of-detail. Animated transitions are used to give the impression of smooth scaling.

Connection reduction The flow diagram can become confusing if all ropes and wires are visible at once. A *connection reduction* can be performed to decrease the number of connections visible at a time. In the reduced state, a connection is invisible.

If one of the nodes has no other connection than the reduced one, the node is invisible as well. To retain a visual correspondence between two connected nodes, each of them displays a small icon representation of the other node (Figure 4.9). If the mouse is moved over the icon representation, the connection is temporarily displayed. By clicking it, the user can make the node and its connections fully visible again. Puppet masters provide an inline button to quickly toggle reduction of all their ropes at once.

4.7.3 Visualization of Control

Any user interaction in the system, such as a change of the perspective in a 3D view or the creation of new tracks in World Lines, causes a modification of parameters in a subset of the nodes. For some nodes parameter changes occur more frequently, while others remain unchanged most of the time. This fact suggests to introduce an importance value for nodes. These values are updated after every control operation. If the node k is directly controlled, the importance value I_k is increased according to the expression

$$I_k = min(1, I_k + c_{inc} \cdot w) \qquad I_k, w, c_{inc} \in [0,1] \tag{4.1}$$

where c_{inc} is an empiric constant (set to 0.05) and w defines the weight of the control operation. In our system, we set this value to 1 for all operations except for changes to the camera perspective in the 3D view, where we assign a value of 0.01. This operation occurs very often and should have a smaller impact to the layout of the flow diagram. All other nodes, i.e., the nodes that are not directly affected by the control operation, receive a small decrease of their importance value I_k according to

$$I_k = max(0, I_k \cdot (c_{dec} + (1-w)^{1-c_{dec}})) \qquad I_k, w, c_{dec} \in [0,1] \tag{4.2}$$

where c_{dec} is a constant we set to 0.95. While interacting, the flow diagram automatically changes shape to direct the user attention to the most important parts. To accomplish this, the aforementioned simplifications are automatically applied according to the importance values using the following lookup:

- $I_k < 0.1$: node k receives LOD 0; apply connection reduction to all ropes connected to node k
- $I_k \in [0.1, 0.5)$: node k receives LOD 0
- $I_k \in [0.5, 0.8]$: node k receives LOD 1
- $I_k > 0.8$: node k receives LOD 2

Meta-communicators are excluded from these automatic LOD modifications. If any LOD value changes a re-layout of the node network is necessary. We try to move a subset of the nodes, if possible, to fill the gained space and if required, to provide the missing space. These automatic movements respect the relative node positioning of the layout to retain the mental model. We apply animated transitions to let users more easily perceive the modifications. If the flow diagram does not fit into the screen

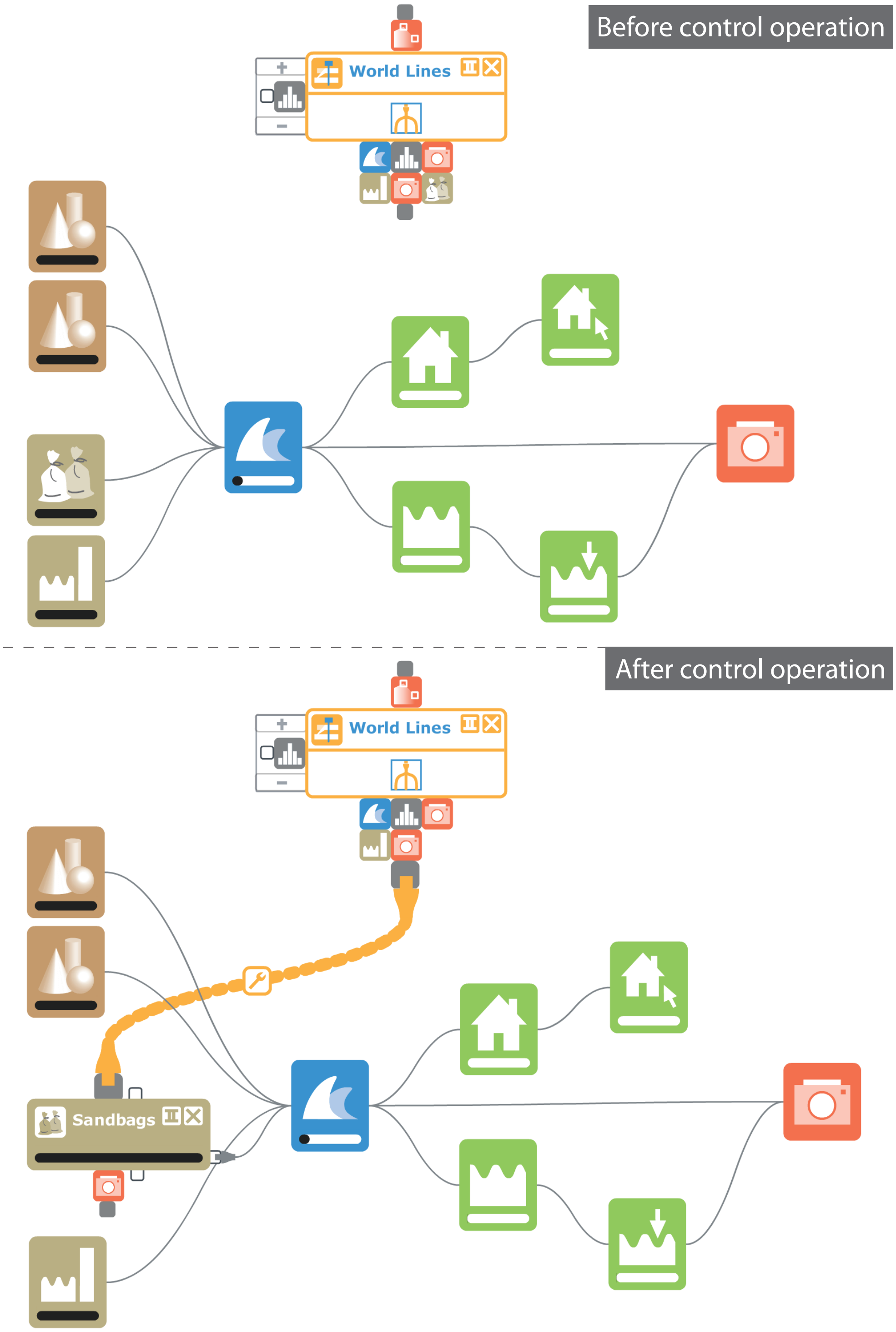


Figure 4.10 – Flow-diagram appearance before and after a control operation. The user modifies sandbag barriers of tracks in the World Lines view.

space of the containing window, we change the window-scroll positions to put the focus on the center of importance $\vec{S}_I$, which is determined as

$$\vec{S}_I = \frac{1}{\sum_n I_k} \cdot \sum_n I_k \cdot \vec{p}_k \tag{4.3}$$

where n is the number of nodes and $\vec{p}_k$ is the screen position of the k-th node. In addition, we alter the thickness of ropes to further emphasize the control-flow. If a meta-connected node is not controlled at all, we apply connection reduction. Figure 4.10 displays an example for a flow diagram before and after a control operation, which involves the configuration of sandbag barriers. With the proposed course of action, the meta-flow supports the user in understanding the internal control operations. Moreover, the importance based display of the meta-flow offers a way to record and visualize work-flow and knowledge of the current user.

4.7.4 Visualization per Wire

In the World Lines view, we can visualize information on a per-time and per-track level. However, this information is restricted to the output of a single connector in the data-flow, or at most, to a derived quantity that describes many outputs. For example the frames can be colored by the number of flooded buildings at the given simulation step. World Lines cannot provide a comprehensive overview of results from several node outputs. We believe that such an overview is best shown in the flow diagram itself. However, the components of the flow diagram can only give restricted information on a per-time and per-track level. Figure 4.11 illustrates how World Lines and the flow diagram can compensate each others shortcomings. In this example, we show progress information in both views: the World Lines view displays which tracks are already simulated, the progress bars of the nodes display which portions of every attribute in the simulation are already computed.

Progress monitoring The presented system utilizes the data-flow processing-algorithms as described by Schindler et al. [114]. This way, we have important frame information about the data elements each node can compute. More specifically, we know for what frames (i.e., time steps and tracks), a node can generate data. If we compare this information to the data that a node has already computed, we can define the progress per node. Figure 4.11 shows how the World Lines view indicates the processed frames by coloring them in yellow. The progress is associated with one node in the data-flow. To give an overview over the system progress, each data-flow node in the diagram is equipped with a progress bar. It is straightforward to display the progress with respect to time. To also give a hint on progress related to tracks, we employ transparency inside the progress bar. This means each track is represented by a semi-transparent rectangular area which are all blended together to give the resulting progress bar. It is noteworthy that the progress bars are updated after every control operation. This way, the user can quickly identify how parameter changes affect the deletion of cached data in a node.

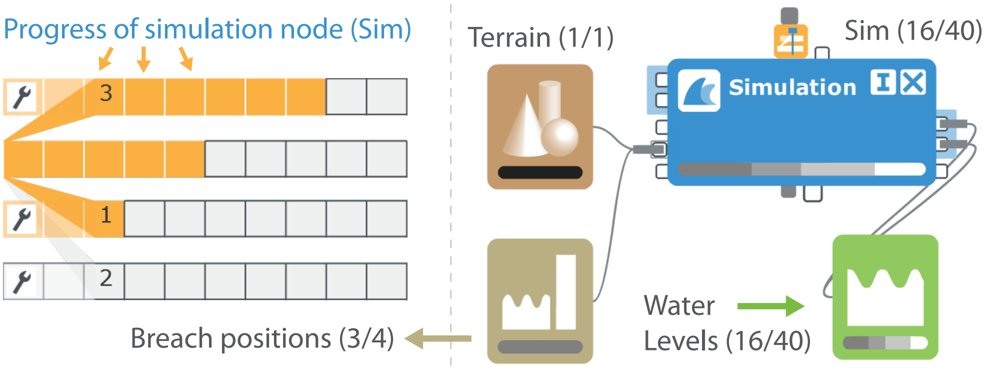


Figure 4.11 – Progress visualization with World Lines (WL) vs. inside the flow diagram. WL comprise 4 tracks for 4 different breach positions. WL show progress on a per-frame level, the flow diagram on a per-node level. The figure labels depict (calculated frames/frames to calculate).

Robustness analysis In a simulation-steering environment, we require the ability to verify whether a solution, i.e., a choice of parameters, is stable with respect to uncertain input. A robust solution is given, when the variation in the output is smaller than the variation in the input. In Section 4.5 we have discussed how to enter a distribution of input values. Via the configurator node, we display the input distribution inside the flow diagram. Now, to evaluate the robustness of a solution, we compute the resulting distributions for all attributes in the simulation which the user selects to be included in the robustness analysis. These output distributions are then visualized inside the flow diagram.

The outputs that have to be included in the robustness analysis, can be selected interactively by selecting output-connectors. This interaction automatically creates and connects a special data-flow node that we term *robustness analyzer*. Figure 4.12 shows robustness analyzers (grey nodes) that are connected to different nodes. Initially, the robustness-analyzer node is only visible through the small icon at the selected output connector, i.e., its connections are reduced (Figure 4.12a). The robustness analyzers are in charge of computing the output distributions. The results of these calculations are visualized in several ways. To get an overview, the thickness of each data-flow wire is changed to reflect the relative standard-deviation value σ of the data which flows through the wire. If this variation is very small, the wire remains reduced and the responsible robustness-analyzer node hidden (Figure 4.12a). The higher the standard deviation, the thicker the wire and the higher the LOD of the associated analyzer node (Figure 4.12b-d). At LOD 1, the analyzer displays a visualization of the distribution of data which flows through the respective wire (Figure 4.12c). This can be done for example using an information visualization such as a histogram or an ensemble-line graph. The latter plots the temporal evolutions of quantities for each track and displays a confidence interval (light orange area). At LOD 2, this

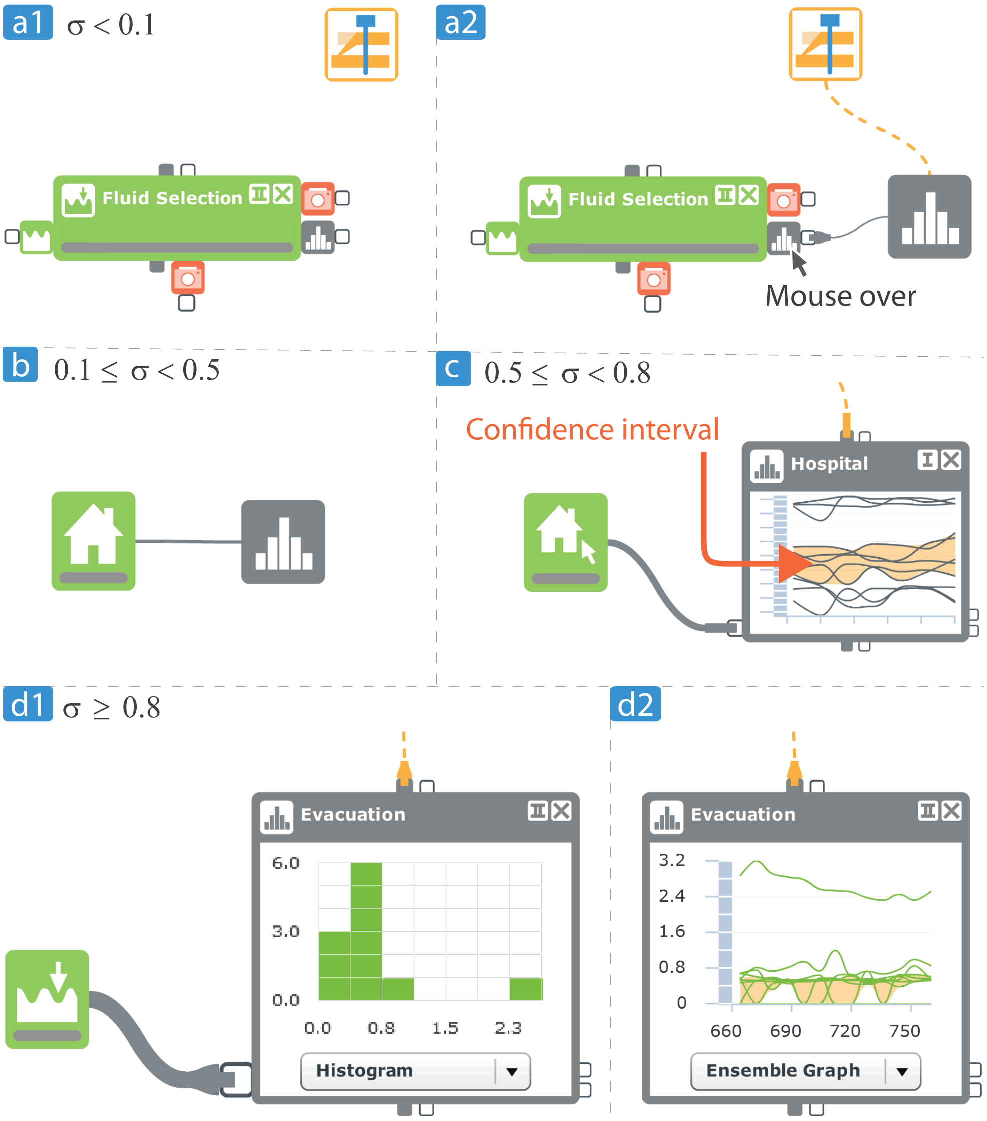


Figure 4.12 – Robustness analyzers (RA) connected to different nodes to visualize the resulting distributions per output. The thickness of the data-wire reflects the relative standard deviation σ of the distribution. (a1) Initially, the RA is hidden. (a2) The user moves the mouse over the icon representation to temporally display the RA. (b) Due to an increased σ-value, the RA is permanently shown. (c) At higher σ-values, the RA shows details about the output distribution using an info-vis view such as a (d1) histogram or an (d2) ensemble-line graph.

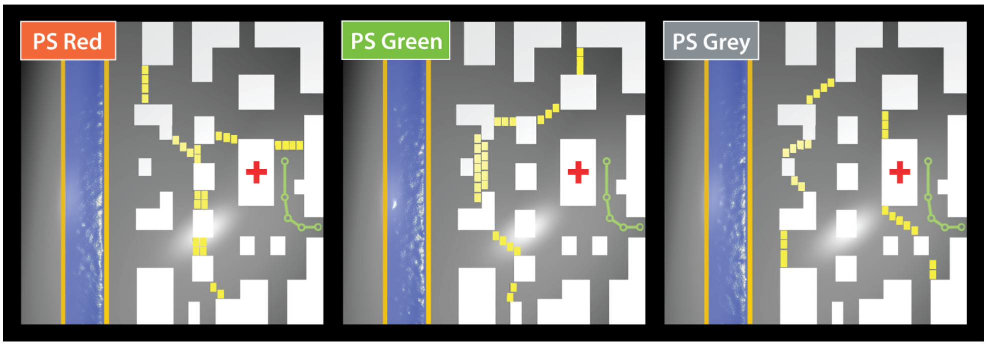


Figure 4.13 – The three competing barrier arrangements which are investigated in the case study. The colored labels refer to the parameter studies in Figure 4.14.

information visualization provides more details (Figure 4.12d). We point out, that this flow-diagram visualization itself is updated during the ensemble simulation.

4.8 Evaluation

In this Section we demonstrate the approach on the flooding scenario introduced in Figure 4.2. For details on the simulation setup, we refer to the World Lines (WL) case study in Section 2.8. For this chapter, we have rearranged the buildings and increased the size of the terrain to $140 \times 140\, m^2$.

4.8.1 Case study tasks

As a preliminary task, the domain expert defines three alternative precautions to protect the neighborhood from flooding. We employ World Lines to create a track for each of these multi-barrier arrangments. Figure 4.13 contains images to show the difference between the arrangments.

