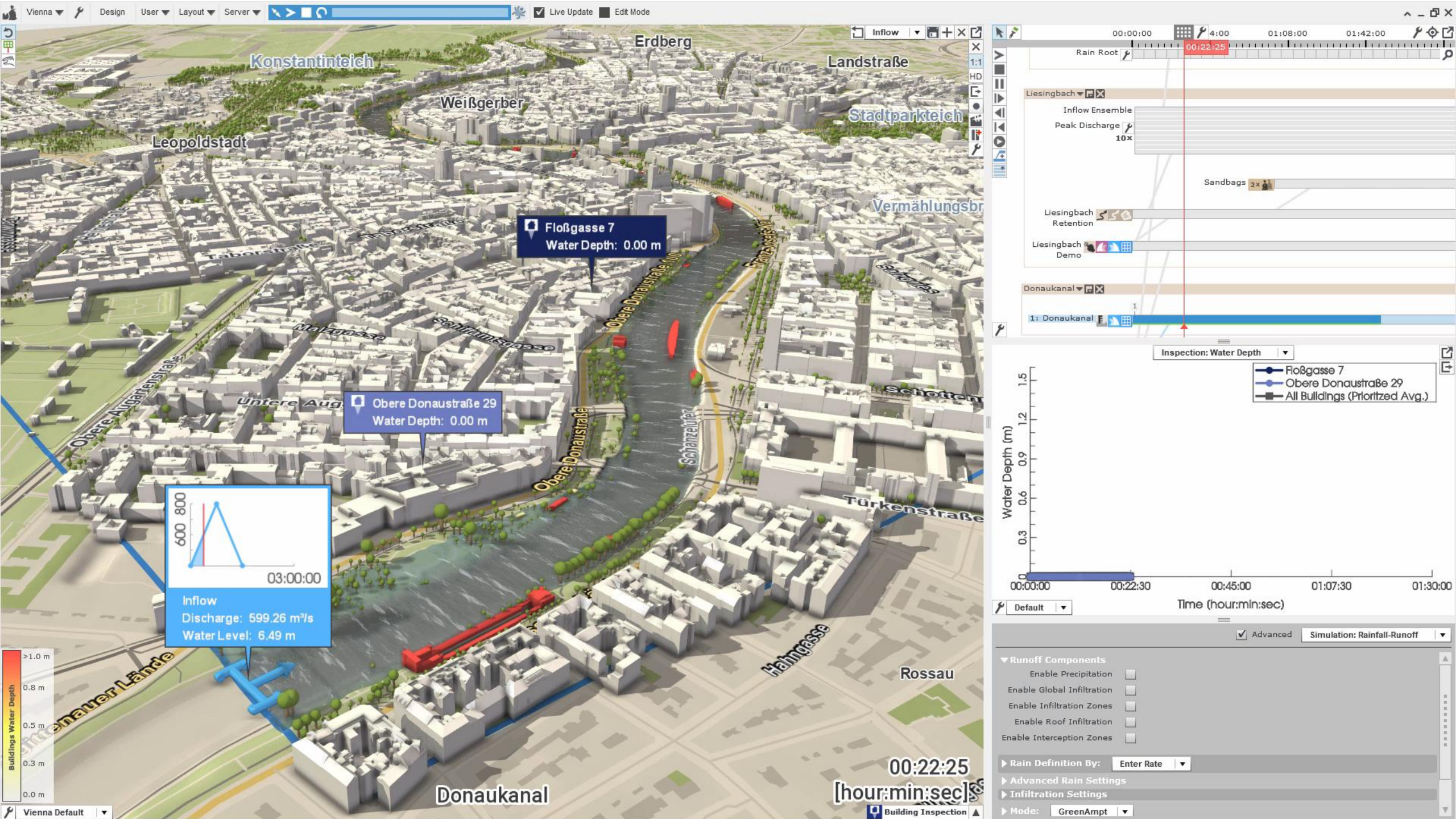
Dataflow in scenarify

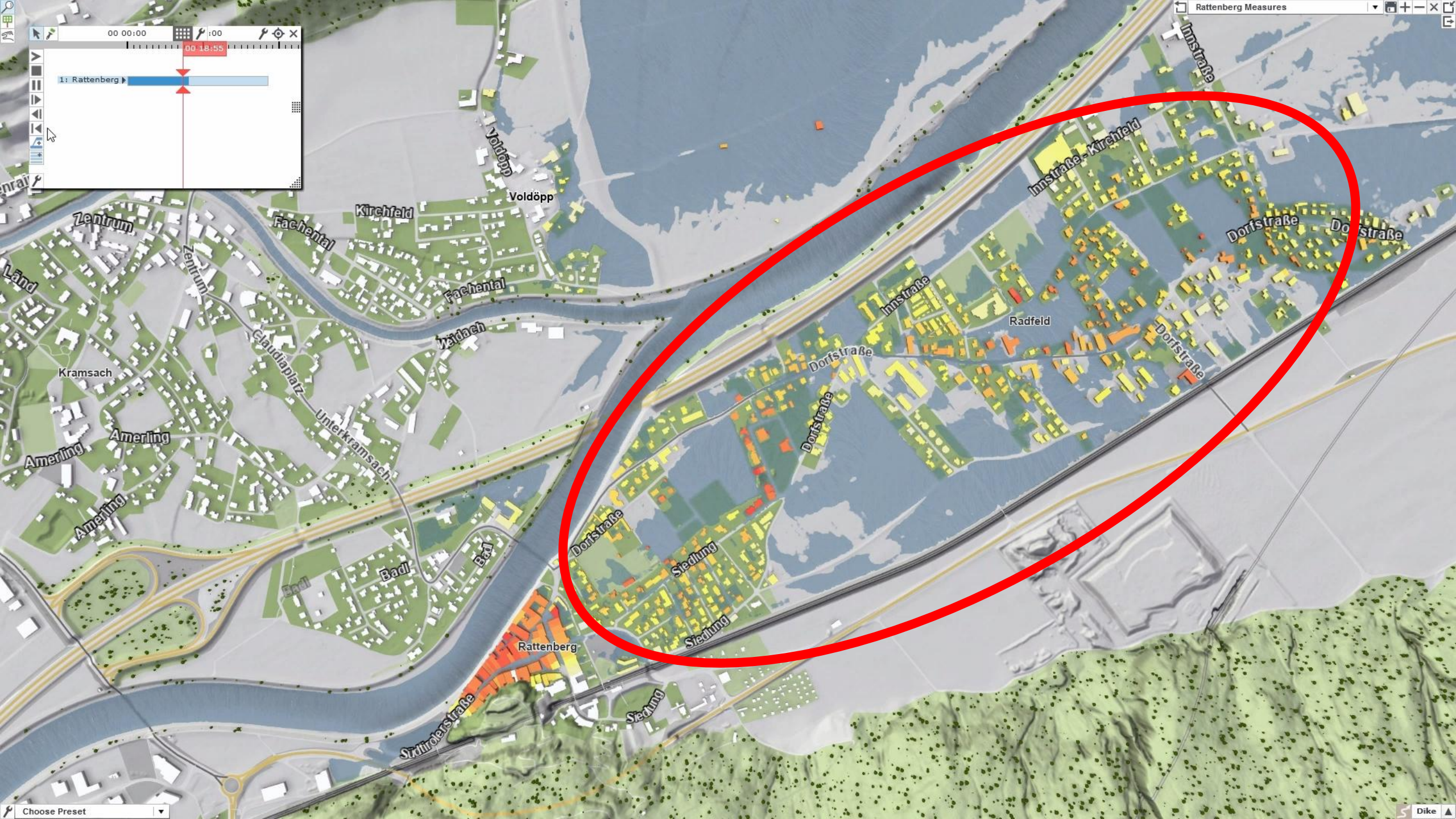
Design and Implementation of a Rendering Engine – 26.11.2024

Daniel Cornel – cornel@vrvis.at

vrvis


Vienna
Design
User
Layout
Server
Live Update
Edit Mode
Inflow
Erdberg
Konstantinteich
Landstraße
Weißgerber
Stadtparkteich
Leopoldstadt
Vermählungsbr
Floßgasse 7
Water Depth: 0.00 m
Obere Donaustraße 29
Water Depth: 0.00 m
Obere Donaustraße
Untere Aug
Malzgasse
Schiffamtsgasse
Schanzelufer
Türkenstraße
Hahngasse
Rossau
Donaukanal
Obere Augartenstraße
Inflow
Discharge: 599.26 m³/s
Water Level: 6.49 m
600 800
03:00:00
Buildings Water Depth
>1.0 m
0.8 m
0.5 m
0.3 m
0.0 m
00:22:25
[hour:min:sec]
Vienna Default
Building Inspection
00:00:00
4:00
01:08:00
01:42:00
00:22:25
Rain Root
Liesingbach
Inflow Ensemble
Peak Discharge
10×
Sandbags
2×
Liesingbach Retention
Liesingbach Demo
Donaukanal
1: Donaukanal
Inspection: Water Depth
Floßgasse 7
Obere Donaustraße 29
All Buildings (Prioritzed Avg.)
Water Depth (m)
0
0.3
0.6
0.9
1.2
1.5
00:00:00
00:22:30
00:45:00
01:07:30
01:30:00
Time (hour:min:sec)
Default
Advanced
Simulation: Rainfall-Runoff
Runoff Components
Enable Precipitation
Enable Global Infiltration
Enable Infiltration Zones
Enable Roof Infiltration
Enable Interception Zones
Rain Definition By:
Enter Rate
Advanced Rain Settings
Infiltration Settings
Mode:
GreenAmpt


Rattenberg Measures
00
01 02:00
00 00:15
1: Rattenberg
Voldöpp
Kirchfeld
Fachental
Zentrum
Länd
Kramsach
Claudiaplatz
Unterkramsach
Amerling
Badl
Waldach
Radfeld
Innstraße
Innstraße - Kirchfeld
Dorfstraße
Siedlung
Rattenberg
Südtirolerstraße
Choose Preset
Dike


Rattenberg Measures
00 00:00
00 18:55
1: Rattenberg
Innstraße
Innstraße - Kirchfeld
Voldöpp
Kirchfeld
Fachental
Zentrum
Länd
Kramsach
Claudiaplatz
Unterkramsach
Amerling
Badl
Waidach
Dorfstraße
Innstraße
Radfeld
Siedlung
Rattenberg
Südtirolerstraße
Choose Preset
Dike


Rattenberg Measures
00 00:00
:00
00 18:55
1: Rattenberg
Voldöpp
Kirchfeld
Fachental
Zentrum
Kramsach
Claudiaplatz
Unterkramsach
Amerling
Badl
Rattenberg
Südtirolerstraße
Siedlung
Dorfstraße
Innstraße
Innstraße - Kirchfeld
Radfeld
Choose Preset
Dike


Rattenberg Measures
00 00:00
01 02:00
1: Rattenberg
2: T46
Voldöpp
Kirchfeld
Fachental
Zentrum
Kramsach
Claudiaplatz
Unterkramsach
Amerling
Badl
Radfeld
Innstraße
Innstraße - Kirchfeld
Dorfstraße
Siedlung
Rattenberg
Südtirolerstraße
Choose Preset
Dike


Rattenberg Measures
00 00:00
01 02:00
1: Rattenberg
2: T46
Aqua Barrier
Delete All
Mobile barrier consisting of pallets and supports
Aqua Barrier Vertical
Delete All
Mobile barrier consisting of pallets and supports. The pallets are installed vertically
Aquariwa
Delete All
Mobile barrier consisting of fibre glass cylinders filled with ballast
Big Bags
Delete All
Large bags filled with sand or water
Concrete Wall
Delete All
A concrete flood wall (long-term construction measure)
Dike
Delete All
A terrain modification to create a natural dam or polder
Hill
Delete All
A terrain modification to create a hill
Mobile Walls
Delete All
Mobile barrier consisting of beams stacked between posts
Retention Basin
Delete All
A terrain modification to store flood water
River Bed
Delete All
A terrain modification to create a river or channel
Sandbags
Delete All
A flexible flood protection barrier
Sidewalk
Delete All
The center line of a sidewalk to be created
Sidewalk around Buildings
Delete All
Select buildings to auto-create a sidewalk around them within the drawn polygon
Dorfstraße
Unterkramsach
Badl
Siedlung
Choose Preset
Buildings
Inspection
Protection
Setup
Sewer
Synth
Water


Voldöpp
Kirchfeld
Kirchfeld
Fachental
Fachental
Fachental
Zentrum
Zentrum
Länd
Claudiaplatz
Weidach
Kramsach
Kuglgasse
Unterkramsach
Amerling
Amerling
Amerling
Badl
Badl
Badl
Badl
Dorfstraße
Dorfstraße
Dorfstraße
Dorfstraße
Dorfstraße
Innstraße
Innstraße - Kirchfeld
Radfeld
Siedlung
Siedlung
Siedlung
Siedlung
Rattenberg
Südtirolerstraße


Mobile Walls Posts: 227
Mobile Walls Beams: 1415
Mobile Walls Posts: 14
Mobile Walls Beams: 115
Sandbags: 1867
Voldöpp
Kirchfeld
Fachental
Zentrum
Weidach
Kramsach
Kuglgasse
Claudiaplatz
Unterkramsach
Amerling
Badl
Rattenberg
Südtirolerstraße
Dorfstraße
Siedlung
Radfeld
Innstraße


Mobile Walls Posts: 227
Mobile Walls Beams: 1415
Mobile Walls Posts: 14
Mobile Walls Beams: 115
Sandbags: 1867
Voldöpp
Kirchfeld
Fachental
Zentrum
Kramsach
Amerling
Kuglgasse
Claudiaplatz
Weidach
Badl
Rattenberg
Siedlung
Dorfstraße
Innstraße
Radfeld
Südtirolerstraße

Concrete Wall — Delete All
A concrete flood wall (long-term construction measure)

Dike — Delete All
A terrain modification to create a natural dam or polder

Hill — Delete All
A terrain modification to create a hill

Mobile Walls — Delete All
Mobile barrier consisting of beams stacked between posts

Retention Basin — Delete All
A terrain modification to store flood water

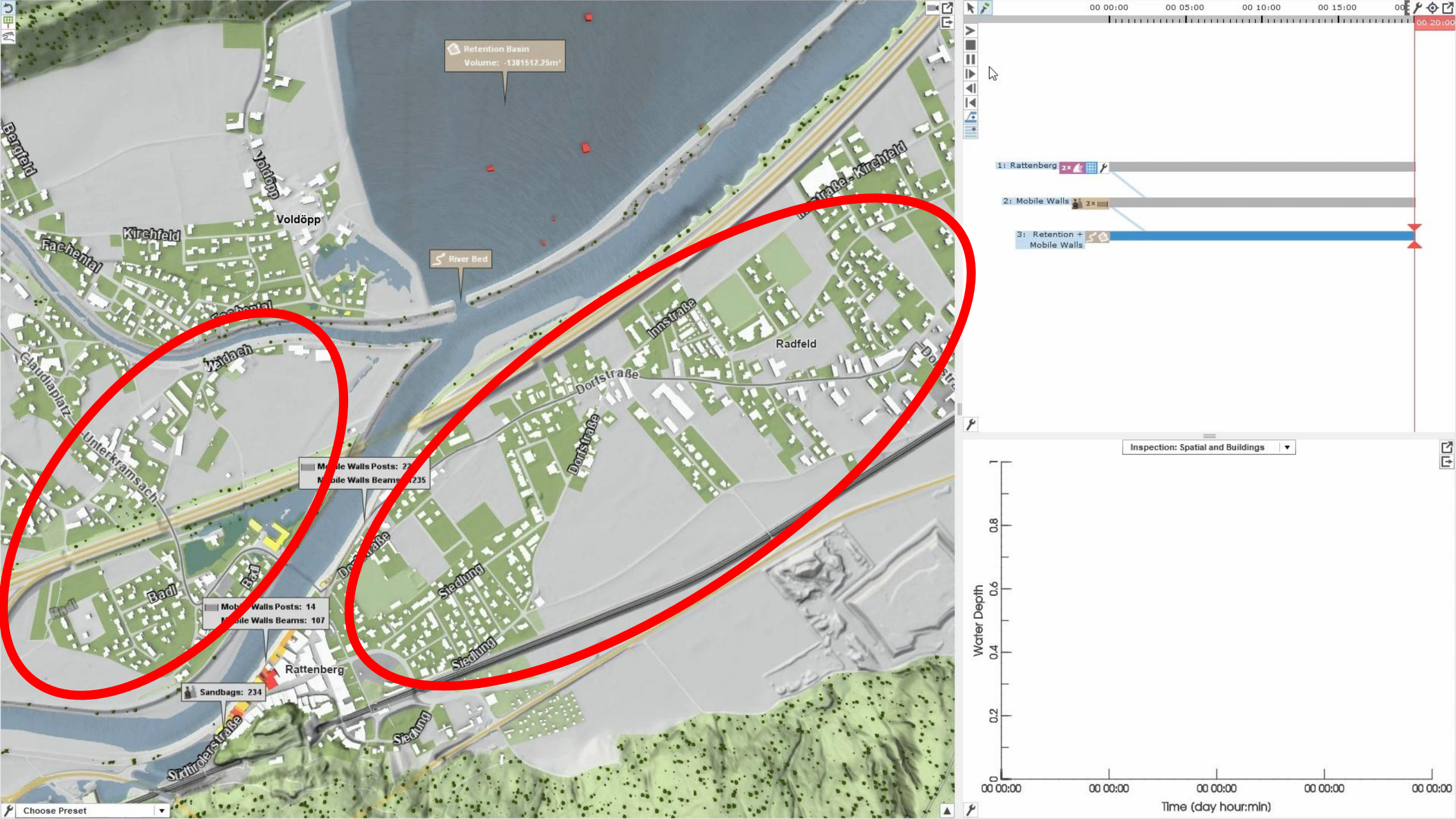
River Bed — Delete All
A terrain modification to create a river or channel

Sandbags — Delete All
A flexible flood protection barrier

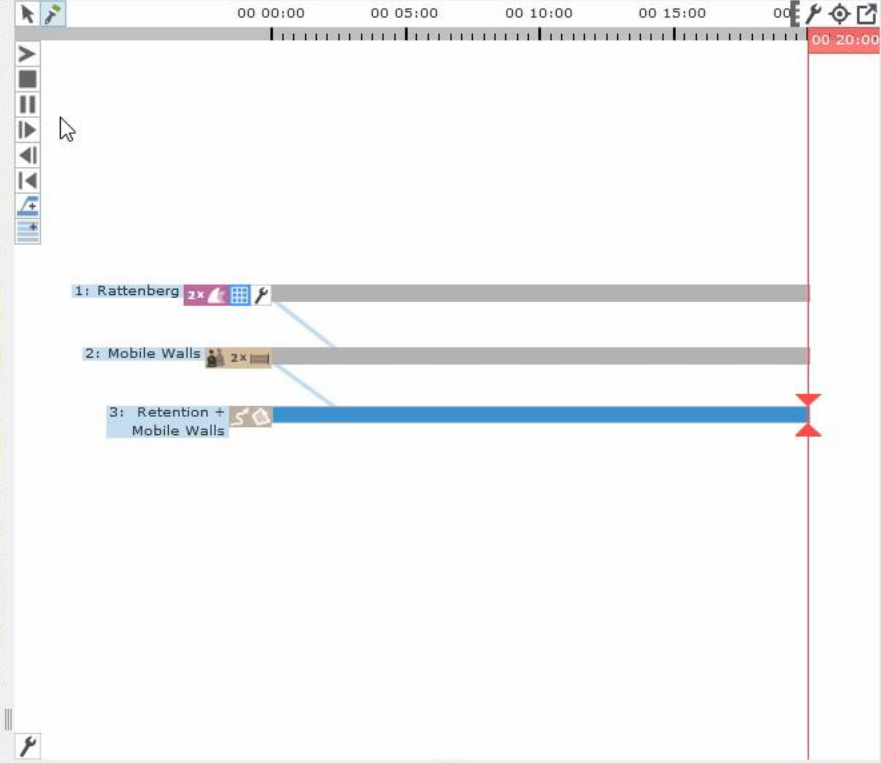
Sidewalk — Delete All
The center line of a sidewalk to be created

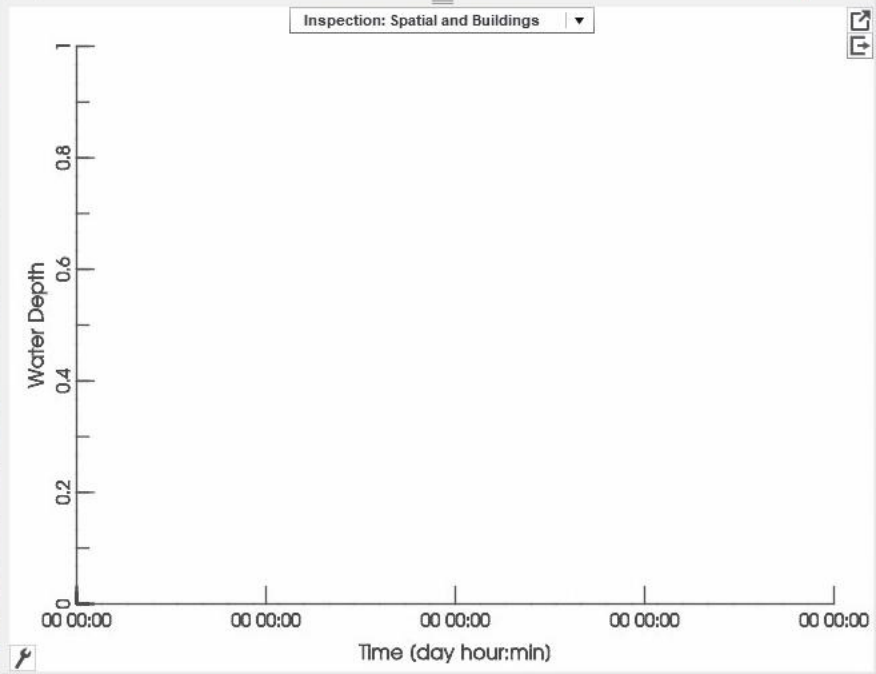
Sidewalk around Buildings — Delete All
Select buildings to auto-create a sidewalk around them within the drawn polygon

Buildings | Inspection | Protection | Setup | Sewer | Synth | Water

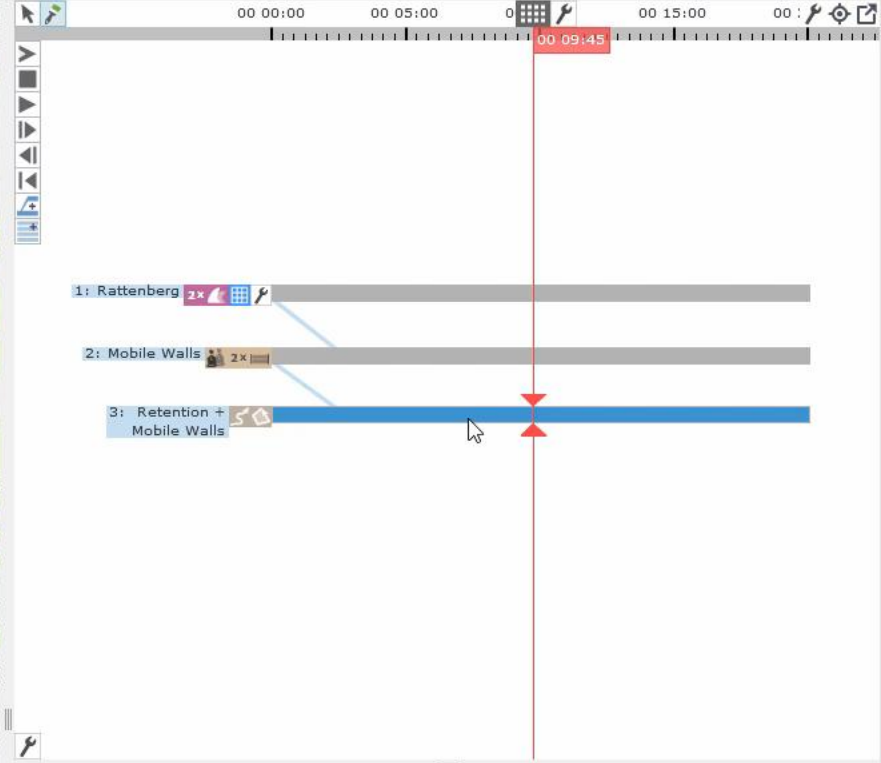

Retention Basin
Volume: -1381512.25m³
River Bed
Voldöpp
Kirchfeld
Fachental
Bergfeld
Innstraße
Radfeld
Dorfstraße
Claudiaplatz
Unterkramsach
Badl
Siedlung
Rattenberg
Sandbags: 234
Südtirolerstraße
Mobile Walls Posts: 14
Mobile Walls Beams: 107
00 00:00
00 05:00
00 10:00
00 15:00
00 20:00
1: Rattenberg
2: Mobile Walls
3: Retention + Mobile Walls
Inspection: Spatial and Buildings
Water Depth
Time (day hour:min)
Choose Preset

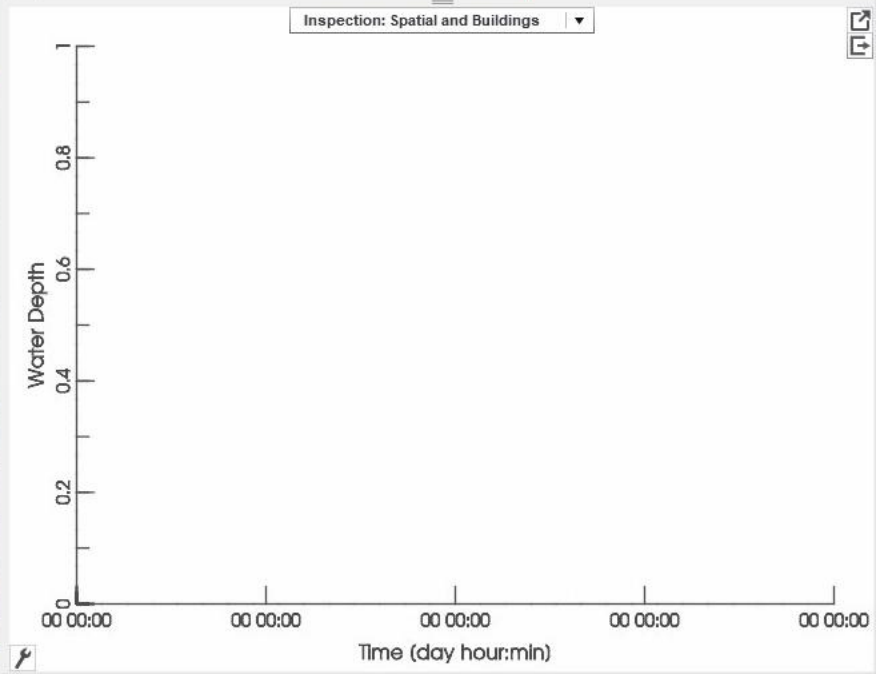

Retention Basin
Volume: -1381512.25m³
River Bed
Mobile Walls Posts: 222
Mobile Walls Beams: 1235
Mobile Walls Posts: 14
Mobile Walls Beams: 107
Sandbags: 234
Bergfeld
Voldöpp
Kirchfeld
Fachental
Innstraße - Kirchfeld
Innstraße
Radfeld
Dorfstraße
Claudiaplatz
Unterkramsach
Badl
Siedlung
Rattenberg
Südtirolerstraße
Choose Preset


