Sketching Uncertainty into Simulations

Hrvoje Ribičić, Jürgen Waser, Roman Gurbat, Bernhard Sadransky, and M. Eduard Gröller

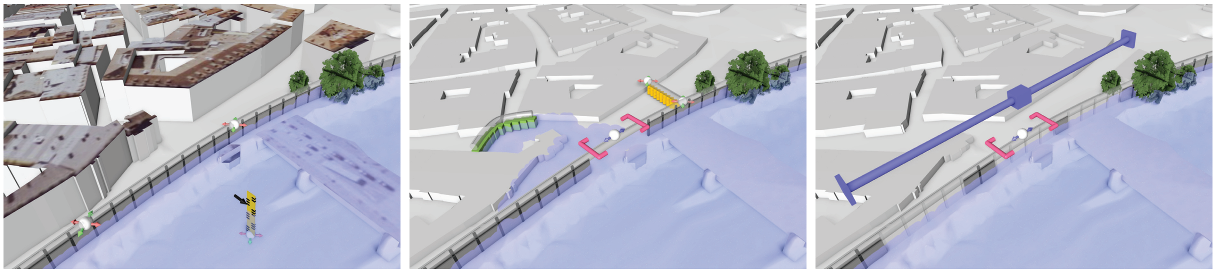


Fig. 1. Screenshots of the interface. (Left) The user has sketched a mobile wall to protect the city from flooding. (Middle) Interactive design of multiple embankments to mitigate the effects of a user-induced dam break. (Right) The user manipulates a blue handle to setup an ensemble simulation with respect to the uncertain breach location. The resulting distribution of position values is visualized as a feedback aggregation directly on the mobile wall.

Abstract—
In a variety of application areas, the use of simulation steering in decision making is limited at best. Research focusing on this problem suggests that most user interfaces are too complex for the end user. Our goal is to let users create and investigate multiple, alternative scenarios without the need for special simulation expertise.

To simplify the specification of parameters, we move from a traditional manipulation of numbers to a sketch-based input approach. Users steer both numeric parameters and parameters with a spatial correspondence by sketching a change onto the rendering. Special visualizations provide immediate visual feedback on how the sketches are transformed into boundary conditions of the simulation models. Since uncertainty with respect to many intertwined parameters plays an important role in planning, we also allow the user to intuitively setup complete value ranges, which are then automatically transformed into ensemble simulations.

The interface and the underlying system were developed in collaboration with experts in the field of flood management. The real-world data they have provided has allowed us to construct scenarios used to evaluate the system. These were presented to a variety of flood response personnel, and their feedback is discussed in detail in the paper. The interface was found to be intuitive and relevant, although a certain amount of training might be necessary.

Index Terms—Emergency/Disaster Management, Interaction Design, Uncertainty Visualization, Sketch-Based Steering, Ensemble Simulation Steering, Integrated Visualization System, Flood Management.

1 Introduction

In the field of flood management, simulations can be used to improve decision making. With a number of flood defenses in place, the people in charge of flood-protection measures often wish to test what would happen if any defenses were to fail. By simulating known and expected hazards, flood-risk maps are generated, which describe the effects of the disaster. Based on these, evacuation measures and second lines of defense can be designed in advance. The simulations are usually performed manually, at great cost for every individual parameter value to be explored. As an alternative, simulation-steering systems can enable the examination of many different parameters, with changes that can be introduced dynamically.

However, the obstacles to using any kind of simulation support often lie in the tools provided to the end user. An examination of the existing tools and how experts use them has found that a lack of simplicity and robustness is to blame for the low incidence of use [22]. Simulations are powerful tools, but the freedom they afford is also a burden. The larger the parameter space, the harder it is to setup or steer a simulation. As a result, only the simplest of simulations are applied in crisis management. To alleviate this problem, this paper proposes an improvement in the user interface and explores if this could result in a more widespread use of simulation-based decision making.

The biggest problem of current interfaces is the gap between the designer and the user of the system. Current input methods are primarily numeric inputs, suitable for a well-trained engineer who can both intuit and understand how changes in parameters are reflected in the results. The emergency-response personnel, drawn from a variety of public institutions, cannot be expected to do the same. More intuitive methods are required, especially for specifying and manipulating the shapes and positions of parameters with a spatial correspondence.

There are very good reasons as to why numeric inputs are still predominant. They do not suffer from issues related to input precision. The settings of one simulation can easily be transferred to another one. The specification of parameter ranges required for ensemble simulations is straightforward. In the field of hydrology, where ensemble simulations are used regularly due to input uncertainty, the last requirement is especially important. While it is impossible to offer an alternative with the exact same properties, any solution aiming to replace current interfaces must at least partially address these requirements.

This paper introduces an intuitive approach based on sketching that addresses the issue of precision and provides a simple way for users to create ensembles. We introduce sketching functionality into a simulation-steering enviroment that uses World Lines (WL) [27]. Using an input device such as a mouse or touchscreen, the user draws onto a rendering of the simulation to change its parameters. The changes can vary from the creation of new objects, such as protec-

- *Hrvoje Ribičić is with VRVis Vienna, E-mail: ribicic@vrvis.at.*
- *Jürgen Waser is with VRVis Vienna, E-mail: jwaser@vrvis.at.*
- *Roman Gurbat is with VRVis Vienna, E-mail: gurbat@vrvis.at.*
- *Bernhard Sadransky is with VRVis Vienna, E-mail: sadransky@vrvis.at.*
- *Eduard Gröller is with TU Vienna, E-mail: groeller@cg.tuwien.ac.at.*

Manuscript received 31 March 2012; accepted 1 August 2012; posted online 14 October 2012; mailed on 5 October 2012.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

tion barriers or water sources, to changes to existing parameters such as the water level of a river. The inputs are interpreted and transformed into new boundary conditions that World Lines can understand.

The process of interpreting the sketches inevitably introduces imprecision. To counter this, we introduce feedback visualizations. As the user is sketching the change, the system produces a context-specific visualization showing how the sketch was interpreted. The visualization allows the users to detect errors caused by sketching before starting an expensive simulation, and to better understand the effect of the introduced change. In some cases, a lower-complexity simulation is performed to allow the user a greater insight into the effects of the chosen parameters.

Every sketch can be modified after being completed, using visual abstractions called handles. Each handle represents an invididual simulation parameter, e.g., size, position, or velocity. As the meaning of every handle is well-defined, the user can introduce input uncertainty into the simulation by selecting handles. In response, the handles morph, representing and controlling a distribution of values rather than a single one. Other mechanisms for creating ensembles are available for input parameters that cannot be described as a single value. Shapes can either be interpolated, or taking advantage of the input method, quickly sketched one after another.

To evaluate whether our interface better meets the needs of expert end-users, we presented our system to a consortium of experts. These range from operational staff handling flooding hazards to flood-simulation experts. We prepared two scenarios based on real-world data showing various features of the solution as employed in an urban and a rural setting. After presenting this data to the experts, we collected responses to a questionnaire. The paper is structured around these scenarios, first introducing the components of the system necessary to understand them, and then presenting feedback visualization and ensemble specification as applied to them. In the end, each of the three scenarios is presented alongside the expert feedback.

In summary, the scientific contributions of this paper are:

- Mechanisms for the translation of sketches into the boundary conditions of a simulation.
- Feedback visualizations for a precise process of translation.
- A mechanism for specifying ensemble simulations through sketching.
- Manipulable visualizations of ensemble-parameter ranges.
- An evaluation of the solution by a consortium of experts.