Before we can create a parameter study with respect to uncertain levee-breach locations as demanded by task T1 (Figure 4.2, T1), we have to extend the parameter space of WL. This is accomplished by establishing a rope from the related puppet master to the levees node. In the auto-created configurator of the rope, we select the breach-position parameter and add it to the parameter space. Now it is possible to account for uncertain levee-breach locations by creating a parallel parameter study for each of the three tracks respectively. Each track in the parameter study has a duration of 12 *min*, 40 *sec*. The breach-position values are sampled according to a Gaussian distribution with a mean value of 52 *m* and a standard deviation of 12 *m*. The sample density is 11, resulting in a total number of 33 tracks to be simulated. The resulting Gaussian distribution can be inspected in the related configurator node. For the robustness-analysis procedures, we need to be able to inspect the results for one parameter study at a time. In the flow diagram, one parameter study is

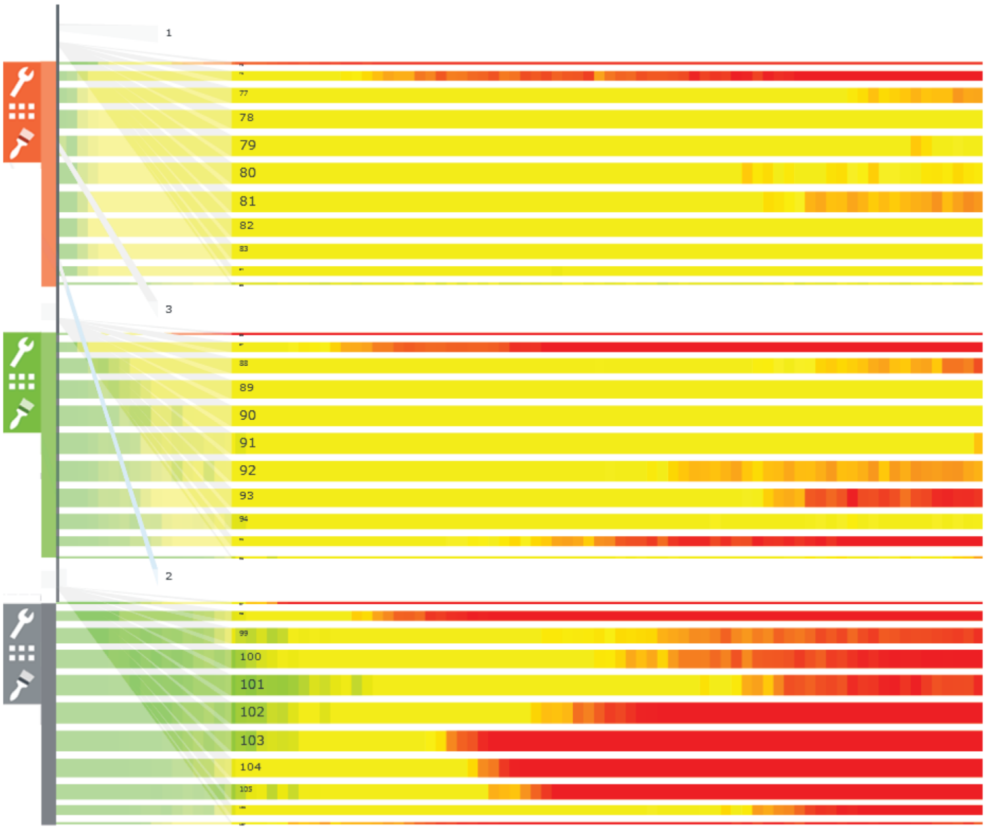


Figure 4.14 – Parameter studies for the robustness analysis of three barrier arrangements to protect a neighborhood from imminent flooding. World Lines (WL) visualize the global damage. The grey parameter study (PS Grey) performs worst. It cannot be read from WL alone whether PS Red or PS Green represents the more robust solution.

considered active at a time, identified by the frames that are currently brushed in the WL view. The three parameter-study cursors are visually distinct with respect to coloring (Figure 4.14). We use this color in the visualizations of the configurator and the robustness analyzers to clearly identify the active parameter study. In the remainder of this section, we refer to a parameter study by the color of its associated cursor and term them PS Grey, PS Red and PS Green respectively.

As a next step, the flow diagram is extended to fulfill the requirements of task T2 (Figure 4.2, T2). We add the required nodes for building-damage estimation and for the calculation of water levels. For both output quantities, we require spatial selection nodes that we meta-connect to the results of the 3D view. This way, we enable inspection of water levels and damage at user-defined points in space. We select the hospital to enable monitoring of its individual damage calculations. The regression model used for building-damage estimation in Euros is based on investigations by

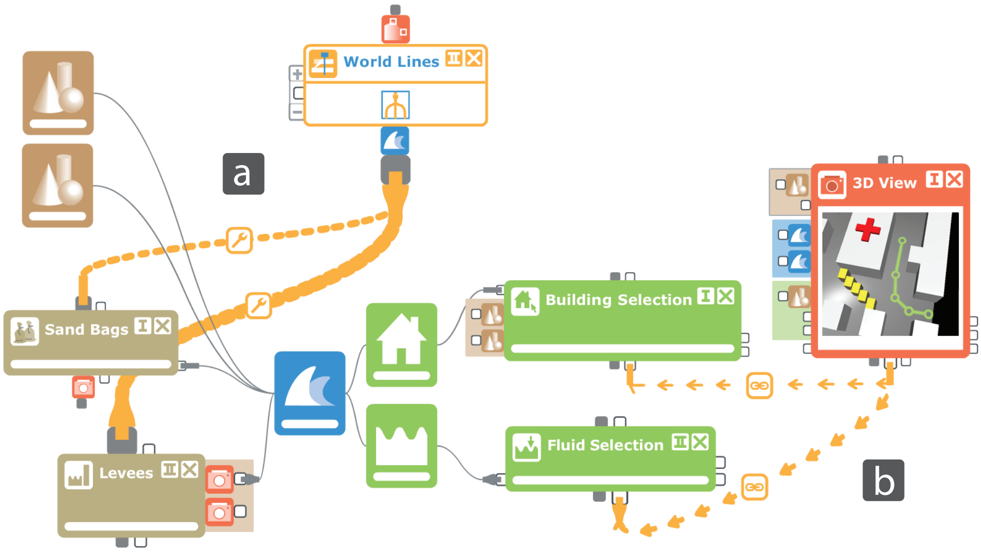


Figure 4.15 – Visualization of control, knowledge and work-flow after the user has setup the system according to the requirements of the case study. (a) The sandbags node and the levees node (ochre) have been steered through World Lines. (b) The user did spatial selections through the 3D View (red) to prioritize the hospital and define the evacuation path.

Thieken et al. [127]. The given formula requires a value of household content which can be entered by the user. We set this value particularly high for the hospital. After this, the domain expert defines the evacuation path through mouse clicks in the 3D view. The related spatial selection node is then capable to output the water levels in meters at the vicinity of the evacuation path. At this point, the flow diagram has changed shape several times to reflect the user work-flow and control operations (Figure 4.15). The emphasis is put on the ropes from the puppet master to the sandbags node and the levees node (Figure 4.15a) which receive a high level-of-detail. The two thick ropes originating from the results connector of the 3D view clearly highlight the importance of spatial selection (Figure 4.15b) . To be able to produce the risk visualizations as required by task T3 (Figure 4.2, T3), we have to create a robustness-analyzer node for each data-flow output of interest. We choose the global damage, the damage to the hospital as well as the water levels along the evacuation path. After this, we are ready for simulation. World Lines provide a scheduling scheme that allows for real-time monitoring of results. During simulation, the risk curves in the flow diagram are regularly updated. This way, we can reason as soon as simulation data is available. The total simulation time is approximately 50% of real-time.

To produce the visualizations in the flow diagram, we brush all frames of a parameter study that lie within the time interval of 620 *sec* to 760 *sec*. For each track in the distribution, the robustness analyzer averages results across this time interval

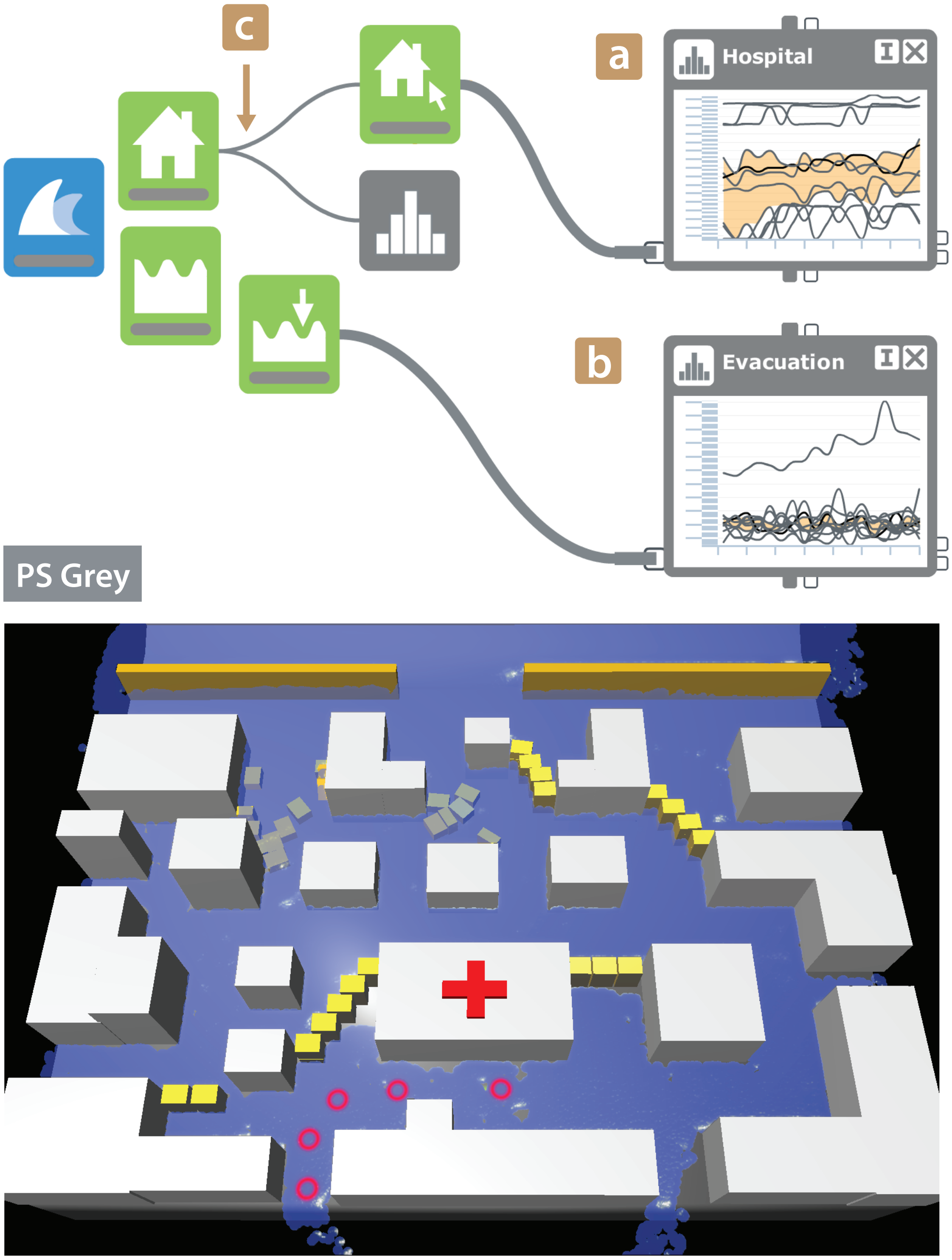


Figure 4.16 – Robustness analysis of PS Grey. The flow diagram shows detailed risk curves for (a) the damage of the hospital and (b) the water levels along the evacuation path. Neither the hospital nor the evacuation path can be considered safe. (c) There are no details about the global damage since in all cases, the damage is high. For these reasons, the barrier arrangement cleary fails with respect to all criteria.

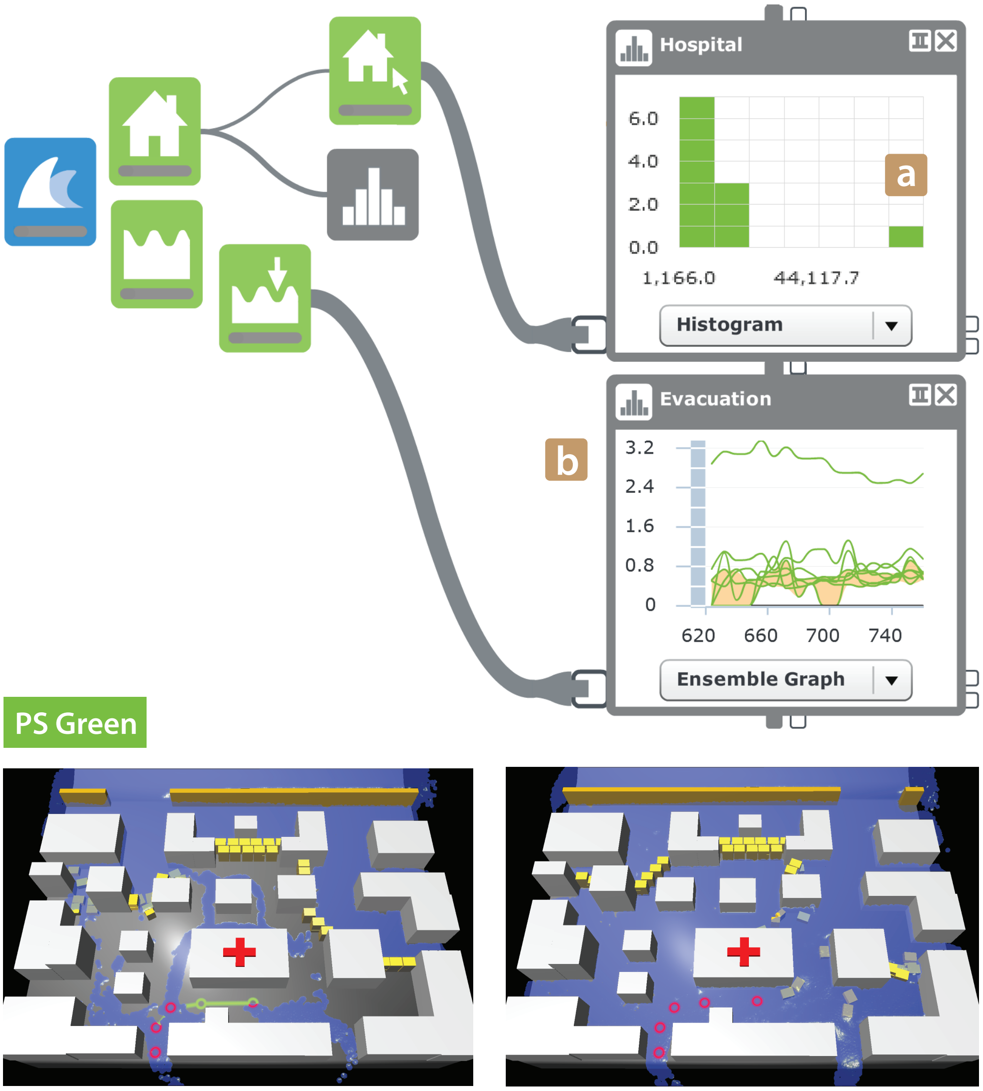


Figure 4.17 – Robustness analysis of PS Green. (a) The multi-barrier system protects the hospital except for cases of unlikely levee-failure. (b) The evacuation path is not secure in all cases. For this reason, the people in the hospital cannot be considered safe.

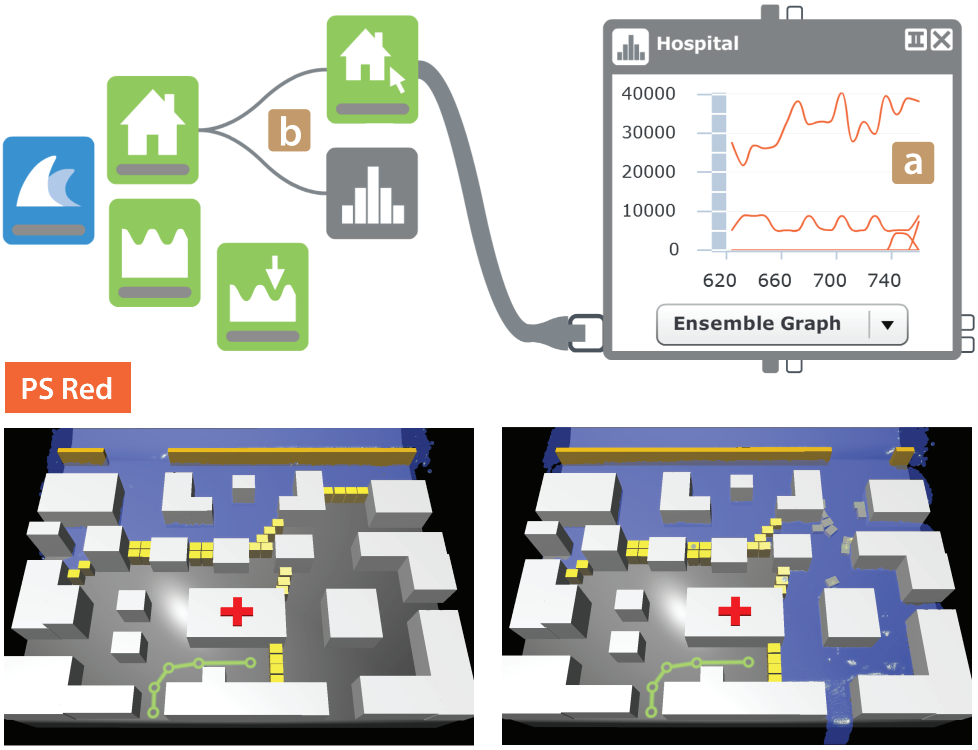


Figure 4.18 – Robustness analysis of PS Red. (a) The hospital is flooded in case of an unlikely levee-failure. (b) There is no detailed information about the evacuation path, since in all cases, the path is water free. This is why the related response measures perform best with respect to damage and safety.

to calculate the standard deviation and the histogram. The ensemble graphs plot the temporal evolutions of quantities for each track and display a confidence interval. The final results are shown in Figures 4.16 (PS Grey), 4.17 (PS Green) and 4.18 (PS Red). PS Grey clearly fails with respect to all requirements. The damage to the hospital is high in most of the tracks (Fig. 4.16a). The evacuation path cannot be considered safe, no matter where the levee-breach takes place (Fig. 4.16b). The standard deviation with respect to the global damage estimation is low (Fig. 4.16c) since in all cases, the neighborhood suffers from severe flooding. For this reason, the related robustness analyzer shows no detailed information. A quick peek into the visualization of the WL view (Fig. 4.14) verifies the found results. Here, frames are colored according to the global damage (green=low, yellow=middle, red=high). Through the WL visualization alone, we cannot say whether the solution tested with PS Red or the one tested with PS Green is more robust. In both cases, the damage to the complete neighborhood is high at breach locations that are farther off the most likely one. The hospital-damage visualizations in both flow diagrams comprise an outlier that puts the hospital in danger (Fig. 4.17a, Fig. 4.18a). However, the precautionary measures tested with PS

Green are incapable of keeping the evacuation path water free (Fig. 4.17b). On the other hand, the flow diagram of PS Red does not show an output distribution for the water levels at the evacuation path at all (Fig. 4.18b). This is due to the fact that the related barrier arrangement protects the evacuation path with respect to each value in the parameter study. Even though we need to choose a response strategy that puts the hospital in danger in case of an unlikely outcome, we are able to keep the people in the hospital safe.