00 00:00
00 05:00
00 10:00
00 15:00
00 20:00
1: Rattenberg
2: Mobile Walls
3: Retention + Mobile Walls


Inspection: Spatial and Buildings
Water Depth
0
0.2
0.4
0.6
0.8
1
00 00:00
00 00:00
00 00:00
00 00:00
00 00:00
Time (day hour:min)


Retention Basin
Volume: -1381512.25m³
River Bed
Mobile Walls Posts: 220
Mobile Walls Beams: 1144
Mobile Walls Posts: 14
Mobile Walls Beams: 99
Rattenberg
Voldöpp
Kirchfeld
Fachental
Radfeld
Dorfstraße
Innstraße
Innstraße - Kirchfeld
Claudiaplatz
Unterkramsach
Badl
Siedlung
Südtirolerstraße


00 00:00
00 05:00
00 15:00
00 09:45
1: Rattenberg
2: Mobile Walls
3: Retention + Mobile Walls


Inspection: Spatial and Buildings
Water Depth
0
0.2
0.4
0.6
0.8
1
00 00:00
00 00:00
00 00:00
00 00:00
00 00:00
Time (day hour:min)

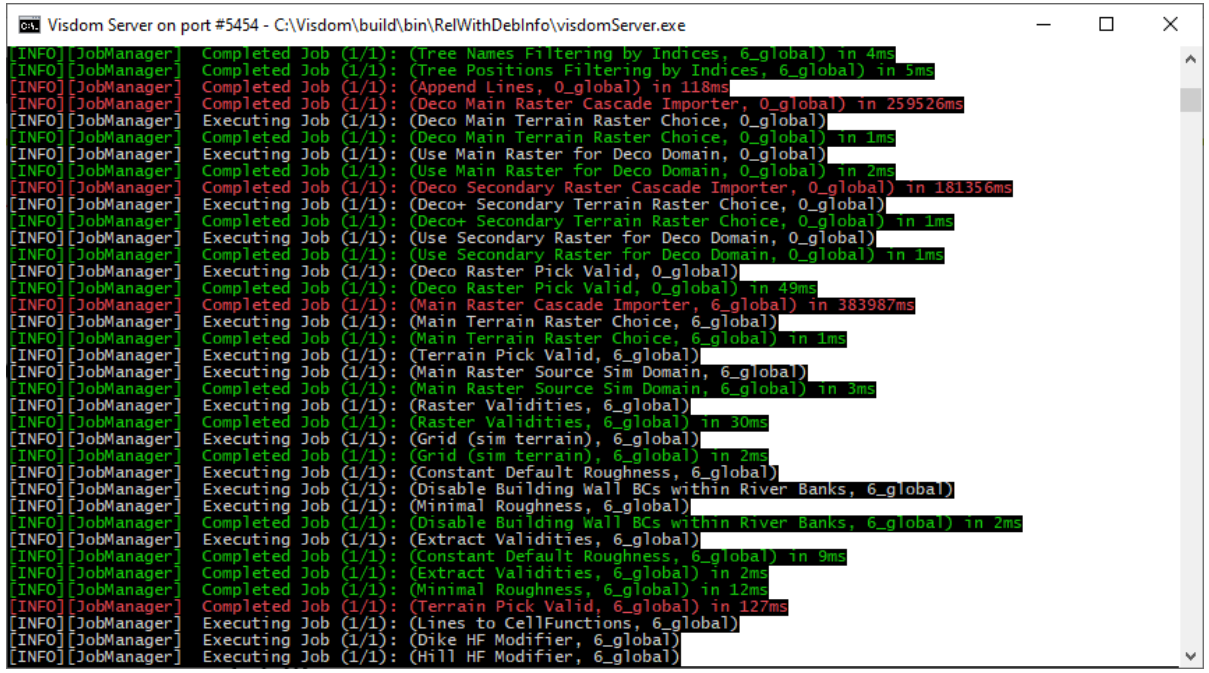
Client/Server Architecture

Local client connected to local or remote server

- Laptop, gaming PC, multi-GPU workstation, cloud

Server-side data storage, simulation, rendering

- Easier to maintain
- Centralized data
- Better data security
- No hardware requirements for client
- Less compatibility issues
- Less scalability to multiple users



Client/Server Architecture

Interactions recorded by client

Requests sent as XML to server

Image or video stream sent back to client

- Bandwidth can be limiting factor

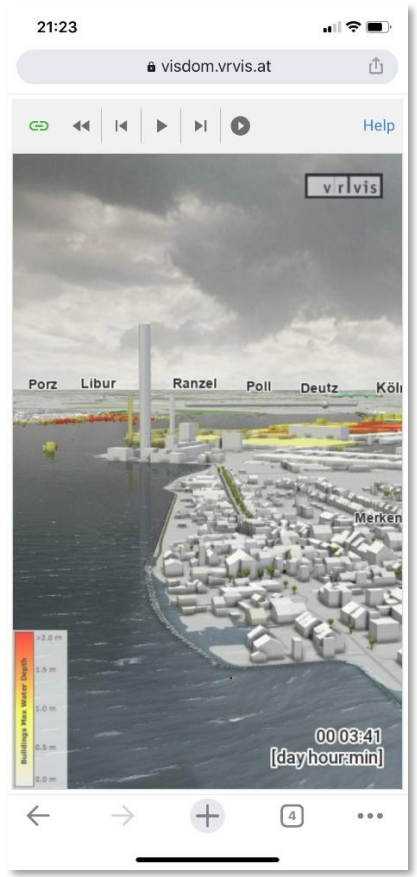
Rendering on demand

- No rendering loop
- Reduced energy consumption and bandwidth usage
- Ideal for multi-instance servers

Client/Server Architecture

Lightweight web client than runs on every modern device

Browser client for flood management experts



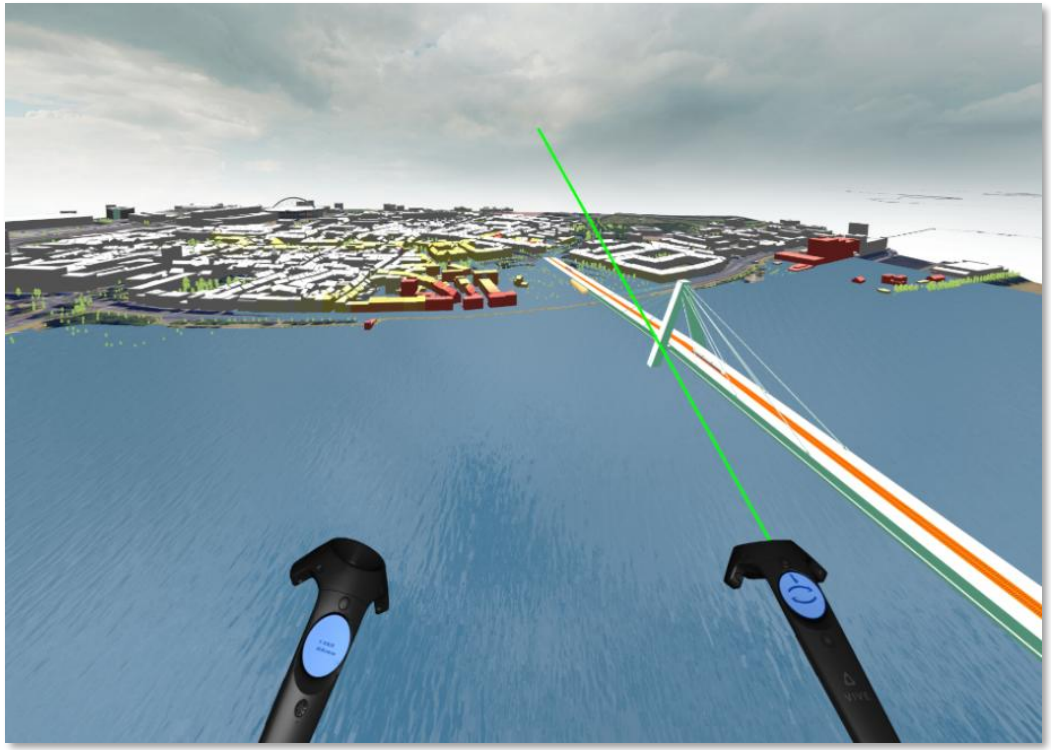
Mobile client for flood prognosis

Client/Server Architecture

Different clients tailored to different tasks and target audiences

App concept supported by modular dataflow system

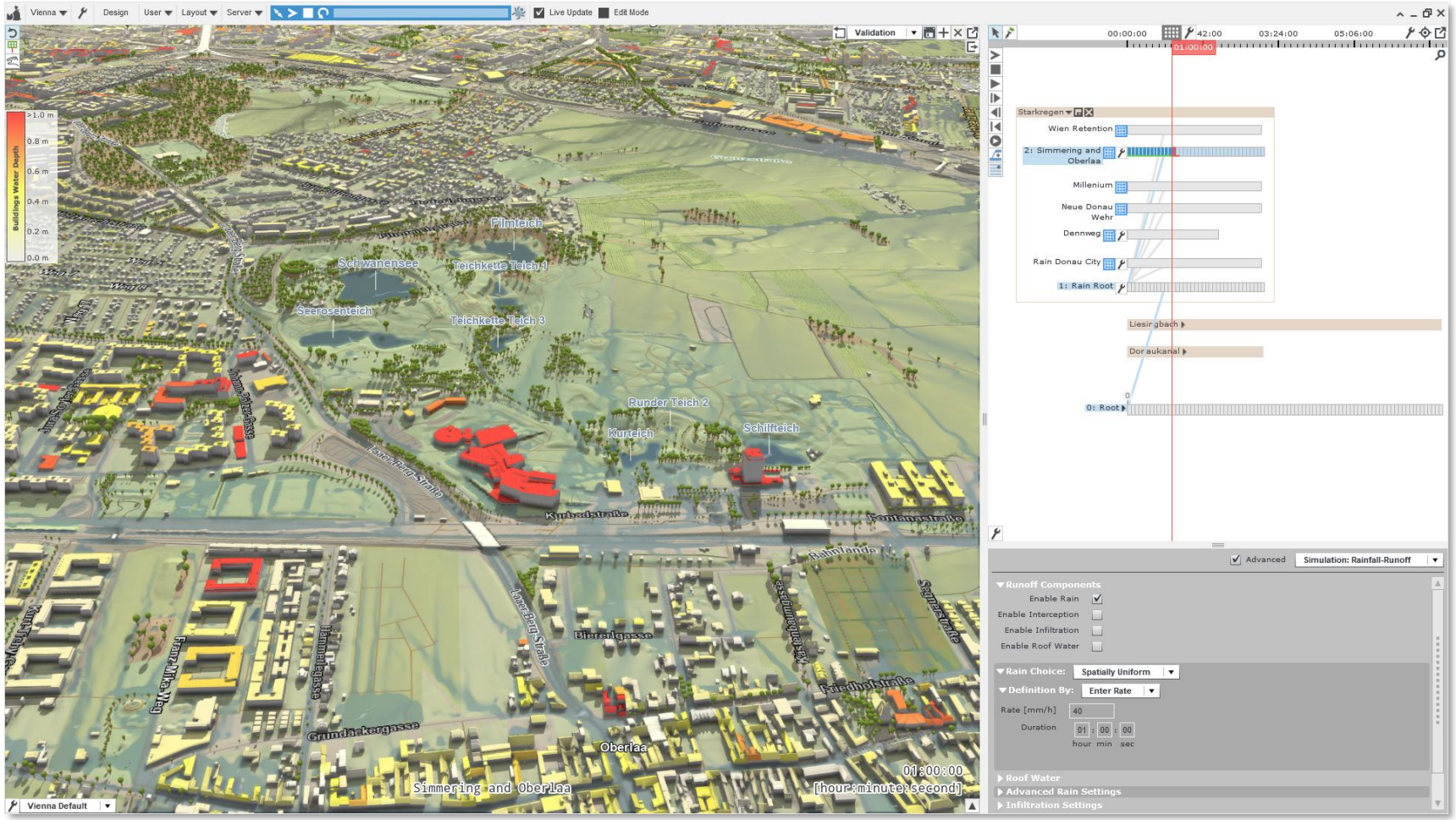
Embedded web app for personalized flood risk

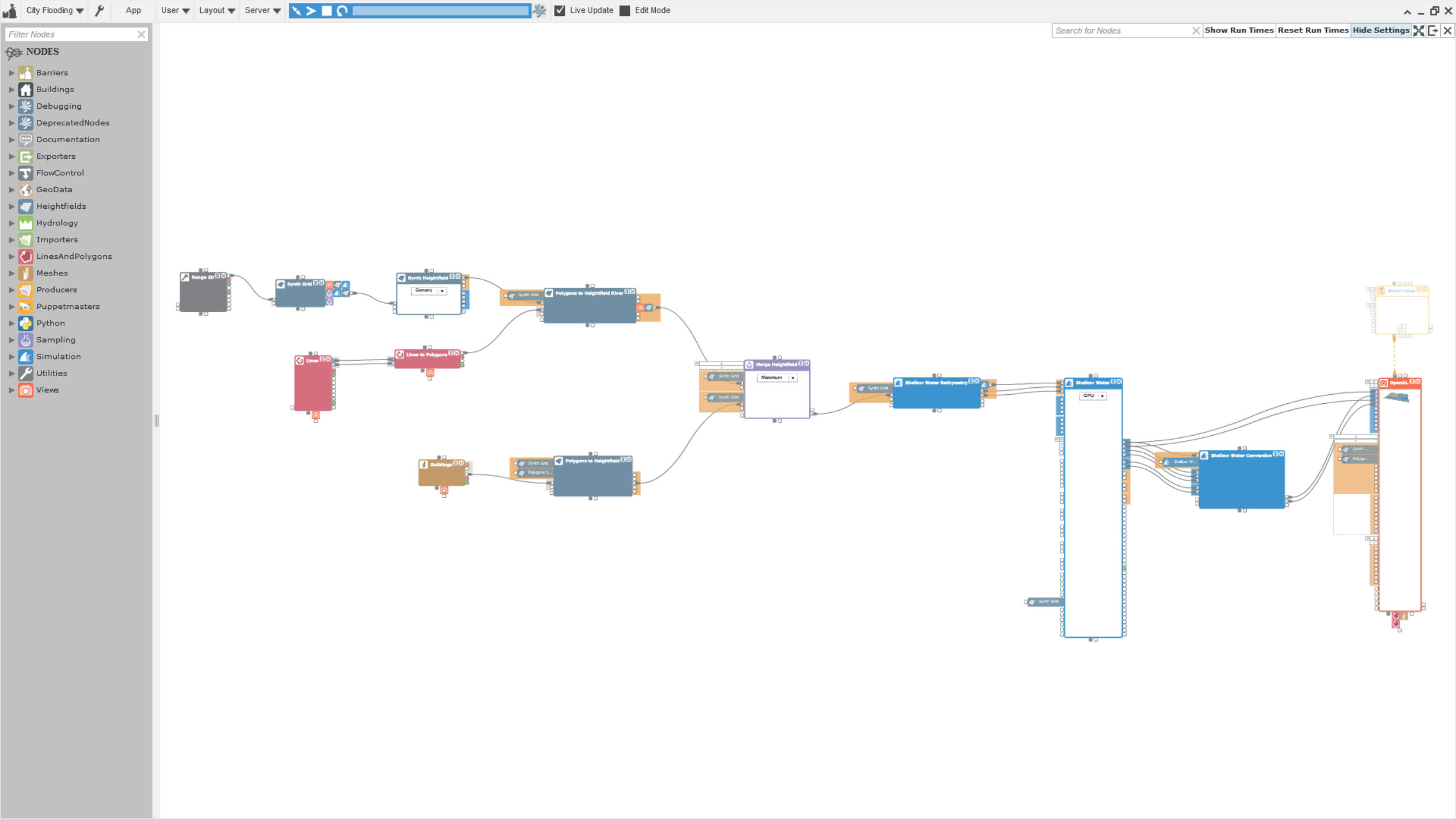


VR client for public communication

Client/Server Architecture

Desktop client for creation of new apps



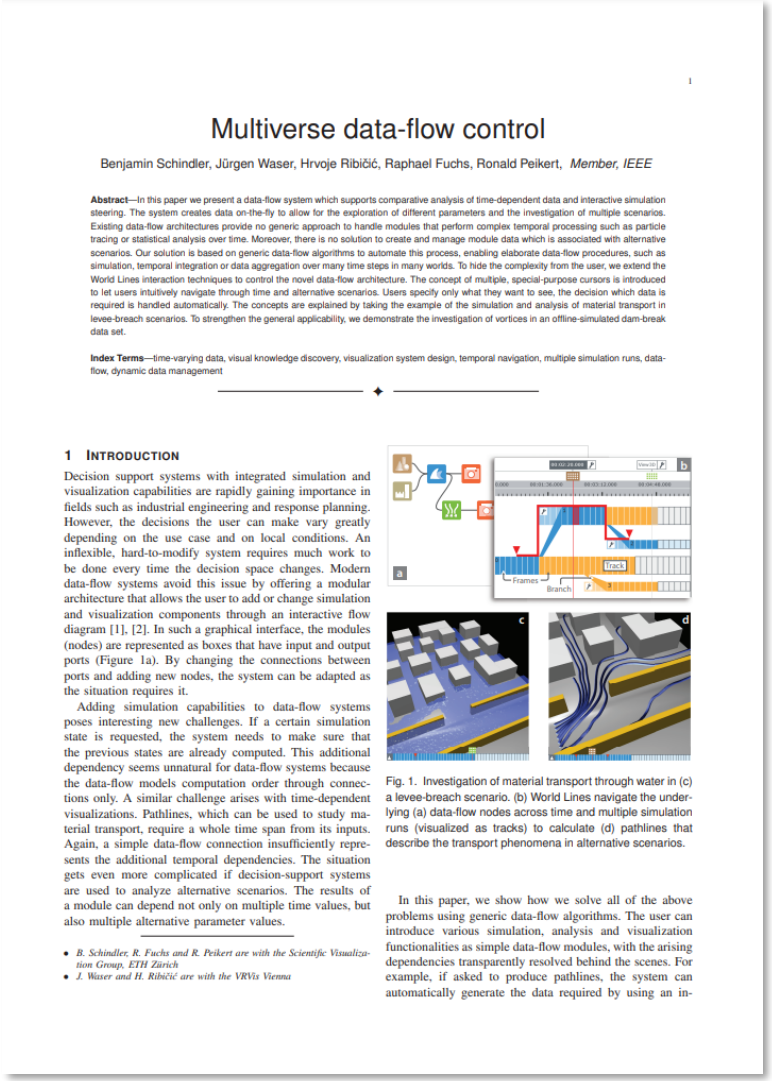

City Flooding
App
User
Layout
Server
Live Update
Edit Mode
Filter Nodes
NODES
Barriers
Buildings
Debugging
DeprecatedNodes
Documentation
Exporters
FlowControl
GeoData
Heightfields
Hydrology
Importers
LinesAndPolygons
Meshes
Producers
Puppetmasters
Python
Sampling
Simulation
Utilities
Views
Search for Nodes
Show Run Times
Reset Run Times
Hide Settings
Range 2D
Synth Grid
Synth Heightfield
Generic
Polygons to Heightfield River
Lines
Lines to Polygons
Buildings
Polygons to Heightfield
Merge Heightfield
Maximum
Shallow Water Bathymetry
Shallow Water
GPU
Shallow Water Conversion
World Lines
OpenGL

Dataflow

Benjamin Schindler, Jürgen Waser, Hrvoje Ribičić, Raphael Fuchs, Ronald Peikert

Multiverse Data-Flow Control

In IEEE Transactions on Visualization and Computer Graphics, 19(6), pages 1005–1019, 2013



Multiverse data-flow control

Benjamin Schindler, Jürgen Waser, Hrvoje Ribičić, Raphael Fuchs, Ronald Peikert, *Member, IEEE*

Abstract—In this paper we present a data-flow system which supports comparative analysis of time-dependent data and interactive simulation steering. The system creates data on-the-fly to allow for the exploration of different parameters and the investigation of multiple scenarios. Existing data-flow architectures provide no generic approach to handle modules that perform complex temporal processing such as particle tracing or statistical analysis over time. Moreover, there is no solution to create and manage module data which is associated with alternative scenarios. Our solution is based on generic data-flow algorithms to automate this process, enabling elaborate data-flow procedures, such as simulation, temporal integration or data aggregation over many time steps in many worlds. To hide the complexity from the user, we extend the World Lines interaction techniques to control the novel data-flow architecture. The concept of multiple, special-purpose cursors is introduced to let users intuitively navigate through time and alternative scenarios. Users specify only what they want to see, the decision which data is required is handled automatically. The concepts are explained by taking the example of the simulation and analysis of material transport in levee-breach scenarios. To strengthen the general applicability, we demonstrate the investigation of vortices in an offline-simulated dam-break data set.

Index Terms—time-varying data, visual knowledge discovery, visualization system design, temporal navigation, multiple simulation runs, data-flow, dynamic data management

1 Introduction

Decision support systems with integrated simulation and visualization capabilities are rapidly gaining importance in fields such as industrial engineering and response planning. However, the decisions the user can make vary greatly depending on the use case and on local conditions. An inflexible, hard-to-modify system requires much work to be done every time the decision space changes. Modern data-flow systems avoid this issue by offering a modular architecture that allows the user to add or change simulation and visualization components through an interactive flow diagram [1], [2]. In such a graphical interface, the modules (nodes) are represented as boxes that have input and output ports (Figure 1a). By changing the connections between ports and adding new nodes, the system can be adapted as the situation requires it.

Adding simulation capabilities to data-flow systems poses interesting new challenges. If a certain simulation state is requested, the system needs to make sure that the previous states are already computed. This additional dependency seems unnatural for data-flow systems because the data-flow models computation order through connections only. A similar challenge arises with time-dependent visualizations. Pathlines, which can be used to study material transport, require a whole time span from its inputs. Again, a simple data-flow connection insufficiently represents the additional temporal dependencies. The situation gets even more complicated if decision-support systems are used to analyze alternative scenarios. The results of a module can depend not only on multiple time values, but also multiple alternative parameter values.

• *B. Schindler, R. Fuchs and R. Peikert are with the Scientific Visualization Group, ETH Zürich*
• *J. Waser and H. Ribičić are with the VRVis Vienna*

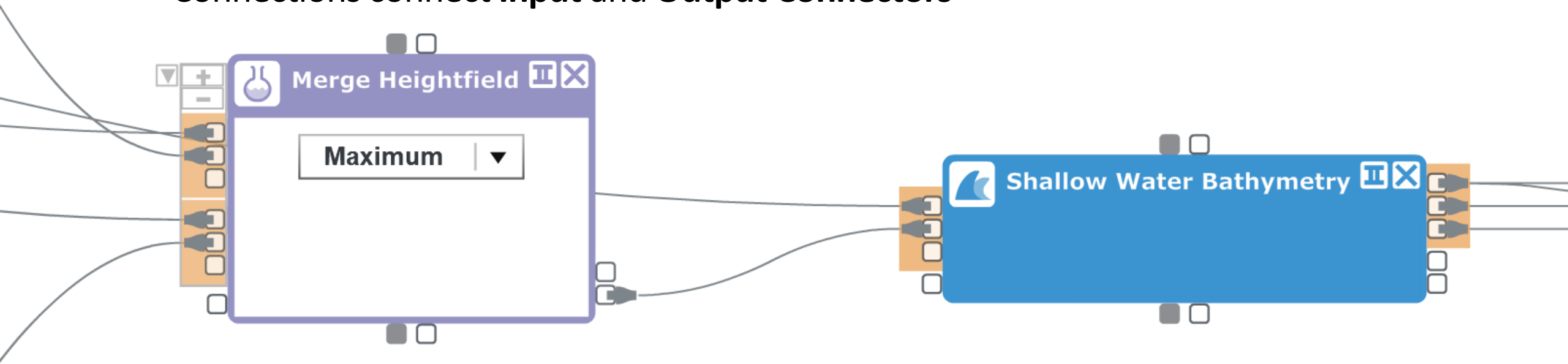
Fig. 1. Investigation of material transport through water in (c) a levee-breach scenario. (b) World Lines navigate the underlying (a) data-flow nodes across time and multiple simulation runs (visualized as tracks) to calculate (d) pathlines that describe the transport phenomena in alternative scenarios.