2 Related Work

Existing disaster-management software is mainly used for coordination and communication during the response phase [13, 14]. Decision makers can access pre-calculated flood-hazard maps to better handle the situation at hand [17]. Since the course of events can only be predicted to a limited degree of certainty [7], numerous alternative scenarios have to be evaluated in advance [6]. In practice, these scenarios are computed in ensemble simulations which consist of many simulation runs for varying parameter values [10]. However, the integration of simulation technology into operational decision-support systems is not yet successful mainly due to the complexity involved in steering the simulations [22]. Recent work on simulation steering [5, 27] has shown that it is possible to create and manage multi-simulation runs without the need for special simulation expertise. However, the presented metaphors for the specification of simulation parameters are still based on time-consuming input mechanisms. To design a flood embankment, users click into an orthogonal representation of the scenario to determine the location of one barrier element at a time. Complete parameter distributions as required by ensemble simulations can only be entered through traditional manipulation of numbers [28]. TanGeoMS [26] employs a tangible interface to simplify the specification of simulation parameters. To alter the topography of a terrain model, users place objects on a surface. The modified model is then imported into a GIS for the simulation of real-world processes.

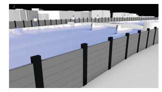
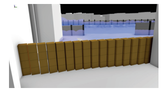
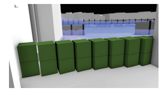
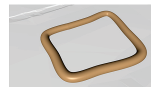
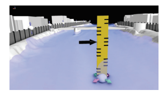
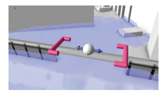
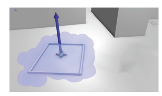
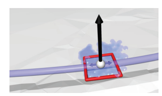
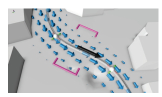
Category	Representation	Name and Description	Primitive
Protection		**Mobile Walls** Mobile barrier consisting of beams stacked between posts	Spline
		**Sandbags** The most flexible flood protection barrier	Spline
		**Big Bags** Large bags filled with sand or water	Spline
		**Polder** A tract of land enclosed by embankments known as dikes	Closed Spline
Incidents		**Rise of the Water Levels** An increase in the water level which causes flooding	Position
		**Breach** A breach in a levee which causes flooding	Position on Spline
		**Sewer Overflow** Sanitary-sewer overflows which cause flooding	Position
		**Seep Water** Seep-water locations where groundwater reaches the surface	Position on Spline
Forces		**Force** A force field to be applied to the water flow	Spline

Fig. 2. List-based version of the action pool which displays all actions that can be deployed. The right-most column lists the sketching primitive required to draw the action into the simulation.

Sketch-based interfaces are a natural way of interacting with computers. Drawing strokes is familiar to almost everyone and thus a convenient and efficient way to express requests [30]. The literature provides comprehensive surveys on sketch-based modeling including descriptions of a common architecture [11, 20]. All sketch-based approaches are based on a pipeline consisting of three stages: sketch acquisition, sketch filtering and sketch interpretation. Sketch-based modeling techniques can be utilized to create low-detail models for rapid prototyping or design work [20]. Smelik et al. [25] introduce a procedural sketching-approach which enables a non-specialist user to create a complete 3D virtual world. Designers can immediately see the effects of their procedural modeling operations. McCrae and Singh [18] try to minimize the set of sketching instructions to support the efficient authoring of road layouts [8]. Pihuit et al. [21] present an approach for modeling vascular systems. Instead of an iterative sketching process, a drawing is transformed into a 3D model at once [24].

Apart from the modeling domain, sketch-based interfaces are found in the field of animation and illustration. Davis et al. [12] present a technique which transforms 2D sketches into an animation. Schroeder et al. [23] use a sketch-based interface to allow artists to draw into an illustrative visualization of a 2D vector field. Zhu et al. [31] present a sketching system that enables interactive illustration of complex fluid behavior. The authors adapt a fluid-simulation model to enhance the illustrations. Ongoing fluid simulations can be directed by user sketches [9]. Gu and Deng [15] generate group formations for crowd simulation by sketching formation boundaries. ForcePAD [1, 16] is a tool for visualizing the behavior of structures subjected to loading and boundary conditions. Users are able to design structures, apply loads and define boundary conditions without knowledge of the underlying finite element model. A heatmap visualizes the resulting stresses and deformation in the material.

The input concepts presented in this work utilize World Lines (WL) for the management and navigation of simulation runs [4, 27]. In the WL view, each run is shown as a track in a horizontal tree-like layout. New decisions are recorded as branches and result in new tracks.

3 Sketching Boundary Conditions

The common theme of all the evaluation scenarios is flood protection. The person in charge of the defenses has an overview of an area where a disaster can happen, and a number of protection measures available. Given that there is only a limited amount of time, materials and personnel that can be used, the selection of the measures to be deployed is a difficult task. The goal of our steering environment is to allow the user to experiment with various options, and to do it as intuitively as possible. The resulting interface is minimal, consisting of an action pool and a toolbar allowing the choice of a type of interaction. An accompanying video demonstrates the interface in action [2].

3.1 Action Pool

The action pool is the tool used to show the user the changes that can be applied to the simulation. Figure 2, column 1-3, depicts the action pool which contains a list of actions accompanied by short descriptions. In our scenarios, the actions can roughly be divided into three categories: Protection, incidents and forces. A rough workflow of a typical session involves the user setting up the initial conditions by sketching incidents, and perhaps some inital protection measures. To explore the scenario, the user applies further incidents, and protection measures designed to thwart them. For example, the user may load a project containing a river flowing through a town. By applying a water-rise action to the river, the town is put under threat by a slowly rising river. To counter the threat, the user chooses a mobile protection-wall action, and sketches two walls along the riverside. To explore what might happen if the defenses fail, a wall can be breached to simulate a collision with debris.

The main benefit of the action pool is that it provides a central location where the user can find all the steerable changes important to a scenario. These are not necessarily all the parameters of a simulation. The designer setting up the simulation chooses a subset of parameters to be manipulated, lowering the complexity for the end user. Other parameters can still be changed through conventional interfaces.

3.2 Sketching Primitives

An important fact to note about the actions is that they are not complete specifications of simulation changes. An action must always be accompanied by a position, a shape, or another entity used to complete a specification. It is up to the user to designate these when an action is introduced. To allow the specification to be consistent for different action types, we introduce sketching primitives. The goal of sketching primitives is to separate what is sketched from how it is sketched.

One of the basic sketching primitives is a spline. Splines describe a curve positioned in space. They are created when the user draws on the terrain, adapting to its shape (Figure 3a). If the user sketches a barrier of big bags (Figure 3b), the system interprets the sketch as a creation of a spline primitive to which a big-bags action is assigned. The representation of the spline is rendered above the created barrier, showing the user the primitive that was generated. If the user chooses to place a water source by clicking on the terrain, e.g., to model a sewer overflow, a position primitive is implicitly created.

The usefulness of the distinction between what is sketched and how it is sketched becomes apparent when the user attempts to explore alternatives. For example, the user may wish to replace a barrier consisting of big bags with a mobile protection wall. Rather than having to specify the shape of the barrier again, possibly introducing errors, the user simply assigns a different action to the spline (Figure 3c,d). This attachment of action to primitive is called *action assignment*, and is used to assign meanings to primitives. It can also be used to assign