4.8.2 Domain expert feedback

The Nodes on Ropes case study was assessed by an expert with experience in flood forecasting and management systems [19]. It was immediately clear to him that the main strength of Nodes on Ropes is the ability to account for the uncertainty in the forecasts and management implications. Uncertainty can come from uncertain weather predictions and uncertain locations of levee breaches, among other factors. Having an understanding of the uncertainty is a key criterion in flood management decisions as one usually chooses robust flood management designs. These are designs that for a range of possible time evolutions of the inputs will still give a reasonable flood mitigation effect. One attempts to minimize the risk of making a poor decision that may possibly aggravate the adverse effects of a flood on infrastructure and people. Nodes on Ropes allows the user to monitor the evolution of the uncertainty in time with respect to different outputs (such as the accessibility of the evacuation path) which makes it ideally suited for the uncertainty assessment by flood managers. Nodes on Ropes is more flexible than World Lines as one can assign boundary conditions (such as the potential levee breach position) as well as a range of other parameters. This is important for real-world applications as they always differ on a case by case basis. Another advantage over World Lines is that the data and control-flow system has now become transparent. The modular design makes it easy for the flood manager to change parameters related to the inundation in real time, such as the location of a potential dike breach. Also, the flow of uncertainty has become transparent which helps to detect the sources of uncertainties in the decision making. The system could be used for real-time application during a flood event. The real-time updating of the ensemble forecasts in a separate window is an interesting feature that helps keep track of the evolution of the forecasts and their uncertainty.

4.9 Implementation

Nodes on Ropes is part of the Visdom [2] system, comprising World Lines and the presented approach. The separation of meta-flow and data-flow is reflected by the client-server architecture of the system. The meta-flow is completely evaluated on the client, the data-flow on the server. Parameters are represented in XML, enabling a flexible and generic implementation of all meta-flow responsibilities. When a control operation on the client side is initiated (e.g., due to a user interaction), the meta-flow is executed first. During this process, a global synchronizer instance bundles the changed data-flow parameters into one XML request structure. When the meta-flow is complete, this request is sent to the server. The server is in charge of executing the data-flow which involves the compute-intensive parts of the process such as simulation or rendering. We leverage GPU power to perform these processes in real-time. Only images and results are sent back to the client. The results are then available for further processing through the meta-flow connector results. The client is a light-weight application implemented in Actionscript that runs in a browser. This way we are working towards our vision for the future to provide mobile decision support on-site.

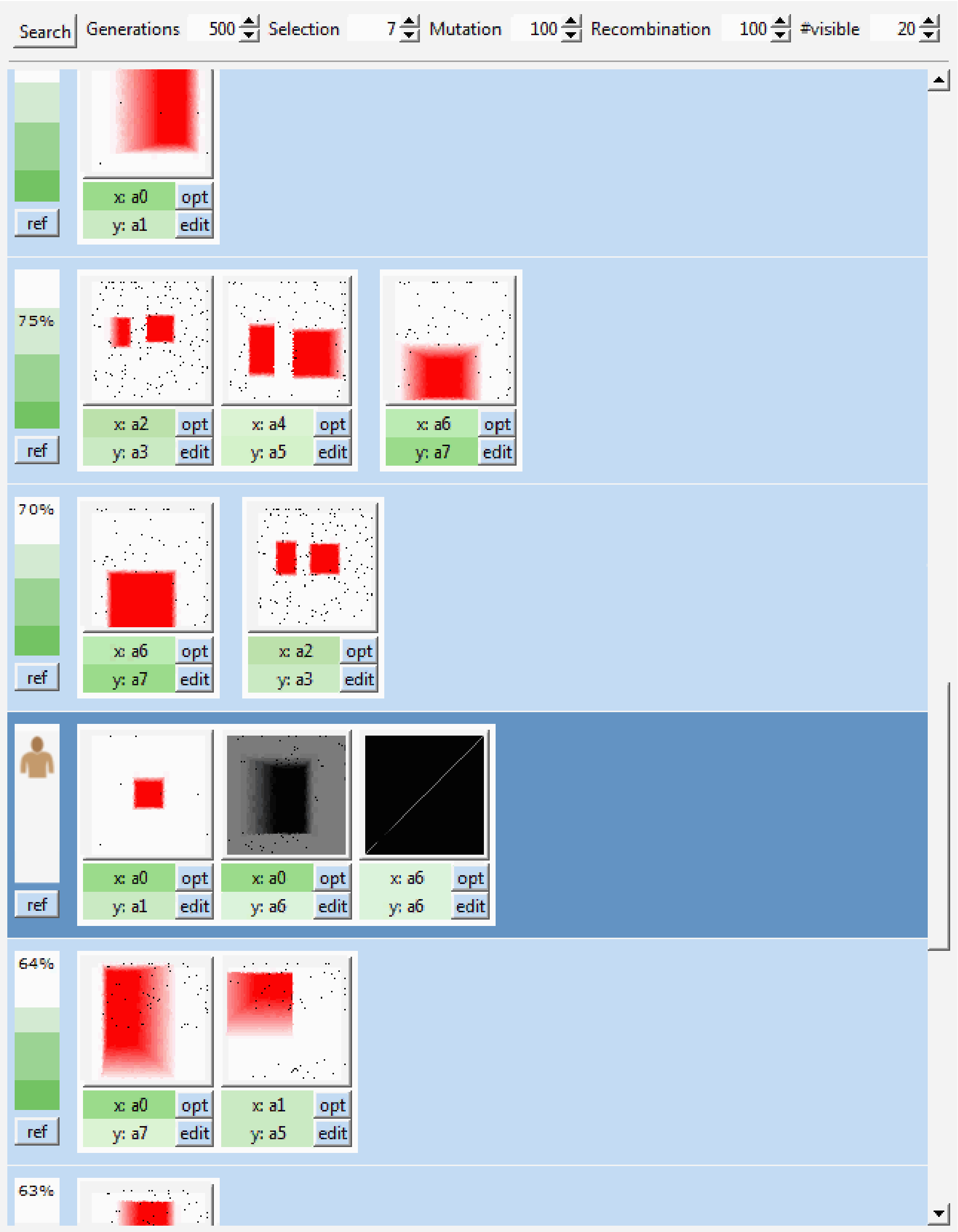
4.10 Discussion

So far simulations have been predominantly concerned with the data-flow. The control-flow was often simple and/or hardwired. Increasingly complex simulations are characterized by an intricate interplay of various heterogeneous components. The elaborate and dynamic nature of the evolving control-flow requires novel visualization and interaction functionality. The goal of this chapter is to tackle three important questions related to interactive simulation steering: How can we handle parameter studies and understand uncertainties? How can we visualize what is going on in a simulation? How can the user modify the simulation setup and extend the explored parameter space? To answer these questions we suggest the extension of the classical data-flow concept by four novel schemes:

First, we propose to add a configurable meta-flow as an abstraction for control- and information-flow between nodes. Second, based on the meta-flow, we suggest interactive parameter studies through World Lines and configurator nodes. One benefit from this approach is the possibility to create customized steering mechanisms which combine visualization, simulation design and parameter space navigation. In this way, the generic data-flow approach can become as user friendly as special purpose turn-key systems. In the general setup we retain the concept of the data-flow in a consistent manner, even though complex node interactions are possible using the meta-flow. Third, we suggest puppet-master nodes which can control other nodes via ropes. The concept of puppet masters is our answer to the rising complexity when interactive steering comes into play. System complexity involves a trade-off between simplicity and power of the system. At one end of the spectrum, using a programming language like C++, one can set up a complete dynamical system and control any detail of its behavior up to the finest degree which the current computing architecture allows. At the other end of the spectrum, there are ready-made tools for simulation and visualization which are easier to handle than a programming language. In this chapter, we suggest a system which minimizes the capabilities required from the user while retaining complex system behavior at the same time. This is possible by allowing the user to perform complex changes via puppet masters. Fourth, we suggest to use the data-flow and meta-flow themselves for visualization. The connectors and wires can display relevant information on the importance of a data element or about statistical distributions in the parameter study which are difficult to show using World Lines alone.

For future work, we consider the automatic analysis of simulation steps as an important tool to control which simulations can be kept running and which should be stopped. For large parameter studies early simulation termination can save a lot of computational resources.

The contributions of this chapter are guided by the basic insight that a more complex framework needs smarter tools to control its features. In the case of the data-flow, this approach translates into the idea to put the nodes on ropes. Extended control also requires new means to support the user in utilizing the new possibilities. We see visualization as an ideal tool to empower the user to take control. In this chapter, we suggest puppet-master nodes for this purpose. There is a wide range of possible future research in the direction of using visualization to steer and control complex applications. The results in this chapter show that a successful visualization approach has to build on the knowledge available in the application domain. By augmenting a common technique such as the data-flow by an elaborate meta-flow, it becomes possible to not only use the data-flow for building the application, but to learn from it as well.



Search Generations 500 Selection 7 Mutation 100 Recombination 100 #visible 20

Figure 5.1 – Interactive visualization of hypotheses that have been found by a genetic search algorithm. A hypothesis is a combination of fuzzy-logic selections to explain features in the simulation data.

A good decision is based on knowledge
and not on numbers.

— *Plato*

CHAPTER

5 Visual Human+Machine Learning

The modularity of the presented system supports the creation of intricate simulation setups for different application domains. In many cases, the simulation system generates large amounts of heterogeneous data the user has to make sense of. In this chapter, we describe a novel method to integrate interactive visual analysis and machine learning to support the insight generation of the user. The suggested approach combines the vast search and processing power of the computer with the superior reasoning and pattern recognition capabilities of the human user. An evolutionary search algorithm is adapted to assist in the fuzzy logic formalization of hypotheses that aim at explaining features inside multivariate, volumetric data.

5.1 Introduction

DUE to the increasing pace and complexity of modern simulation systems, users are often confronted with huge data sets that are difficult to make sense of. Usually, the results contain multivariate sets of physical attributes such as temperature or fluid velocity, often with hidden relations between them. Even for domain experts, it can be tedious to search for explanations for features in the data. In the past, machine learning (ML) algorithms have been adapted to assist in the exploration process. In the context of database systems, we find methods that produce logic rules which are valid for many items in the database. Based on labels inside the data set, these methods can build classifiers from the data. However, these methods require knowledge that is contained in the data itself. That is, the machine learns from the data only, producing hypotheses that are restricted to given relations between data items inside. Therefore these techniques are often not applicable in the field of engineering simulations and computational fluid dynamics (CFD). The simulation simply does not generate labels for the data items and a classification can only be found interactively during visual analysis. A mechanism is thus needed to let engineers add information that is based on their knowledge and experience. Interactive visual analysis (IVA) supports the formation of flexible user selections. The main idea of IVA is to depict various attributes using multiple views and to allow the engineer to interactively select (brush) a subset of the data in these views. All corresponding

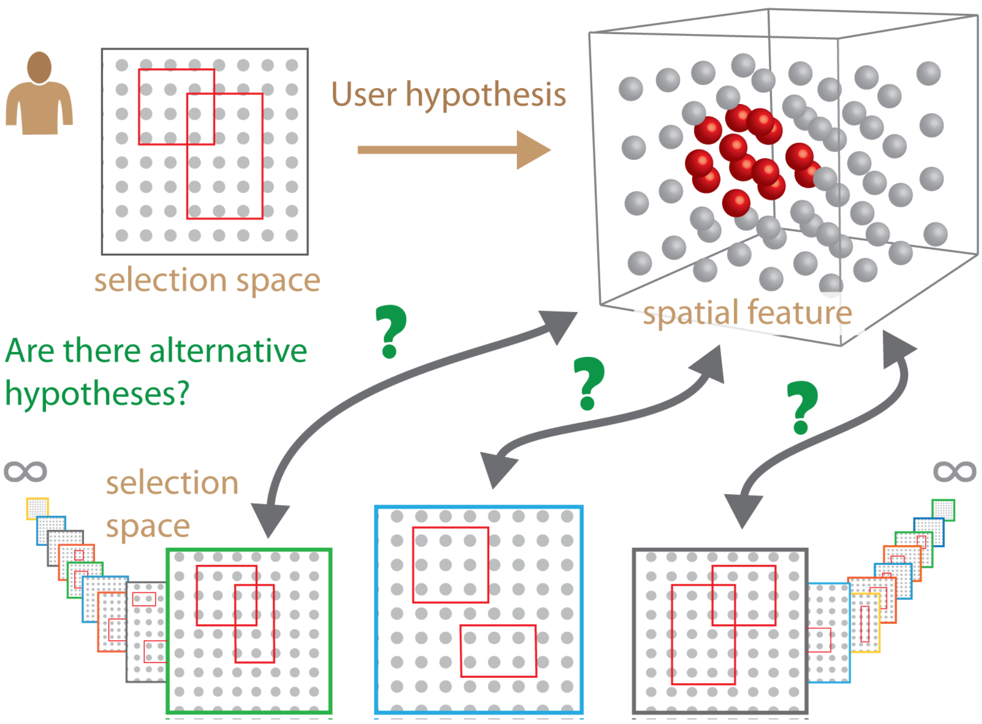


Figure 5.2 – Curse of dimensionality for IVA. In the space of all possible hypotheses it is not clear if the current one is the only candidate. This is a common problem domain experts encounter when they use standard IVA methods (linking and brushing) for making sense of multidimensional simulation results. Secondly, searching for a hypothesis to explain a given spatial feature can be almost impossible for the user.

data items in linked renderings are highlighted as well, providing the analyst with information about the interplay of the attributes involved. As an example, consider a simulation of an industrial prototype where regions of high temperatures are found. In the temperature attribute view, the engineer selects the critical value ranges. During visual inspection in linked views, the user finds that the areas of critical temperature are related to regions of high vorticity values and low velocities. This way, the user has formed the hypothesis that ’critical temperature’ corresponds to ’high vorticity values’ AND ’low fluid velocities’.

When we present the general concepts of IVA to our application partners from research and industry, we are often confronted with scepticism. We can formulate the critical issues in the following questions (see also Figure 5.2):

1. How do I know if my selection is the only one that relates to the spatial feature? Are there alternative explanations?

2. Is there a reasonable hypothesis that corresponds to a spatial feature I am interested in?

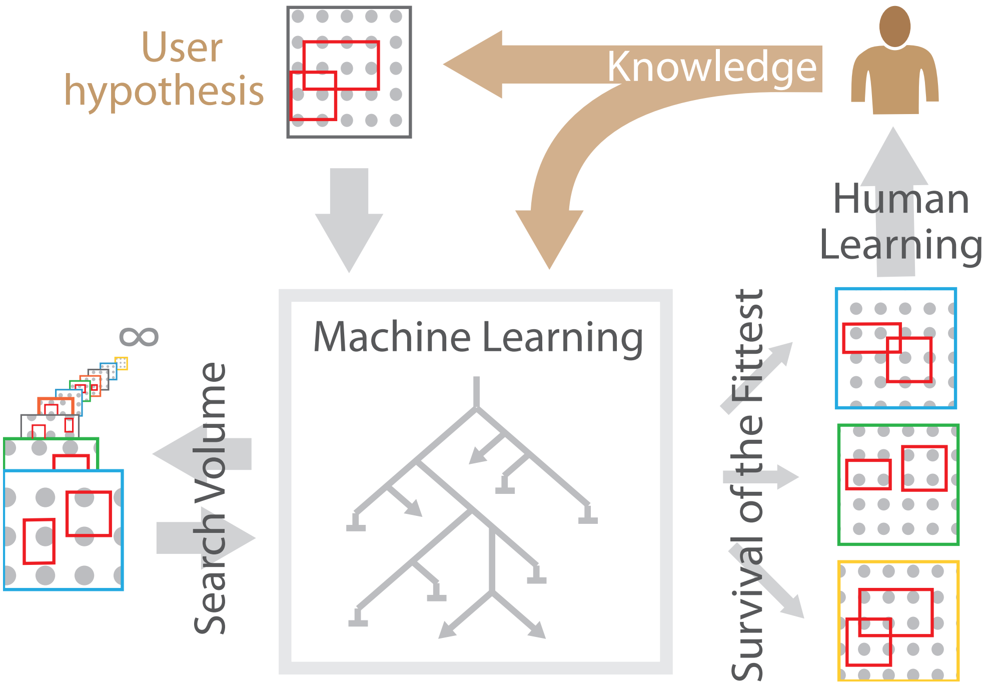


Figure 5.3 – We propose the extension of human learning with machine learning, comprising a genetic algorithm that efficiently searches for the best hypotheses available.

Considering the infinite number of possible selections and logical combinations thereof, it soon becomes clear that this criticism is justified. This is often referred to as the *curse of dimensionality* for IVA which complicates reasoning when many attribute dimensions are available. To overcome these problems we propose a combination of IVA and machine learning. See Figure 5.3 for an illustration. We present a set of fitness criteria which filter the best hypotheses out of the vast search space using a genetic algorithm. While searching, both the machine and the human user are able to learn from and contribute to the current set of hypotheses. We present an interactive framework with adequate visualizations that enable the user to analyze and modify the generated hypotheses before feeding them back into the machine learning algorithm.

5.2 Related Work

Shneiderman [119] suggests to combine information visualization with data mining and gives additional recommendations for future research: novel methods should allow the user to specify what he/she is looking for, results should be easily reportable and the human responsibility should be respected. Keim et al. [68] describe the visual analytics process to be characterized by interaction between data, visualizations, models about the data, and the users in order to discover knowledge. They emphasize the importance of the human in the data analysis process. Seo and Shneiderman [117] present the rank-by-feature framework to find features in high-dimensional data. Keim [69] provides a taxonomy of visual data mining techniques and gives an insightful overview of visual data exploration techniques. He suggests that the generation of hypotheses can be done either by automatic techniques from statistics or with ML or visually. In this chapter we show that these approaches can all work together.