In this paper, we show how we solve all of the above problems using generic data-flow algorithms. The user can introduce various simulation, analysis and visualization functionalities as simple data-flow modules, with the arising dependencies transparently resolved behind the scenes. For example, if asked to produce pathlines, the system can automatically generate the data required by using an in-

Dataflow

Modular architecture

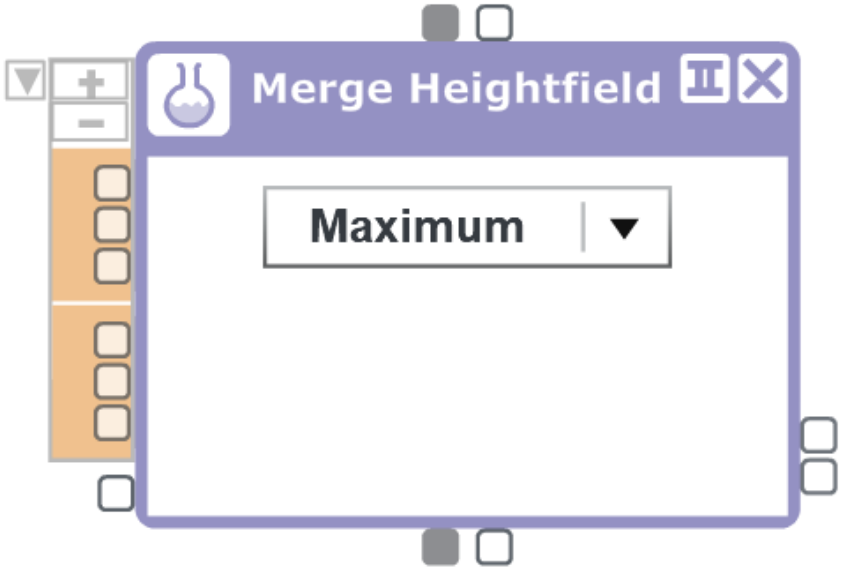
- Directed acyclic graph
- Functionality encapsulated in **Nodes**
- Data flows through **Connections**
- Connections connect **Input** and **Output Connectors**



Nodes

Small or larger unit of functionality

- Importers
- Data transformations
- Simple and complex algorithms
- Simulations
- Visualization
- User interactions
- Dataflow control
- …
- 550+ nodes in 20+ categories



Nodes

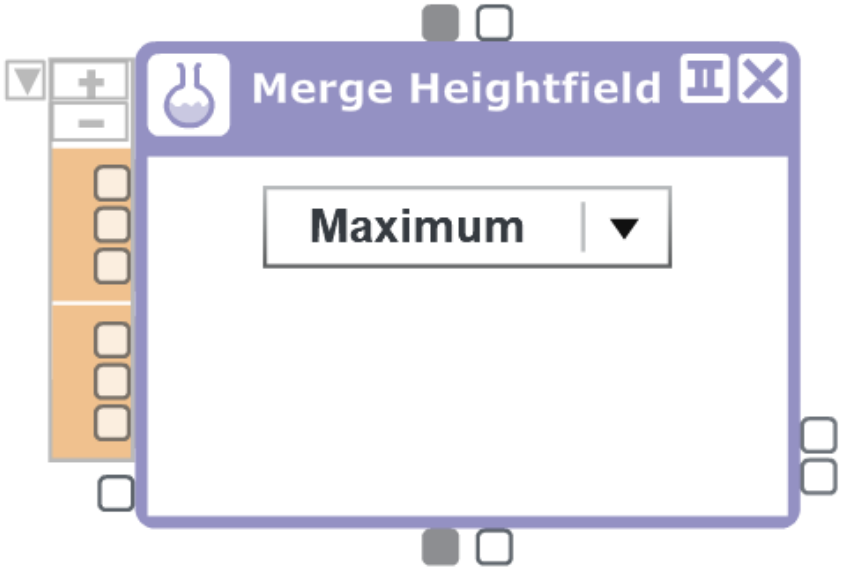
Reusable code

Easy to use

Combination of nodes for complex algorithms

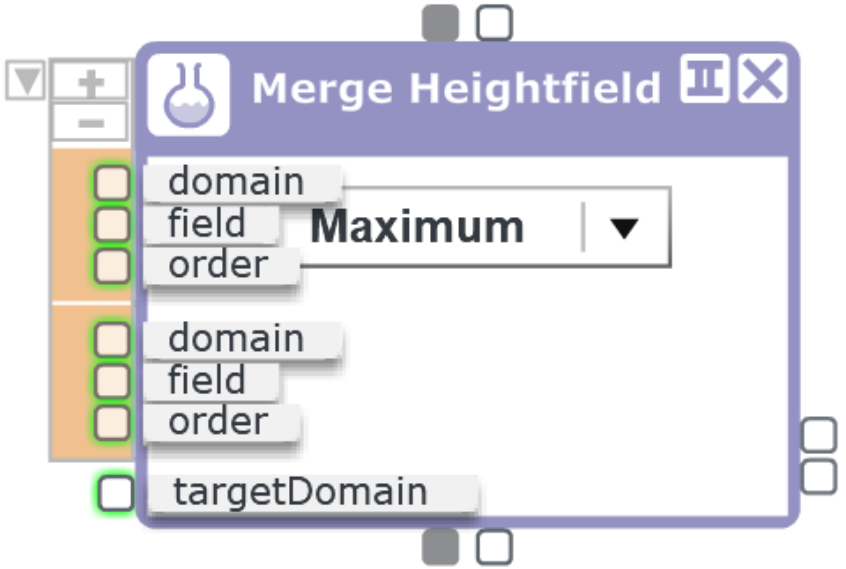
Break down problems into smaller ones

Each node has a unique internal ID



Input Connectors

Specifies input data from previous nodes



Required and optional connectors

- Required: Data for transformations
- Optional: Additional/alternative functionality
- No input needed for generators and importers

Groups and sequences of connectors

- Each input is uniquely defined by sequence index and name

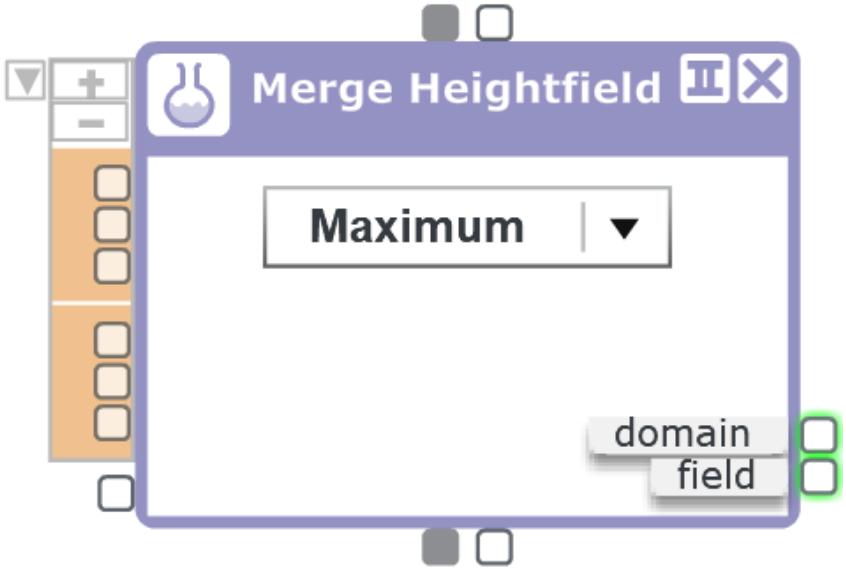
Input data are immutable

Output Connectors

Specifies output data for subsequent nodes

Output is optional

- Computation only if needed
- No output needed for exporters and visualization



Each output is uniquely identified by name

Computed data stored in hash table with name as key (Data cache)

HDD caching enabled per node by developer

Data

AbstractData base class

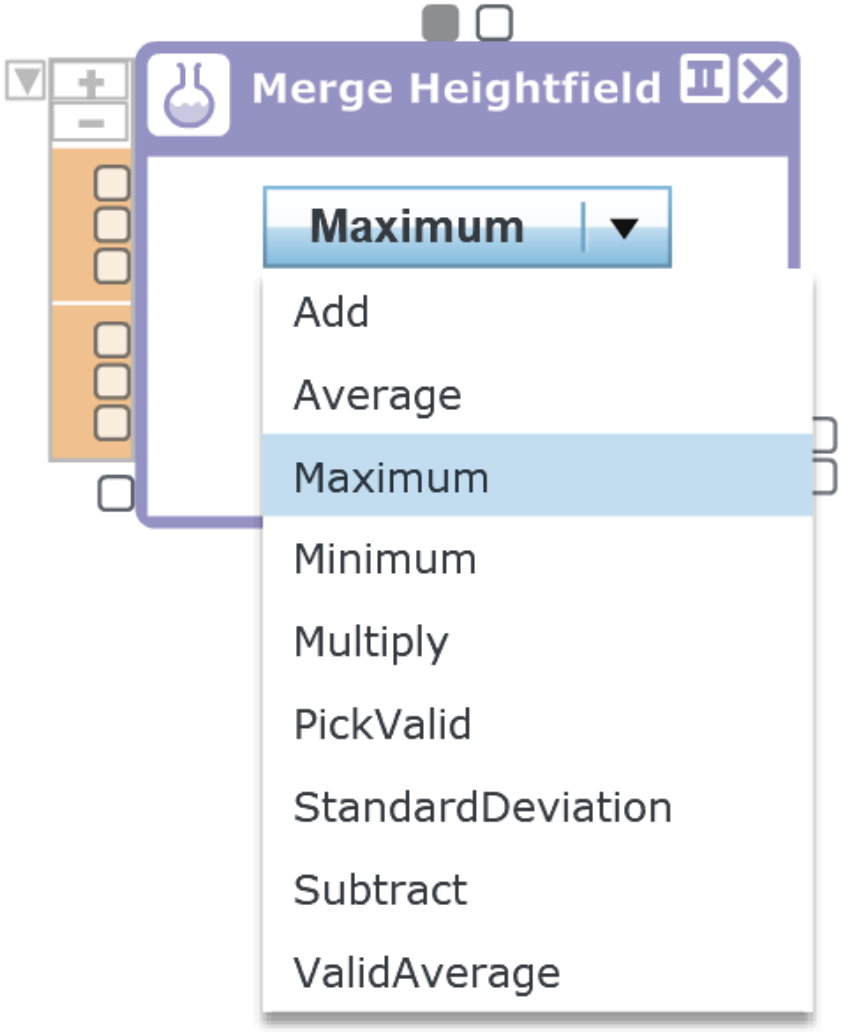
- Vector
- GridDomain
- IComposite
- Mesh
- Image
- Lines
- …

Serializable to binary format to enable HDD caching

Node Settings

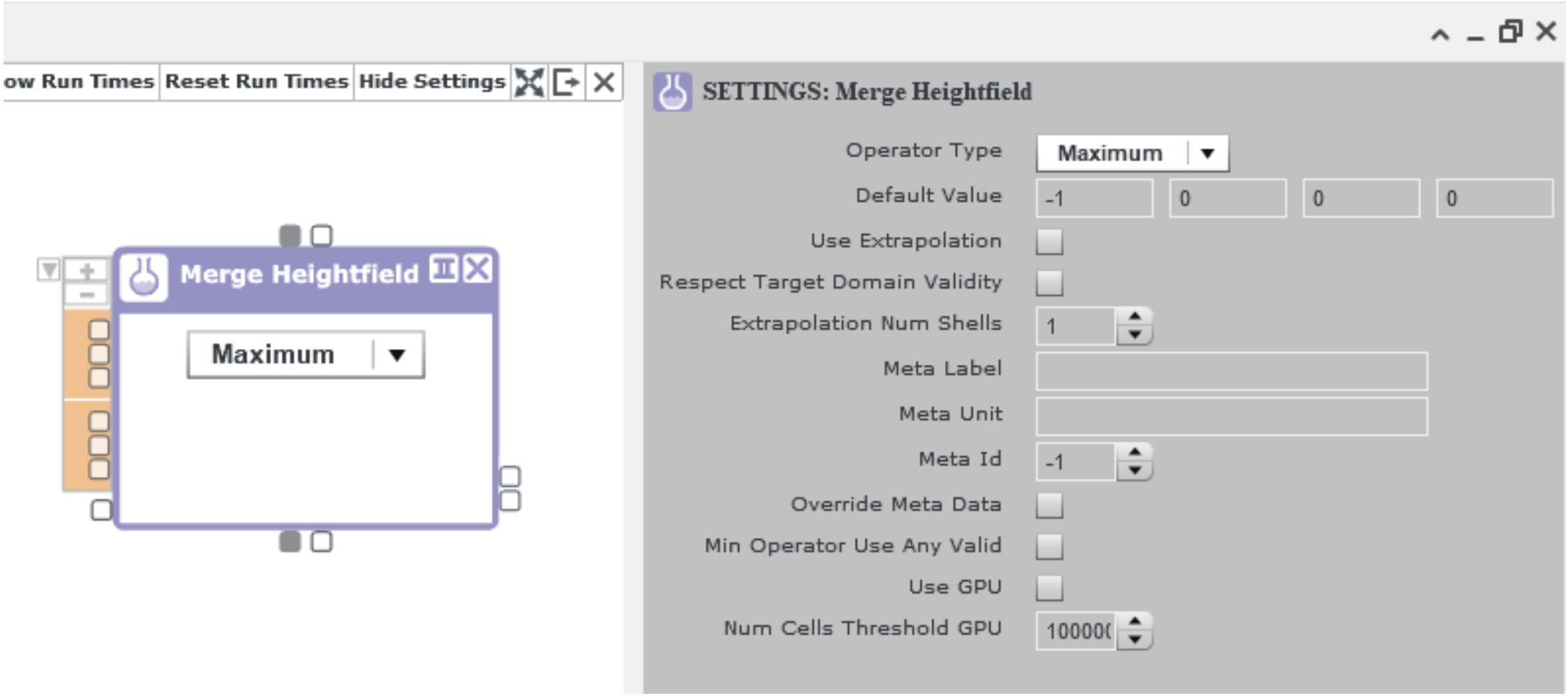
Parameters

Steer behavior of algorithms



Node Settings

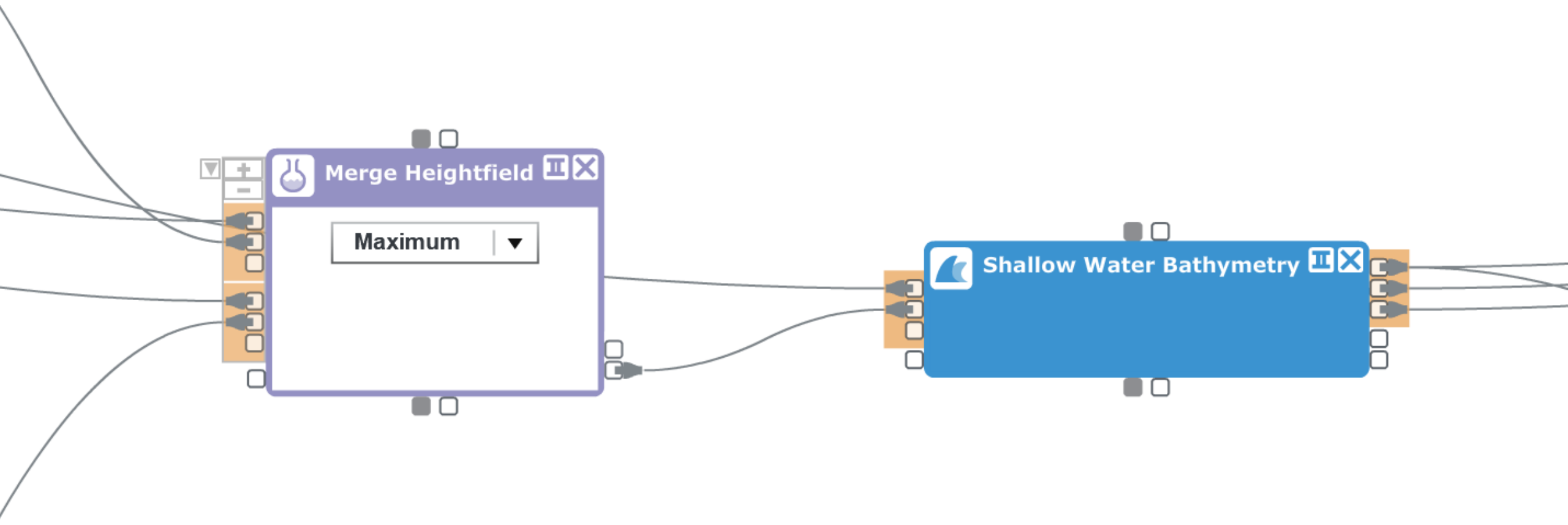
GUI elements specified by node developer



Connections

Visual representation of dataflow between nodes

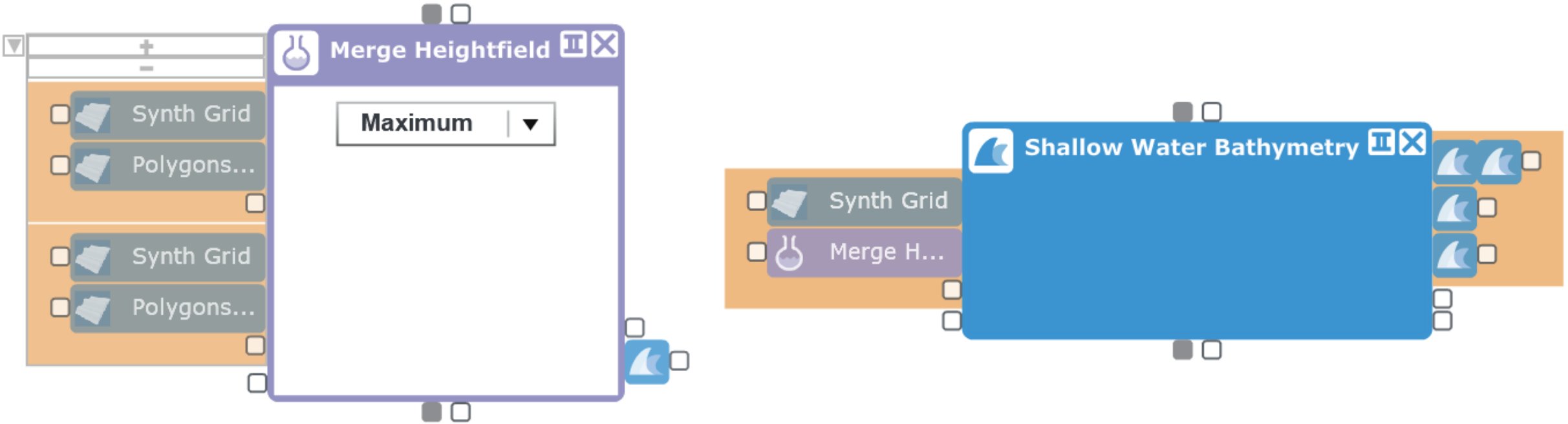
Realized internally as pair of references to node IDs and connector names



Connections

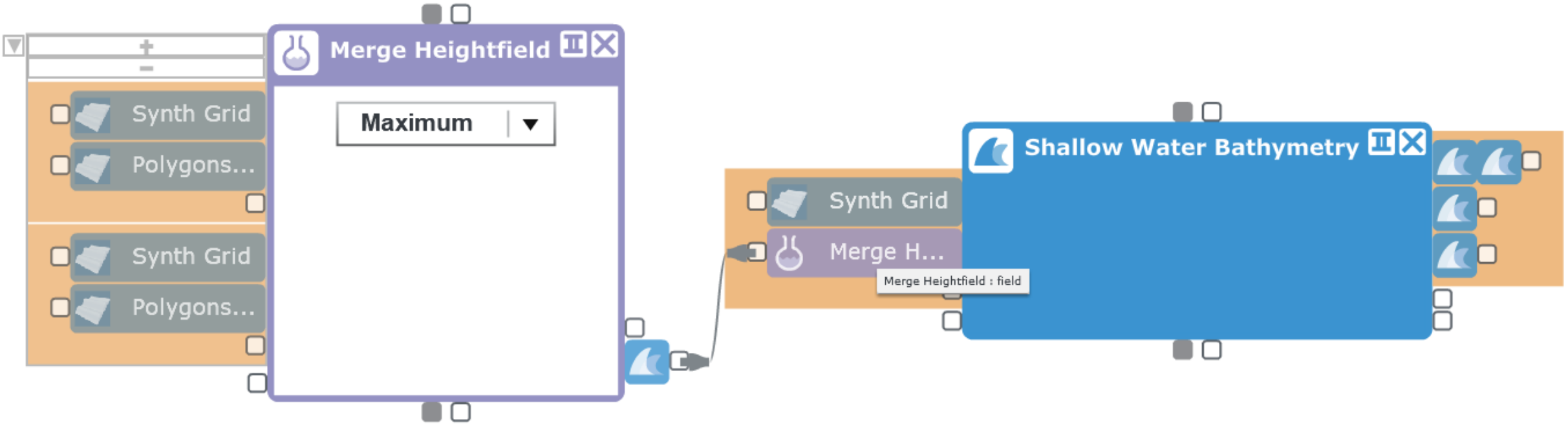
Can be hidden individually to reduce complexity of node diagram

Replaced by icons and input node names



Connections

Hidden connections can be inspected on mouse-over



Jobs

A job is a node execution

Managed by a job manager with a pool of worker threads

Jobs can only be executed if all connected input is available

Traditional Dataflow Execution

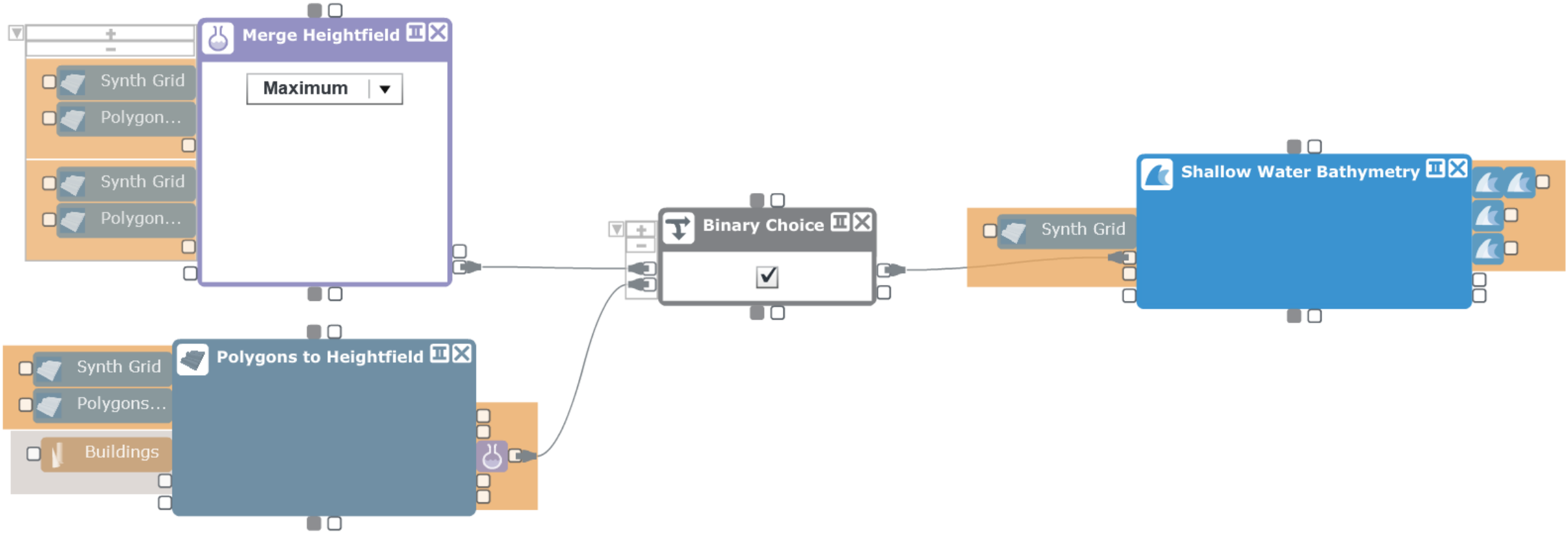
Single downstream pass

Traverse dataflow from source nodes to view nodes

Push jobs to queue and execute concurrently

Traditional Dataflow Execution

Executes the jobs of all nodes, even disconnected ones



Optimized Dataflow Execution

First pass: Traverse dataflow from sink nodes (view, exporter) to source nodes

- Gather dependencies of all nodes based on connectivity and flow control nodes
- Check if required data already exist in node‘s cache
- Mark nodes that require execution

Second pass: Traverse dataflow from source nodes to sink nodes

- Create jobs only for the previously marked nodes
- Avoids unnecessary computation

World Lines

Jürgen Waser, Raphael Fuchs, Hrvoje Ribičić, Benjamin Schindler, Günter Blöschl, Meister Eduard Gröller

World Lines

In IEEE Transactions on Visualization and Computer Graphics 16(6), pages 1458–1467, 2010

World Lines

Jürgen Waser, Raphael Fuchs, Hrvoje Ribičić, Benjamin Schindler, Günther Blöschl, and M. Eduard Gröller

Abstract—In this paper we present World Lines as a novel interactive visualization that provides complete control over multiple heterogeneous simulation runs. In many application areas, decisions can only be made by exploring alternative scenarios. The goal of the suggested approach 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. 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. To demonstrate this technique we present a flooding scenario and show the usefulness of the integrated approach to support informed decision making.

Index Terms—Problem solving environment, decision making, simulation steering, parallel worlds, CFD, smoothed particle hydrodynamics.

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 1). 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 [19] has identified as one of the most important research problems in scientific visualization.

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 [41]. Obviously we can 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 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 [27]. 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) which we present in this paper integrates simulation, visualization, and interactive analysis into a unified system.

2 Related Work

In this paper 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.

Simulation Steering Mulder et al. [31] 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. [20] point out major topics when building a simulation-based problem solving environment: control structures, data distribution, data presentation, and user interfaces. Treeck et al. [42] present a steering system that enables modification of geometry via basic transformations. Matkovic et al. [27] suggest to combine CFD simulation and visualization by writing out multiple simulation runs as ensemble data and comparing these runs using the COMVis System.

History, Provenance and Processes GRASPARC [5] and Hyperscribe [45] 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 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 [11, 3]. AsbruView [22, 23] 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 illustrates a genealogical tree as lines in a horizontal layout [1]. VisTrails [38] adapts a history tree for capturing and reusing provenance in a visual exploration system. Graph-based layouts have been adopted for data exploration [26] and process visualization [28, 35]. Business process visualization deals with complex

- *Jürgen Waser is with VRVis Vienna, E-mail: jwaser@vrvis.at.*
- *Raphael Fuchs is with ETH Zurich, E-mail: raphael@inf.ethz.ch.*
- *Hrvoje Ribičić is with VRVis Vienna, E-mail: ribicic@vrvis.at.*
- *Benjamin Schindler is with ETH Zürich, E-mail: bschindl@inf.ethz.ch.*
- *Günther Blöschl is with TU Vienna, E-mail: bloeschl@hydro.tuwien.ac.at.*
- *Eduard Gröller is with TU Vienna, E-mail: groeller@cg.tuwien.ac.at.*

Manuscript received 31 March 2010; accepted 1 August 2010; posted online 24 October 2010; mailed on 16 October 2010.
For information on obtaining reprints of this article, please send email to: tvcg@computer.org.