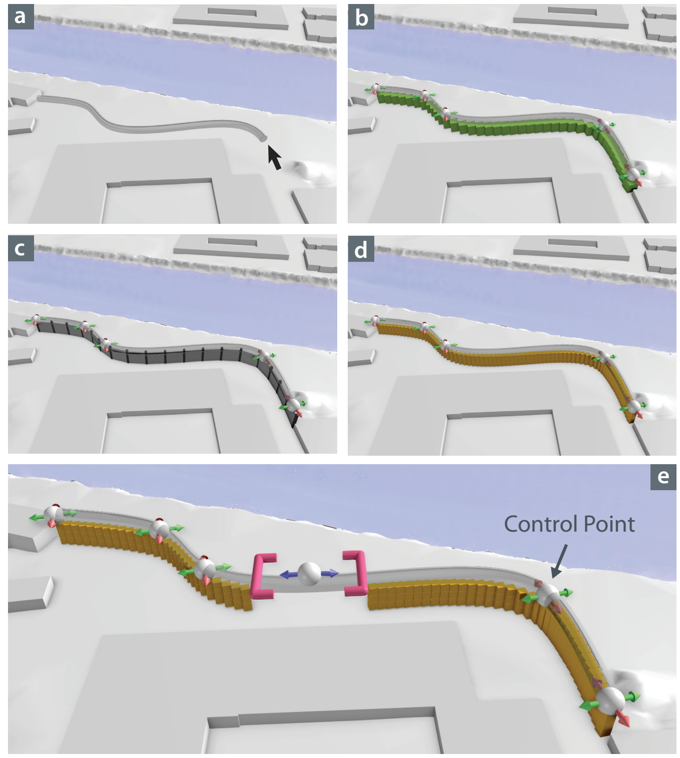


Fig. 3. Sketching and action assignment. (a) The user is drawing a spline while the big-bags action is selected. (b) Immediately after completion, the system displays feedback on how the sketch is interpreted. The action associated with the spline is replaced by (c) mobile walls or (d) sandbags to test alternative measures. (e) Multiple actions can be assigned to a single sketch, e.g., to attach a barrier and a breach.

multiple actions to a spline, e.g., when introducing a barrier action and a breach action (Figure 3e). The decoupling also allows for a consistent way of manipulating objects. Should the user decide to change the shape of a barrier, the control points drawn on the spline can be moved to achieve this (Figure 3e). The spline can represent very different simulation objects, such as shapes, position ranges, directions, or areas, and still be manipulated in the same intuitive and consistent way. With this functionality at hand, the user is able to design a mobile protection system preventing the city from flooding (Figure 4).

The basic user interface thus consists of two tools - sketching and action assignment. If an action is currently selected in the action pool, it is automatically assigned to a newly sketched primitive, in case the two are compatible. However, the user can also create primitives that have no action assigned to them to signal unexpected uses of actions. For example, let us assume the user wants to model a dam, under which water can leak due to seeping behavior. The user can create the spline primitive to specify a range of positions where a leak may happen, and assign a seep-water action, i.e., a water source, to it. The usual behavior of sketching with this action selected would be to implicitly create a position primitive, not a spline. Explicit assignment creates a movement constraint - the water source is created on the spline, and can be moved along the spline only. This constraint can also be directly introduced as a different action within the action pool. For this reason, we differentiate between the sewer-overflow incident and the seep-water action, both of which are based on water-source sketching. To better illustrate the abilities of the system, Figure 2 shows a list of actions and primitives used in our scenarios.

3.3 Edit Mode

One issue of using sketching as an input method is that the user has to find a perspective appropriate for a sketch. This requirement is in conflict with another of the experts' requirements. For presentation pur-

Fig. 4. City protected by mobile walls. The user has drawn barriers on each side of the river. The simulation of rising water levels confirms the robustness of the chosen design.

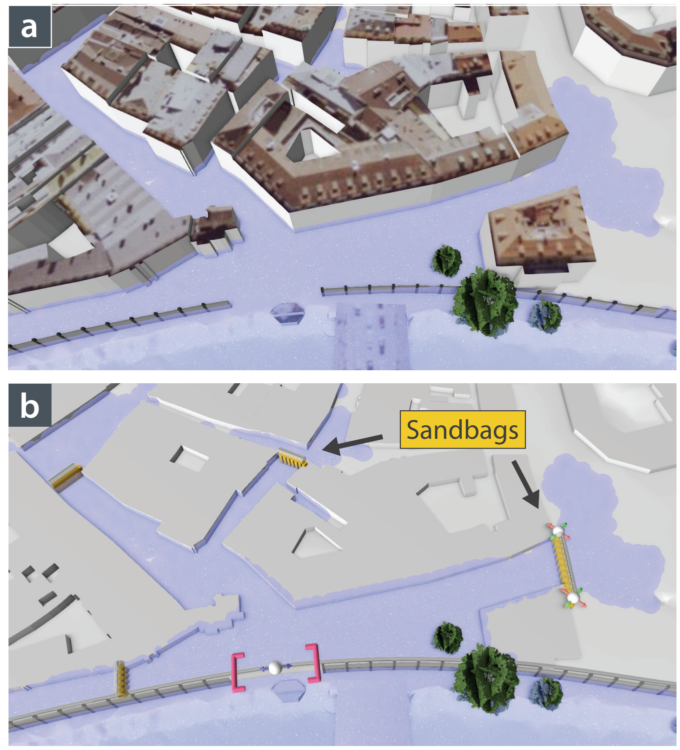


Fig. 5. Edit mode and sketching context. (a) Examination of a breach scenario in the simulation mode. The system performs simulation tasks while the user navigates across time. (b) Design of a breach closure in the edit mode. The system fetches a previous simulation state which displays the water spread in a simplified sketching context.

poses, they require that the appearance of the rendering should look as realistic as possible. This allows the results to be used to inform the decision makers and those likely to be affected by the flooding. Unfortunately, when the rendering of the scenario is realistic, many visual details that otherwise enhance and enrich the user's experience may distract the user when drawing. Examining the results of the simulation and setting it up have different rendering requirements.

We solve this issue by introducing two modes of operation. The simulation mode (Figure 5a) is the normal World Lines mode of operation. The user can navigate across scenarios and time, examining results with the simulation being performed as necessary. The edit mode (Figure 5b) differs from the simulation mode in two distinct ways. The first difference is related to the simulation mechanism. Creating actions through branching requires that the user navigates to the time where the action is to be introduced. In simulation mode, a simulation executes to prepare the data related to this time step. The edit mode attempts to use the last available simulation state as the context for sketching instead. By doing so, the user can sketch entire decision trees, and simulate them later, preferably in an overnight simulation.

The other difference is visual. The edit mode offers a simpler view of the scenario. Textures and decorations are removed, and to allow easier drawing of barriers in tight spaces, buildings are flattened. These changes are not preset, and can be customized to adapt the edit mode to the scenario. The goal is to have a representation where it is possible to clearly identify what can be manipulated, and to easily sketch new primitives and add new actions. This allows actions and primitives to be clearly visible in edit mode.

3.4 Handles

In the previous sections we have shown how primitives and actions are introduced through the interface, and how the rendering is adjusted to make them as visible as possible. However, the first placement of these objects is not necessarily the right one. The user has to be able to alter their properties, and to do so easily, intuitively, and consistently. To achieve this goal, we introduce different types of handles. A handle is a rendered object that represents a single property of an action or primitive, and allows the user to manipulate it using dragging. One example of a handle is a control point on a water-source object, which can be seen in Figure 6a. The control point shows the position of the source, which can be manipulated by simply dragging it from one location to another one. The same image shows five more handles - one arrow handle that can be dragged up and down, altering the rate of water emission, and four cylinder handles that control the size of the source when moved towards or from its center.

As can be seen from the examples, handles differ in the way they can be moved. These movement constraints are a necessity as two-dimensional screen-space movement cannot be translated into a three-dimensional one without additional information. We use three distinct movement constraints for handles:

- Two-dimensional movement on a surface,
- One-dimensional movement along a line,
- Movement along a spline.

The appearance of handles is designed to reflect a movement constraint, with the goal of the user understanding how to alter a property upon seeing the handle for the first time. However, the restriction alone does not describe the type of the handle. Consider the example of an emission-rate handle and a height handle. Both control one number, and can thus be realized by a one-dimensional movement constraint. However, the height has a natural meaning and rendering context and can be displayed as a point only, while the emission rate must be shown as the length of an arrow. In the end, the appearance of a handle is a function of the property it represents. In some cases, this property can change, resulting in a different handle appearance. When manipulating three-dimensional positions using handles, this is the case. The handle allows for two-dimensional movements when observed from above, and one-dimensional ones when observed from the side. This perspective-aware manipulation concept is implemented for the control points of a spline. This way, the user can modify the local height of a protection wall when looking at it from the side (Figure 6b).