Hao et al. [49] have presented a powerful method to find attributes correlating to a selected feature. They propose queries which allow the user to select hot spots within one attribute and to find correlated attributes. Yang et al. [146] present a system to extract hidden features in a data set interactively. They define distance metrics for the user hypotheses to filter out similar hypotheses using clustering. Müller et al. [95] discuss how principal component analysis can enhance the visualization with a focus on scatter-plot matrices. Doleisch et al. [35] present the feature definition language which we have adopted for the interactive generation of user hypotheses. Kehrer et al. [67] apply this approach to create insight into weather simulation data. Rautek [106] uses a fuzzy rule base for the automatic generation of visualization parameters in volume renderings.

Hertzmann [57] gives an introduction to ML suitable for the visualization researcher. Chen [28] believes that the integration of ML and information visualization is a fruitful approach. In this spirit, Rossi [109] suggests to favor user intervention and control during ML over direct interaction with the visualization. Ward [137] describes how visual clues allow users to monitor dimension reduction and clustering techniques in order to check whether important information has been removed. Jänicke et al. [60] have presented an approach to find relevant regions inside large and complex data sets based on statistical complexity.

Laine [79] discusses three criteria for machine learning to be applicable in an industrial environment. These are supervised operation, robustness and understandability. He stresses that statistical significance is no measure for relation to the problem of interest and presents examples where statistical properties are not sufficient for knowledge generation. Ma [85] discusses seminal work that allows the user to select samples of relevance from which a machine learning algorithm builds a classifier which then segments the rest of the data (see also the work of [130, 131]).

Another direction of research focuses on the improvement, verification or substantiation of data mining results with visualization and interaction. These works show how classifiers can be improved or analyzed by interaction and visualization. Examples are: decision tree construction, support vector machine algorithms and association rules - all improve when supported by an IVA environment [9, 104, 25, 89, 105].

5.3 Visual Learning

In this section we describe different aspects of the visual framework which allow the user to monitor and steer the ML algorithm. We start with a general overview and give an example. Then we discuss several important facts which are relevant when combining human and machine learning. Details are discussed in later sections.

The genetic algorithm can be considered to be learning from the population of available hypotheses which encapsulate the generated information about the data set. When the user or the machine interact with the hypotheses in the population, using editing, optimization or combined operations, the population grows while generating novel information. Based on fitness evaluation or user interaction, the low performing hypotheses or parts thereof are removed which is also an important learning step.

We structure the presented approach according to different types of interaction:

- Genes: When working with selections the user creates novel information which is treated by the algorithms as genes. In the presented framework a single selction is a gene. Handling of genes is discussed in Subsection 5.3.2.
- Hypotheses: Complex hypotheses are composed of selections. A hypothesis can be evaluated for each data item and linked to the volume rendering for inspection. Based on background knowledge and information about other hypotheses in the population, the user can analyze the influence of the selections and recombine and modify the hypotheses before reiterating the heuristic search. The interaction with hypotheses is discussed in Subsection 5.3.3.
- Fitness: To exercise a global type of control over the development of the population, the user is able to specify different parameters such as the weights for the computation of fitness values (see Subsection 5.3.4).
- Optimization: It can be very difficult to maximize the fitness of a hypothesis interactively. In Subsection 5.3.5, we describe a solution to this problem and explain why interaction is still necessary.
- Global Parameters: These parameters steer the ML algorithm and control the rate of mutation, recombination and selection (see Section 5.5).

5.3.1 Considerations on Learning

In this section we present some facts we consider relevant when human and machine learning are to be combined.

Confirmation bias Cognitive science shows that there is a tendency to search for new information in a way that confirms the current hypothesis and to irrationally avoid information and interpretations which contradict prior beliefs [61].

No free lunch From theory we know that there is no universally good search/learning configuration [38]. Changing one parameter may improve the performance for some problems, but usually reduces it for others. Human knowledge is required to specify good parameters and to steer the machine learning algorithm into the right direction. For example, the selection in Figure 5.4c can be locally optimal, but the user understands that disabling parts of the hypothesis makes sense nevertheless.

Strengths and weaknesses The most important strength of the human user is the ability to understand the problem at hand. Therefore the user can often decide when to perform directed search (optimization) and how to limit the search space. Furthermore, based on the visualization, the user is often able to extract information from noisy or incomplete data. The user can give meaning to the features detected in the data. The machine on the other hand has many capabilities the user lacks: its work is inexpensive, fast and tireless. It is able to perform large searches and we know that heuristic search algorithms can perform quite well in large and unstructured search spaces.

No Labels Flow features can be fuzzy for multiple reasons. Often, their boundaries are not sharp. Larger features (such as vortices) are composed of smaller structures which can be features themselves. When exploring simulation data it is not pre-determined which data ranges are important. Only after the engineer has gained understanding of the situation we obtain labels for the data (e.g., 'too hot').

User Requirements The three central requirements are control, robustness and understandability. Control enables the engineer to define what he/she considers interesting and to steer the machine learning process in the right direction. Robustness is the insensitivity to small errors or deviations from assumptions. The third requirement is understandability. This means straightforward visualization of results and their textual representation to facilitate reporting.

5.3.2 Visual Genes

We call the smallest item which carries information a gene. In the present context, we consider a single one-dimensional fuzzy selection a gene. It is a tuple of the type $(fuzzy_{min}, full_{min}, full_{max}, fuzzy_{max}, a_i)$, where the first four values define a fuzzy selection and a_i is the relevant attribute. Detailed information about selections is visible in small scatter-plot icons. See Subsection 5.4.1 for a formal definition. These icons act as toggle buttons to allow deactivation of all selections in one attribute view. For example, in Figure 5.4c, two genes have been switched off. For more specific interactions, the user can enlarge these scatter plots to edit, move, add or remove selections. In the first steps of data exploration, one of the most important tasks is to find the data attributes which are relevant for the problem at hand. Also, when a very large number of attributes is available, the heuristic algorithm will learn faster when only relevant attributes are considered. Therefore, we augment the user interface by a simple color-based clue describing which *attributes* are common in the gene pool. For each attribute we compute a frequency value and integrate this information as the background color into the display of hypotheses. For example, in Figure 5.4b, the important attributes a_0 and a_1 are frequent in the gene pool and receive darker coloring than the others.

5.3.3 Visual Hypotheses

A hypothesis is a fuzzy-logic combination of selections. See Subsection 5.4.1 for a formal definition. Each hypothesis is visualized as a horizontal frame of interactive attribute views. Disjunctivly combined views (clauses) are placed side-by-side without separating space (see Figure 5.4b). Clauses themselves are placed in separate frames (see Figure 5.5e). A conjunction of clauses forms a hypothesis. The user hypothesis is indicated by a small icon on the left and further highlighted by a different background color. This way, the hypotheses in a population can be viewed and compared. To specify novel genes, selections from linked views can be transferred to the user hypothesis. To analyze spatial properties of a given hypothesis, the user can display the resulting fuzzy selection in a linked 3D rendering, where fuzzy membership values can be included into the transfer function, e.g., as a mapping of the degree of selection to opacity or color.

5.3.4 Visual Fitness

The goal is to find hypotheses that lead to features which are similar in physical space and different in selection space. These hypotheses should be as simple as possible. We therefore require measures for feature similarity, complexity and individuality to define the fitness of a hypothesis. See Section 5.4 for detailed explanations. Setting the weights of these measures provides control over what to look for at the current learning step.

High fitness values may result from good performance in one or several of the three components feature similarity, individuality and complexity. Since this is important information these three components are shown as different shades of green in the fitness bar to the left of each hypothesis. For example, in Figure 5.4, the increase in fitness between (b) and (c) is due to the lower complexity (lowest dark green bar), whereas the difference between (c) and (d) is due to the better match between the features.

5.3.5 Visual Optimization

A crucial weakness of optimization algorithms, which lack the support of background knowledge, is the fact that they often cannot discriminate between local and global optima. Therefore they are prone to get stuck at local optima where the fitness of a hypothesis is relatively high but far from a global optimum. The question whether a local optimum is meaningful or not in the current context cannot be answered by the machine. On the other hand, even if including the local neighborhood only, an extensive search often can not be performed by the human user. For these reasons, we add an optimize button to each clause such that the user can decide when and where to search. We have implemented a simple hillclimbing operator which performs a gradient ascent towards the local optimum: The operator moves each boundary element of every selection and calculates the fitness for all changes. From all possible modifications, the one with the maximal improvement is accepted. This is done for decreasing

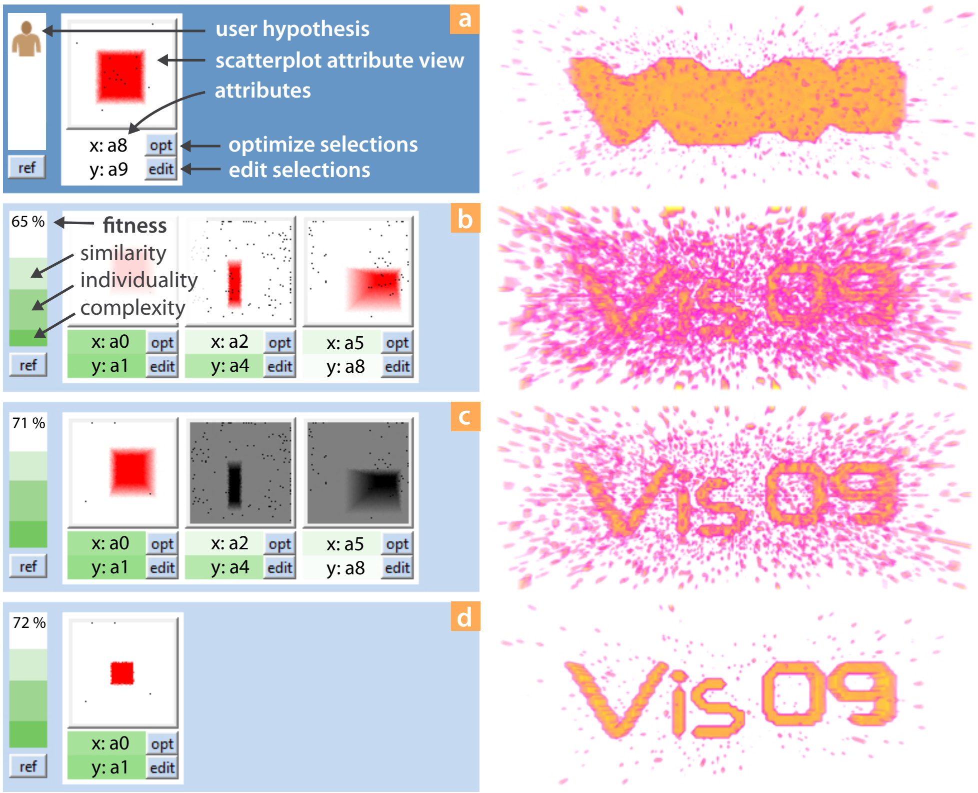


Figure 5.4 – Interaction example (part 1). (a) The user has the idea that attributes a_8 and a_9 describe a feature of interest. The resulting feature is shown in a rendering to the right. (b) The user runs the machine learning algorithm to search for hypotheses. The best resulting hypothesis has 65% fitness. The coloring (dark green) shows that attributes a_0 and a_1 are important in the entire population. (c) The user deactivates genes on other attributes and the fitness increases. (d) Via interactive visual analysis, the user improves the hypothesis.

step widths until the best improvement falls below some user specified margin. For example, consider a one-dimensional non-smooth selection $S = (s_{min}, s_{max})$ contained in a hypothesis. For a given step δs, the fitness of the hypothesis changes when S is replaced by $S_1 = (s_{min} - \delta s, s_{max})$, $S_2 = (s_{min} + \delta s, s_{max})$, $S_3 = (s_{min}, s_{max} - \delta s)$ or $S_4 = (s_{min}, s_{max} + \delta s)$.

5.3.6 The Vis 09 example

This example is based on a synthetic data set containing a well distinguishable feature ('Vis09') which is related to multiple attributes in the data (see Figures 5.4 and 5.5). Over a 3D domain, we define ten scalar attributes. Voxels outside the feature have random attribute values in the $[0, 1]$ interval. The data values inside the feature are

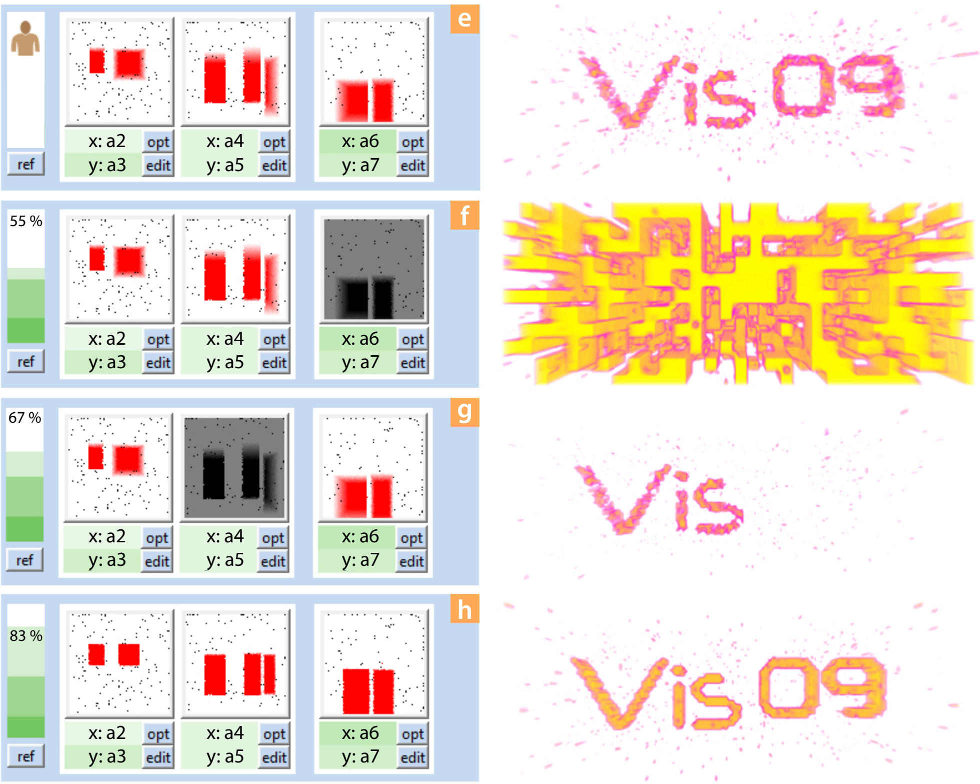


Figure 5.5 – Interaction example (part 2). It is still unclear whether there are alternative hypotheses. (e) A larger search has found an alternative hypothesis. (f) The alternative explanation consists of two clauses combined by conjunction (AND). If one of the two clauses is missing, the feature vanishes. (g) The first clause contains the disjunction (OR) of multiple selections. Attributes a_2, a_3 select the shape "Vis" only. (h) Automatic local optimization via hill climbing finds the optimal selections.

chosen such that the feature is describable by two different hypotheses. The first hypothesis comprises a selection on attributes a_0 and a_1. The second hypothesis is defined among attributes a_2, a_3, a_4, a_5, a_6 and a_7 which have to be combined in a specific way. The 'Vis' part of the feature can be selected by brushing the a_2 vs. a_3 scatter plot in conjunction with a_6 vs. a_7. The '09' part can be selected by brushing the a_4 vs. a_5 scatter plot in conjunction with a_6 vs. a_7. Neither a_2 vs. a_3 nor a_6 vs. a_7 alone result in a useful hypothesis. Therefore the 'Vis09' feature is of the form $((a_2 \wedge a_3) \vee (a_4 \wedge a_5)) \wedge (a_6 \wedge a_7)$. Such a feature is almost impossible to find for the human user, even though it is still rather simple from the viewpoint of the machine. The synthetic data set contains a third feature defined in attributes a_8, a_9 which is easily found and selected interactively. It has been arranged to contain the neighborhood of the 'Vis09' feature (see Figure 5.4a). In Figure 5.4a the user has figured out that attributes a_8 and a_9 might describe a feature of interest. This is a common situation where one attribute describes roughly what an engineer is looking for (such as low pressure is roughly related to vortices). The resulting feature in Figure 5.4a is not very sharp though. In Figure 5.4a, the user runs the ML algorithm to search for alternative hypotheses. The best resulting hypothesis has low fitness but the user is able to see that the machine has found something valuable. In Figure 5.4c the user deactivates genes on other attributes by toggling the corresponding parts of the machine hypothesis, and the fitness increases. In Figure 5.4d the user has found a novel hypothesis. Interactive visual analysis allows the user to relate the novel hypothesis to the reference hypothesis and generate a good explanation even though there is some noise left.

At this point, the user has gained some information. However, it is still unclear if there are alternative hypotheses and so the user triggers another search. In Figure 5.5e the algorithm reveals an alternative hypothesis. In Figure 5.5f the alternative consists of two clauses combined by conjunction ($\wedge$). If one of the two clauses is missing, the feature vanishes as already discussed above. Figure 5.5g: Based on the interactive framework, the user analyzes the impact of the different parts of the hypothesis and is able to understand the relation of the attributes to the 'Vis' and '09' parts of the feature. Figure 5.5h: Finally, it is desirable to reduce the amount of noise in the selection. However, this is difficult to do by hand. A simple optimization scheme helps the user to find the optimal selections in relation to the feature of interest.

5.4 Hypotheses

In this section we describe the formalization of hypotheses and measures to quantify their properties.