World Lines

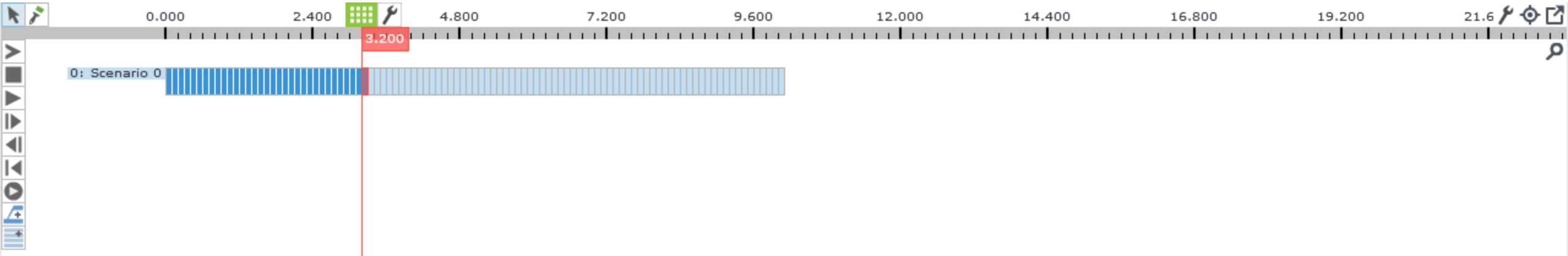
Simulation data are time-dependent

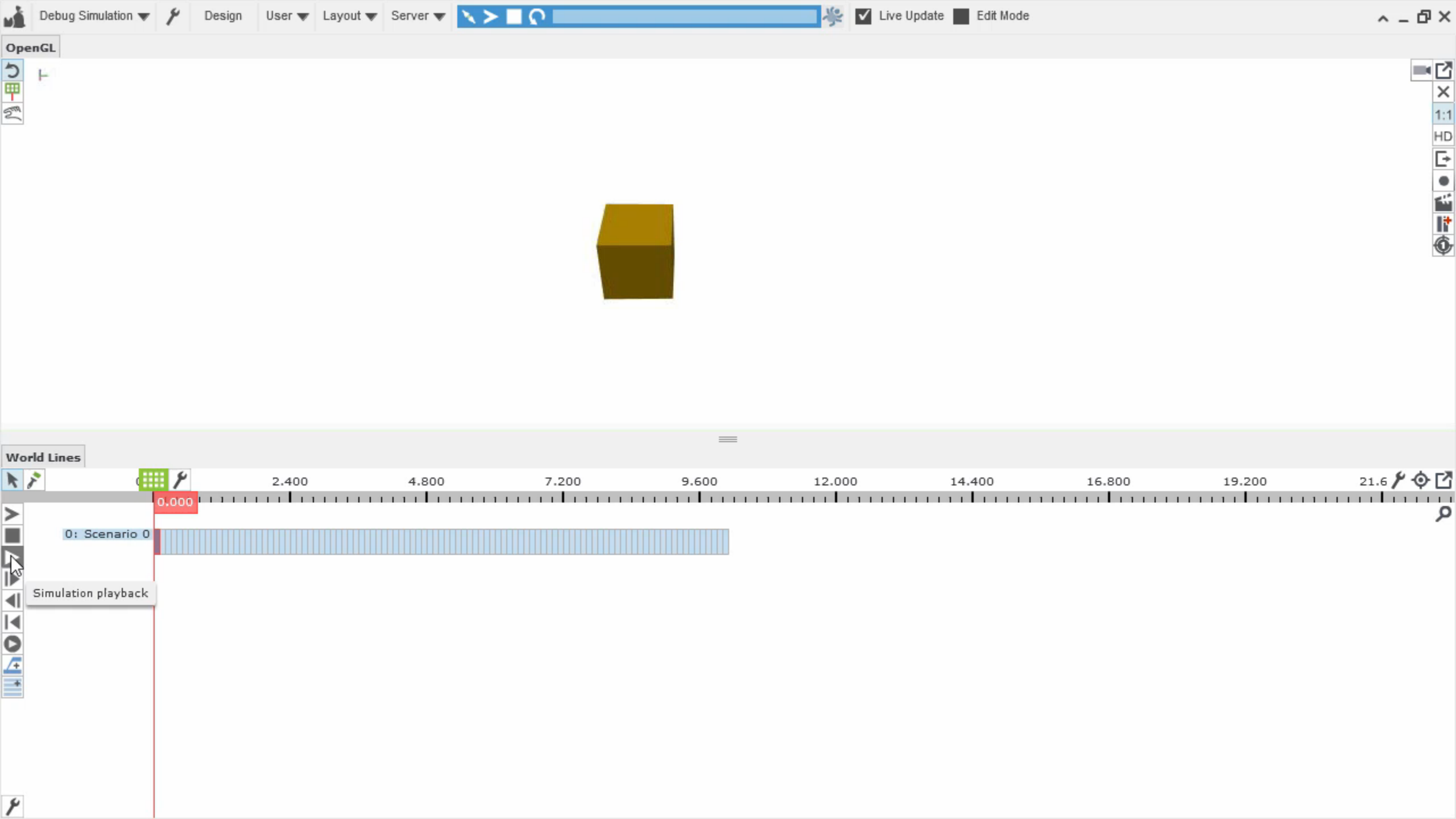
Temporal representation and navigation required

Time track specifies event duration

Time step specifies temporal resolution

Cursor is linked to view nodes and controls current time step




Debug Simulation
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
1:1
HD
World Lines
2.400
4.800
7.200
9.600
12.000
14.400
16.800
19.200
21.6
0.000
0: Scenario 0
Simulation playback

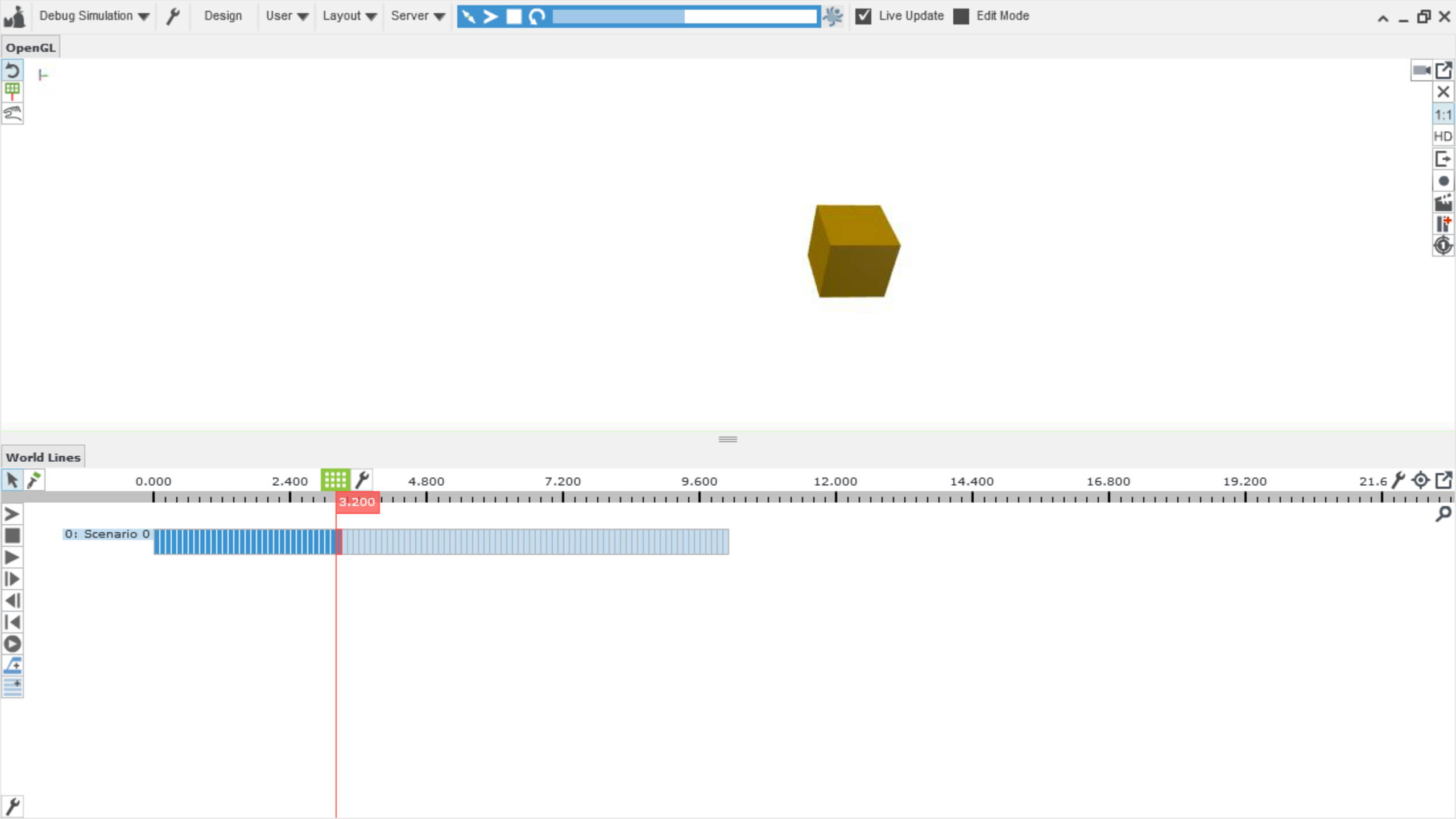
Scenarios

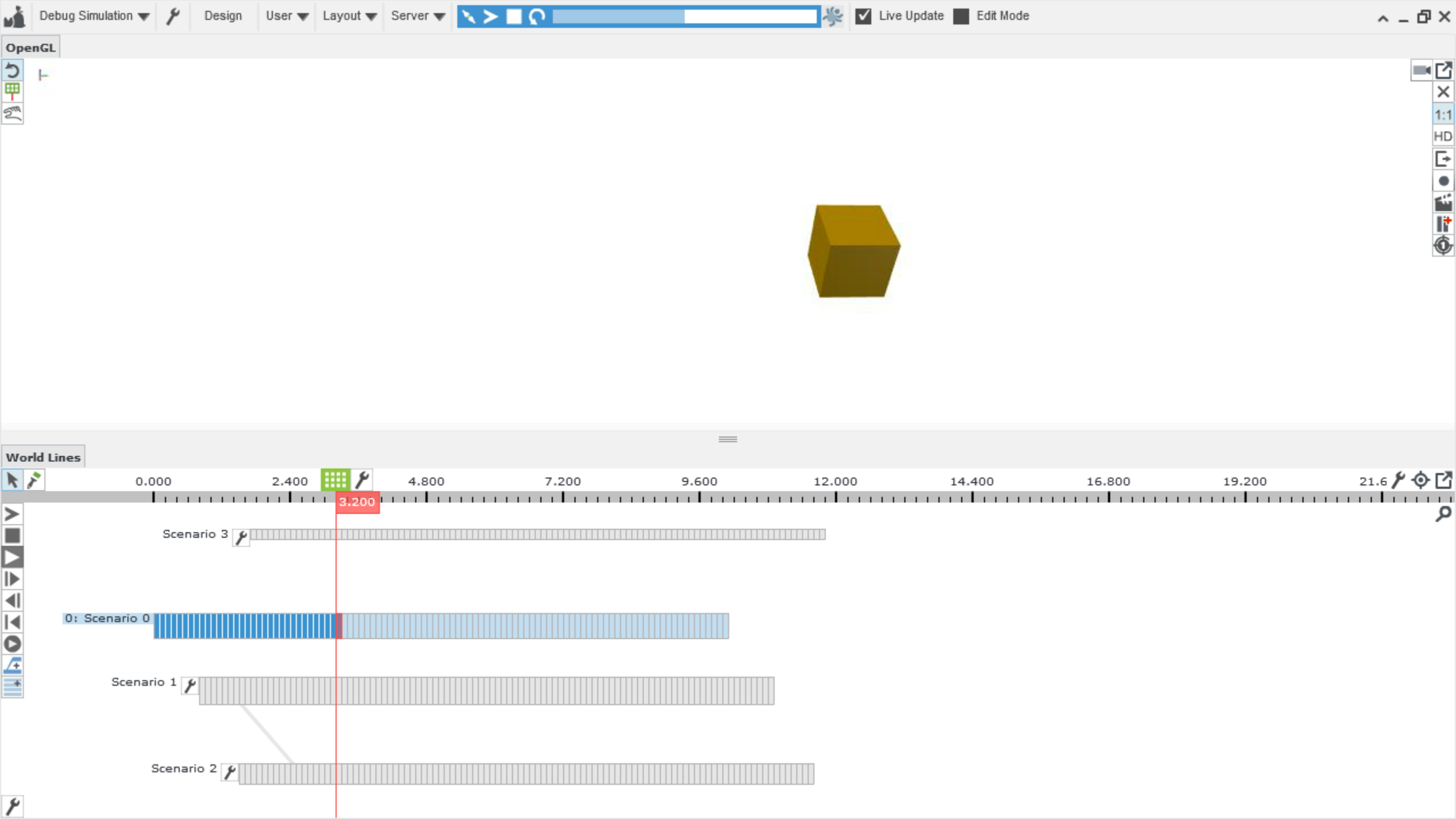
A scenario is a possible reality in the system

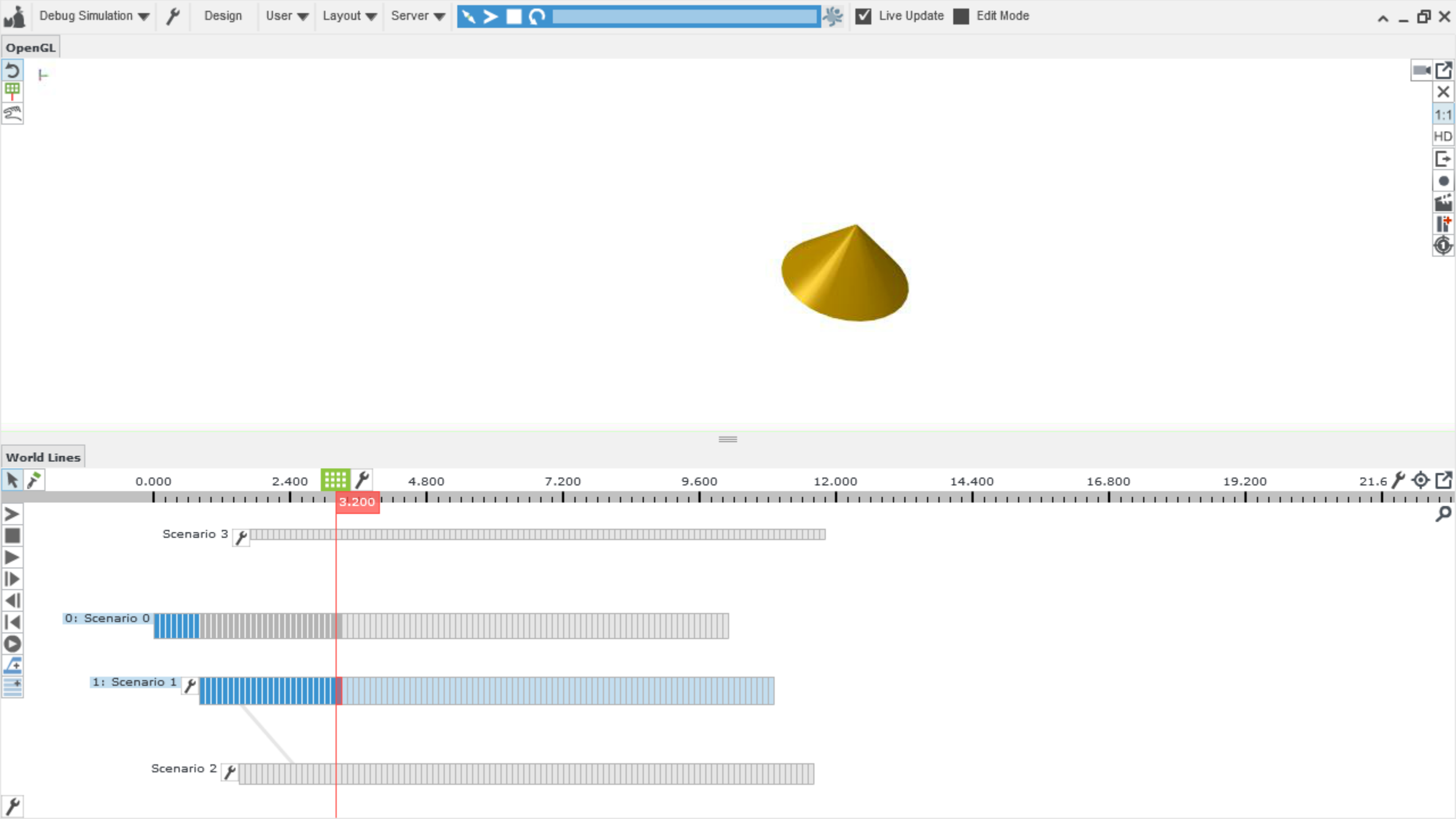
Defined by its settings

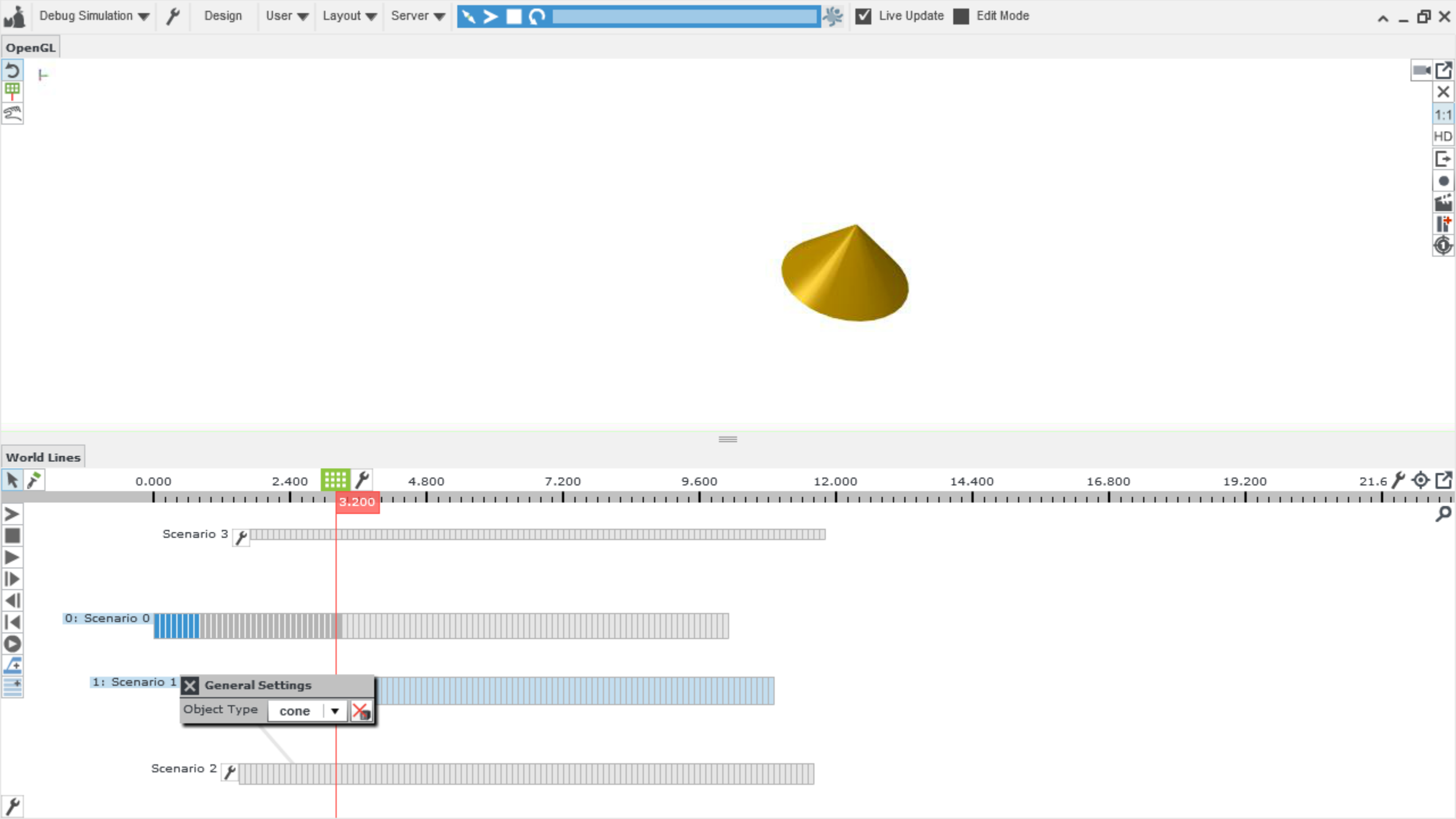
Associated with one time track

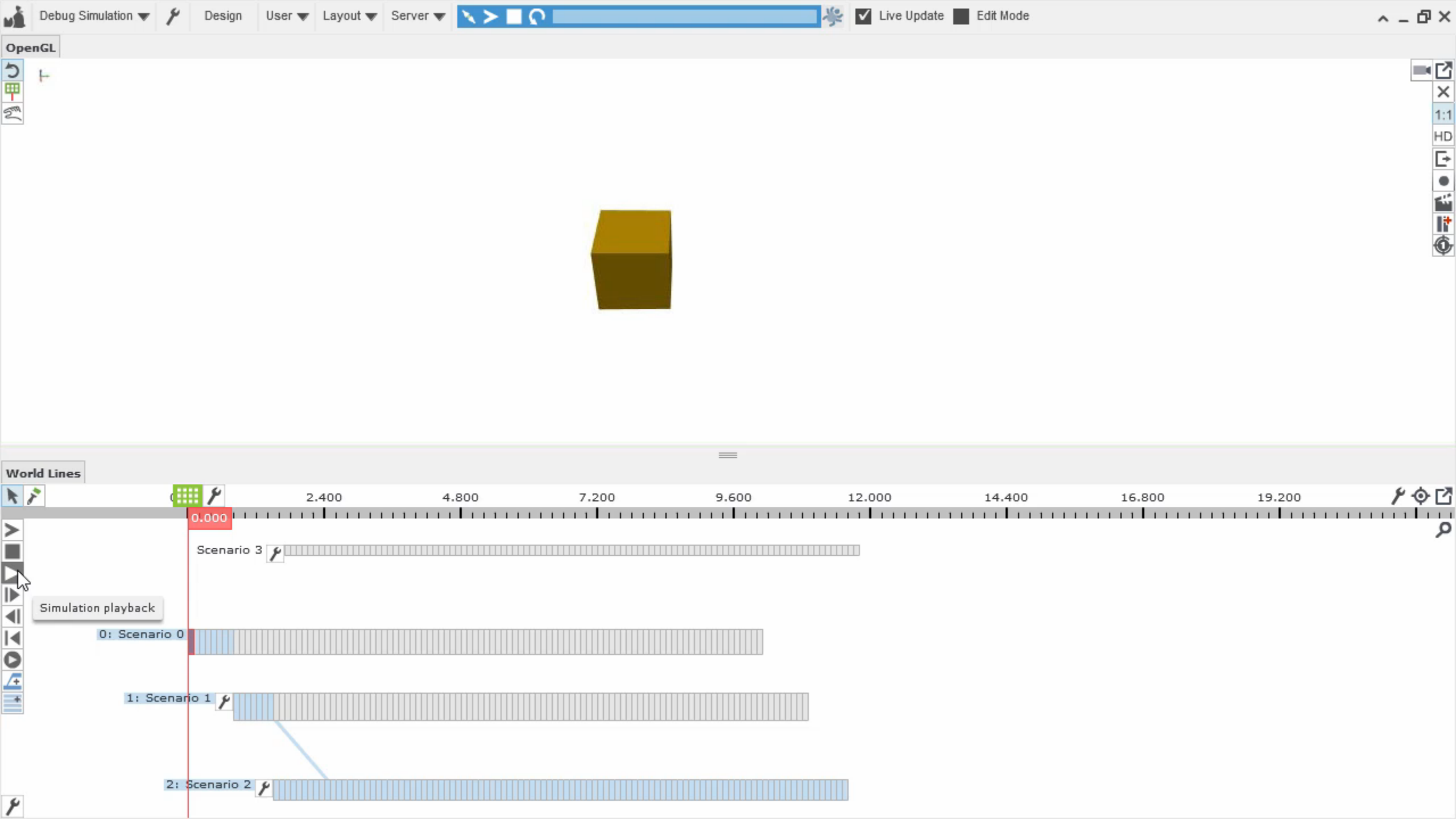
Multiple tracks enable investigation of multiple scenarios at once


Debug Simulation
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
1:1
HD
World Lines
0.000
2.400
4.800
7.200
9.600
12.000
14.400
16.800
19.200
21.6
3.200
0: Scenario 0


Debug Simulation
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
1:1
HD
World Lines
0.000
2.400
4.800
7.200
9.600
12.000
14.400
16.800
19.200
21.6
3.200
Scenario 3
0: Scenario 0
Scenario 1
Scenario 2


Debug Simulation
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
1:1
HD
World Lines
0.000
2.400
4.800
7.200
9.600
12.000
14.400
16.800
19.200
21.6
3.200
Scenario 3
0: Scenario 0
1: Scenario 1
Scenario 2


Debug Simulation
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
1:1
HD
World Lines
0.000
2.400
4.800
7.200
9.600
12.000
14.400
16.800
19.200
21.6
3.200
Scenario 3
0: Scenario 0
1: Scenario 1
General Settings
Object Type
cone
Scenario 2


Debug Simulation
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
1:1
HD
World Lines
2.400
4.800
7.200
9.600
12.000
14.400
16.800
19.200
0.000
Scenario 3
Simulation playback
0: Scenario 0
1: Scenario 1
2: Scenario 2

Implications for Nodes

A single node in the dataflow can have different sets of settings

- Different execution behavior
- Different output data

Aggregation might require simulation of multiple scenarios at once

- Concurrent jobs with different settings and data

Scopes

Unique identifier of a context throughout the dataflow

- int scenarioId: Identifies the scenario
- double timeStep: Optionally identifies the time step
- bool global: Flag to switch between scenario scope and time step scope

Used as key in hash tables

- Hash function computes a uint64 from scenarioId and timeStep
- Per-node output data
- Node settings
- Jobs