Employing consistent representations of handles has an additional advantage. For example, the breach action does not add objects into the simulation. Instead, it simply punches a hole into an existing barrier. Given that the breached barrier might be confused for two individual barriers, we use handles to accentuate the breach and to manipulate it. One positional handle is created, accompanied by two dimensional handles. These determine the position and width of the breach, and show it to the user in an understandable way (Figure 6c).

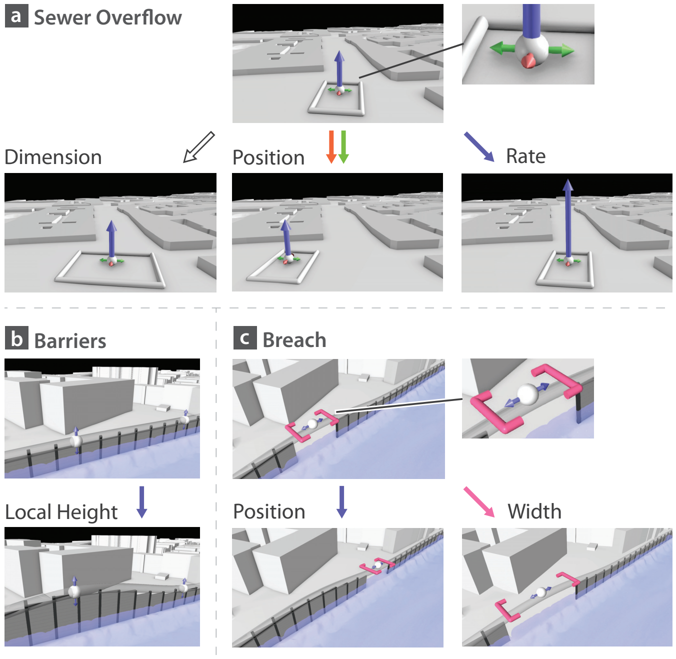


Fig. 6. Handles to alter action parameters. (a) Water-source handles. (b) Control points on a spline to adjust the local height of a barrier. (c) Breach handles with movement constraints along a spline.

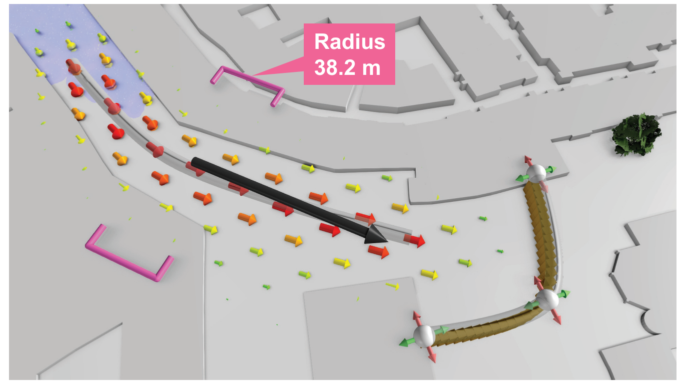


Fig. 7. Force field to test the stability of a barrier. The pink handles control the radius of influence, the black arrow the strength of the force field. An arrow plot visualizes the resulting force field. The floating label reveals the exact radius of influence.

4 Feedback Visualization

One of the goals of this paper is to make providing inputs via sketching less prone to interpretation errors. It is possible to rephrase this goal: to allow the user to introduce the change he or she has in mind. If the user knows the exact numeric value that should be set as the parameter, a sketching solution cannot outperform a conventional input method. But if the user does not have this knowledge, displaying additional information as interaction is taking place can make the setup of parameters easier. This is the idea of feedback visualization - to create a visual representation of input changes that supports the user in reaching the desired simulation setup.

4.1 Handles as feedback visualizations

The simplest feedback visualizations present in the system have already been described in the previous section. Both spline sketching and handles satisfy the criteria we outlined for feedback visualizations. They change in response to the user's input, showing a new state resulting from the changes. However, not all handles can be considered to be effective feedback visualizations. If the user is altering parameters with a spatial correspondence, handles do supply useful additional information. The object changes shape or position as a part of a rendering, and the visual context can allow for a better understanding of relative sizes or distances. The feedback is valuable when altering mobile walls or other actions assigned to a shape primitive.

When manipulating more abstract parameters, the user can still benefit from the handle representations, but this way of examining them is less efficient than being shown a numerical value. Additionally, the user may want to examine successfully placed handles and retrieve the exact numerical values to be able to communicate them to others, or record them. For this purpose, the handles support a label-based feedback visualization that can be turned on as necessary. Whenever a handle is manipulated, the semantic information encoded within can be used to create a label describing the handle's properties. This label can be rendered on-screen to provide numerical information regarding the manipulated properties, as can be seen in Figure 7.

The labels are useful, but their continued presence may contribute to visual clutter. To remedy this, they can be disabled completely, and when enabled, they appear only as necessary. A label is shown when the user is hovering the mouse over a handle, or manipulating a handle. Should the user want a label to persist, clicking on a handle with a special tool toggles its persistence. The positioning of labels is also important, as they should not interfere with the manipulation of objects and obscure important details. The semantic information encoded within the handle allows us to approximate where the handle may be moved, and place the label away from these zones.

While the labels do offer concrete numeric values, exactly how they affect the simulation may not be obvious from the values themselves. To remedy this, other feedback visualizations have to be used.

4.2 Local effect visualizations

An issue with some actions introduced into the system is that their effect on the simulation is difficult to infer from the sketch used. One example of such an action is a directed force field. Force fields are actions that can be assigned to a spline, and which are translated into additional forces acting upon simulation objects. They are useful in two cases: when employed to imitate natural influences such as wind, and when used to introduce artificial influences called pseudoforces into the simulation. These allow an expert to exert some influence over the simulation by inducing a behavior to occur. For example, an expert may wish to model the collision of floating debris with a barrier to test its stability, but may have no way of creating floating objects within the simulation. To approximate such a behavior, the expert can produce a force acting upon a barrier (Figure 7).

Once a force field is assigned to a spline, however, its extent, strength, and orientation are visible to the user only through handles, which do not show how the force affects the space around the spline. To allow the user insight into the effects of the sketch, the task is then to visualize the vector field produced by the force field. In order to realize this, methods from flow visualization can be used. An overview is provided by Weiskopf and Erlebacher [29]. For our purpose, we base the feedback visualization on a point-based direct visualization where visual representations for points are created. We choose to map the information to glyphs shaped like arrows and render them in the 3D view in order to visualize the characteristics of the force field. A so-called arrow plot is easy to understand and a clear and simple, but effective, way to present vector fields (Figure 7). A transfer function is used to color the glyphs according to the magnitude of the forces, allowing an insight into the exact values used.

We call feedback visualizations like these local effect visualizations. Any type of visualization or representation can be used to produce them, but their primary characteristic is that they appear in certain contexts only. If no force field is present, or the user navigates to a scenario without one, no arrow plot is shown. The visualization of the effect is local both in the spatial sense, as the visualizations are related to a certain spatial property, and in the causal sense, as they never appear if a relevant action is not introduced. Local effect visualizations

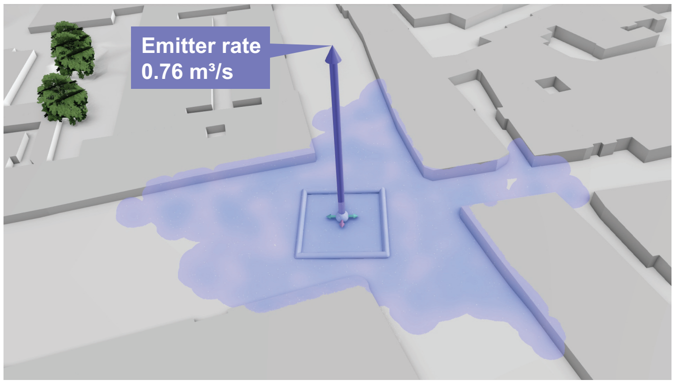


Fig. 8. Sewer-overflow feedback. A preview simulation provides hints on the resulting water expansion during customization of the emission rate. The floating label reveals the exact rate of water flow.

allow us to integrate many different visualization types into the interface to help the user understand parameters, while not suffering from visual clutter or information overload.