5.4.1 Definitions

A sample point in the n-dimensional data set is given by the attribute vector $\mathbf{a} = (a_1, \ldots, a_n)$. Each selection $S(\mathbf{a})$ is a mapping from $\mathbb{R}^n$ to fuzzy membership values in the range $[0,1]$. A data element $\mathbf{a}$ is assigned a value of $S(\mathbf{a}) = 1$ if it is fully selected. A view $V(\mathbf{a})$ is a mapping that presents a subspace of the attributes to the user and can have specific interactions to select data points interactively. The suggested method could potentially combine multiple data selection metaphors, but we focus on smooth brushing on two-dimensional scatter plots. Without loss of generality, we will restrict these to rectangular areas to simplify the discussion. A single smooth rectangular selection $S(\mathbf{a})$ can be defined by two rectangular areas $R_1 \subset R_2$ on the drawing area of the view (see transparent regions in the layers of Figure 5.6). Each data element is then assigned a fuzzy membership value of 1 if it was mapped onto a position $p \in R_1$. A data element has fuzzy membership value of zero if it is mapped outside of R_2 and is assigned fractional membership value inside the region $R_2 \setminus R_1$. We define this fractional membership value as $d(p, \partial R_2)/(d(p, \partial R_1) + d(p, \partial R_2))$ with $d(x,R) := \min\{|x-r|, r \in R\}$ where ∂ is the boundary operator. This definition has the benefit of being directly extensible to non-rectangular selections. Users may

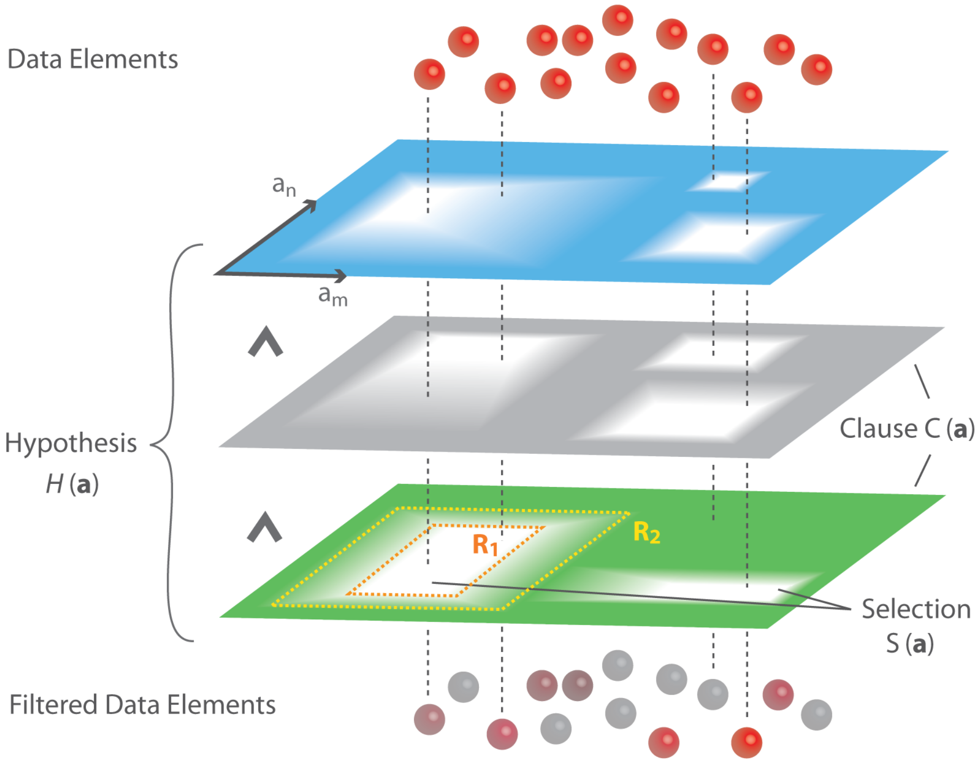


Figure 5.6 – Fuzzy logic hypotheses. The hypothesis is visualized as a stack of filter layers (clauses) with partially-transparent regions (fuzzy selections). The overlapping layers have a selective filter effect on the data items (spheres) above. While fully selected in the beginning (red), some of the spheres are being re-colored when falling through the stack. Less selected items receive higher desaturation, becoming grey if deselected.

specify multiple selections within a single view which are then combined as fuzzy disjunctions. For more complex hypotheses, different views can be connected to each other via fuzzy logic operators. The resulting terms are Boolean expressions which are transferred into conjunctive normal form. This modification is necessary, as we bias towards simple, efficient and interpretable terms for both the genetic algorithm and interactive visualizations. With this transformation at hand, we formally define the hypothesis $\hat{H}(\mathbf{a})$ as a conjunction of N clauses $C_i(\mathbf{a})$

$$\hat{H}(\mathbf{a}) := \bigwedge_{i=1}^{N} \underbrace{\bigvee_{j=1}^{M_i} S_{ij}(\mathbf{a})}_{clause\,C_i} \tag{5.1}$$

where each clause is a disjunction of M_i selections S_{ij}. The straightforward way to evaluate Equation 5.1 is to translate the operators to the point-wise operations

’minimum’ and ’maximum’. This leads to the expression

$$H(\mathbf{a}) = \min_{i=1}^{N}(\max_{j=1}^{M_i} S_{ij}(\mathbf{a})). \tag{5.2}$$

Within a clause, a data point is selected according to the largest value in all contained selections. In the final hypothesis, each data point receives the minimum selection value of all clauses. Figure 5.6 illustrates the quantities defined so far. A clause can be regarded as a layer with transparent regions that correspond to the disjunctively combined selections. To visualize the selective effect of the clause, we imagine the data items as spheres that fall through such a layer. In analogy to a filter process, each sphere is then re-colored depending on the location it fell through. The color intensity corresponds to the fuzzy selection value. In this picture, a complete hypothesis is modeled as a stack of several overlapping layers. The search for an alternative hypothesis is thus equivalent to the search for a different stack of selective data filters. To qualify the resulting hypothesis we require new quantities as defined in the next sections.

5.4.2 Feature Similarity

The feature similarity $\mathscr{F}(H_1,H_2)$ between two hypotheses H_1 and H_2 measures the point-wise distance between the feature membership values resulting from Equation 5.2. We distinguish between selected data items (fuzzy value > 0) denoted by $H_i^+ := \{\mathbf{a}|H(a) > 0\}$ and deselected items (fuzzy value$= 0$) that are given by $H_i^- := \{\mathbf{a}|H(a) = 0\}$ with $i = 1,2$, and compare these two sets independently. This is necessary since the number of points belonging to the feature (H_i^+) can be a lot smaller than the number of data points outside the feature (H_i^-). Using this approach we can compute the directed similarity $\mathscr{F}_d$ between H_1 and H_2. The first term of $\mathscr{F}_d$ in Equation 5.3 accumulates the distance of all selected elements, whereas the second term accounts for the non-selected parts.

$$\mathscr{F}_d(H_i,H_j) = \frac{\sum_{H_i^+} 1-|H_i-H_j|}{|H_i^+|} \cdot \frac{\sum_{H_i^-} 1-|H_i-H_j|}{|H_i^-|}$$

$$\mathscr{F}(H_1,H_2) = \min(\mathscr{F}_d(H_1,H_2),\mathscr{F}_d(H_2,H_1)) \tag{5.3}$$

5.4.3 Complexity

The complexity of a hypothesis is a measure for the number of involved views v and the number of selections s in the hypothesis. More complex hypotheses are conceptually harder to understand. We thus need an expression, that allows for choosing upper limits for the number of views and selections, beyond which the complexity of a hypothesis strongly increases. Therefore, the complexity $\mathscr{C}(H)$ has the negative Gaussian shape

$$\mathscr{C}(H) := 1 - e^{-\left(\frac{s}{v\cdot\sigma_1}\right)^2 - \left(\frac{v}{\sigma_2}\right)^2} \tag{5.4}$$

where σ_1 and σ_2 control the number of selections (s) and views (v), respectively. We set $\sigma_1 = 10$ and $\sigma_2 = 5$ for all evaluations in this chapter.

5.4.4 Individuality

The individuality measures how unique a hypothesis is in the set of all hypotheses in the current population. For this purpose, we require a function that compares two hypotheses according to their resemblance

$$\mathscr{R}(H_1,H_2) := \frac{1}{\max(\#H_1,\#H_2)} \cdot \sum_{i=1}^{\#H_1} \max_{j=1}^{\#H_2}(\mathscr{R}_c(C_i,C_j)) \tag{5.5}$$

where $\#H_i$, $i = 1,2$ refers to the number of clauses in a hypothesis. This definition is motivated by the 'stack of filters' understanding of a hypothesis as shown in Figure 5.6. From this viewpoint the resemblance of two hypotheses can be understood as the average resemblance of filters, where we always compare the two most resembling filters (clauses). The benefit of this approach is that two hypotheses can have high resemblance if only a subset of their genes (selections) is similar. The resemblance $\mathscr{R}_c(C_1,C_2)$ between two clauses is

$$\mathscr{R}_c(C_1,C_2) := \frac{1}{\max(\#C_1,\#C_2)} \cdot \sum_{i=1}^{\#C_1} \max_{j=1}^{\#C_2}(\mathscr{R}_s(S_i,S_j)) \tag{5.6}$$

where $\#C_i$, $i = 1,2$, refers to the number of selections in a clause. A selection S as defined in Subsection 5.4.1 can be considered as a 2D fuzzy set with cardinality $|S| := \int_{\mathbb{R}\times\mathbb{R}} S(x)dx$. The resemblance of two selections $\mathscr{R}_s(S_1,S_2)$ is now defined as the cardinality of fuzzy set intersection of S_1 and S_2 divided by the cardinality of their fuzzy set union.

$$\mathscr{R}_s(S_1,S_2) := \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}. \tag{5.7}$$

This can be understood as dividing the area of the intersection of S_1 and S_2 by the area of their union.

In order to allow the heuristic algorithm to create populations of sufficiently diverse hypotheses the following individuality ranking is of crucial importance. When there are multiple closely resembling hypotheses we want to rank them in a way that the one with the locally highest feature similarity also has highest individuality. Using Equations 5.5-5.7 we first compute the local neighborhood of a given hypothesis H which is defined as the list of hypotheses that are resembling H most. A hypothesis H' is contained in the local neighborhood of H if it fulfills the threshold condition $\mathscr{R}(H,H') \leq \mathscr{R}_{limit}$. This threshold is set to 0.6 in our evaluations. To calculate the individuality, all hypotheses in the local neighborhood are ranked by their feature similarity $\mathscr{F}$ with respect to the user's hypothesis. If the feature similarity of two hypotheses is equal, the less complex hypothesis receives higher rank. Figure 5.7 illustrates the local individuality ranking for a given hypothesis (blue podium). The resulting rank r is used to define the individuality $\mathscr{I}(H)$ as follows

$$\mathscr{I}(H) := (1-\mathscr{R}(H,H_{user})) \cdot (1-\varepsilon)^r \tag{5.8}$$

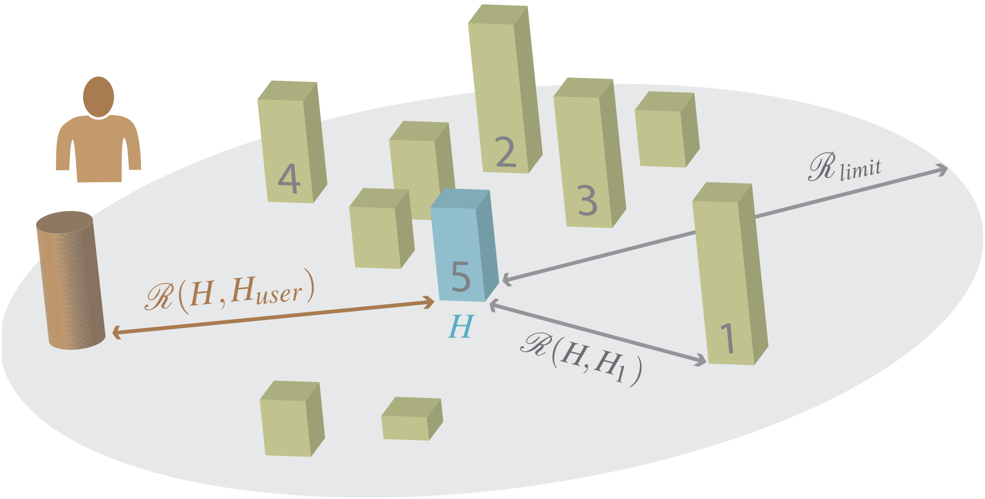


Figure 5.7 – Local ranking. For evaluating the rank of a hypothesis (grey number on the podiums), the current hypothesis (blue) is competing with its nearest neighbors in resemblance space. The higher its feature similarity (higher podium), the higher its rank. The individuality is then computed as the resemblance to the user's hypothesis (brown), weighted with respect to the evaluated rank.

where the resemblance to the user hypothesis H_{user} is given special attention. The parameter ε controls the influence of the ranking value and is set to 0.25 in all our tests.

5.5 Machine Learning Algorithm

This section discusses our adaptation of a genetic search algorithm to find alternative hypotheses. A genetic algorithm consists of three basic steps: *selection*, *mutation* and *recombination*. In the selection process, the fittest individuals are chosen to be included in the next population without modification. A mutation step changes an individual randomly, thus widening the diversity of the population. Finally, in the reproduction step, individuals of a generation are recombined to produce members of the next generation.

The population size in our experiments contains 100 hypotheses. Ten are transferred via selection, 45 via mutation and 45 via reproduction. Setting these values can be important when the user wants different search strategies at different points in the analysis process. In the beginning, a configuration of high mutation rate, low selection and large population size might be good. Later, when several good candidate hypotheses are already present, low mutation rates and a very high recombination level can be appropriate to find interesting larger hypotheses consisting of the genes already present in the population.

5.5.1 Fitness and Selection

The fitness of an individual is defined in relation to the user hypothesis H_{user} and all other hypotheses of the population. We define the fitness of an individual H as

$$Fitness(H) = w_f \cdot \mathscr{F}(H, H_{user}) + w_i \cdot \mathscr{I}(H) - w_c \cdot \mathscr{C}(H). \tag{5.9}$$

where w_f, w_i and w_c weight the influence of feature similarity, individuality and complexity, respectively. For the evaluations in this chapter, we set $w_f = 0.5$, $w_i = 0.3$ and $w_c = 0.2$. If the user is not completely sure about the shape and position of a given feature, there are two ways to take this into account: First, by a lower weight of the feature similarity component. Second, by reducing the fuzzy selection values in the reference hypothesis. We have found that overly complex hypotheses can require a large portion of the computational resources. We thus remove hypotheses of very high complexity scores without a complete fitness evaluation.

When an individual of the population has to be selected, we employ an acceptance-rejection method. This way, we achieve a distribution of random selections that is in accordance to the fitness values. A random integer i in the range from one to the size of the population and a random real value from the $f = [0,1]$ interval are generated. If the i-th individual in the population has fitness higher than f, this individual is selected. Otherwise the process is repeated.

5.5.2 Mutation

In standard definitions of genetic algorithms, mutations change bits of the string which defines the individual. We use a more semantic mutation operator to change the elements of a hypothesis. In the mutation step one individual is chosen as discussed in the previous section. A selection can mutate in four different ways: edit, move, crawl and replace. A selection is defined by a number of vertices (eight in the case of rectangles). The edit-mutation either selects and changes one of the float values which describe the position of one vertex or it changes the data attribute a_i randomly. The move-mutation changes the position of all vertices of a selection in the same random direction. The crawl-mutation edits the fuzzy boundaries with higher probability than the sharp boundaries. The replace-mutation generates a novel random selection. We use equal probabilities for all four mutation methods.

The Boolean operators in a hypothesis mutate by reduction and flip. The reduction-mutation randomly removes one of the arguments of the selected operator. For example $\vee(s_1, s_2, s_3)$ could be reduced to $\vee(s_2, s_3)$. The flip operator changes a disjunction to a conjunction and vice versa. We do not use an insertion operation.

5.5.3 Recombination

In the reproduction step, two individuals are selected as discussed in Section 5.5.1 in order to be combined for generating new offspring. We suggest to internally represent each hypothesis as an operator tree of selections according to Equation 5.2. For both selected individuals, a node in the operator tree is chosen and two offspring individuals are created by swapping the subtrees attached to the selected nodes. Each node has the same probability to get selected. This way, the number of possible recombinations of the two individuals is not artificially limited. This approach benefits from a continuous evolution of hypotheses, which is in the spirit of interactive control over the evolutionary process.

5.6 Evaluation

In this section we describe evaluations of the visual human+machine learning approach.

5.6.1 Measurements

This section analyzes the number of generations and the time necessary to find a sharply defined feature inside synthetic volume data. We have set up a spherical feature in attributes a_0 and a_1 of a ten-dimensional data set that consists of 32^3 voxels. More exactly, for voxels with $v_x^2 + v_y^2 + v_z^2 < 1$ the attributes a_0, a_1 contain random values in the $[0, 0.1]$ interval, all other voxels contain random values in the $[0, 1]$ interval, but either a_0 or a_1 has to be larger than 0.1. The alternative hypothesis to be found by the algorithm has been defined in two other attributes a_2 and a_3 in the same way. This (as well as the 'Vis09' example) can be considered as a worst case scenario since a slight deviation from the correct selection in a_2, a_3 performs poorly, whereas in practice, features in CFD data tend to be smoothly defined such that approximately correct hypotheses already perform well.

To study the robustness of the search algorithm, we have applied different noise levels to the attribute values. For noise level w we generate a random value r between -1 and 1 and modify the attribute value $a \leftarrow a + 0.01 \cdot w \cdot r$. Figure 5.8 shows the number of generations and time needed until a feature similarity of 0.8 has been reached. The algorithm has been rerun 100 times per setting to evaluate the error bars. While performing well at low noise levels and producing results after several minutes, the algorithm becomes less reliable the higher the noise level. This behavior can be expected, as the feature becomes less pronounced the more voxels contain random attribute values.

5.6.2 Case Study: Analysis of a Cooling Jacket

In the following subsection, we discuss an application of the suggested technique on a real world data set. A detailed description of the cooling jacket can be found in previous work [80, 52, 23]. It is sufficient to know that the most important attribute of this data set is the fluid temperature. Many simulation variables (19 attributes) are related to the fluid temperature. Also, there are additional important derived attributes such as the λ_2 vortex detector or velocity, pressure and density gradients. In sum, there

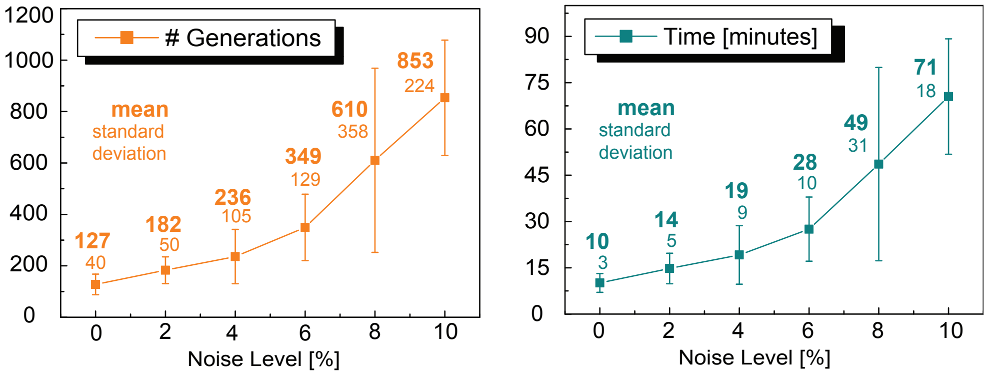


Figure 5.8 – Performance analysis: The search algorithm has been tested on synthetic data at different noise levels. The left graph shows the number of generations needed until a feature similarity of 0.8 has been reached. The right plot shows the related timing results.

are 35 possible attributes to consider inside a data set consisting of approximately 1.5 million 3D cells.