Scope-Based Settings

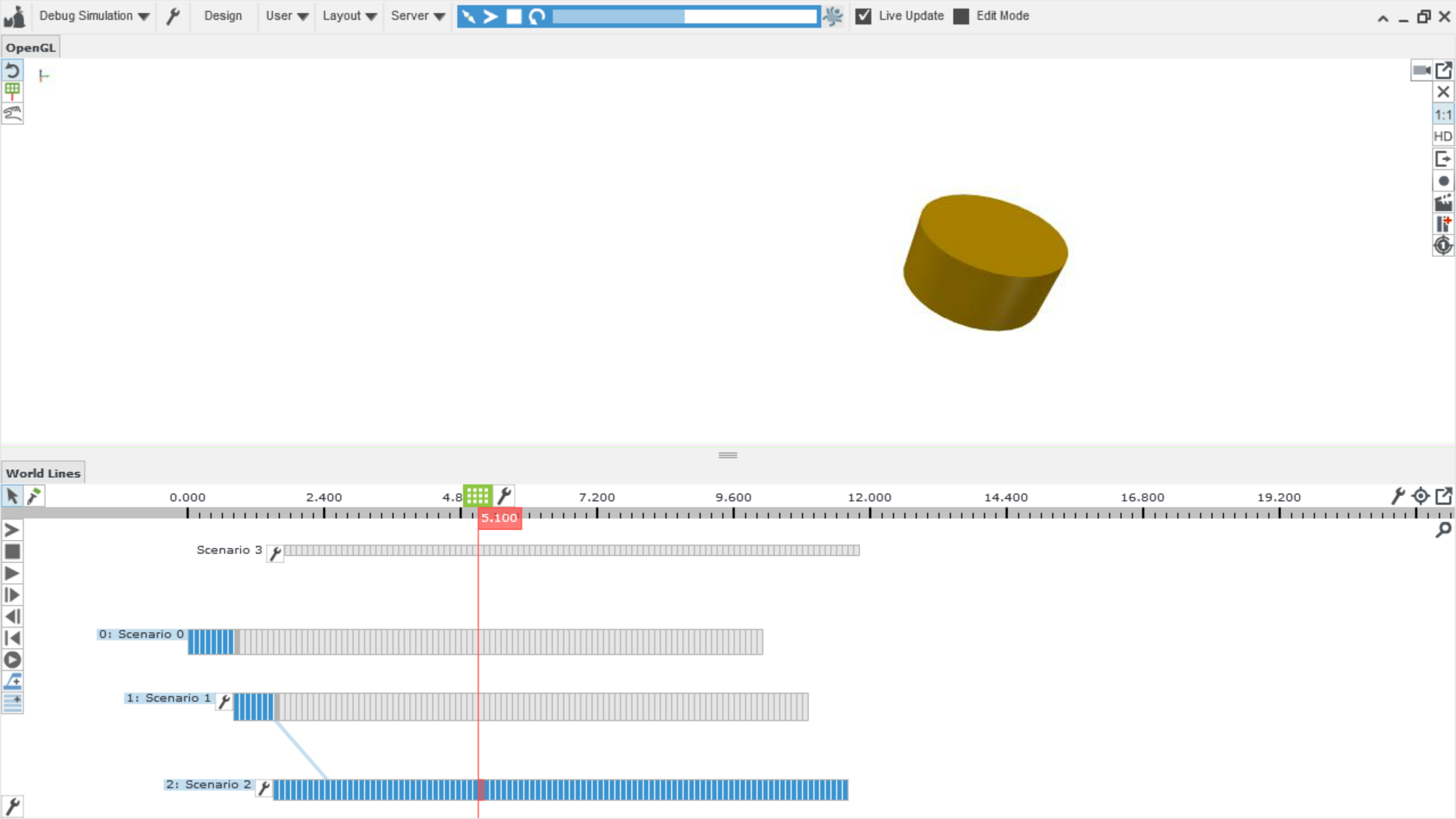
Node settings are defined for a scope

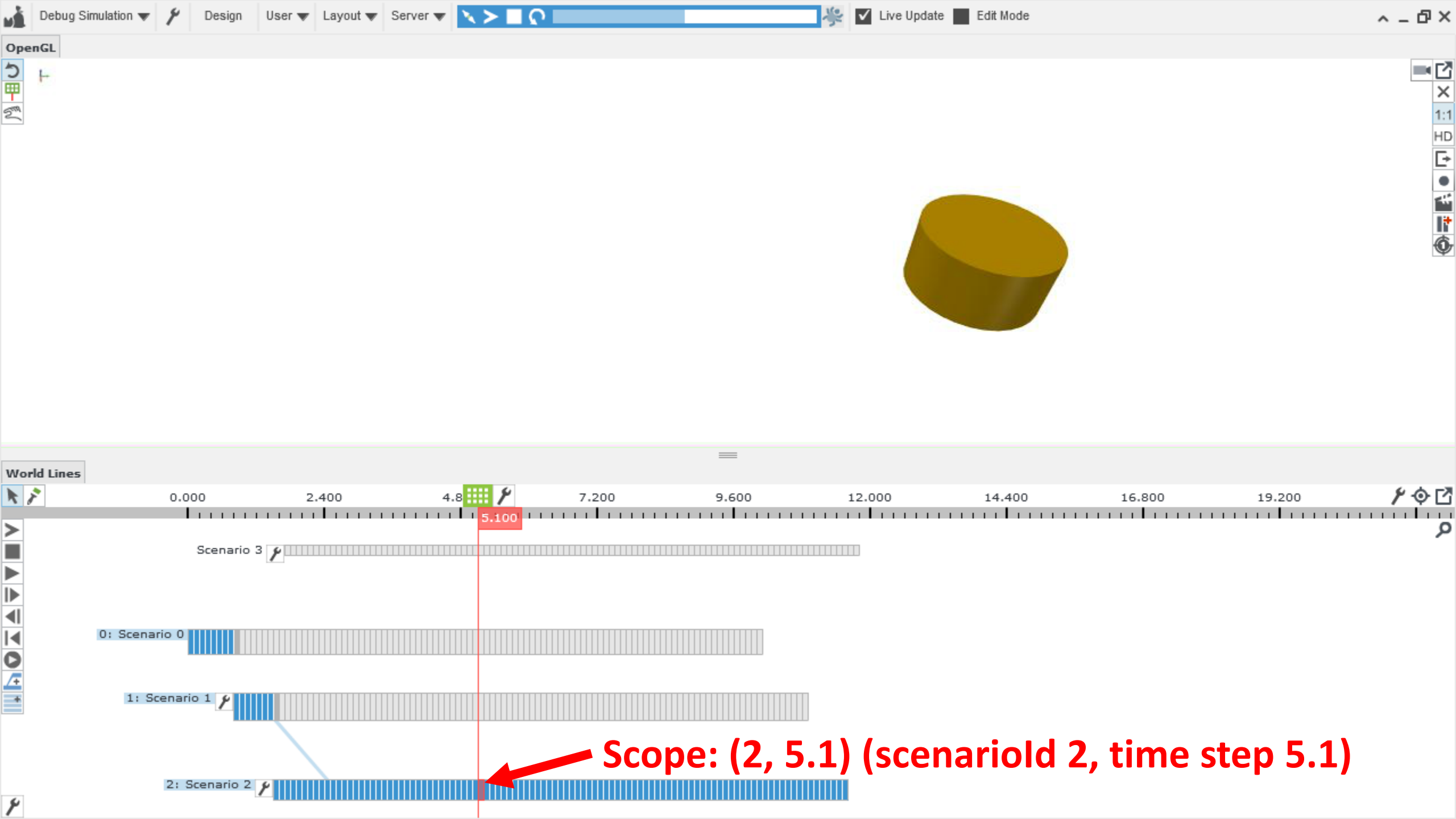
- Per scenario
- Per time step

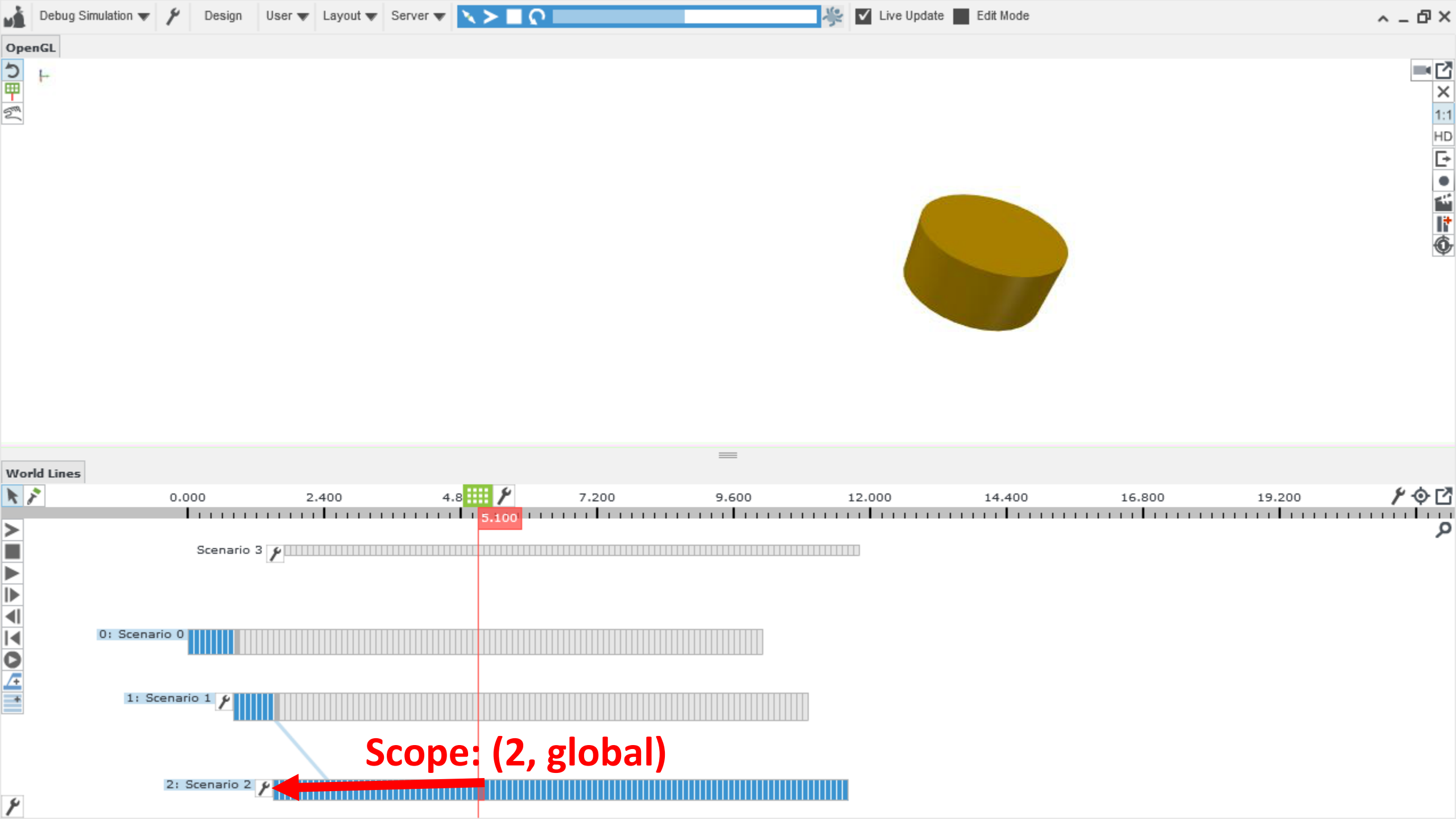
Assignment is specified by developer during app creation

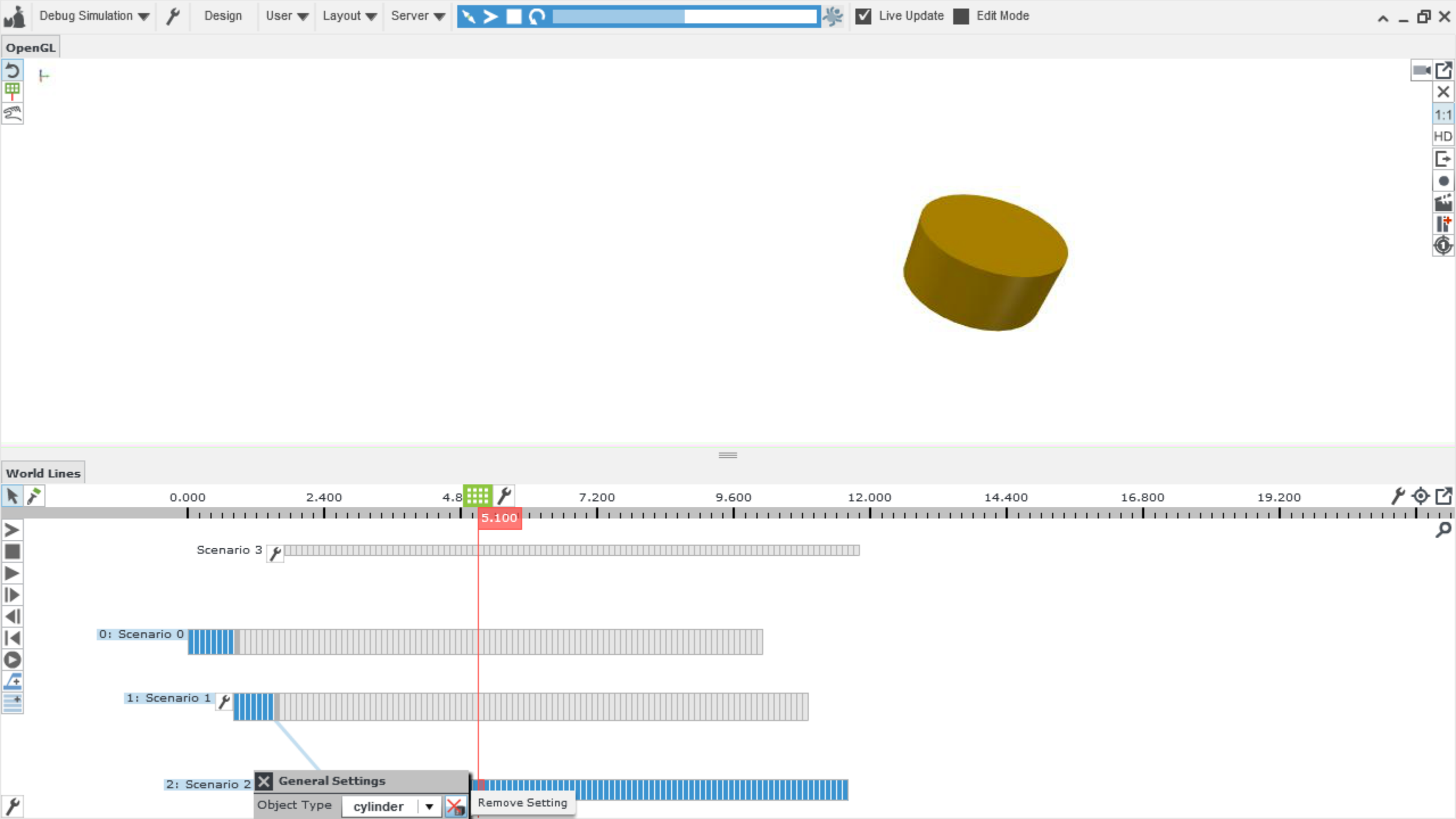
Node holds a settings container

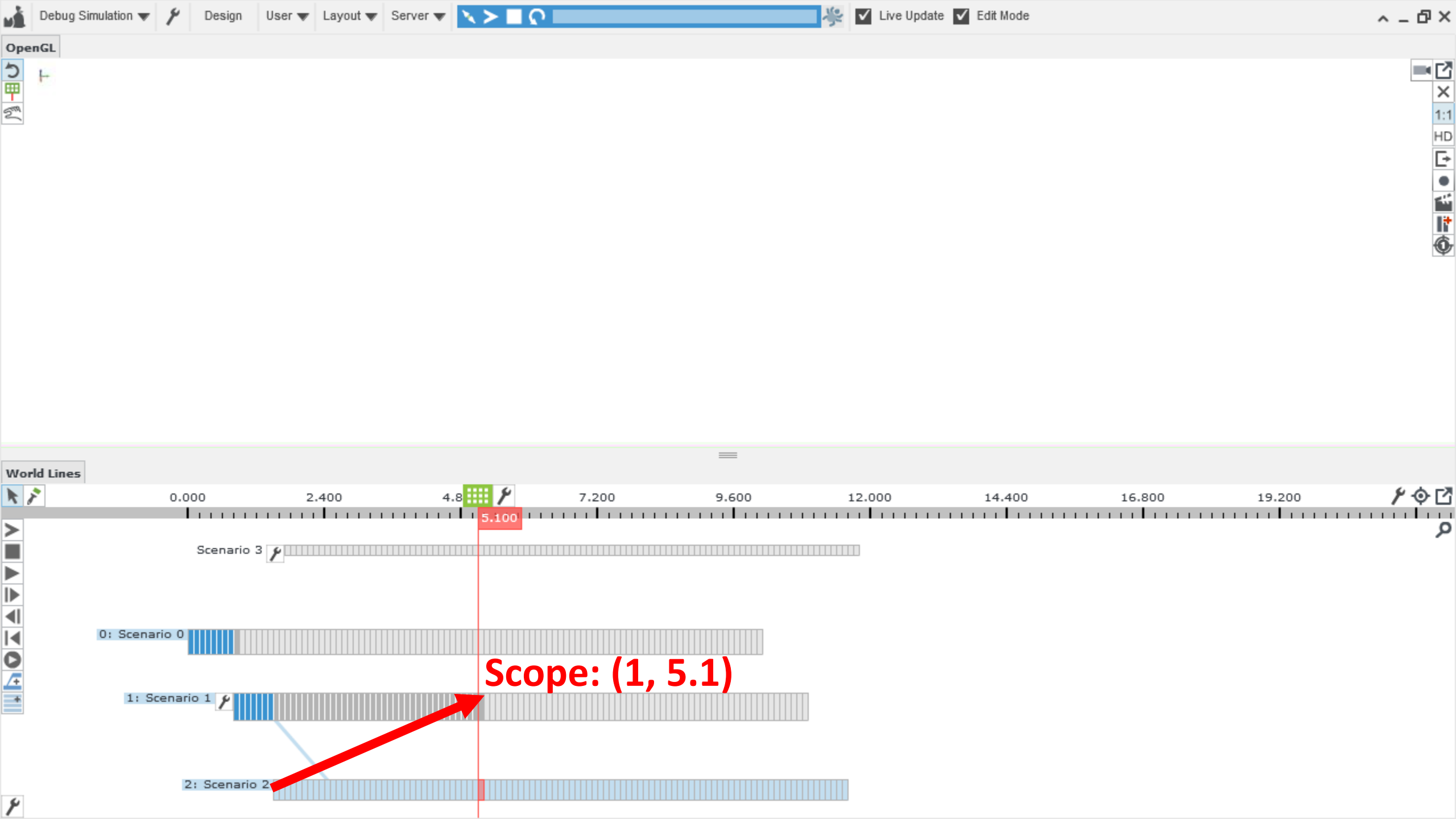
- Hash table indexed by scope hash
- Contains list of settings with name and type
- Not every setting might have a value for each scope

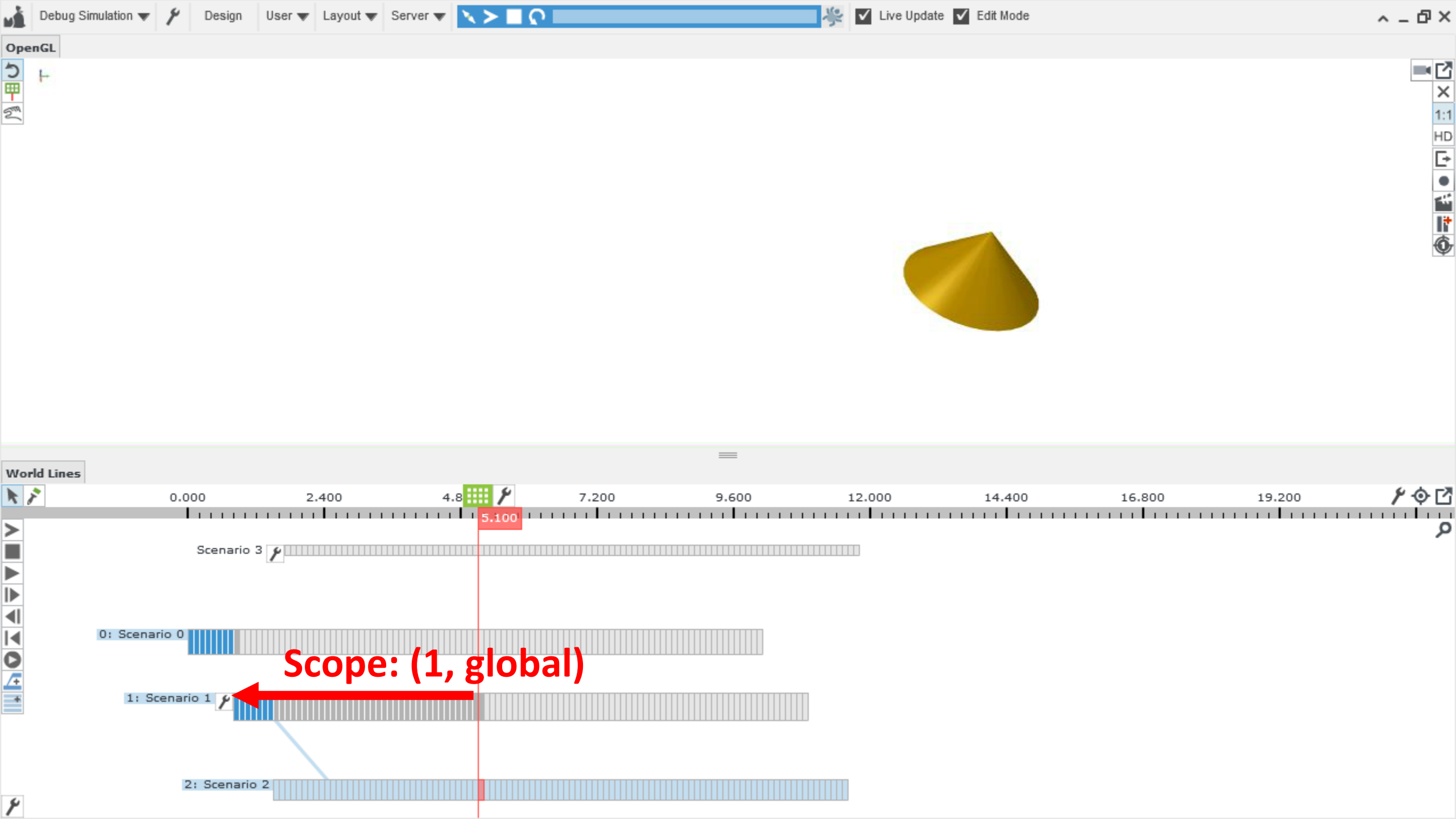

Debug Simulation
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
1:1
HD
World Lines
0.000
2.400
4.8
7.200
9.600
12.000
14.400
16.800
19.200
5.100
Scenario 3
0: Scenario 0
1: Scenario 1
2: Scenario 2

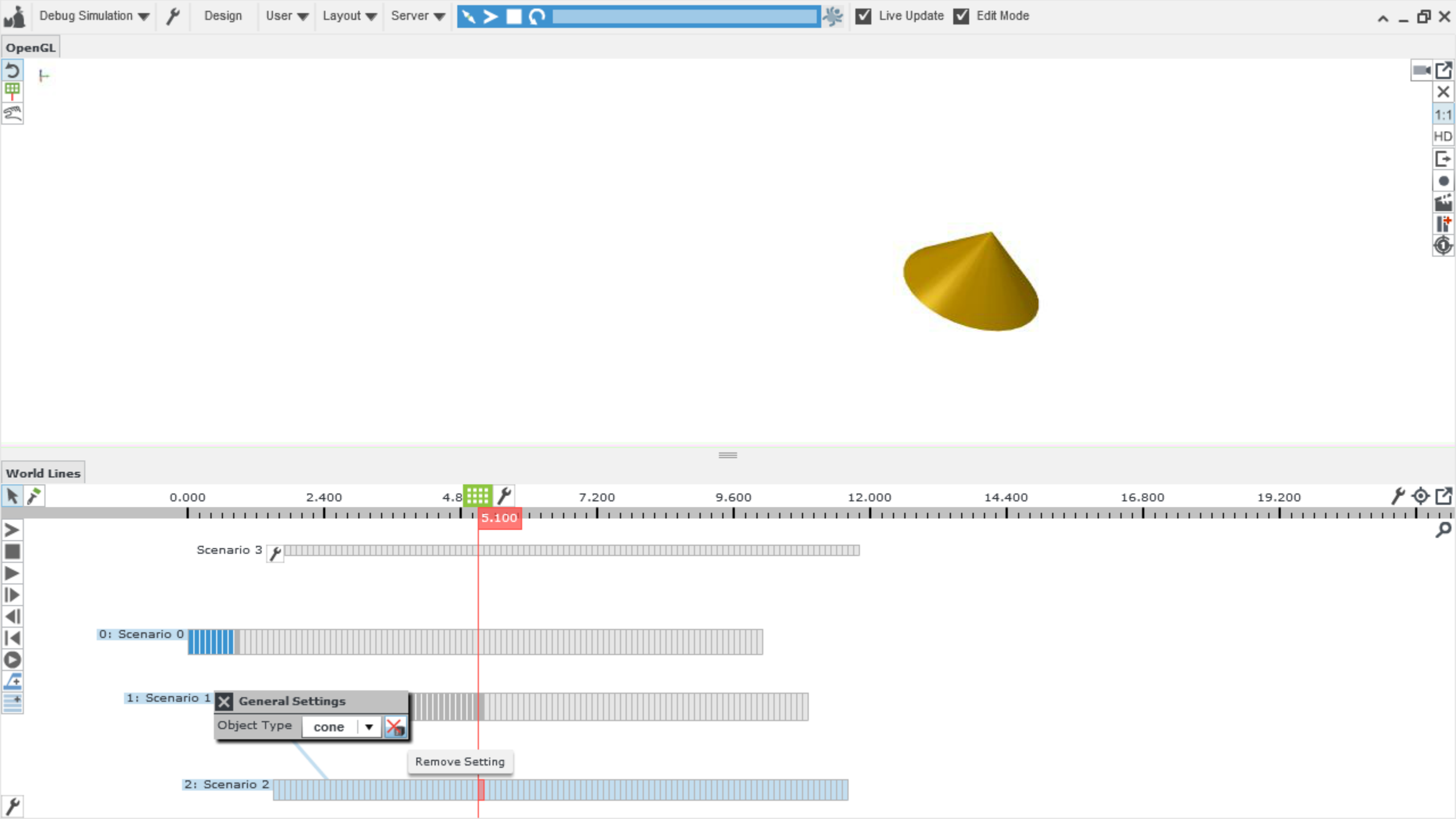

Debug Simulation
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
1:1
HD
World Lines
0.000
2.400
4.8
7.200
9.600
12.000
14.400
16.800
19.200
5.100
Scenario 3
0: Scenario 0
1: Scenario 1
2: Scenario 2
Scope: (2, 5.1) (scenarioId 2, time step 5.1)