4.3 Preview simulations

Although visualizations such as arrow plots are useful for showing more information about introduced parameter changes, they do not aid the user directly in understanding how these changes affect the simulation. This is to be expected - after all, if such knowledge could be gained automatically without performing a simulation, the simulation itself would not be necessary. However, for the manipulation of more abstract parameters, it is useful to provide the user with at least a hint of what might be expected from a parameter change. To achieve this, we use preview simulations when manipulating handles.

A preview simulation is a simulation of lower accuracy that can be performed quickly enough to give the user additional information when manipulating a handle. One example is the water-source action. If the user places a water-source action, its effects are not seen immediately. The quantity of the water produced is determined by the emission rate, but it is hard to visualize the extent the produced water will have, or how it will flow upon being produced. To show these effects, we perform a lower-accuracy simulation initialized by the current state of the system that runs for a certain number of time steps, and integrate its results into the rendering (Figure 8). This allows the user a glimpse of what might happen when the change is introduced. The information gained can be used to alter the action further, until the desired result is achieved. Of course, even with a much faster simulation, it is unreasonable to expect that updates can happen at interactive rates. The preview simulation is started after the user has stopped manipulation by dragging, and after a slight delay it produces updates. The preview simulation is also a form of a local effect visualization, meaning that once the user has navigated away from the scenario in which a water source was introduced, its effects on the rendering disappear.

5 Sketching Ensembles

While feedback visualizations to some extent can help the user make better parameter choices, they cannot provide more than a hint in exploring the input-parameter space. To explore the complex effects of changing an input parameter, we use ensemble simulations. The specification of these is usually done by selecting a numerical parameter, and choosing a range of values that it can take. As such interaction is exactly what we have aimed to avoid by introducing a sketching interface, we provide substitute mechanisms for introducing ensembles.

5.1 Handle Ensemblization

The basic mechanism for introducing ensemble simulations into our system is based on a property of the handles described earlier. Every handle, given a certain perspective, controls exactly one simulation parameter. The parameter can be a scalar or a vector (such as position), but for the time being we will focus on scalar parameters only. To exploit this relationship between handles and properties, we introduce a new tool into the interface.

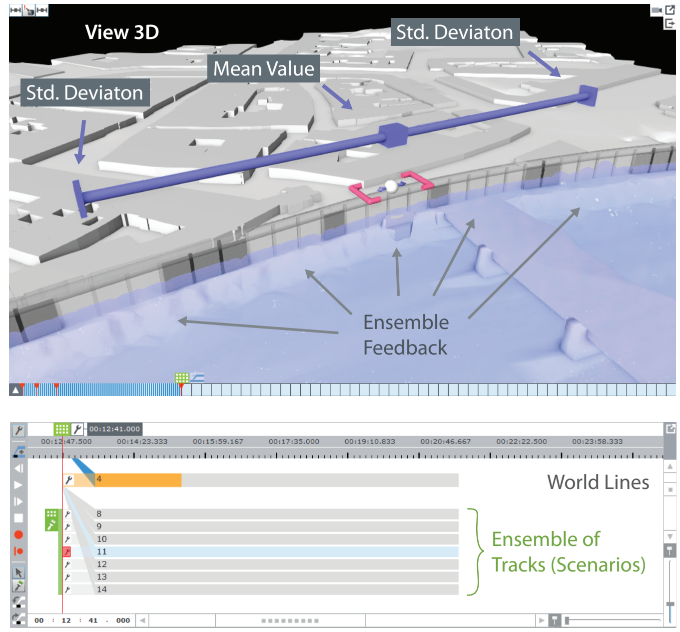


Fig. 9. Ensemblization according to breach position. (Bottom) The process auto-creates a group of tracks, each of which represents a breach location. (Top) The distribution of breach positions is modified via the ensemble handle (blue). An immediate feedback aggregation displays the resulting values on the barrier.

The *ensemblization* tool allows the user to specify an ensemble simulation by simply clicking on a handle representing the property whose input uncertainty is to be explored. After the click is made, the user can see two changes happen. Based on the description of the property present in the handle, a default distribution of values is generated and delivered to World Lines. World Lines respond by creating a track representing an alternative scenario for every value in the distribution, allowing a simulation to be run for each one of them. Figure 9 shows a screenshot of the created tracks after ensemblization of the breach position. The other change manifests itself in the rendering. A special ensemble handle that allows the user to control the ensemble is created. As can be seen in Figure 9 and Figure 10, the relationship between the parent handle and the ensemble handle is visible in the coloring and the positioning of the ensemble handle.

The appearance of the ensemble handle is based on the box plot glyph [19]. This allows us to show the distribution in a way end users have seen or can quickly comprehend. As in the original box plot, we show the average value and the quartiles, however, instead of showing the extremes, we opt to show the standard deviation of the distribution instead. The feedback visualizations to be described later already show the extremes, and the standard deviation allows the same degree of control while taking up a smaller amount of screen space. The user can manipulate the ensemble handle by moving the central part of the box plot, changing the mean value, or by extending its sides, changing the deviation. The handle inherits the movement constraints of the original handle. For example, if the position of a spline breach is ensemblized, the user can slide the mean value across the spline in the very same way the original breach position was manipulated (Figure 9). Another example is shown in Figure 11a. A black ensemble handle is moved along a level gauge to setup a distribution of water levels.

For the time being, we limit the use of this mechanism to creating single-parameter ensembles. Positions and the like can be ensemblized given a one-dimensional movement constraint based on a spline, but, e.g., ensemblizing the water source dimensions is not yet supported. We have not implemented multi-dimensional ensembles yet due to a

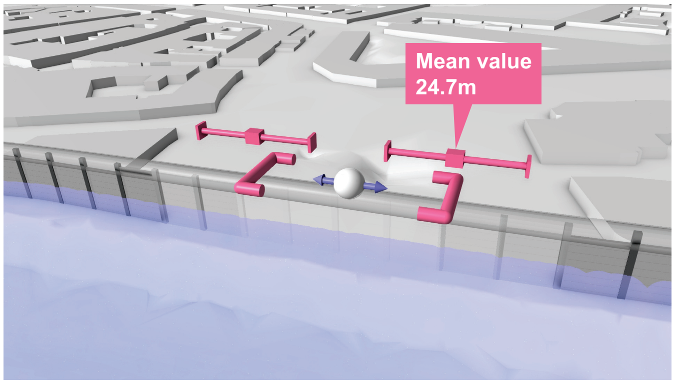


Fig. 10. Ensemblization according to breach width. The ensemble handle receives the coloring of the handle which is associated with the breach width.

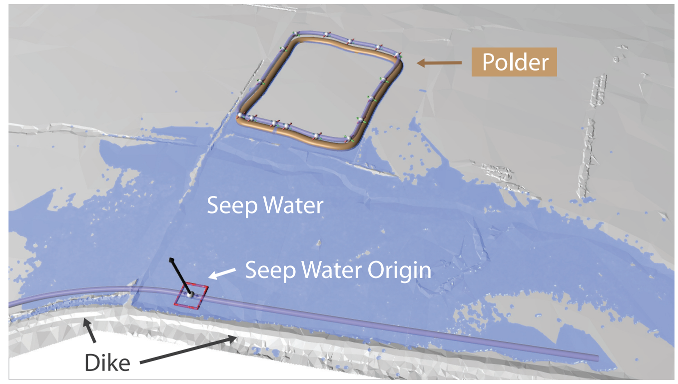


Fig. 12. Rural scenario. User-controlled seep water reaches the surface behind a dike and floods the landscape. An area is protected by a sketched polder.