We start the analysis with the assumption that high temperatures ranging from 363.0 K to 372.151 K with a fuzzy border of 1 K correspond to a feature (see Figure 5.9 and 5.10). A quick initial search with 20 generations and a population of 50 individuals reveals a connection between high temperature values and a certain range of fluid densities. We use hill climbing to find the optimal density interval d and reach an almost exact (97%) feature similarity to the original hypothesis as seen in Figure 5.9b. Based on this information, the engineer might guess that the high temperatures are related to the parts of the fluid where high pressure hinders the hot parts of the fluid to expand. Therefore we extend our hypothesis to ”high temperature and density in d”, i.e., high saturation pressure (Figure 5.9c). After a few minutes of interactive analysis it is clear that it is not possible to find related attribute ranges interactively.

We continue with a larger search and after 300 generations the machine has generated a large number of suggestions, but none of them has very high feature similarity. The visualization of global attribute frequencies shows that selections on certain attributes are very common in the population though, so we can still assume that there are hidden features to be found. We focus on the attributes related to the successful genes and deactivate all other genes from the best selections. Now, a visual comparison of the volume rendered features related to the three best hypotheses explains the non-perfect feature similarity: the generated hypotheses are related to three different but partially overlapping subsets of the fluid. See also Figure 5.10. Hypothesis A ”density is medium and pressure is high” is actually the explanation the engineer was looking for, but it was impossible to verify this hypothesis as the only one. There are two other possible explanations B, C for different hot parts in the fluid. There are also sections where a lot of heat is transferred from the wall,

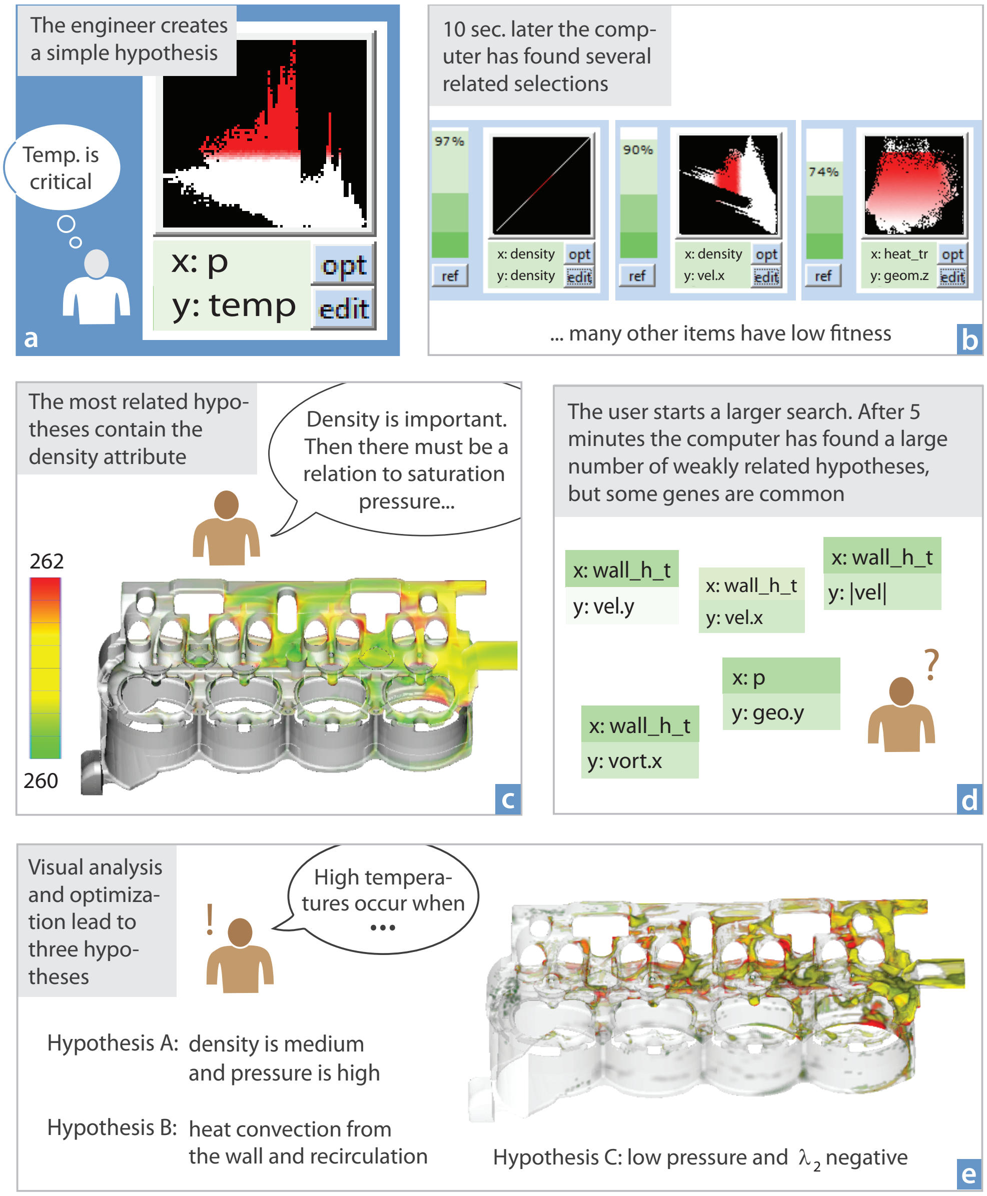

The engineer creates a simple hypothesis
Temp. is critical
x: p
y: temp
opt
edit
a
10 sec. later the computer has found several related selections
97%
ref
x: density
y: density
opt
edit
90%
ref
x: density
y: vel.x
opt
edit
74%
ref
x: heat_tr
y: geom.z
opt
edit
... many other items have low fitness
b
The most related hypotheses contain the density attribute
Density is important. Then there must be a relation to saturation pressure...
262
260
c
The user starts a larger search. After 5 minutes the computer has found a large number of weakly related hypotheses, but some genes are common
x: wall_h_t
y: vel.y
x: wall_h_t
y: vel.x
x: wall_h_t
y: |vel|
x: p
y: geo.y
x: wall_h_t
y: vort.x
?
d
Visual analysis and optimization lead to three hypotheses
!
High temperatures occur when ...
Hypothesis A: density is medium and pressure is high
Hypothesis B: heat convection from the wall and recirculation
Hypothesis C: low pressure and λ₂ negative
e

Figure 5.9 – Case study story: The engineer knows that there are critical temperatures inside the data set. An initial search shows that density and temperature are highly related since vaporized coolant has low density. The user adjusts the hypothesis to *hot AND vapor* and starts a larger search, but there is no single best solution. The user analyzes the attribute ranges with the highest fitness and finds multiple hypotheses which can explain the hot temperatures at different locations in the data set. Hypothesis A: where pressure is high, the vapor could be compressed too much. Hypothesis B: where recirculation areas appear, the fluid transport could be hindered. Hypothesis C: vortex regions could trap hot fluid inside.

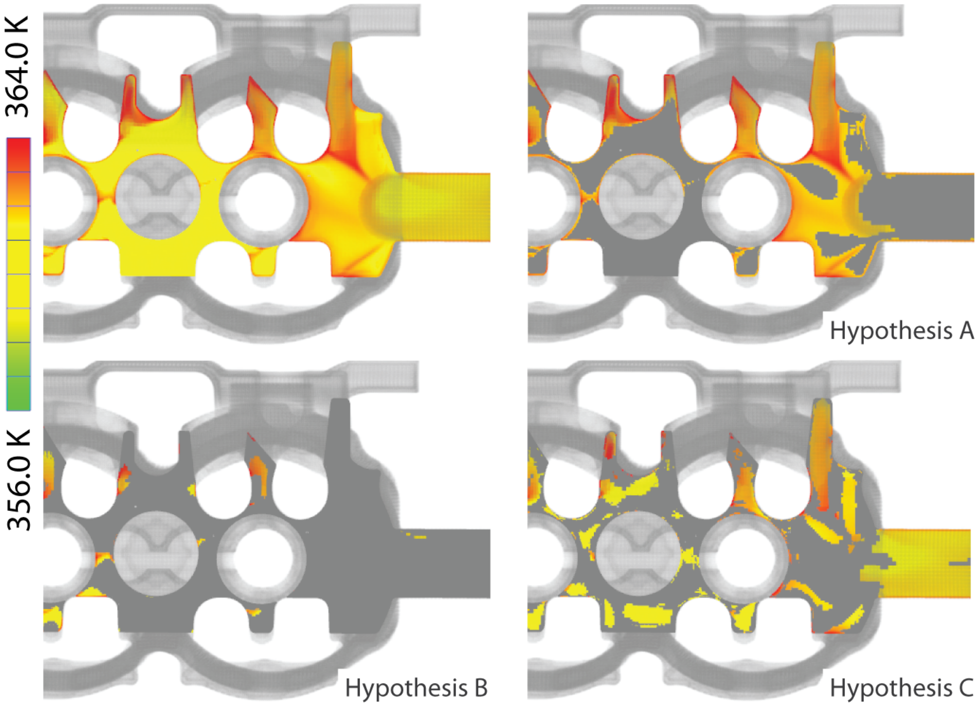


Figure 5.10 – Feature comparison: (top left) Temperature distribution in a plane. (Hypothesis A-C) Selected areas correspond to the findings in Figure 5.9. We can see that the different variables explain different but partially overlapping subsets of the fluid.

but the material transport is jammed due to recirculating flow. Also, there are parts where vortices (”low pressure and $\lambda_2 < 0$”) are trapping hot parts of the fluid. This combination of hypotheses is a novel result, which none of the previous case studies was able to generate since the interplay of six attributes in the data set (temperature, density, pressure, wall heat convection, inverse flow and λ_2) is almost impossible to find using interactive analysis alone.

5.7 Implementation

The presented approach becomes possible since the compute intensive parts of the fitness evaluation are implemented on the GPU. The operations which require the traversal of all data elements, i.e., the evaluation of hypotheses (Equation 5.2) and feature similarity (Equation 5.3) can be implemented in Nvidia’s Cuda [99]. Hypothesis evaluation is a straightforward data traversal to evaluate a function consisting of min and max operations. For the required data handling the ’histogram’ example, available with Nvidia’s Cuda SDK [99], is a good starting point. Using the Cuda API it is possible to stream chunks of data to the GPU such that the size of GPU memory is not a limiting factor.

The computation of feature similarity is a simple modification of the 'reduction' example available in the Cuda SDK, where the computation of the sum is replaced by the term $\mathscr{F}_d$ from Equation 5.3. The computation of these values and the corresponding data handling require about 98% of the computation time. The genetic algorithm itself requires less than 1% of computation time and is implemented using the C++ standard template library. An operator tree based representation of hypotheses simplifies the otherwise complicated operations that are needed by the evolutionary algorithm. Especially mutation, recombination and the translation of hypotheses to conjunctive normal form is simplified by this approach.

5.8 Discussion

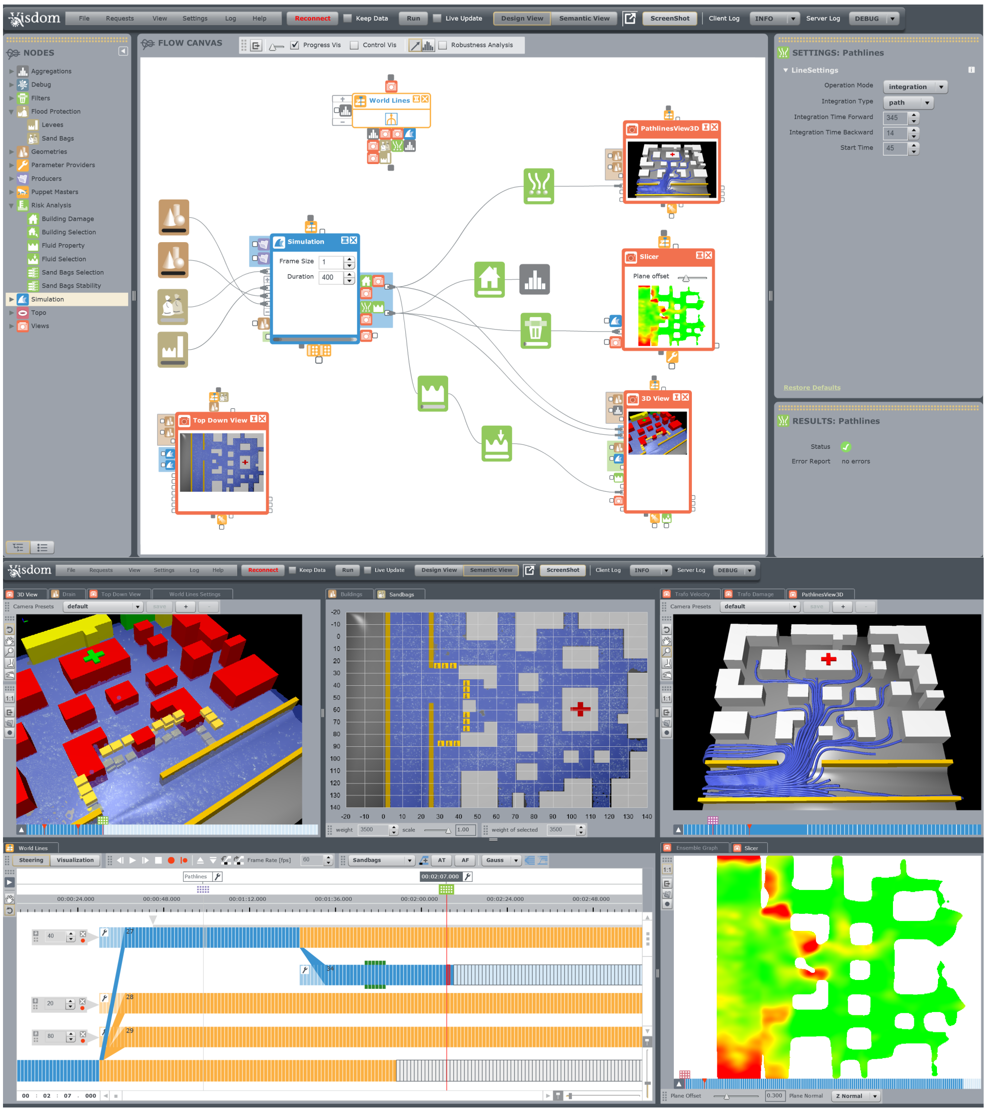
The main goal of this chapter is to answer the questions related to the verification and generation of hypotheses. The outcome is a framework for generating hypotheses by combining human and machine learning. Instead of replacing the inferential abilities of the human, the machine stimulates them by producing alternatives which can be evaluated by the human. A heuristic search algorithm which generates related features helps the engineer to evaluate theories and find alternatives in the vast hypothesis space of multi-variate data. The generated hypotheses are automatically added to the visualization and are available for further refinement and search steps.

In the presented framework, a hypothesis is simply a set of fuzzy selections together with boolean combinations. This simple format is easily transferred to text and facilitates reporting (e.g., "temperatures larger than 360 K coincide with densities in the range 0.3 to 0.5 kg/m^3 and negative velocities."). This is easier to understand than the weights of a neuronal net for example.

We consider the understandability of the generated hypotheses and the suggested incremental human+machine learning approach as the key selling point of this chapter. As Shneiderman has suggested, our approach allows the user to specify what he/she is looking for. Foremost, the visual human+machine learning approach respects the human responsibility. The integration of the machine learning step can prevent the user from missing important relations in the data.

There are several questions remaining. Since it is not possible to search the vast space of all possible hypotheses, we cannot guarantee that we are not missing an interesting hypothesis. A statistical approach to measure the decreasing probability that important selections of a given complexity have been overlooked would be an important improvement, not only to the presented approach but also to IVA in general.

There are several improvements we propose for future work. First, we have discussed the evaluation of spatial features in three dimensions, but it would be possible to apply the presented technique in a more general setting, i.e., when features are not structured in space. Non-spatial features could benefit from the presented approach using information visualization techniques only. Second, the presented approach is not limited to one particular gene type. Other visualization parameters (e.g., a viewpoint or a transfer function) and user inputs could be included into the learning process. Third, in real world data, there can be a lot of samples which do not change the fitness of a given hypothesis. To improve performance it could be sufficient to evaluate the fitness only for a randomly selected subset of the data. Fourth, the selections presented in this chapter are all based on brushing scatter plots. The presented approach is extensible to more complex selections which can be defined differently in various types of views and modified by appropriate mutation and recombination operators. Finally, the presented framework is limited to orthogonal projections. It would be interesting to study if the combination of the grand tour [13] with visual human+machine learning reveals additional meaningful hypotheses.



SETTINGS: Pathlines

LineSettings

Operation Mode: integration

Integration Type: path

Integration Time Forward: 345

Integration Time Backward: 14

Start Time: 45

RESULTS: Pathlines

Status: ✓

Error Report: no errors

Figure 6.1 – Visdom [2] integrates simulation, steering and visual analysis into a unified, modular application. At its core is a data-flow network that is controlled by World Lines to enable interactive decision making.

The minimum we should hope for with any display technology is that it should do no harm.

— *Edward Tufte*

CHAPTER 6

Summary and Conclusions

THE adverse effects of climate change on people and infrastructure is one of the largest challenges humanity has to cope with. The frequency of natural disasters such as floods is already high [26] and may increase in the near future [101]. Modern simulation technology can help to understand the cause, the nature and the impact of these phenomena. Moreover, we can exploit simulation systems to prepare response measures which aim at reducing the risk. Today, much effort is put into faster and more accurate simulation components to increase both computational efficiency and fidelity of the results. At the same time, the simulation methods become more and more complex due to a rising number of input parameters and heterogeneous data results. Interactive decision making is based on changing simulation parameters to investigate their effects on the simulation outcome. An understanding of the relations between input and output parameters is crucial and defines the ultimate goal of simulation steering. However, a comprehensive system which faciliates these efforts is largely missing. This gap is the topic of this thesis.

Initially, an intuitive concept is needed which allows the user to test alternative options without the need for special simulation expertise. World Lines is a management strategy to create and compare multiple related simulation runs in a horizontal tree-like visualization. In the World Lines view, a simulation run is represented as a track. Users insert and test new decisions based on the concept of branching. To account for uncertain knowledge about the input parameters, we provide the ability to create and inspect a complete ensemble of simulation runs. Via multiple cursors, users navigate a system of multiple linked views through time and alternative scenarios. In this way, the system supports comparative visual analysis of many simulation runs.