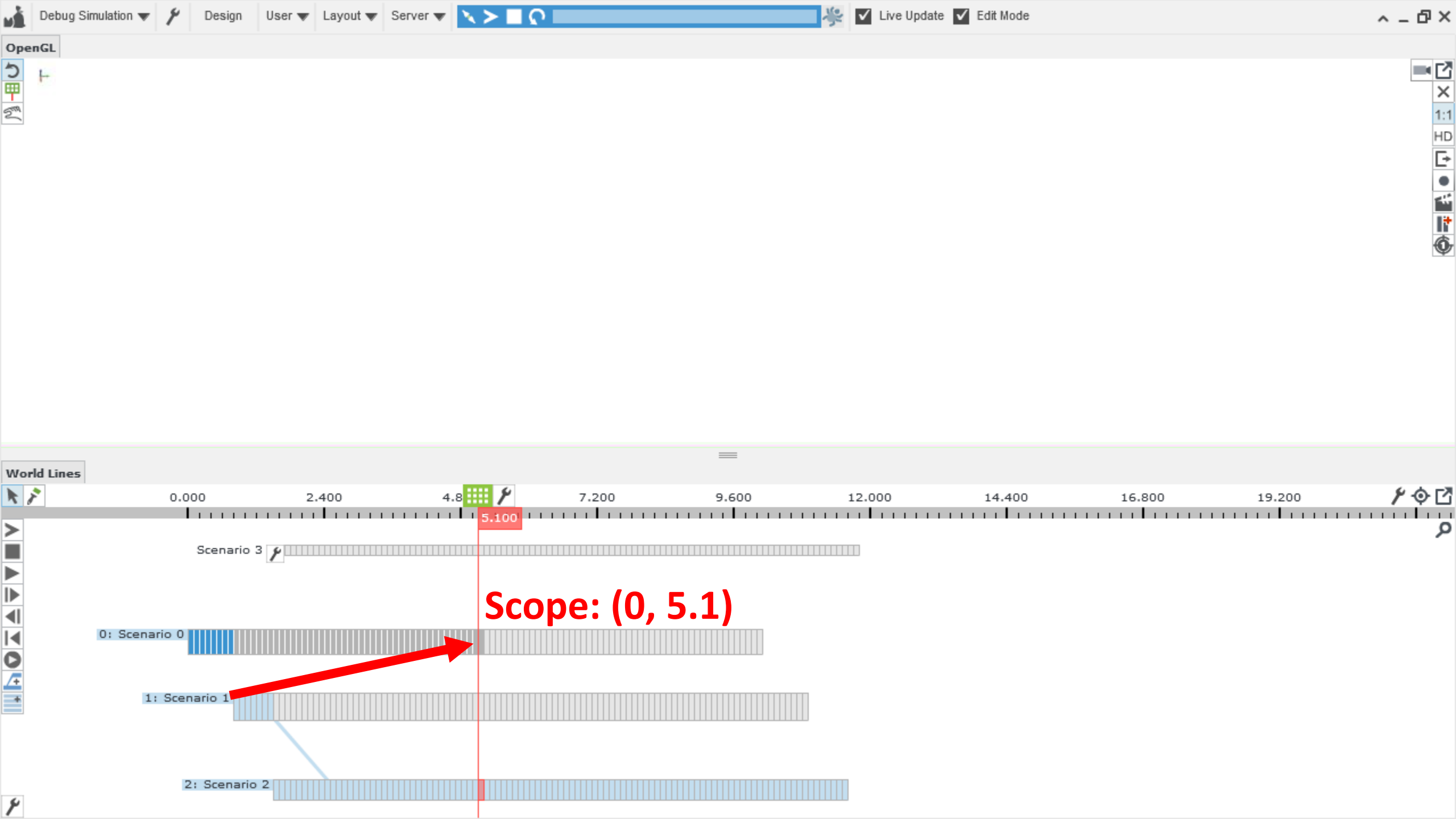

Debug Simulation
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
1:1
HD
World Lines
0.000
2.400
4.8
7.200
9.600
12.000
14.400
16.800
19.200
5.100
Scenario 3
0: Scenario 0
1: Scenario 1
2: Scenario 2
Scope: (2, global)

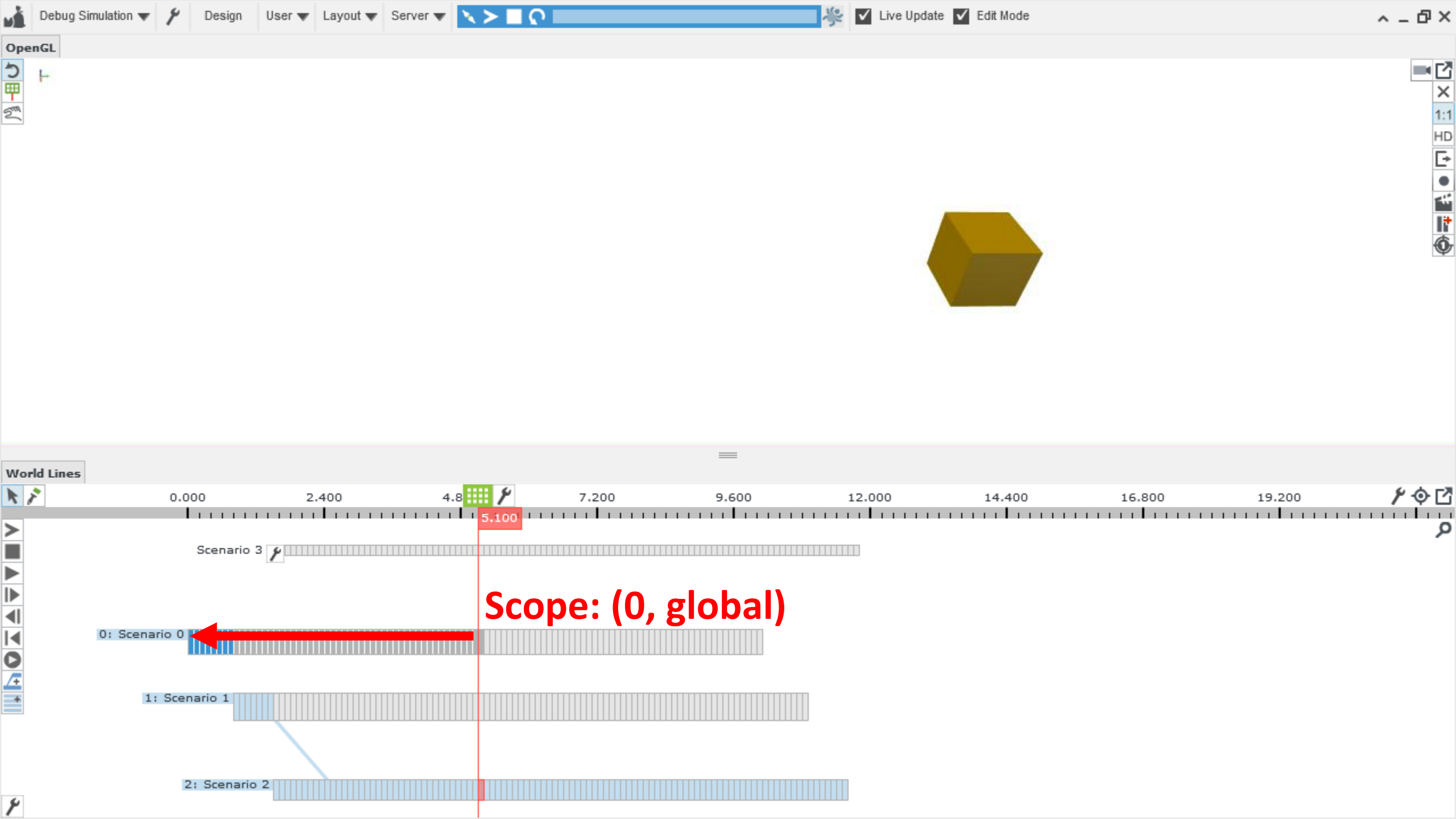

Debug Simulation
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
1:1
HD
World Lines
0.000
2.400
4.8
7.200
9.600
12.000
14.400
16.800
19.200
5.100
Scenario 3
0: Scenario 0
1: Scenario 1
2: Scenario 2
General Settings
Object Type
cylinder
Remove Setting

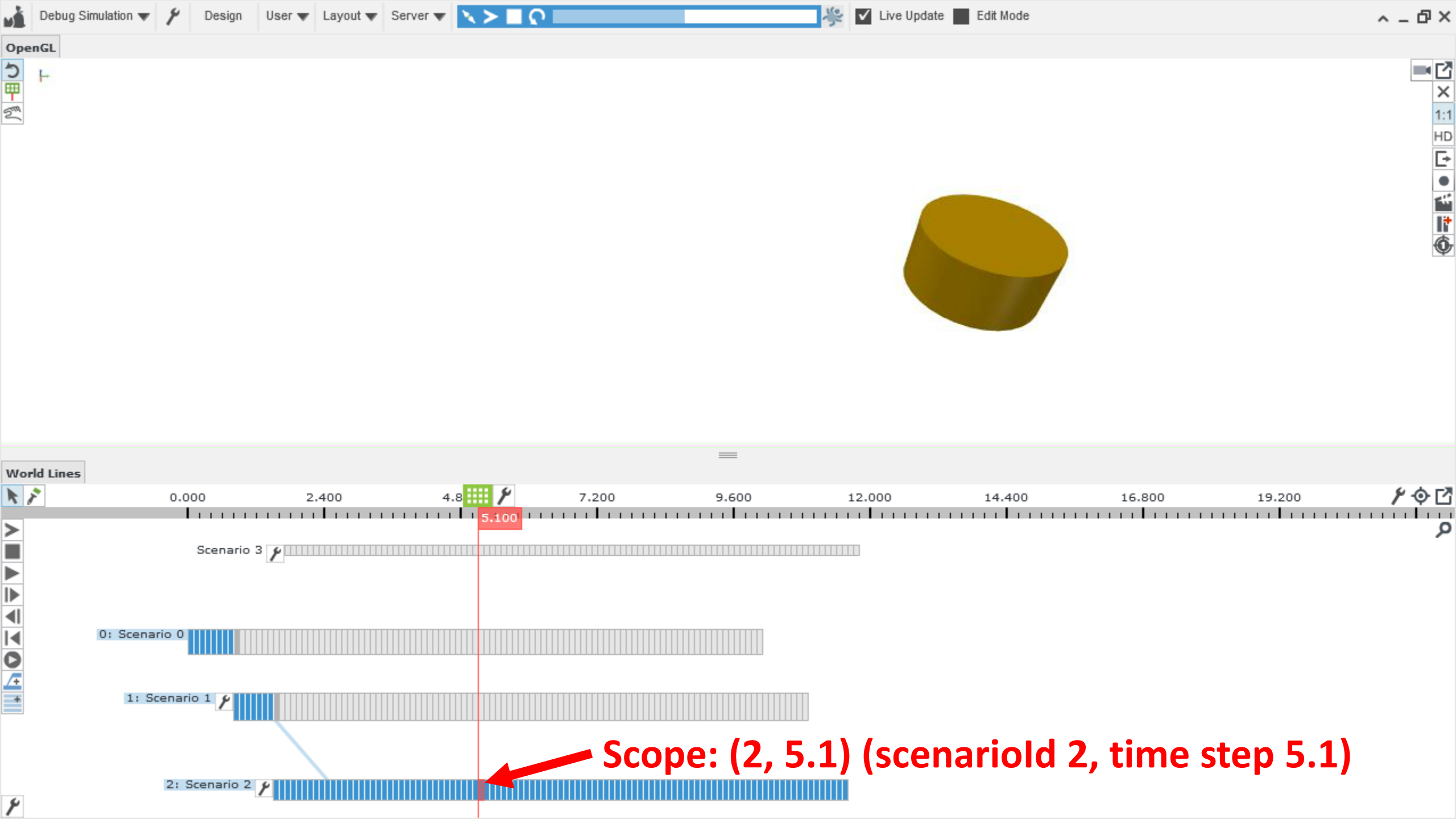

Debug Simulation
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
1:1
HD
World Lines
0.000
2.400
4.8
7.200
9.600
12.000
14.400
16.800
19.200
5.100
Scenario 3
0: Scenario 0
1: Scenario 1
2: Scenario 2
Scope: (1, 5.1)


Debug Simulation
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
1:1
HD
World Lines
0.000
2.400
4.8
7.200
9.600
12.000
14.400
16.800
19.200
5.100
Scenario 3
0: Scenario 0
Scope: (1, global)
1: Scenario 1
2: Scenario 2


Debug Simulation
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
1:1
HD
World Lines
0.000
2.400
4.8
7.200
9.600
12.000
14.400
16.800
19.200
5.100
Scenario 3
0: Scenario 0
1: Scenario 1
General Settings
Object Type
cone
Remove Setting
2: Scenario 2

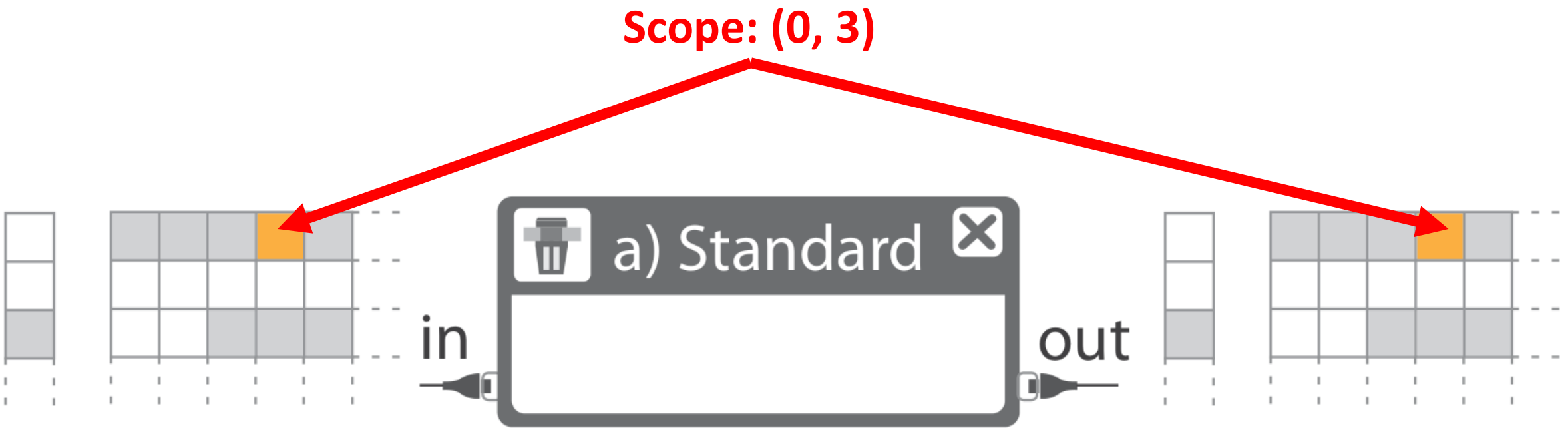

Debug Simulation
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
1:1
HD
World Lines
0.000
2.400
4.8
7.200
9.600
12.000
14.400
16.800
19.200
5.100
Scenario 3
Scope: (0, 5.1)
0: Scenario 0
1: Scenario 1
2: Scenario 2


Debug Simulation
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
1:1
HD
World Lines
0.000
2.400
4.8
7.200
9.600
12.000
14.400
16.800
19.200
5.100
Scenario 3
Scope: (0, global)
0: Scenario 0
1: Scenario 1
2: Scenario 2


Debug Simulation
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
1:1
HD
World Lines
0.000
2.400
4.8
7.200
9.600
12.000
14.400
16.800
19.200
5.100
Scenario 3
0: Scenario 0
1: Scenario 1
2: Scenario 2
Scope: (2, 5.1) (scenarioId 2, time step 5.1)

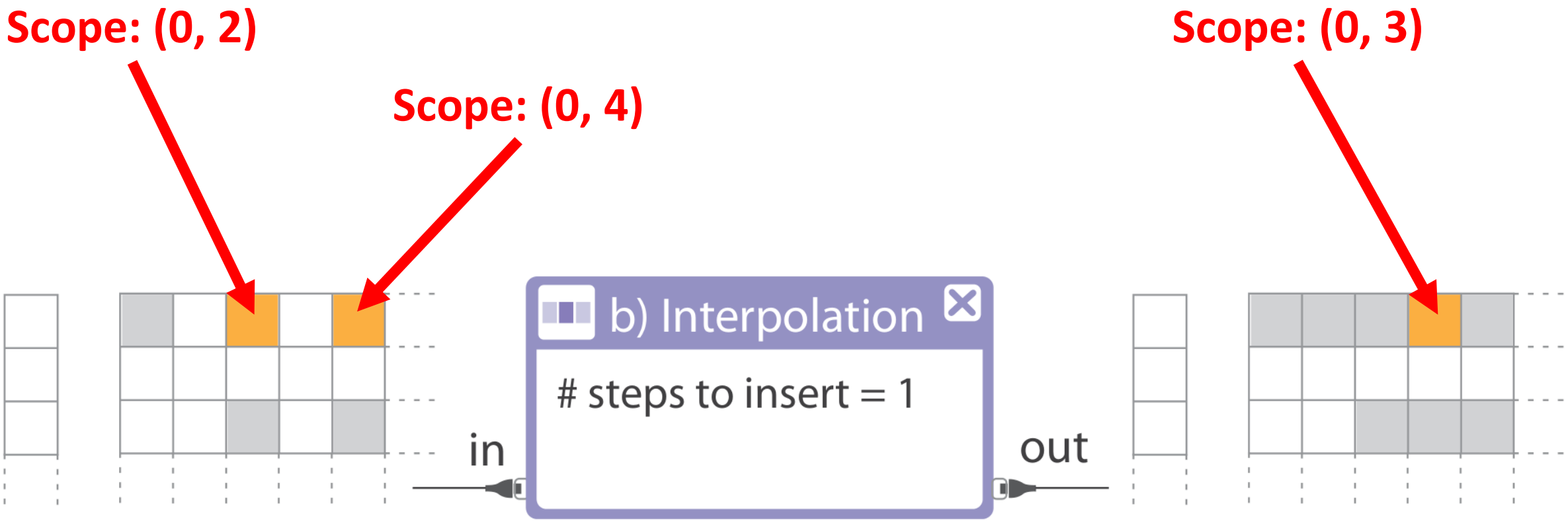
Required Scopes of Nodes

Usually: Required scope of output (scope of job) is scope of input



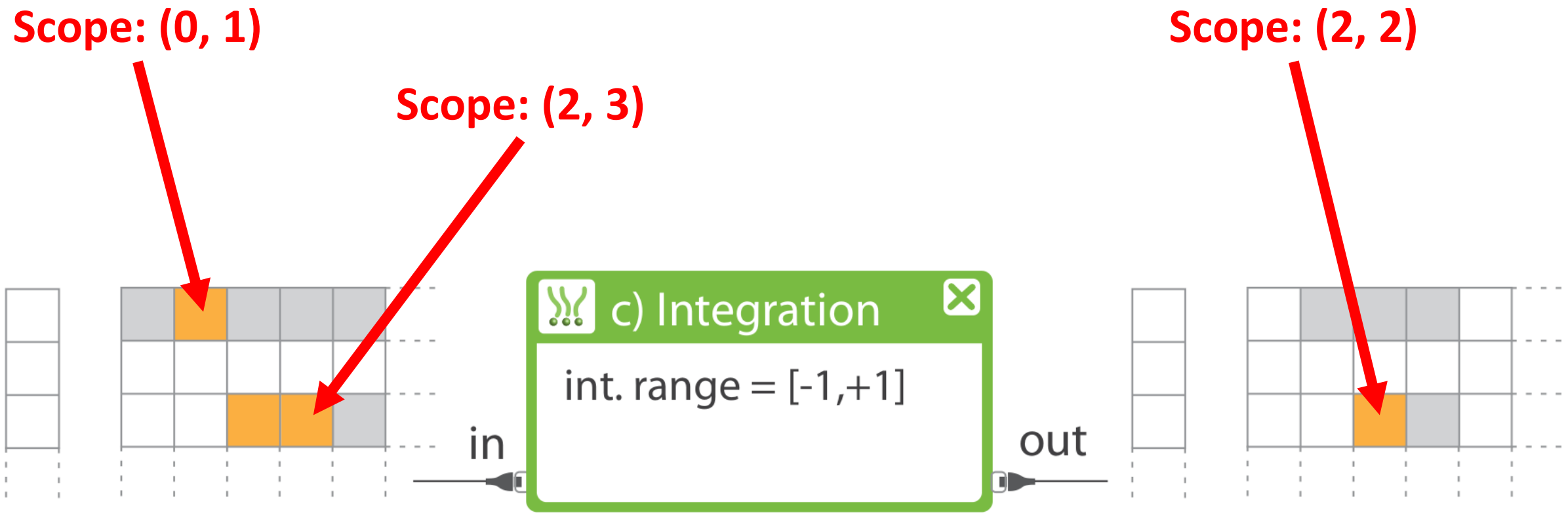
Special Nodes

Time interpolation: Requires input of *n* scopes before and after the active scope



Special Nodes

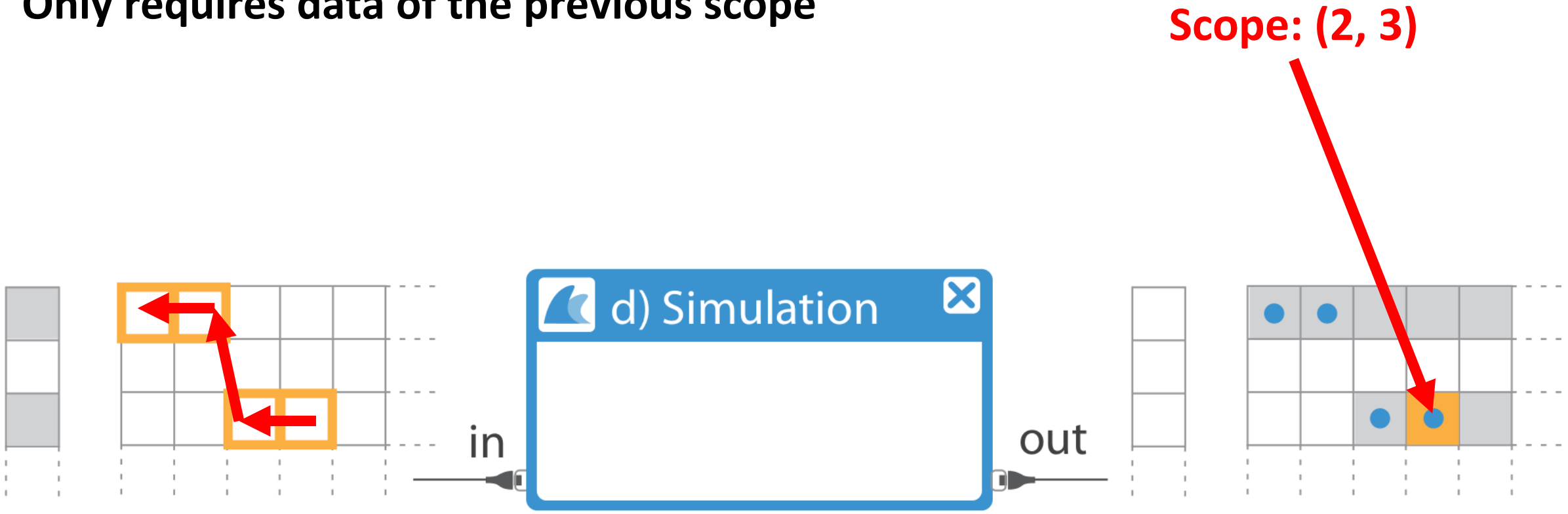
Time integration: Requires data of all scopes within the specified range



Special Nodes

Simulation: Requires all previous scopes to be computed first

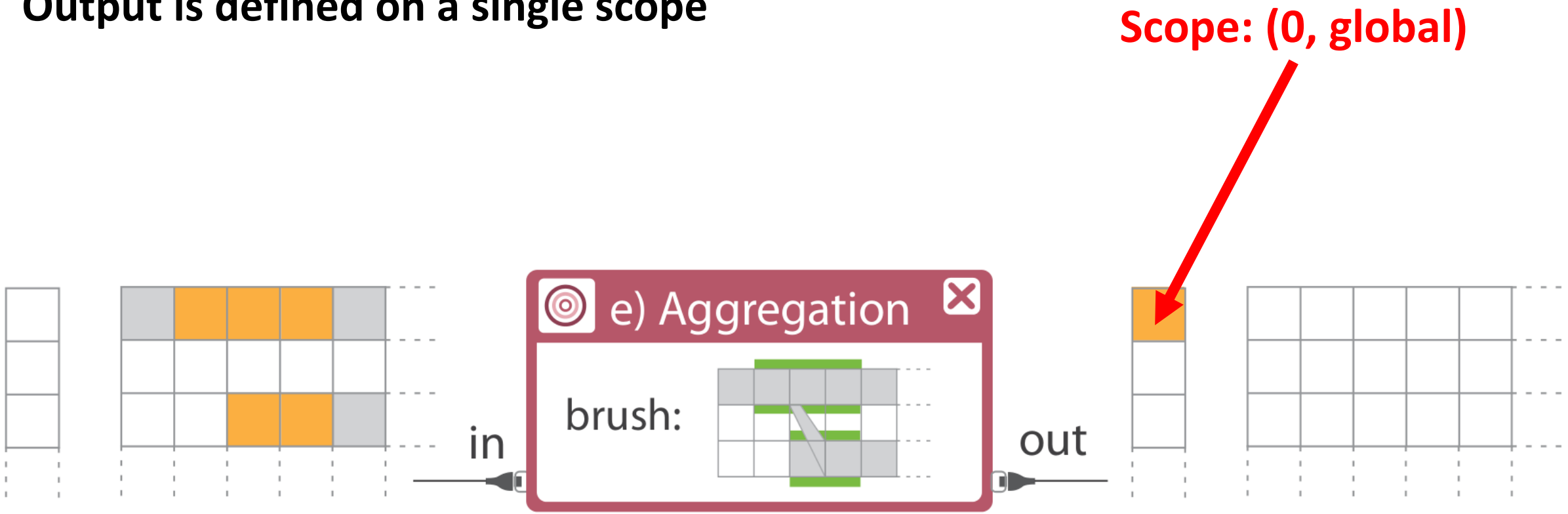
Only requires data of the previous scope