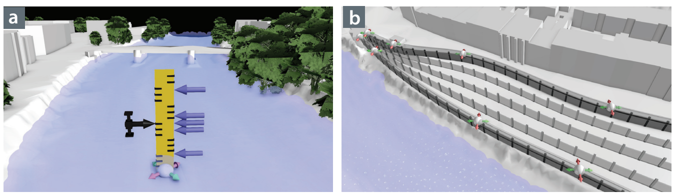


Fig. 11. (a) Ensemblization according to the water level. For each water level contained in the ensemble, a blue arrow is placed on the right side of the yellow gauge. (b) Interpolation of two mobile walls to investigate a range of alternative barrier arrangements.

lack of World Lines support, but there is no reason why the ensemblization mechanism would not be useful here. Multiple ensemble handles can coexist in the rendering, each specifying a single dimension of a range, or a component of a vector. The absence of these features has not proven to be an issue for our evaluation scenarios, but we plan on extending World Lines to allow for them to be used.

5.2 Spline Ensemblization

While the previous section explained how simple parameters can be ensemblized, we would like to interpolate more complex parameters as well. One requirement in particular is the interpolation of barriers, to test various placement positions and heights. As barriers are splines, this requires a way of specifying a spline ensemble. We allow the user two ways to do so. The first way is a manual ensemble creation. The World Lines can be instructed to enter a manual ensemble mode. In this mode, the user can sketch a change, such as a spline, and press a button to save the change within the track. Another track ready for additional sketching is created, and the user can continue until an ensemble has been specified. While this input method would be cumbersome using standard input interfaces, a sketching approach allows it to be performed by simply drawing multiple lines on the rendering.

The second method is more similar to the ensemblization procedure, using the same tool for specification. The process, shown in Figure 11b, begins with the user drawing two barriers - the limits of the barrier range to be produced. The ensemblization tool is then used to select the splines in any order, resulting in the generation of multiple splines between the original two. The splines are generated by interpolating the shapes they consist of. We have found that a simple pairwise interpolation of spline control points suffices for our purposes.

5.3 Ensemble Feedback Visualizations

As all the other changes that can be introduced by sketching, the created ensembles have accompanying feedback visualizations. Unlike the ensemblization mechanism, these are not generic and must be designed for each parameter separately. This is because they are based on the feedback visualizations of individual parameters, which can vary greatly. However, the guiding principle of all of them is based on data aggregation. Some of these visualizations are shown in Figures 9- 11.

The first example, which is given in Figure 9, shows the feedback visualization of a barrier breach-position ensemble. Barrier breaches do not usually require special feedback visualizations, as they manifest themselves within the rendering of the breached barrier. However, when multiple breaches are present, some might overlap, creating an illusion of breaches differing in width as well as position. To counter this, we render each breach with a measure of transparency. If breaches are overlapping, the overlapping area is rendered more transparently than the non-overlapping ones. The mechanism can be controlled by a transfer function that takes the number of overlapping breaches as a parameter, and outputs an alpha value. The user can see all the values at the same time, and alter the ensemble accordingly.

Another example is spline aggregation. As seen in Figure 11b, all of the splines are rendered at the same time. To allow for better visibility, all but the two that are the basis for the interpolation are rendered transparently. As the user adjusts one of the interpolated splines, the entire feedback visualization updates as well.

In summary, while the ensemble-feedback visualizations do differ from case to case, they follow a similar pattern of attempting to show multiple values at the same time. They are also based on a familiar single-parameter feedback visualization, allowing the user to understand them more easily. The ensemble-feedback visualization is also a local effect visualization - it is shown only in the ensemble tracks, and can be turned off to depict individual feedback visualizations. If the ensemble visualization does not contain enough information, this enables a detailed examination of ensemble members.

6 Evaluation

To find out if the created interface succeeded at the goals we set out to achieve, we performed an extensive evaluation.

6.1 People

Finding the right set of people ready to participate in an evaluation was more difficult than expected. The number of experts working in this field is limited, and we wanted to use people who had not been exposed to our system before. Our difficulties were resolved by a collaboration with the flood protection centre in Cologne, Germany. The city of Cologne had encountered two devastating floods in 1995 and 1997. These events prompted the development of multiple flood defenses, managed by a flood protection agency. After these floods the city had not encountered a flood strong enough to seriously threaten their defenses. To test whether they are effective, the agency employs simulations. However, they do so for a small number of possible risks, mostly barrier breaches. The results are static risk and flooding maps that are useful, but not flexible enough to account for changes in dangers. Due to the costs associated with hiring engineers to perform

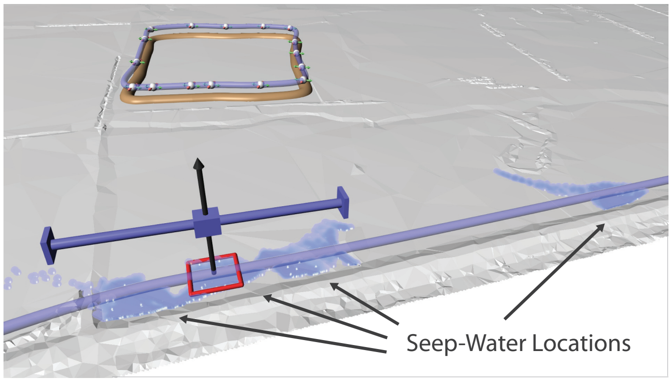


Fig. 13. Ensemble with respect to seep-water locations. Multiple preview simulations are aggregated to provide immediate feedback.

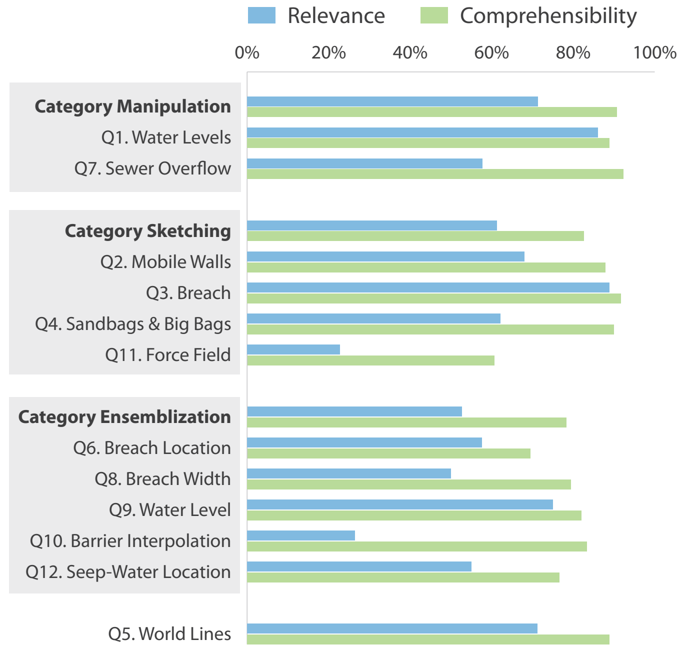


Fig. 14. Evaluation results per question and per question category. The ratings are averaged among the participants.

more simulations, they were open to the idea of evaluating a solution that could help them cut costs and improve protection measures.

All in all, 12 people from Cologne participated in the evaluation. They work on many different functions but are all involved in flood protection. These include the representatives of the flooding protection agency, public services such as firemen, and the local authorities in charge of traffic control and evacuation plans. To allow for some differentiation, we divided them in two categories: Flood management and response personnel. To make the evaluation more comprehensive, we enlisted a flood-protection consultant experienced with simulations. This person forms a third category: Consulting engineer.