The system needs to be capable of adapting, since real-world applications differ case-by-case. World Lines is built on top of a data-flow network to provide a high level of modularity. To accomplish this, we suggest an interface that allows the data-flow nodes to report what data they can produce, and which enables the user to interactively define what data the nodes have to compute. We provide special purpose navigation in World Lines to compute features that require complex time-dependent processing without having to worry about the internal data-flow logic. In addition, we decouple the control-flow of parameters from the standard data-flow in order to deal with the often opposing requirements of a simulation setup to be flexible and comprehensible at the same time. This concept is termed meta-flow and has a visual characteristic in the flow diagram by putting the nodes on ropes. Special meta-flow nodes, termed puppet masters, are able to gain full control over all nodes that are connected via ropes. World Lines are an example of a sophisticated puppet master to perform simulation steering. Via the meta-flow, parameter changes are streamed down from World Lines to the affected nodes. The ropes visualize this process to direct the user attention to the most relevant parts. The same idea is applied to the data-flow to highlight the most important results inside the flow diagram.

The steering process generates a huge amount of heterogeneous information the user has to make sense of. We employ the computer to support the user in the insight generation. To find correlations inbetween the data, we suggest an evolutionary algorithm which is capable of searching the vast space of hypotheses in a reasonable time. Through an interactive visualization of the search results, the user can optimize the found theories.

Most of the presented concepts are independent of the application domain and simulation technology. Industrial prototyping or surgery planning could also benefit from a technique that allows the user to compare design decisions interactively. In addition, the combination of World Lines with the extended data-flow allows for the comparative analysis of data that is loaded from files. However, we believe that the future lies in the integration of all tasks into a single tool which the presented Visdom [2] system aims for. This involves simulation, steering, visual analysis and presentation.

The next logical step is to test the contributions on real-world scenarios. In case of flood management, we require input methods for state-of-the-art response strategies, such as the installation of mobile protective walls. Real-world applications demand the computation of parameter studies that comprise hundreds of simulation runs in order to thoroughly investigate uncertainties. To cope with this, World Lines require an automated clustering method which is based on track collapsing. To save computational resources during scheduling, we need more advanced progress monitoring with the option to automatically terminate simulation runs if certain criteria are not fulfilled.

The lyrics of 'Brothers in Arms' by Dire Straits include *"There's so many different worlds. So many different suns. And we have just one world. But we live in different ones."* Visdom and World Lines will help us to see beyond.

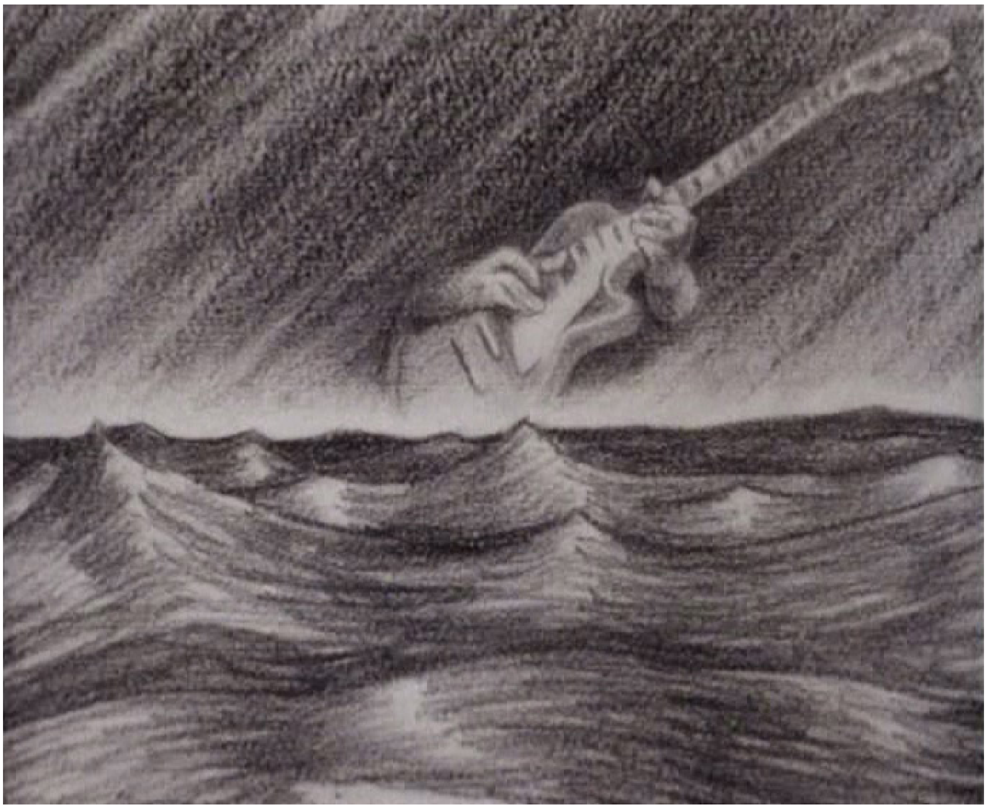


Figure 6.2 – A still taken from the music video ”Brothers in arms” by Dire Straits.

Bibliography

[1] Mevislab: A development environment for medical image processing and visualization. http://www.mevislab.de/ (last visited on 17 March 2011).

[2] Visdom - An integrated visualization system. http://visdom.at (last visited on 11 May 2011).

[3] World Lines Video. http://visdom.at/media/slides/world_lines.mp4 (last visited on 25 March 2011).

[4] *Zeittafel der Weltgeschichte. Den letzen 6000 Jahren auf der Spur*. Ullmann/Tandem, 2001.

[5] Adobe Systems Incorporated. Flex: An open source framework for developing web applications. http://www.adobe.com/products/flex/ (last visited on 3 August 2010).

[6] Advanced Visual Systems Inc. AVS - Advanced Visual System. http://www.avs.com/ (last visited on 21 March 2011).

[7] W. Aigner, S. Miksch, B. Thurnher, and S. Biffl. Planning lines: Novel glyphs for representing temporal uncertainties and their evaluation. In *Proceedings IEEE Symposium on Information Visualization 2005 (InfoVis 2005)*, 2005.

[8] A. Amirkhanov, C. Heinzl, M. Reiter, and M. E. Gröller. Visual optimality and stability analysis of 3dct scan positions. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1477 –1487, 2010.

[9] M. Ankerst. *Visual Data Mining*. PhD thesis, Faculty of Mathematics and Computer Science, University of Munich, 2000.

[10] ANSYS, Inc. *Explicit Dynamics, Chapter 10: Optimization Studies*. http://www.cadfamily.com/download/CAE/ANSYS-Explicit/Explicit_Dynamics_Optimization_Studies.pdf (last visited on 2 March 2011), 2009.

[11] H. Apel, A. Thieken, B. Merz, and G. Blöschl. Flood risk assessment and associated uncertainty. *Natural Hazards and Earth System Science*, 4:295–308, 2004.

[12] H. Apel, A. Thieken, B. Merz, and G. Blöschl. A probabilistic modelling system for assessing flood risks. *Natural Hazards*, 38:79–100, 2006.

[13] D. Asimov. The grand tour: a tool for viewing multidimensional data. *SIAM J. Sci. Stat. Comput.*, 6(1):128–143, 1985.

[14] L. Bartram, A. Ho, J. Dill, and F. Henigman. The continuous zoom: A constrained fisheye technique for viewing and navigating large information spaces. pages 207–215. ACM Press, 1995.

[15] L. Begnudelli, B. F. Sanders, and S. F. Bradford. Adaptive Godunov-based model for flood simulation. *ASCE Journal of Hydraulic Engineering*, 134(6):714–725, 2008.

[16] Bestiario. Impure - A web-based visual programming tool written in ActionScript to visualize data from the internet. `http://www.impure.com/` (last visited on 8 February 2011).

[17] J. Biddiscombe, B. Geveci, K. Martin, K. Moreland, and D. Thompson. Corrections to 'time dependent processing in a parallel pipeline architecture'. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):pp. 241, 2008.

[18] I. Bitter, R. Van Uitert, I. Wolf, L. Ibanez, and J.-M. Kuhnigk. Comparison of four freely available frameworks for image processing and visualization that use itk. *IEEE Transactions on Visualization and Computer Graphics*, 13(3):483–493, 2007.

[19] G. Blöschl. Flood warning - on the value of local information. *Intl. J. River Basin Management*, 6(1):41–50, 2008.

[20] G. Blöschl, C. Reszler, and J. Komma. A spatially distributed flash flood forecasting model. *Environ. Model. Softw.*, 23(4):464–478, 2008.

[21] K. Brodlie, L. Brankin, A. Poon, G. Banecki, H. Wright, and A. Gay. GRASPARC - A problem solving environment integrating computation and visualization. In *Proceedings IEEE Visualization 1993*, pages 102–109, 1993.

[22] S. Bruckner and T. Möller. Result-driven exploration of simulation parameter spaces for visual effects design. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1467–1475, 2010.

[23] R. Bürger, P. Muigg, M. Ilcìk, H. Doleisch, and H. Hauser. Integrating local feature detectors in the interactive visual analysis of flow simulation data. In *Proceedings of the 9th Joint IEEE VGTC - EUROGRAPHICS Symposium on Visualization (VisSym 2007)*, pages 171–178, 2007.

[24] G. Cameron. Modular visualization environments: past, present, and future. *ACM SIGGRAPH Computer Graphics 1995*, 29(2):3–4, 1995.

[25] D. Caragea, D. Cook, H. Wickhamand, and V. Honavar. Visual methods for examining SVM classifiers. *Visual Data Mining: Theory, Techniques and Tools for Visual Analytics*, pages 136–153, 2008.

[26] Centre for Research on the Epidemiology of Disasters (CRED). 2010 disasters in numbers. `http://cred.be/sites/default/files/PressConference2010.pdf` (last visited on 3 March 2011), 2011.

[27] P. Chatelain, M. Bergdorf, and P. Koumoutsakos. Large scale, multiresolution flow simulations using remeshed particle methods. In *Meshfree Methods for Partial Differential Equations IV*, pages 35–46, 2008.

[28] C. Chen. Top 10 unsolved information visualization problems. *IEEE Computer Graphics and Applications*, 25(4):12–16, 2005.

[29] M. Chen, D. Ebert, H. Hagen, R. S. Laramee, R. van Liere, K.-L. Ma, W. Ribarsky, G. Scheuermann, and D. Silver. Data, information, and knowledge in visualization. *IEEE Computer Graphics and Applications*, 29:12–19, January 2009.

[30] H. Cloke and F. Pappenberger. Ensemble flood forecasting: a review. *Journal of Hydrology*, 375(3-4):613–626, 2009.

[31] R. A. Dalrymple and A. Hérault. Levee breaching with GPU-SPHysics code. In *Proceedings of the 4th SPHERIC International Workshop*, pages 352–355, 2009.

[32] Deltares and the Delft GeoSystems. Levee patroller - the levee inspection simulator. `http://www.delftgeosystems.nl/leveepatroller/` (last visited on 18 May 2011).

[33] Department of Energy (DOE) Advanced Simulation and Computing Initiative (ASCI). VisIt - An interactive parallel visualization tool for viewing and animating visualizations from scientifc data. `https://wci.llnl.gov/codes/visit/home.html` (last visited on 9 February 2011).

[34] DHI Group. Decision support system for real-time flow forecasting. `http://www.dhigroup.com/` (last visited on 18 May 2011).

[35] H. Doleisch, M. Gasser, and H. Hauser. Interactive feature specification for focus+context visualization of complex simulation data. In *Proceedings of the 5th Joint IEEE VGTC - EUROGRAPHICS Symposium on Visualization (VisSym 2003)*, pages 239–248, 2003.

[36] H. Doleisch, M. Gasser, and H. Hauser. SimVis: Interactive visual analysis of large and time-dependent 3D simulation data. In *Proceedings of the 2007 Winter Conference on Simulation*, pages 712–720, 2007.

[37] C. Elliott, V. Vijayakumar, W. Zink, and R. Hansen. National instruments labview: A programming environment for laboratory automation and measurement. *Journal of the Association for Laboratory Automation*, 12(1):17–24, 2007.

[38] T. M. English. Optimization is easy and learning is hard in the typical function. In *Proc. Congress on Evolutionary Computation: CEC00*, pages 924–931, 2000.

[39] EU. Directive 2007/60/EC of the European Parliament and of the Council of 23 October 2007 on the assessment and management of flood risks. Official Journal of the European Union L 288/27., 2007.

[40] W. Felger and F. Schröder. The visualization input pipeline - enabling semantic interaction in scientific visualization. *Computer Graphics Forum*, 11:139–151, 1992.

[41] R. P. Feynman. Space-time approach to non-relativistic quantum mechanics. *Rev. Mod. Phys.*, 20(2):367–387, 1948.

[42] Fraunhofer Institut, Techno- und Wirtschaftsmathematik. Flood simulation package for the design of drainage systems conforming to European standards. `http://www.itwm.fraunhofer.de/fileadmin/ITWM-Media/Abteilungen/SMS/Pdf/SMS_Flyer_Hochwasser_DE_nCD_A3_web.pdf` (last visited on 18 May 2011).

[43] R. Fuchs and H. Hauser. Visualization of multi-variate scientific data. *Computer Graphics Forum*, 28(6):1670–1690, 2009.

[44] R. Fuchs, J. Waser, and M. E. Gröller. Visual human+machine learning. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1327–1334, 2009.

[45] H. L. Gantt. Work, wages and profit. *The Engineering Magazine, New York*, 1910. Republished as Work Wages and Profits, Easton, Pennsylvania, Hive Publishing Company, 1974.

[46] J. N. Ghazali and A. Kamsin. A real time simulation and modeling of flood hazard. In *Proceedings of the 12th WSEAS international conference on Systems*, pages 438–443, 2008.

[47] M. Gouda, K. Karner, and R. Tatschl. Dam flooding simulation using advanced CFD methods. In *WCCM V, Fifth World Congress on Computational Mechanics*, 2002.

[48] R. B. Haber and D. A. McNabb. Visualization idioms: A conceptual model for scientific visualization systems. *IEEE Visualization in Scientific Computing*, pages 74–93, 1990.

[49] M. Hao, U. Dayal, D. Keim, D. Morent, and J. Schneidewind. Intelligent visual analytics queries. In *Proceedings IEEE Symposium on Visual Analytics Science and Technology 2007 (Vast 2007)*, pages 91–98, 2007.

[50] M. C. Hao, D. A. Keim, U. Dayal, and J. Schneidewind. Business process impact visualization and anomaly detection. *Information Visualization*, 5(1):15–27, 2006.

[51] H. Hauser. Generalizing focus+context visualization. In G.-P. Bonneau, T. Ertl, and G. Nielson, editors, *Scientific Visualization: The Visual Extraction of Knowledge from Data*, pages 305–327. Springer Berlin Heidelberg, 2005.

[52] H. Hauser, R. Laramee, and H. Doleisch. Topology-based versus feature-based flow analysis - challenges and an application. In *Proceedings of TopoInVis 2007: Topology-Based Methods in Visualization*, pages 79–90, 2007.

[53] J. Heer and G. G. Robertson. Animated transitions in statistical data graphics. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1240–1247, 2007.

[54] C. Henze. Feature detection in linked derived spaces. In *Proceedings IEEE Visualization 1998*, pages 87–94, 1998.

[55] A. Hérault, G. Bilotta, and R. A. Dalrymple. GPU-SPHysics: A GPU-based Smoothed Particle Hydrodynamics model for free surface flows. `http://www.ce.jhu.edu/dalrymple/GPU/` (last visited on 3 August 2010), 2008.

[56] A. Hérault, A. Vicari, C. D. Negro, and R. A. Dalrymple. Modeling water waves in the surf zone with GPU-SPHysics. In *Proceedings of the 4th SPHERIC International Workshop*, pages 77–84, 2009.

[57] A. Hertzmann. Machine learning for computer graphics: A manifesto and tutorial. In *Proc. Pacific Graphics 2003*, page 22. IEEE Computer Society, 2003.

[58] R. Hoetzlein. Fluids v.2 - A Fast, Open Source, Fluid Simulator. `http://www.rchoetzlein.com/eng/graphics/fluids.htm` (last visited on 3 August 2010).

[59] IBM. Open visualization data explorer. `http://www.research.ibm.com/dx/` (last visited on 3 May 2011).

[60] H. Jänicke, A. Wiebel, G. Scheuermann, and W. Kollmann. Multifield visualization using local statistical complexity. *IEEE Transactions on Visualization and Computer Graphics*, 13(5):168–175, 2007.

[61] M. Jeng. A selected history of expectation bias in physics. *American Journal of Physics*, 74:578–583, 2006.

[62] M. H. Jo, Y. W. Jo, D. H. Shin, H. S. Kim, and J. S. Kim. Implementing the web based 3d coast flood disaster simulation system. In *25th ACRS, Chiang Mai, Thailand*, 2004.

[63] C. Johnson. Top scientific visualization research problems. *IEEE Computer Graphics and Applications*, 24(4):13–17, 2004.

[64] C. Johnson, S. G. Parker, C. Hansen, G. L. Kindlmann, and Y. Livnat. Interactive simulation and visualization. *Computer*, 32(12):59–65, 1999.

[65] W. M. Johnston, J. R. P. Hanna, and R. J. Millar. Advances in dataflow programming languages. *ACM Computing Surveys*, 36(1):1–34, 2004.

[66] M. Kass and G. Miller. Rapid, stable fluid dynamics for computer graphics. *ACM Trans. Graph.*, 24(4):49–55, 1990.

[67] J. Kehrer, F. Ladstadter, P. Muigg, H. Doleisch, A. Steiner, and H. Hauser. Hypothesis generation in climate research with interactive visual data exploration. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1579–1586, 2008.

[68] D. Keim, F. Mansmann, J. Schneidewind, J. Thomas, and H. Ziegler. Visual analytics: Scope and challenges. *Lecture Notes In Computer Science*, pages 76–90, 2008.

[69] D. A. Keim. Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):1–8, 2002.

[70] S. Kim, Y. Jang, A. Mellema, D. Ebert, and T. Collins. Visual analytics on mobile devices for emergency response. In *Proceedings IEEE Symposium on Visual Analytics Science and Technology 2007 (Vast 2007)*, pages 35–42, 2007.

[71] P. Kipfer and R. Westermann. Realistic and interactive simulation of rivers. In *ACM Proceedings of Graphics Interface 2006*, pages 41–48, 2006.

[72] Kitware. Paraview - an open source, multi-platform data analysis and visualization application. `http://www.paraview.org/` (last visited on 17 March 2011).