Special Nodes

Aggregation: Requires data of all brushed (selected) scopes across all scenarios

Output is defined on a single scope



Node Implementation

AbstractNode base class

- std::vector<Scope> getRequiredInputScopes(Scope)
- void setData(std::string, Scope, std::shared_ptr<const AbstractData>)
- std::shared_ptr<const AbstractData> getData(std::string, Scope)
- void clearCalculatedScopes()

AbstractNodeExecutor

- void setActiveScope(Scope)
- bool loadDataFromCache()
- **virtual void execute()**

```
void PolygonVertexOrderExecutor::execute()
{









}
```

```
void PolygonVertexOrderExecutor::execute()
{
    const Polygons& inputPolys = mInput->fetch<Polygons>("polygons");









}
```

```
void PolygonVertexOrderExecutor::execute()
{
    const Polygons& inputPolys = mInput->fetch<Polygons>("polygons");
    std::shared_ptr<Polygons> outputPolys = std::make_shared<Polygons>(inputPolys);









}
```

```
void PolygonVertexOrderExecutor::execute()
{
    const Polygons& inputPolys = mInput->fetch<Polygons>("polygons");
    std::shared_ptr<Polygons> outputPolys = std::make_shared<Polygons>(inputPolys);

    auto settings = getSettings<PolygonVertexOrderSettings>();









}
```

```
void PolygonVertexOrderExecutor::execute()
{
    const Polygons& inputPolys = mInput->fetch<Polygons>("polygons");
    std::shared_ptr<Polygons> outputPolys = std::make_shared<Polygons>(inputPolys);

    auto settings = getSettings<PolygonVertexOrderSettings>();
    bool shouldBeCCW = (settings.getVertexOrder().getEnumValue() == VertexOrder::ccw);









}
```

```
void PolygonVertexOrderExecutor::execute()
{
    const Polygons& inputPolys = mInput->fetch<Polygons>("polygons");
    std::shared_ptr<Polygons> outputPolys = std::make_shared<Polygons>(inputPolys);

    auto settings = getSettings<PolygonVertexOrderSettings>();
    bool shouldBeCCW = (settings.getVertexOrder().getEnumValue() == VertexOrder::ccw);

    for (index_t poly = 0; poly < inputPolys.getPolygons().size(); poly++)
    {
        Polygon2D& polygon = outputPolys->getPolygons().at(poly);

        if (polygon.isCCW() != shouldBeCCW)
        {
            polygon.reverseInPlace();
        }
    }


}
```

```
void PolygonVertexOrderExecutor::execute()
{
    const Polygons& inputPolys = mInput->fetch<Polygons>("polygons");
    std::shared_ptr<Polygons> outputPolys = std::make_shared<Polygons>(inputPolys);

    auto settings = getSettings<PolygonVertexOrderSettings>();
    bool shouldBeCCW = (settings.getVertexOrder().getEnumValue() == VertexOrder::ccw);

    for (index_t poly = 0; poly < inputPolys.getPolygons().size(); poly++)
    {
        Polygon2D& polygon = outputPolys->getPolygons().at(poly);

        if (polygon.isCCW() != shouldBeCCW)
        {
            polygon.reverseInPlace();
        }
    }

    setData("polygons", outputPolys);
}
```

```cpp
void PolygonVertexOrderExecutor::execute()
{
    const Polygons& inputPolys = mInput->fetch<Polygons>("polygons");
    std::shared_ptr<Polygons> outputPolys = std::make_shared<Polygons>(inputPolys);

    auto settings = getSettings<PolygonVertexOrderSettings>();
    bool shouldBeCCW = (settings.getVertexOrder().getEnumValue() == VertexOrder::ccw);

    for (index_t poly = 0; poly < inputPolys.getPolygons().size(); poly++)
    {
        Polygon2D& polygon = outputPolys->getPolygons().at(poly);

        if (polygon.isCCW() != shouldBeCCW)
        {
            polygon.reverseInPlace();
        }
    }

    setData("polygons", outputPolys);
}
```

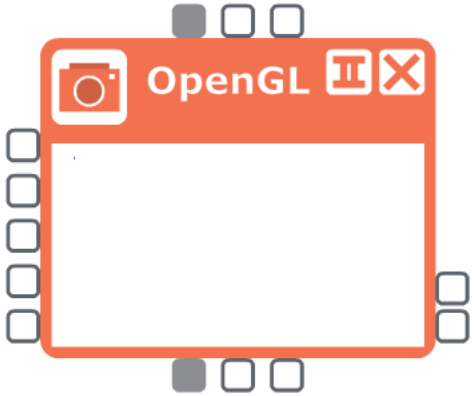
City Flooding | App | User | Layout | Server | Live Update | Edit Mode

Filter Nodes

NODES

- Barriers
- Buildings
- Debugging
- DeprecatedNodes
- Documentation
- Exporters
- FlowControl
- GeoData
- Heightfields
- Hydrology
- Importers
- LinesAndPolygons
- Meshes
- Producers
- Puppetmasters
- Python
- Sampling
- Simulation
- Utilities
- Views
  - Client Info Vis
  - Client Rendering
  - Composite Client Info
  - Console
  - Map
  - OpenGL
  - Plotting
  - Remote View

Search for Nodes | Show Run Times | Reset Run Times | Hide Settings

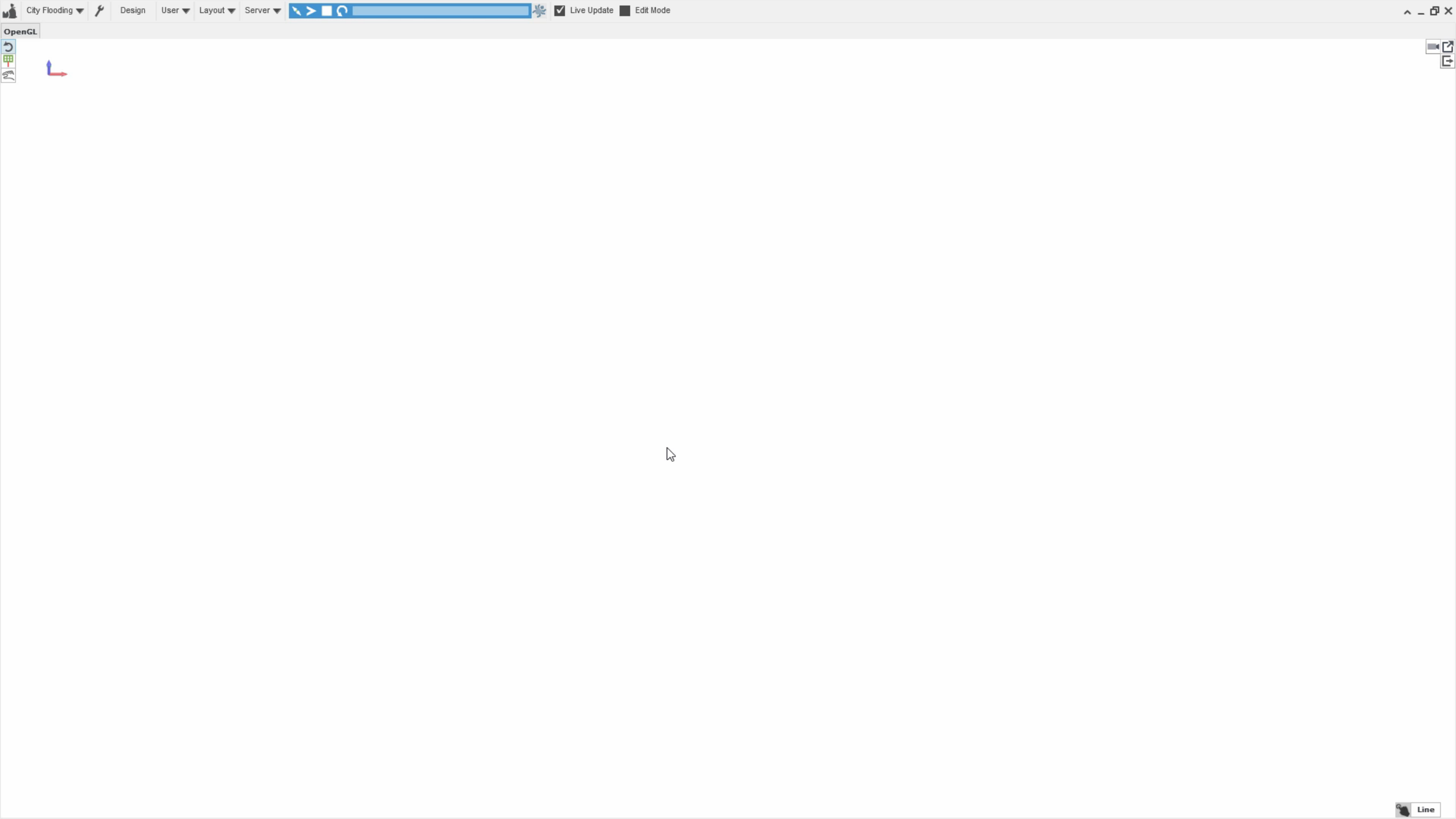


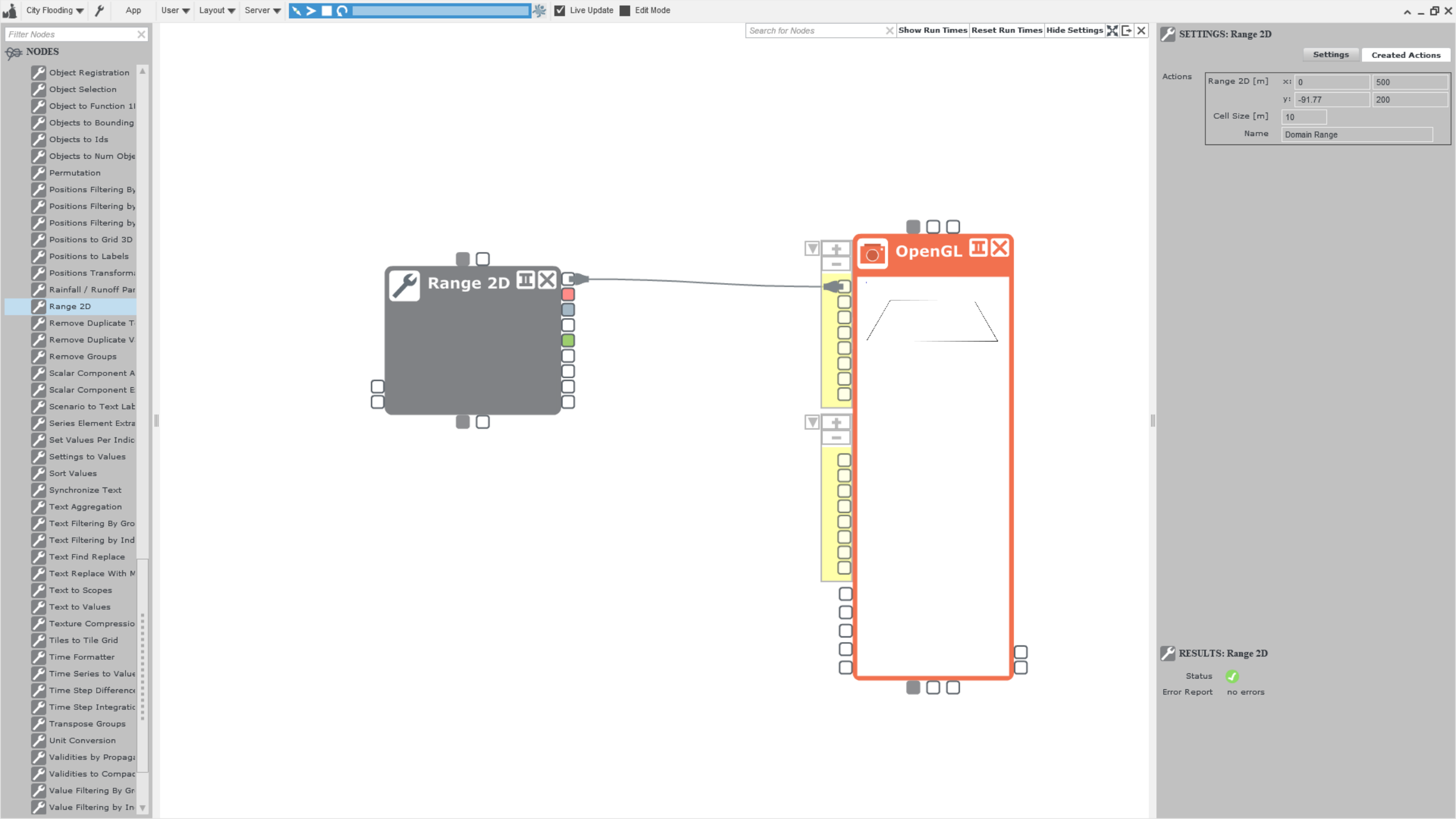
SETTINGS: OpenGL

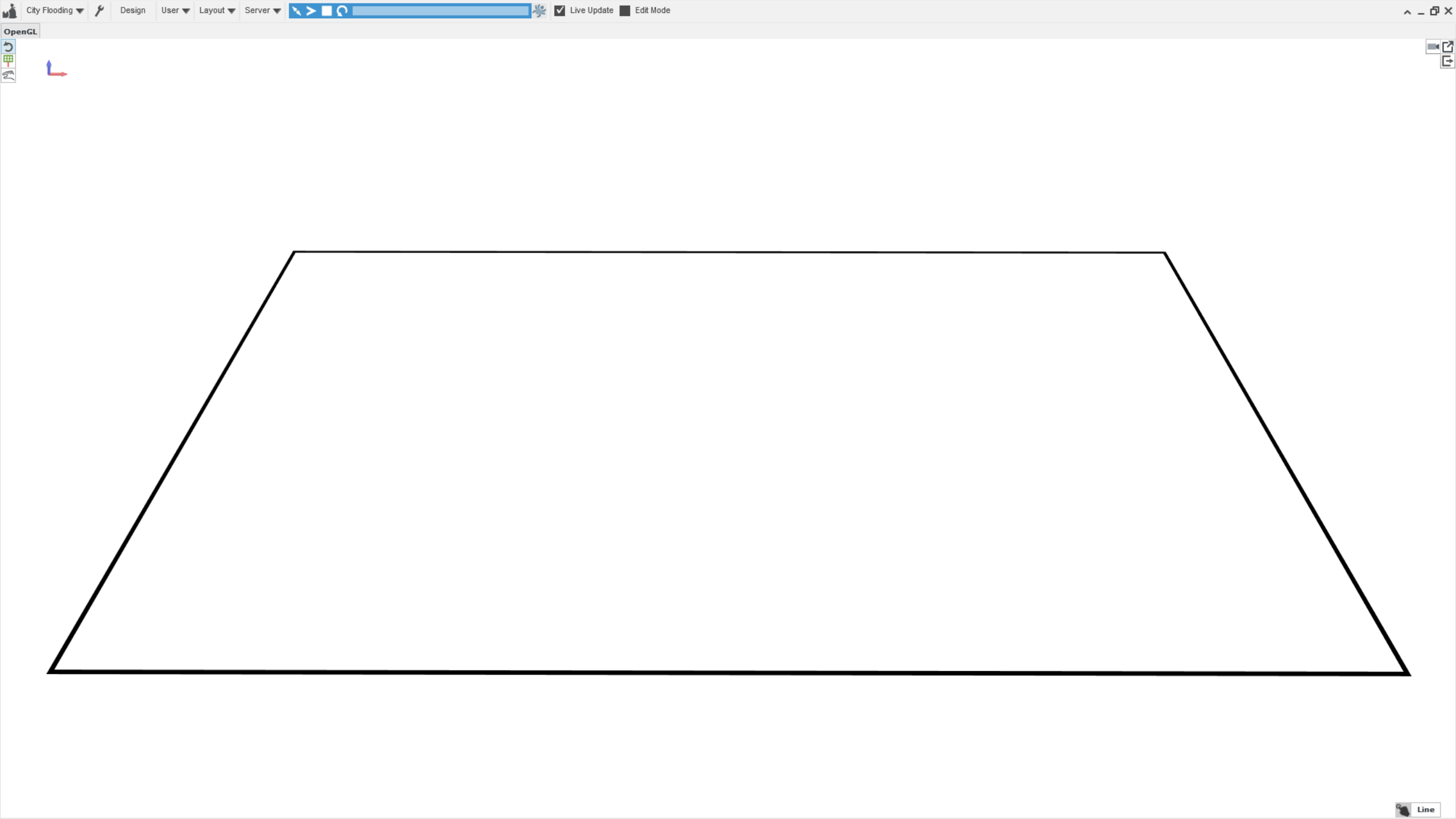
Activated Renderables

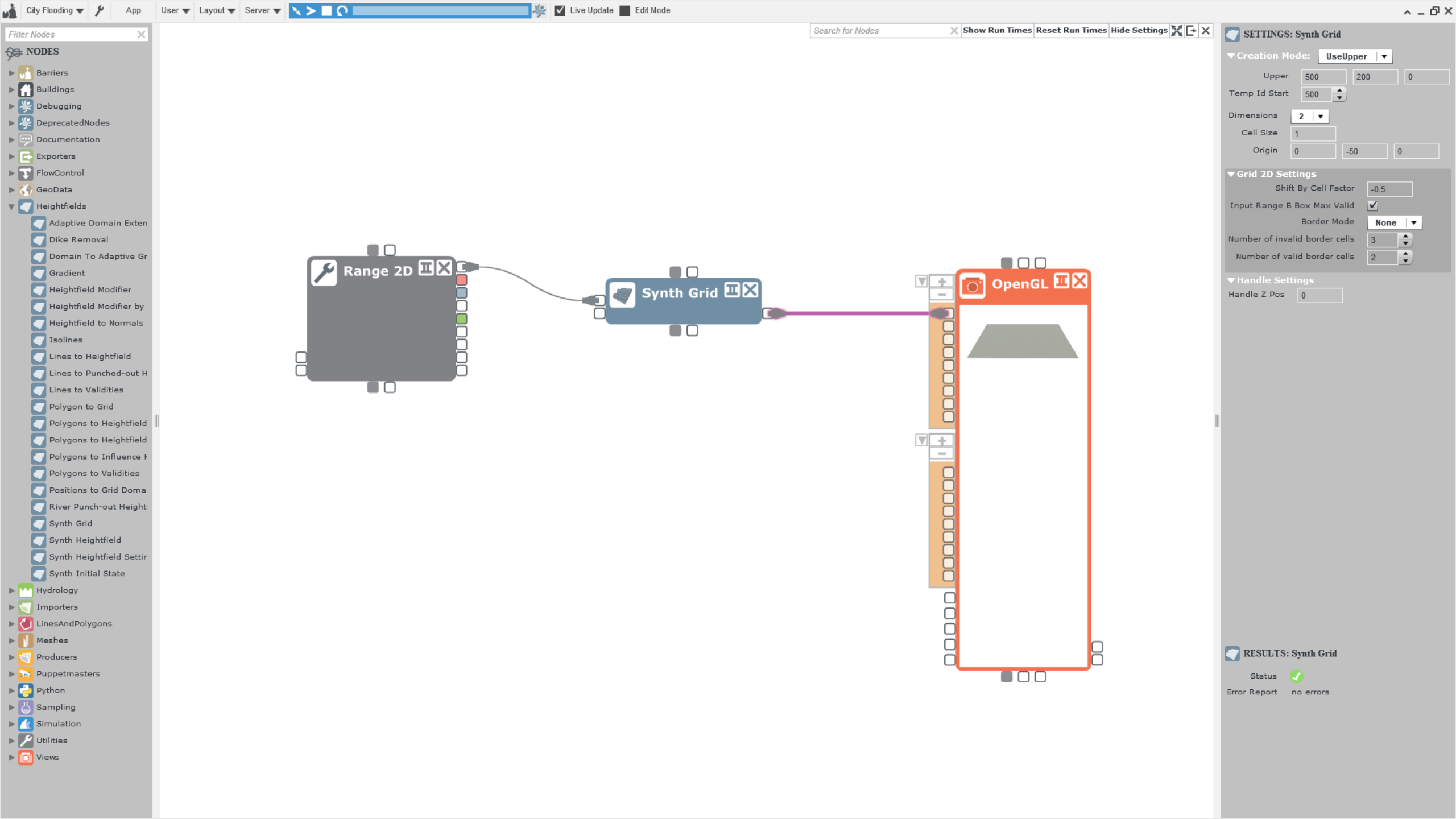
- Agent Renderer
- Background Renderer
- Bounding Box Renderer
- Compass Renderer
- Coord Gizmo Renderer
- Decal Renderer
- Domain Inspector
- Focus Renderer
- Fog Renderer
- Glyph Renderer
- Handle Renderer
- Heightfield Renderer
- Image Renderer
- Label Renderer
- Layer Renderer
- Line Renderer
- Mesh Renderer
- Origin Sphere Renderer
- Overview Renderer
- Particle Point Sprite Renderer
- Point Renderer
- Scale Bar Renderer
- Screen Space Ambient Occlusion Renderer
- Screen Space Anti Aliasing Renderer
- Shadow Renderer
- Skybox Renderer
- Street Label Renderer
- Terrain Renderer
- Text Renderer
- Volume Renderer