6.2 Evaluation methodology

Although we had secured the cooperation of the Cologne experts, the evaluation was to be done as a part of a larger meeting where experts discussed various topics from the area of flood protection. While an ideal way to evaluate the interface would have been to let the users interact with it, we had to settle for showing recordings of various use cases, and gathering the users' opinions with a questionnaire. To at least partially remedy this, we organized an additional session with one of the users who had participated in the questionnaire. By having him interact with the system, complete tasks related to questions, and then reevaluate his answers, we hoped to gain an idea as to how accurate our original evaluation was. The second session also provided an opportunity to evaluate an additionally added requested feature, namely the floating feedback-visualization numerical labels.

Two scenarios were shown - one involving a flood happening in an urban setting, with the Cologne protections available. All the figures present earlier in the text were taken from this scenario. The second one explores the effects of variations in seep-water locations in a rural setting, where the user can deploy polders to protect populated areas. The second scenario can be seen in Figures 12 and 13. Both scenarios were modeled using real-world GIS data, and are based on scenarios the experts we were collaborating with had encountered. While the simulation has not been verified to be physically accurate, experts have concluded that it behaves realistically enough for our purposes.

The evaluation proceeded with a member of our team presenting each recording with a short description of what was happening, and then pausing to let the users respond to two questions related to the current part of the questionnaire. The answers were to be given as grades from 1 to 4, with the option to add written comments. After a minute had passed, a verbal discussion followed to gather more information. The questions used and answers offered were:

- How relevant is this feature to your areas of responsibility?
(1 - Not relevant, 2 - Rather irrelevant, 3 - Rather relevant, 4 - Relevant)

- How comprehensible is this feature?
(1 - Concept is comprehensible and intuitive, 2 - Concept is comprehensible but requires training, 3 - Concept is hard to understand and not intuitive, 4 - Concept is unclear to me)

The individual questions (Q1-Q12) are listed and categorized in Figure 14. The video, the questionnaire, and the gathered data in CSV format are available as accompanying material for the paper.

6.3 Results

The evaluation yielded two different but complementary types of data. The quantitative data set is based on the questionnaire and consists of 24 data columns, two for each presented feature. One column describes the relevance of a feature to a user, and the other how well the user comprehends a feature. To even out the inconsistent scaling, we normalized the values to a range between zero and one, with one representing a very relevant or a well-understood feature. The figure 14 shows the mean ratings of the features. The questions (Q1-Q12) are grouped according to the type of functionality they represent, and within the groups they are sorted by order of presentation.

As was to be expected, the more basic features have been better received by the users. Manipulation can be seen to be the most intuitive, followed by World Lines navigation, the sketching mechanisms, and finally the ensembles. Regardless of the ordering coinciding with complexity, almost all of the evaluated features have been received quite well. The force field and the breach location ensemblization received grades suggesting that they are intuitive, albeit a certain amount of training might be required. However, the comprehension score of the breach location ensemblization might be misleading. All of the other ensemblization features received higher grades, including another breach-related ensemble. As the breach location was the first ensemble to be shown to the users, we assume that multiple ensembles needed to be shown before the users understood the concept.

The relevance of the features follows the same ordering as the comprehension, but with lower values and greater individual score variation. This was to be expected, as the users have varying responsibilities, leading them to consider features more or less relevant. To explore whether there exists a connection between how well a user understands a feature and how relevant it is for him/her, we analyzed the correlation between the two variables for each feature. Our results show that there is a non-existent to weak correlation (-0.17 to +0.21) for the better understood features such as manipulation and navigation.

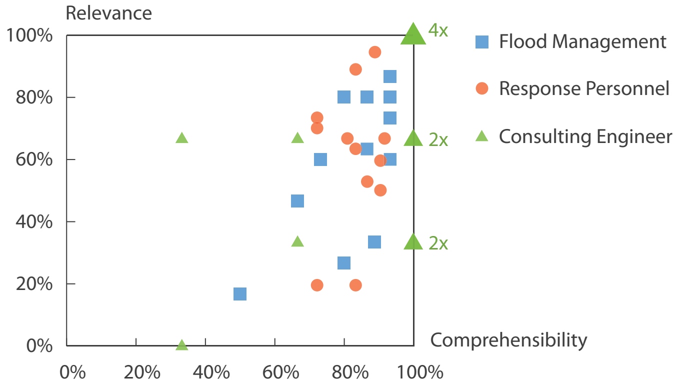


Fig. 15. Evaluation results per user category. Each data point represents the average rating of a question.

For those judged to be less intuitive, a medium-strength correlation (+0.37 to +0.67) exists. While a connection exists, based on this data alone we cannot say that understanding is caused by a user's expertise.

Finally, we tried to see whether there is a difference in how users perceive features based on the category that we placed them in. Our hypothesis was that consulting engineers would understand features the best, but might not see them as useful as the other groups because they can already perform a simulation setup without issues. As flood management staff often performs more abstract tasks such as planning and considering alternatives, we expected them to score better as far as intuitiveness was concerned. We also assumed the response personnel would have a lower average relevance score, given their expertise in particular areas only, and perhaps a weaker understanding of more advanced features. Figure 15 shows that we were at least partially wrong. The consulting engineer fulfilled our expectations, reporting better marks for intuitiveness, but lower relevance values. However, the flood management and response personnel gave very similar grades to features, to the point where we could find no significant difference between the two. An explanation for this result might lie in the fact that both groups contained a similar proportion of engineers and technical personnel versus administrative personnel.

While examining the data set did reveal enough to consider our interface successful, the conversations with the users that occurred after the evaluation helped us to get a more complete impression. For example, one of the comments regarding the breach-position ensemble revealed that a lack of handles in the feedback visualization might be the reason why it was considered less intuitive. Breaches are made visible by handles, and a breach-position ensemble does not contain multiple handles for every breach present. Likewise, a conversation about the force-fields feature and pseudo forces in general revealed that users found them interesting for other purposes, e.g., simulating a lower water pressure due to controlled flooding. As they did so, their opinion and comprehension of the feature improved. This does suggest that the relevance of a feature to the user positively affects the understanding of the feature.

Our second evaluation, performed with a single expert, focused on letting the user interact with the system and seeing if the feedback differed from the original evaluation. The user had previously completed the questionnaire without having interacted with the system, and had given the system good scores. The second evaluation involved a three-hour long interaction session in which a separate simulation setup task was performed for every question in the questionnaire. We provided the minimal amount of guidance necessary, answering the user's questions and giving him an overview of the interface.

After every task, the user was asked to grade the question again after using the system. The reevaluated questionnaire revealed that the user's grades had not changed for any of the questions. Although the evaluation was longer than expected, most of the time was spent discussing various features and possible applications. The user repeatedly stated that the system allows him to setup simulations much quicker than any of the tools he used before. The biggest complaint that surfaced only when using the system was that the floating labels cannot be used to enter numerical values. While he appreciated the sketching functionality, he felt that a hybrid mode in which it would be possible to also use numerical inputs would increase his speed even further.

Overall, the users found the biggest advantage of the system to be the ability to greatly expand the number of simulations they can perform without requiring an engineer's assistance. This lowers the cost of a simulation and allows a large number of pre-simulated scenarios to be available during crises. Should an incident similar enough to a prepared scenario occur, the simulated data allows for easier evacuation and defense planning. Apart from that, having a tool in which incidents could be introduced easily allows for the modelling of hazards different from barrier breaches, e.g., canal or metro floodings.