[73] Y. Koike, A. Sugiura, and Y. Koseki. Timeslider: an interface to specify time point. In *Proceedings of the 10th annual ACM symposium on user interface software and technology (UIST)*, pages 43–44, 1997.

[74] J. Komma, C. Reszler, G. Blöschl, and T. Haiden. Ensemble prediction of floods - catchment non-linearity and forecast probabilities. *Natural Hazards and Earth System Sciences*, 7:431–444, 2007.

[75] R. Kosara and S. Miksch. Visualization techniques for time-oriented, skeletal plans in medical therapy planning. In *Proceedings of the Joint European Conference on Artificial Intelligence in Medicine and Medical Decision Making (AIMDM 1999)*, pages 291–300, 1999.

[76] R. Kosara and S. Miksch. Metaphors of movement: A visualization and user interface for time-oriented, skeletal plans. In *Artificial Intelligence in Medicine, Special Issue: Information Visualization in Medicine*, pages 111–131, 2001.

[77] P. Koumoutsakos, G.-H. Cottet, and D. Rossinelli. Flow simulations using particles: bridging computer graphics and CFD. In *ACM SIGGRAPH 2008 classes*, pages 1–73, 2008.

[78] O. Kreylos, A. Tesdall, B. Hamann, J. Hunter, and K. Joy. Interactive visualization and steering of CFD simulations. In *Proceedings of the 4th Joint IEEE VGTC - EUROGRAPHICS Symposium on Visualization (VisSym 2002)*, pages 25–34, 2002.

[79] S. Laine. *Using visualization, variable selection and feature extraction to learn from industrial data.* PhD thesis, TU Helsinki, 2003.

[80] R. Laramee and H. Hauser. Geometric flow visualization techniques for cfd simulation data. In *Proceedings of the 21st Spring Conference on Computer graphics*, pages 213–216, 2005.

[81] Q. Liang, A. G. L. Borthwick, and G. Stelling. Simulation of a dam- and dyke-break hydroynamics on dynamically adaptive quadtree grids. *Int. J. Numer. Meth. Fluids*, 46:127–162, 2004.

[82] C. Liu and T. Zhang. Distributed data visualization tools for multidisciplinary design optimization of aero-crafts. In *Advances in Multimedia Information Processing - PCM 2006*, pages 953–960, 2006.

[83] D. C. Luckham. *The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001.

[84] K.-L. Ma. Image graphs - a novel approach to visual data exploration. In *Proceedings IEEE Visualization 1999*, pages 81–88, 1999.

[85] K.-L. Ma. Machine learning to boost the next generation of visualization technology. *IEEE Computer Graphics and Applications*, 27(5):6–9, 2007.

[86] M. Marttila-Kontio and R. Honkanen. Not-so-free data flow in a visual data flow programming language. *Computer Science and Information Technology, International Conference on*, pages 613–619, 2009.

[87] K. Matković, D. Gracanin, M. Jelovic, and H. Hauser. Interactive visual steering - rapid visual prototyping of a common rail injection system. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1699–1706, 2008.

[88] K. Matković, H. Hauser, R. Sainitzer, and M. E. Gröller. Process visualization with levels of detail. In *Proceedings IEEE Symposium on Information Visualization 2002 (InfoVis 2002)*, pages 67–70, 2002.

[89] T. May and J. Kohlhammer. Towards closing the analysis gap: Visual generation of decision supporting schemes from raw data. *Computer Graphics Forum*, 27(3):911–918, 2008.

[90] J. Meyer-Spradow, T. Ropinski, J. Mensmann, and K. H. Hinrichs. Voreen: A rapid-prototyping environment for ray-casting-based volume visualizations. *IEEE Computer Graphics and Applications*, 29(6):6–13, 2009.

[91] H. Minkowski. Raum und Zeit. *Physikalische Zeitschrift*, 10:104–111, 1909. (Lecture delivered before the Versammlung Deutscher Naturforscher und Ärzte, Cologne, September 21, 1908.) Reprinted in Blumenthal 1913. English translation in Lorentz et al. 1952. Page numbers refer to this last edition.

[92] J. J. Monaghan. Smoothed particle hydrodynamics. *Reports on Progress in Physics*, 68:1703–1759, 2005.

[93] J. D. Mulder, J. J. van Wijk, and R. van Liere. A survey of compuational steering environments. *Future Generation Computer Systems*, 15:119–129, 1999.

[94] M. Müller, D. Charypar, and M. Gross. Particle-based fluid simulation for interactive applications. In *ACM SIGGRAPH Symposium on Computer Animation (SCA)*, pages 154–159, 2003.

[95] W. Müller, T. Nocke, and H. Schumann. Enhancing the visualization process with principal component analysis to support the exploration of trends. In *Asia Pacific Symposium on Information Visualisation*, volume 60, pages 121–130, 2006.

[96] National Instruments (NI). LabVIEW: A graphical programming environment for engineers to develop measurement, test, and control systems. `http://www.ni.com/labview` (last visited on 21 March 2011).

[97] T. M. Nguyen, J. Schiefer, and A. M. Tjoa. Sense & response service architecture (saresa): an approach towards a real-time business intelligence solution and its use for a fraud detection application. In *DOLAP '05: Proceedings of*

the 8th ACM international workshop on Data warehousing and OLAP, pages 77–86, 2005.

[98] Numerical Algorithms Group (NAG). IRIS Explorer: A visual programming environment to develop, prototype and build visualization applications. `http://www.nag.co.uk/Welcome_IEC.asp` (last visited on 21 March 2011).

[99] NVidia Corporation. CUDA: Parallel computing architecture. `www.nvidia.com/object/cuda_home.html` (last visited on 18 May 2011).

[100] NVidia Corporation. PhysX: Physics Simulation Toolkit. `http://developer.nvidia.com/object/physx.htm` (last visited on 3 August 2010).

[101] I. P. on Climate Change (IPCC). Ipcc fourth assessment report: Climate change 2007 (ar4). `http://www.ipcc.ch/publications_and_data/publications_and_data_reports.shtml` (last visited on 17 March 2011), 2007.

[102] S. G. Parker and C. R. Johnson. SCIRun: A scientific programming environment for computational steering. In *Proceedings of the 1995 ACM/IEEE conference on Supercomputing*, page 52, 1995.

[103] A. M. Pérez, J. M. Gómez, and R. C. Pérez. Semantic interaction in enterprise data-flow visualization environments: An exploratory study. In *ICT Innovations 2009*, pages 217–226. Springer Berlin Heidelberg, 2010.

[104] F. Poulet. SVM and graphical algorithms: a cooperative approach. In *ICDM '04. Fourth IEEE International Conference on Data Mining*, pages 499–502, 2004.

[105] F. Poulet. *Visual Data Mining*, chapter Towards Effective Visual Data Mining with Cooperative Approaches, pages 389–406. Springer Berlin / Heidelberg, 2008.

[106] P. Rautek. *Semantic Visualization Mapping for Volume Illustration*. PhD thesis, Vienna University of Technology, 2009.

[107] S. Rinderle, R. Bobrik, M. Reichert, and T. Bauer. Business process visualization - use cases, challenges, solutions. In *ICEIS 2006 - Proceedings of the Eigth International Conference on Enterprise Information Systems: Databases and Information Systems Integration, Paphos, Cyprus, May 23-27, 2006*, pages 204–211, 2006.

[108] F. Rioux, F. Bernier, and D. Laurendeau. Multichronia - a framework for the exploration of parameter, simulation, data and visual spaces. In *The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC)*, 2008.

[109] F. Rossi. Visual data mining and machine learning. In *European Symposium on Artificial Neural Networks (ESANN2006)*, pages 251–264, 2006.

[110] E. A. Rundensteiner, M. O. Ward, J. Yang, and P. R. Doshi. XmdvTool: visual interactive data exploration and trend discovery of high-dimensional data sets. In *SIGMOD '02: Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, pages 631–631, 2002.

[111] E. Santos, L. Lins, J. Ahrens, J. Freire, and C. Silva. VisMashup: Streamlining the creation of custom visualization applications. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1539–1546, 2009.

[112] A. Sattar, A. Kassem, and M. Chaudhry. 17th street canal breach closure procedures. *Journal of Hydraulic Engineering*, 134(11):1547–1558, 2008.

[113] C. Scheidegger, H. Vo, D. Koop, J. Freire, and C. Silva. Querying and creating visualizations by analogy. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1560 –1567, 2007.

[114] B. Schindler, J. Waser, R. Fuchs, and R. Peikert. Multiverse data-flow control. Technical Report 720, ETH Zürich Computational Science, 2010.

[115] W. Schroeder, K. Martin, and B. Lorenson. *The Visualization Toolkit, An Object Oriented Approach to 3D Graphics*. Kitware Inc., fourth edition, 2004.

[116] Scientific Computing and Imaging Institute (SCI). SCIRun: A scientific computing problem solving environment. `http://www.scirun.org` (last visited on 21 March 2011).

[117] J. Seo and B. Shneiderman. A rank-by-feature framework for interactive exploration of multidimensional data. *Information Visualization*, 4(2):96–113, 2005.

[118] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Proceedings of the IEEE Symposium on Visual Languages 1996*, pages 336–343, 1996.

[119] B. Shneiderman. Inventing discovery tools: combining information visualization with data mining. *Information Visualization*, 1(1):5–12, 2002.

[120] C. T. Silva, J. Freire, and S. P. Callahan. Provenance for visualizations: Reproducibility and beyond. *Computing in Science and Engineering*, 9(5):82–89, 2007.

[121] SimVis Gmbh. SimVis - A novel visualization solution for simulation results. `http://www.simvis.at` (last visited on 18 May 2011).

[122] M. G. Stewart and R. E. Melchers. *Probabilistic risk assessment of engineering systems*. Chapman and Hall, London, 1997.

[123] S. Succi. *The Lattice Boltzmann Equation for Fluid Dynamics and Beyond*. Oxford University Press: Oxford, 2001.

[124] M. Suntinger, H. Obweger, J. Schiefer, and M. E. Gröller. The event tunnel: Interactive visualization of complex event streams for business process pattern analysis. In *Proceedings of IEEE PacificVis 2008*, pages 111–118, 2008.

[125] A. Telea and J. J. van Wijk. Vission: An object oriented dataflow system for simulation and visualization. In *Proceedings of IEEE VisSym*, pages 95–104. Springer, 1999.

[126] A. Telea and J. J. van Wijk. Smartlink: An agent for supporting dataflow application construction. In *Proceedings of the 2th Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym 2000)*, pages 189–198, 2000.

[127] A. H. Thieken, M. Müller, H. Kreibich, and B. Merz. Flood damage and influencing factors: New insights from the August 2002 flood in Germany. *Water Resources Research*, 41:16pp, 2005.

[128] J. Thomas and K. Cook. A visual analytics agenda. *IEEE Computer Graphics and Applications*, 26(1):10 – 13, 2006.

[129] M. Tritthart and D. Gutknecht. 3-d computation of flood processes in sharp river bends. *Water Management*, 160(4):233–247, 2007.

[130] F.-Y. Tzeng, E. B. Lum, and K.-L. Ma. An intelligent system approach to higher-dimensional classification of volume data. *IEEE Transactions on Visualization and Computer Graphics*, 11(3):273–284, 2005.

[131] F.-Y. Tzeng and K.-L. Ma. Intelligent feature extraction and tracking for visualizing large-scale 4d flow simulations. In *Proc. of the 2005 ACM/IEEE conference on Supercomputing*, page 6, 2005.

[132] United States Army Corps of Engineers (USACE). Interagency performance evaluation taskforce (IPET) final report. `https://ipet.wes.army.mil/` (last visited on 14 April 2011), 2007.

[133] C. Upson, T. Faulhaber, Jr., D. Kamins, D. H. Laidlaw, D. Schlegel, J. Vroom, R. Gurwitz, and A. van Dam. The application visualization system: A computational environment for scientific visualization. *IEEE Computer Graphics and Applications*, 9:30–42, July 1989.

[134] W. J. van der Laan, S. Green, and M. Sainz. Screen space fluid rendering with curvature flow. In *Proceedings of the Symposium on Interactive 3D Graphics and Games 2009 (I3D 2009)*, pages 91–98, 2009.

[135] R. van Liere. Computational steering. In *High-Performance Computing and Networking*, pages 696–702. Springer-Verlag, 1996.

[136] C. van Treeck, P. Wenisch, A. Borrmann, M. Pfaffinger, M. Egger, O. Wenisch, and E. Rank. Towards interactive indoor thermal comfort simulation. In *European Conference on Computational Fluid Dynamics*, 2006.

[137] M. Ward. Finding needles in large-scale multivariate data haystacks. *IEEE Computer Graphics and Applications*, 24(5):16–19, 2004.

[138] J. Waser, R. Fuchs, H. Ribičić, and G. Blöschl. Visuelle Aktionsplanung im Hochwassermanagement mit Visdom. In *Forum für Hydrologie und Wasserbewirtschaftung/FgHW*, volume 30, pages 280–286, 2011.

[139] J. Waser, R. Fuchs, H. Ribičić, B. Schindler, G. Blöschl, and M. E. Gröller. World Lines. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1458–1467, 2010.

[140] C. Weaver. Building highly-coordinated visualizations in improvise. In *Proceedings IEEE Symposium on Information Visualization 2004 (InfoVis 2004)*, pages 159–166, 2004.

[141] C. Weaver. Visualizing coordination in situ. In *Proceedings IEEE Symposium on Information Visualization 2005 (InfoVis 2005)*, pages 165–172, 2005.

[142] M. Wohlfart and H. Hauser. Story telling for presentation in volume visualization. In K. Museth, T. Möller, and A. Ynnerman, editors, *Proceedings of EuroVis 2007*, pages 91–98, 2007.

[143] H. Wright, K. Brodlie, and M. Brown. The dataflow visualization pipeline as a problem solving environment. In *Proceedings of the Eurographics Workshop on Virtual Environments and Scientific Visualization 1996*, pages 267–276, 1996.

[144] H. Wright and J. Walton. Hyperscribe: A data management facility for the dataflow visualization pipeline. In *IRIS Explorer Technical Report IETR/4, NAG Ltd*, 1996.

[145] S. Yamaguchi, T. Ikeda, K. Iwamura, K. Naono, A. Ninomiya, K. Tanaka, and H. Takahashi. Development of GIS-based flood-simulation software and application to flood-risk assessment. In *Proceedings of the European Conference on Flood Risk Management*, 2007.

[146] D. Yang, E. A. Rundensteiner, and M. O. Ward. Analysis guided visual exploration of multivariate data. In *Proceedings IEEE Symposium on Visual Analytics Science and Technology 2007 (Vast 2007)*, pages 83–90, 2007.

Curriculum Vitae

DI Jürgen Waser
jwaser@vrvis.at

Personal Data

Date of birth	December 22, 1980
Birthplace	Wels
Citizenship	Austria
Military Service	fulfilled

Education

10/2000 - 06/2008

Vienna, University of Technology
Technical physics, diploma with distinction.

- Focus: Simulation, Animation and Visualization.
- Diploma thesis: *v4flair - A scientific visualization tool for the FLAPW package FLAIR* (development of an interactive application for the visualization of simulation data)

08/2006 - 05/2007

Dalhousie University, Halifax, Canada
Two terms overseas study funded by TASSEP (Trans Atlantic Science Student Exchange Program).

09/1990 - 06/1999

Humanistisches Stiftsgymnasium, Kremsmünster
Completed with distinction

since 1988

Video and Animation
Autodidact, courses at the university of applied arts, Vienna

Professional Experience

since 10/2008

Researcher at the VRVis, Research Center for Virtual Reality and Visualization, focus on GPGPU programming, visualization and computational steering

since 2007

Web Programmer websites, e.g., for the Center for Computational Materials Science http://www.cms.tuwien.ac.at

since 2002

Video Producer DVDs, e.g., for TCG Unitech, Gruber Extrusionstechnik, Waser Gmbh, Hoba company

Professional Experience (continued)

since 2004	**Photographer** pictures for advertising, e.g., for Eckermann Drums: http://www.eckermanndrums.com
1996-2008	**Roofer, Tinsmith, Carpenter** at Fa. Hartl

Activities

since 10/2008	Co-founder and core developer of the integrated visualization system Visdom http://visdom.at/
10/2007	Contributions to CISCI (Cinema and Science): • *Generation of ultrahort laser pulses* http://www.cisci.net/pulse • *Pulse Compression* http://www.cisci.net/compression
10/2005 - 06/2008	Educational movie and animations about ultrashort physics: *Femto*

Publications

2011	J. Waser, H. Ribičić, R. Fuchs, C. Hirsch, B. Schindler, G. Blöschl, and M. E. Gröller. Nodes on Ropes: A comprehensive Data and Control Flow for Steering Ensemble Simulations. *submitted to IEEE Transactions on Visualization and Computer Graphics (Proceedings IEEE Vis 2011)*
	B. Schindler, R. Fuchs, J. Waser and R. Peikert. Marching Correctors - Fast and Precise Polygonal Isosurfaces of SPH Data. *In Proceedings of the 6th International SPHERIC Workshop (to appear)*
	B. Schindler, J. Waser, R. Fuchs, and R. Peikert. Multiverse data-flow control. *Technical Report 720, ETH Zürich Computational Science*, 2010
	J. Waser, R. Fuchs, H. Ribičić and G. Blöschl. Visuelle Aktionsplanung im Hochwassermanagement. *Forum für Hydrologie und Wasserbewirtschaftung/FgHW*, 30.11:280-286, 2011
2010	J. Waser, R. Fuchs, H. Ribičić, B. Schindler, G. Blöschl, and M. E. Gröller. World Lines. *IEEE Transactions on Visualization and Computer Graphics (Proceedings IEEE Vis 2010, honorable mention)*, 16(6):1458-1467, 2010
	R. Fuchs, J. Kemmler, B. Schindler, J. Waser, F. Sadlo, H. Hauser and R. Peikert. Toward a Lagrangian Vector Field Topology. *Computer Graphics Forum (Proceedings EuroVis 2010)*, 29:1163-1172, 2010
2009	R. Fuchs, J. Waser and M. E. Gröller. Visual Human+Machine Learning. *IEEE Transactions on Visualization and Computer Graphics (Proceedings IEEE Vis 2009)*, 15(6):1327-1334, 2009

Languages

German	native speaker
English	fluent in spoken and written
French	basic knowledge (4 years in school)
Latin	qualification in latin

Interests

sports (badminton, volleyball, basketball, tennis, swimming), music, literature, movies, nature, camping, craft, fossils