7 Implementation

The sketching interface was implemented into Visdom [3], a steerable integrated visualization system. Doing so gave us access to the various features used throughout the paper, such as the advanced rendering functionalities, a simulation module, different visualization types, and advanced interfaces such as World Lines. The framework allowed us to build a powerful system by reusing functionality. As the simulation engine itself is a modular component in the Visdom data-flow, we also allow for different simulation types and modules to be used. We plan on using this modularity to explore different flood-simulation models.

One of the more interesting aspects of Visdom is the mobile client. Visdom is implemented as a server-client system with a thin client running on various mobile devices. All the simulation and rendering is performed on the server, leaving the client to handle user interactions only, making the user interface the most important part of the mobile client. As today's mobile devices use touch screens, in the future we hope to apply our sketching interface to such devices.

8 Conclusion

In this paper, we set out to create a new type of user interface for simulation steering - simple enough for a non-expert to use, yet powerful enough to compete with current interfaces. The evaluation suggests we were successful. Even the advanced features such as ensemblization were evaluated to be both intuitive and relevant. While it would have been useful to have more users interact with the system, the second evaluation indicates that a lack of interaction did not affect the results.

There are multiple ways to proceed. One involves a reworking of World Lines to allow for multi-dimensional ensembles. The modularity of the system allows us to integrate other types of simulations as well. We are interested in investigating whether our techniques work for simulations in the area of traffic management or climate research. Another important direction involves resource management and operations research. The current system offers no information about constraints related to available resources. Allowing the users to create plans and monitor resources, while still using functionality such as ensemble simulations, is a promising route for future work.

The very positive feedback we received suggests that improved interfaces can make simulation steering more accessible to end users. Such small changes could pave the way for a more widespread use of simulations in decision making. While our system will require many more improvements and domain-specific input before it is useable in such a context, we believe that it is a research direction worth pursuing. After all, in an age where catastrophic events affect great numbers of people, even small improvements can save lives.

Acknowledgments

This work was supported by grants from the Austrian Science Fund (FWF): P 22542-N23 (Semantic Steering), and the Vienna Science and Technology Fund (WWTF) project Scale-VS (No. ICT08-040). We would also like to thank the Flood Protection Centre of the Stadtentwässerungsbetriebe Cologne, Geoconsult, Michael Greiner, and Jürgen Komma.

REFERENCES

[1] ForcePAD - A tool for visualising the behavior of structures subjected to loading and boundary conditions. http://http://forcepad.sourceforge.net/ (last visited on February, 15th 2011).

[2] Sketching Uncertainty Video. http://visdom.at/videos/#RESEARCH/Sketching_Uncertainty/ (last visited on June, 27th 2012).

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

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

[5] S. Afzal, R. Maciejewski, and D. Ebert. Visual analytics decision support environment for epidemic modeling and response evaluation. *Proceedings IEEE Symposium on Visual Analytics Science and Technology (Vast)*, pages 191–200, 2011.

[6] G. L. Andrienko, N. V. Andrienko, P. Jankowski, D. A. Keim, M.-J. Kraak, A. M. MacEachren, and S. Wrobel. Geovisual analytics for spatial decision support: Setting the research agenda. *International Journal of Geographical Information Science*, 21(8):839–857, 2007.

[7] 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.

[8] C. S. Applegate, S. D. Laycock, and A. M. Day. A sketch-based system for highway design. In *Proceedings of the Eighth Eurographics Symposium on Sketch-Based Interfaces and Modeling*, SBIM '11, pages 55–62, New York, NY, USA, 2011. ACM.

[9] J. Barbič and J. Popović. Real-time control of physically based simulations using gentle forces. In *ACM SIGGRAPH Asia 2008 papers*, SIGGRAPH Asia '08, pages 163:1–163:10, New York, NY, USA, 2008. ACM.

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

[11] M. T. Cook and A. Agah. A survey of sketch-based 3-d modeling techniques. *Interacting with Computers*, 21(3):201–211, 2009.

[12] J. Davis, M. Agrawala, E. Chuang, Z. Popović, and D. Salesin. A sketching interface for articulated figure animation. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '03, pages 320–328, Aire-la-Ville, Switzerland, 2003.

[13] C. D. Gooijer, L. R. Wentholt, and E. Langkamp. FLIWAS: Flood information and warning system. In *Proceedings of the 5th Annual Mekong Flood Forum*, 2007.

[14] M. Gretzschel, R. Jüpner, M. Grafe, and R. Leiner. Application of flood management systems in Germany. In *Proceedings of the First European IAHR Congress*, 2010.

[15] Q. Gu and Z. Deng. Formation sketching: an approach to stylize groups in crowd simulation. In *Proceedings of Graphics Interface*, pages 1–8, 2011.

[16] J. Lindemann, G. Sandberg, and K. Olsson. An approach to teaching architectural and engineering students utilizing computational mechanics software ForcePAD. *ITcon Special Issue ICT Supported Learning in Architecture and Civil Engineering*, 9:219–228, 2004.

[17] M. Marjolein. Frameworks for flood event management. Technical report, *FLOODsiteConsortium*, 2008.

[18] J. McCrae and K. Singh. Sketch-based path design. In *Proceedings of Graphics Interface*, GI '09, pages 95–102, Toronto, Ont., Canada, Canada, 2009. Canadian Information Processing Society.

[19] R. McGill, J. W. Tukey, and W. A. Larsen. Variations of box plots. *The American Statistician*, 32(1):12–16, 1978.

[20] L. Olsen, F. F. Samavati, M. C. Sousa, and J. A. Jorge. Sketch-based modeling: A survey. *Computers & Graphics*, 33(1):85–103, February 2009.

[21] A. Pihuit, M. P. Cani, and O. Palombi. Sketch-based modeling of vascular systems: a first step towards interactive teaching of anatomy. In *Proceedings of the Seventh Sketch-Based Interfaces and Modeling Symposium*, SBIM '10, pages 151–158, Aire-la-Ville, Switzerland, 2010.

[22] M. Rob. Review of flood event management decision support systems. Technical Report T19-07-01, *FLOODsiteconsortium*, 2007.

[23] D. Schroeder, D. Coffey, and D. Keefe. Drawing with the flow: a sketch-based interface for illustrative visualization of 2d vector fields. In *Proceedings of the Seventh Sketch-Based Interfaces and Modeling Symposium*, SBIM '10, pages 49–56, Aire-la-Ville, Switzerland, 2010.

[24] H. J. Shin and T. Igarashi. Magic canvas: interactive design of a 3-d scene prototype from freehand sketches. In *Proceedings of Graphics Interface 2007*, GI '07, pages 63–70, New York, NY, USA, 2007. ACM.

[25] R. M. Smelik, T. Tutenel, K. J. de Kraker, and R. Bidarra. Interactive creation of virtual worlds using procedural sketching. In *Eurographics (Short papers)*, pages 29–32, 2010.

[26] L. Tateosian, H. Mitasova, B. Harmon, B. Fogleman, K. Weaver, and R. Harmon. TanGeoMS: tangible geospatial modeling system. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1605 – 1612, nov.-dec. 2010.

[27] 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.

[28] 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. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):1872–1881, 2011.

[29] D. Weiskopf and G. Erlebacher. Overview of flow visualization. In C. D. Hansen and C. R. Johnson, editors, *The Handbook of Visualization*, pages 261–278. Elsevier, Amsterdam, 2005.

[30] E. Wiese, J. H. Israel, A. Meyer, and S. Bongartz. Investigating the learnability of immersive free-hand sketching. In *Proceedings of the Seventh Sketch-Based Interfaces and Modeling Symposium*, SBIM '10, pages 135–142, Aire-la-Ville, Switzerland, Switzerland, 2010. Eurographics Association.

[31] B. Zhu, M. Iwata, R. Haraguchi, T. Ashihara, N. Umetani, T. Igarashi, and K. Nakazawa. Sketch-based dynamic illustration of fluid systems. *ACM Trans. Graph.*, 30:134:1–134:8, 2011.