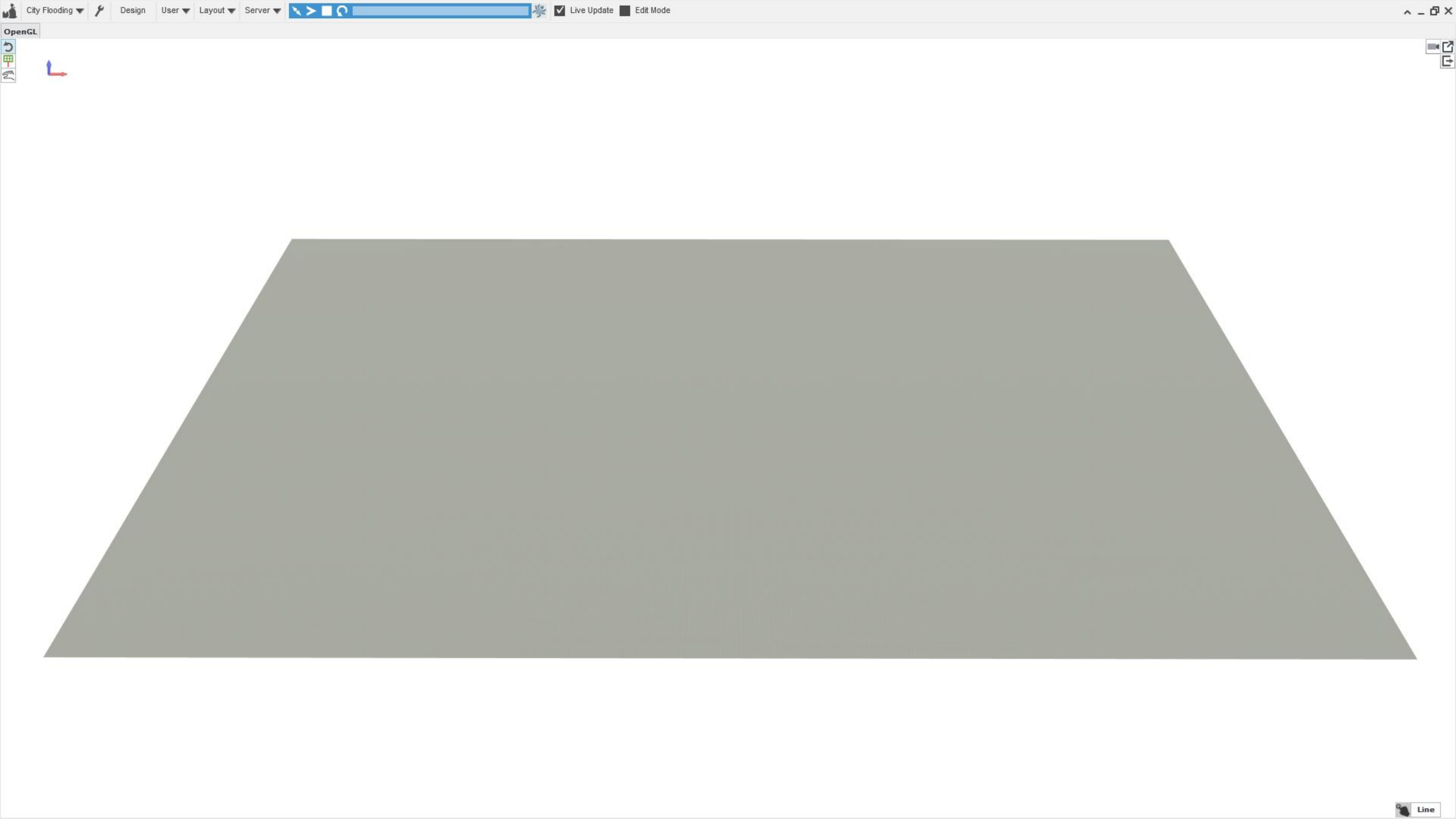
RESULTS: OpenGL

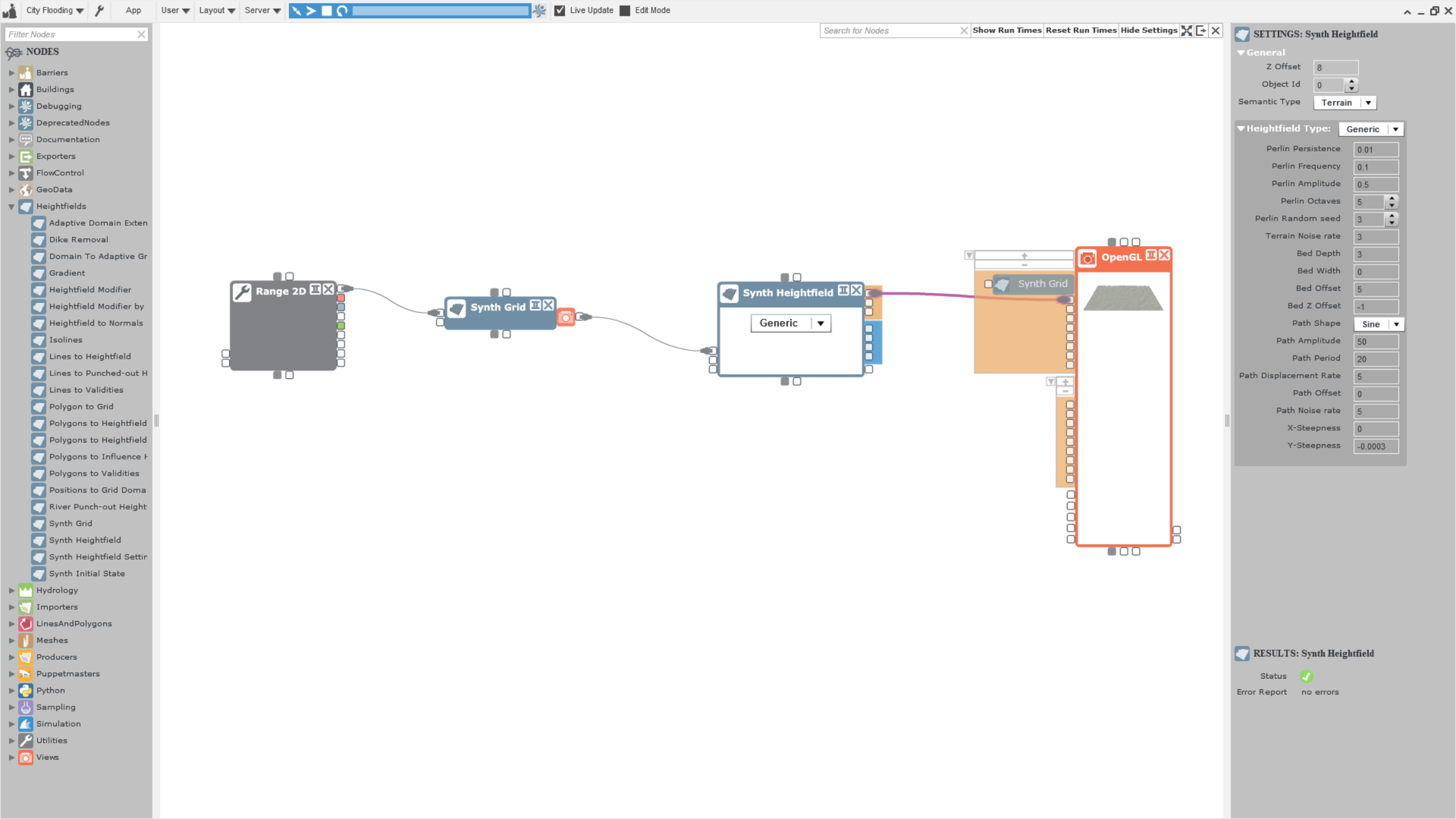

City Flooding
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
Line

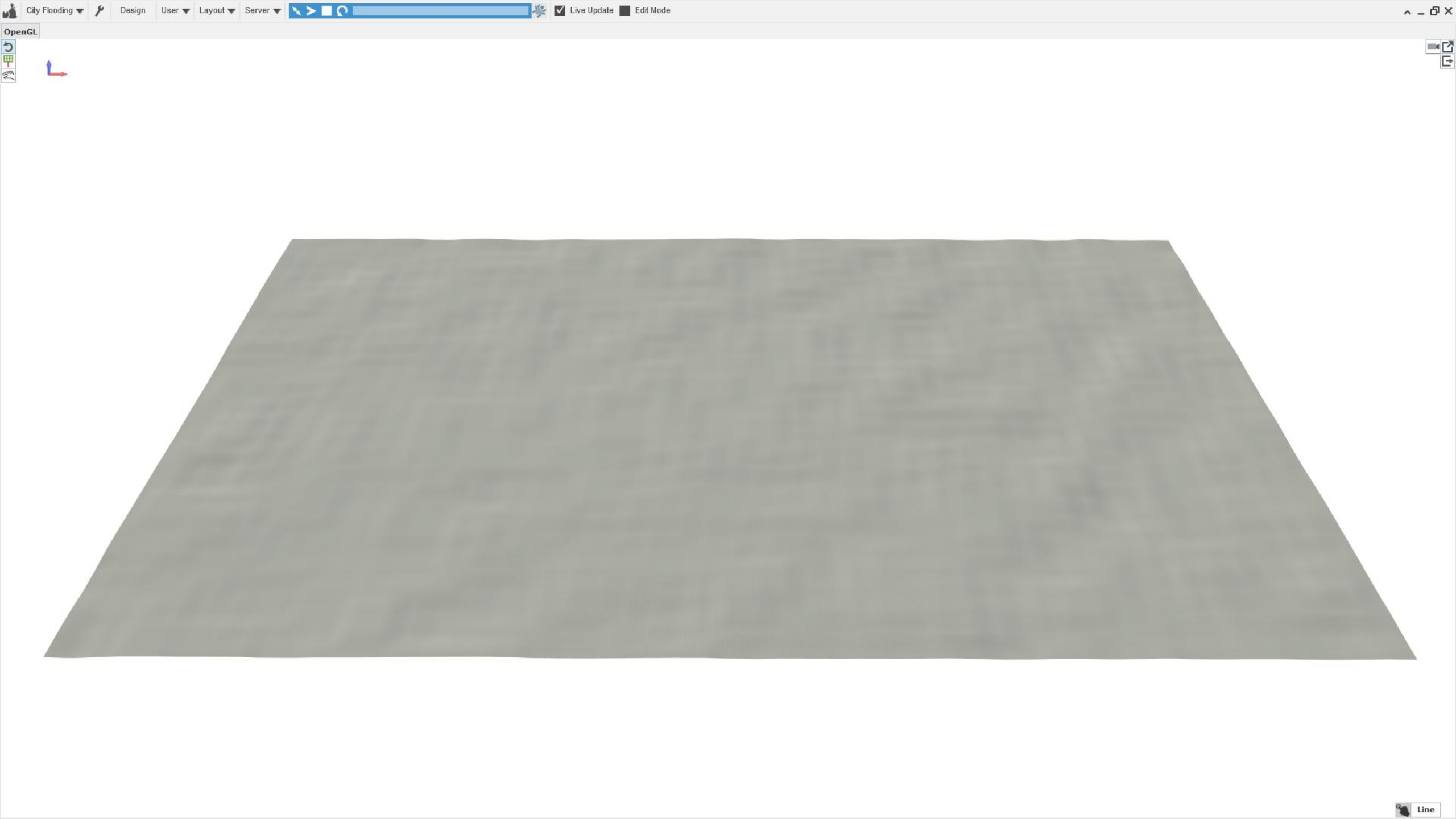

City Flooding
App
User
Layout
Server
Live Update
Edit Mode
Filter Nodes
NODES
Object Registration
Object Selection
Object to Function 1I
Objects to Bounding
Objects to Ids
Objects to Num Obje
Permutation
Positions Filtering By
Positions Filtering by
Positions Filtering by
Positions to Grid 3D
Positions to Labels
Positions Transform
Rainfall / Runoff Par
Range 2D
Remove Duplicate T
Remove Duplicate V
Remove Groups
Scalar Component A
Scalar Component E
Scenario to Text Lab
Series Element Extra
Set Values Per Indic
Settings to Values
Sort Values
Synchronize Text
Text Aggregation
Text Filtering By Gro
Text Filtering by Ind
Text Find Replace
Text Replace With M
Text to Scopes
Text to Values
Texture Compressio
Tiles to Tile Grid
Time Formatter
Time Series to Value
Time Step Difference
Time Step Integratic
Transpose Groups
Unit Conversion
Validities by Propaga
Validities to Compac
Value Filtering By Gr
Value Filtering by In
Search for Nodes
Show Run Times
Reset Run Times
Hide Settings
Range 2D
OpenGL
SETTINGS: Range 2D
Settings
Created Actions
Actions
Range 2D [m]
x: 0
500
y: -91.77
200
Cell Size [m]
10
Name
Domain Range
RESULTS: Range 2D
Status
Error Report
no errors

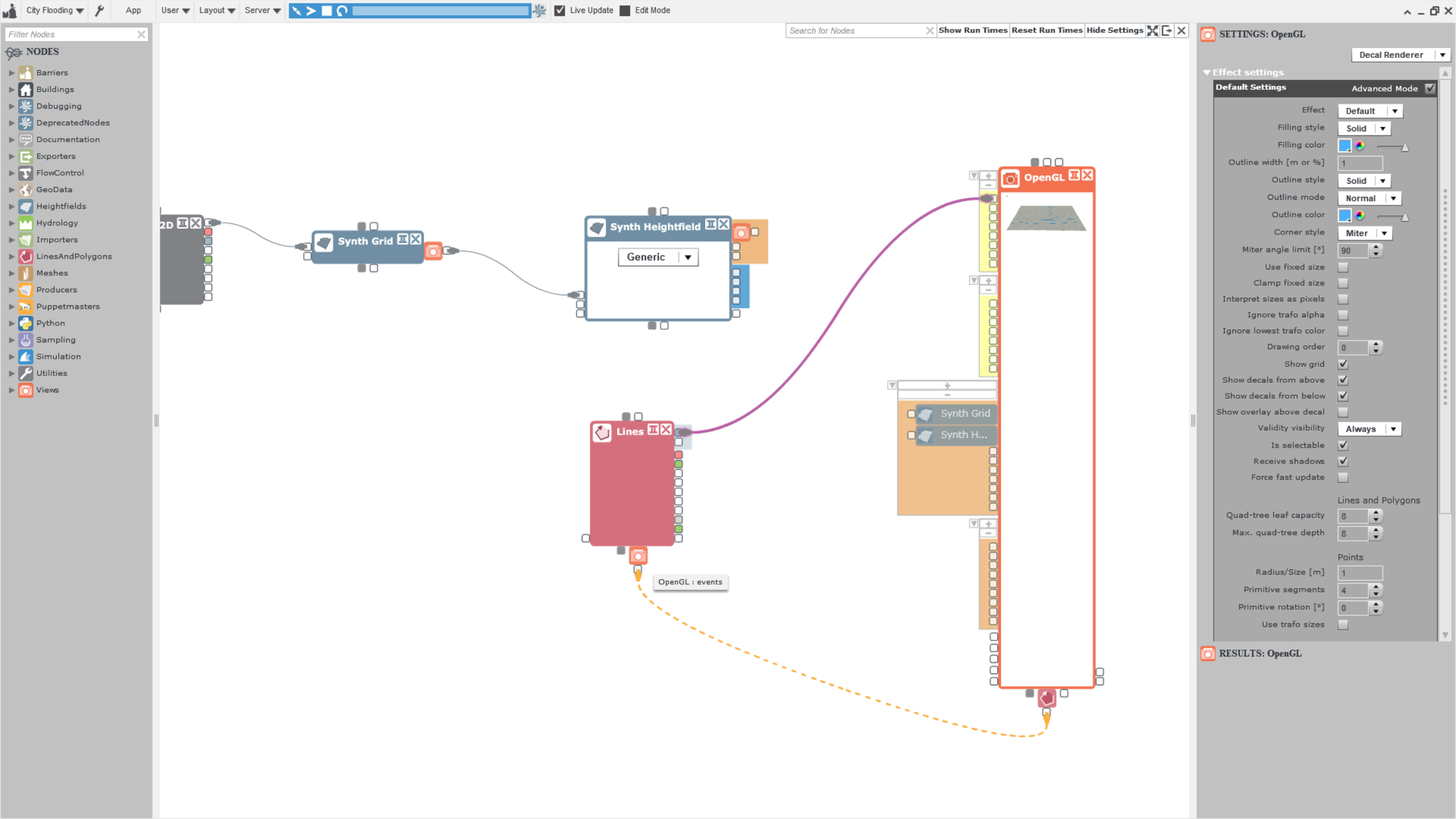

City Flooding
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
Line

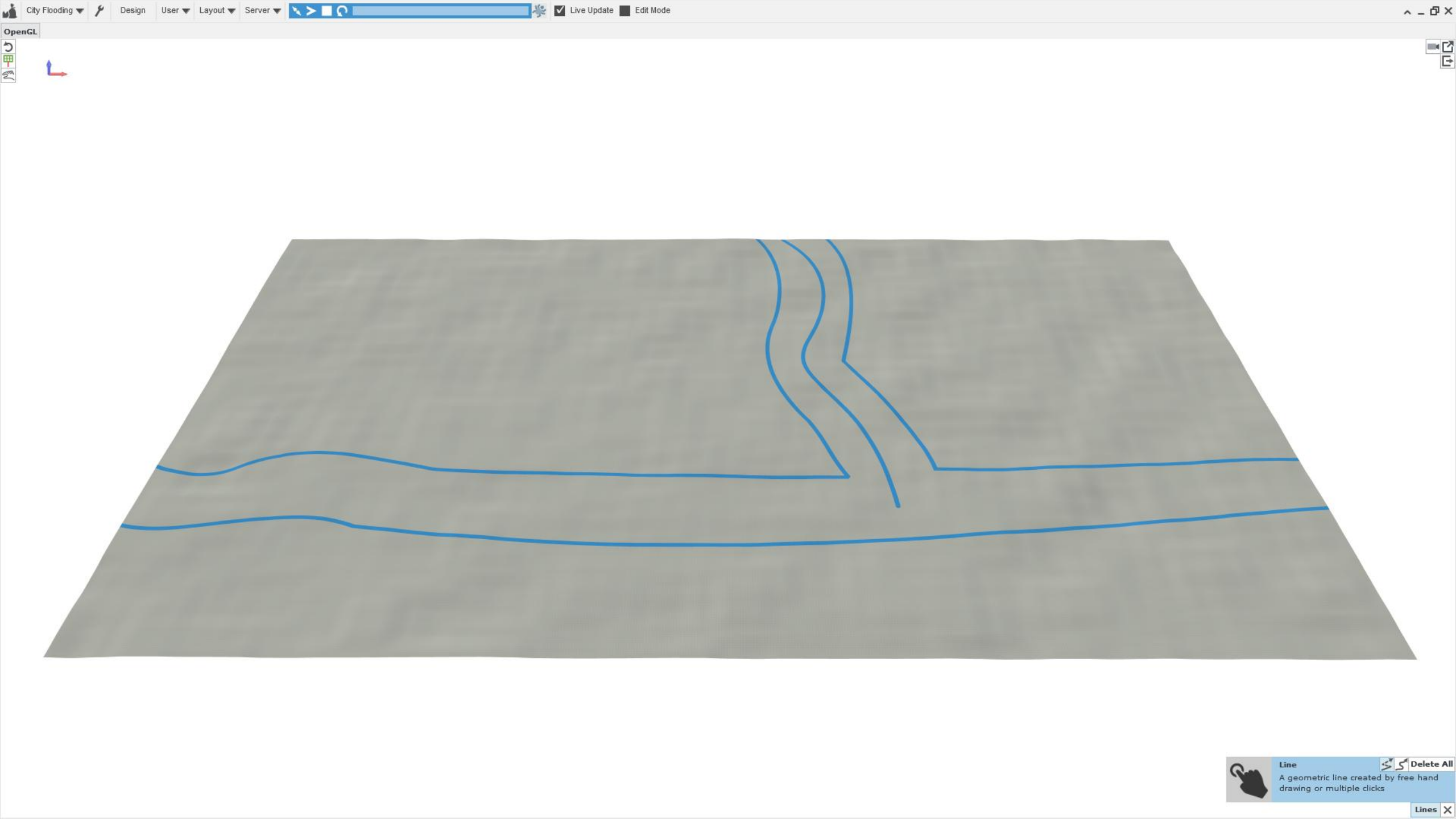

City Flooding
App
User
Layout
Server
Live Update
Edit Mode
Filter Nodes
NODES
Barriers
Buildings
Debugging
DeprecatedNodes
Documentation
Exporters
FlowControl
GeoData
Heightfields
Adaptive Domain Exten
Dike Removal
Domain To Adaptive Gr
Gradient
Heightfield Modifier
Heightfield Modifier by
Heightfield to Normals
Isolines
Lines to Heightfield
Lines to Punched-out H
Lines to Validities
Polygon to Grid
Polygons to Heightfield
Polygons to Heightfield
Polygons to Influence H
Polygons to Validities
Positions to Grid Doma
River Punch-out Height
Synth Grid
Synth Heightfield
Synth Heightfield Settir
Synth Initial State
Hydrology
Importers
LinesAndPolygons
Meshes
Producers
Puppetmasters
Python
Sampling
Simulation
Utilities
Views
Search for Nodes
Show Run Times
Reset Run Times
Hide Settings
Range 2D
Synth Grid
OpenGL
SETTINGS: Synth Grid
Creation Mode:
UseUpper
Upper
500
200
0
Temp Id Start
500
Dimensions
2
Cell Size
1
Origin
0
-50
0
Grid 2D Settings
Shift By Cell Factor
-0.5
Input Range B Box Max Valid
Border Mode
None
Number of invalid border cells
3
Number of valid border cells
2
Handle Settings
Handle Z Pos
0
RESULTS: Synth Grid
Status
Error Report
no errors

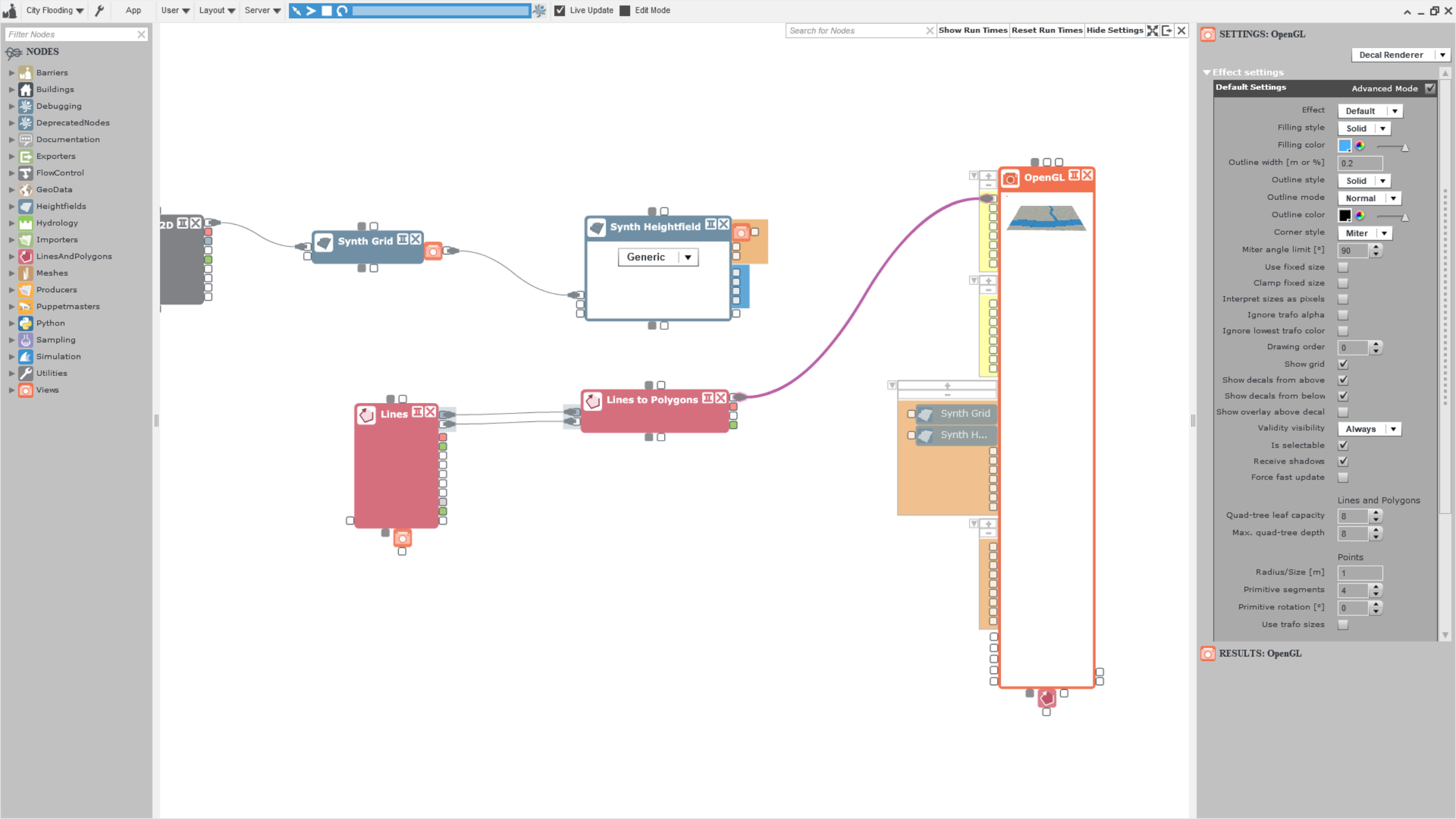

City Flooding
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
Line

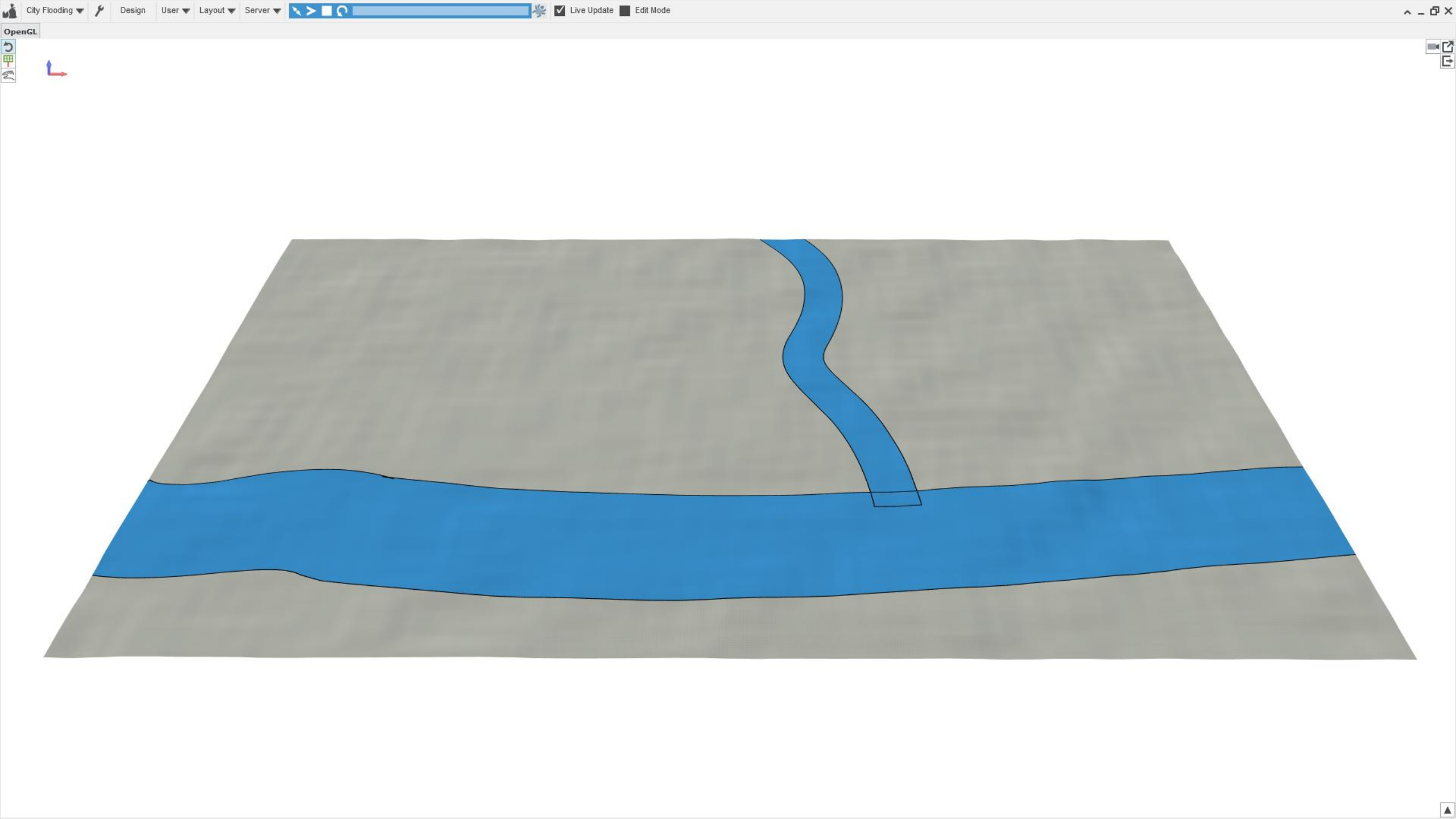

City Flooding
App
User
Layout
Server
Live Update
Edit Mode
Filter Nodes
NODES
Barriers
Buildings
Debugging
DeprecatedNodes
Documentation
Exporters
FlowControl
GeoData
Heightfields
Adaptive Domain Exten
Dike Removal
Domain To Adaptive Gr
Gradient
Heightfield Modifier
Heightfield Modifier by
Heightfield to Normals
Isolines
Lines to Heightfield
Lines to Punched-out H
Lines to Validities
Polygon to Grid
Polygons to Heightfield
Polygons to Heightfield
Polygons to Influence H
Polygons to Validities
Positions to Grid Doma
River Punch-out Height
Synth Grid
Synth Heightfield
Synth Heightfield Settir
Synth Initial State
Hydrology
Importers
LinesAndPolygons
Meshes
Producers
Puppetmasters
Python
Sampling
Simulation
Utilities
Views
Search for Nodes
Show Run Times
Reset Run Times
Hide Settings
Range 2D
Synth Grid
Synth Heightfield
Generic
Synth Grid
OpenGL
SETTINGS: Synth Heightfield
General
Z Offset
8
Object Id
0
Semantic Type
Terrain
Heightfield Type:
Generic
Perlin Persistence
0.01
Perlin Frequency
0.1
Perlin Amplitude
0.5
Perlin Octaves
5
Perlin Random seed
3
Terrain Noise rate
3
Bed Depth
3
Bed Width
0
Bed Offset
5
Bed Z Offset
-1
Path Shape
Sine
Path Amplitude
50
Path Period
20
Path Displacement Rate
5
Path Offset
0
Path Noise rate
5
X-Steepness
0
Y-Steepness
-0.0003
RESULTS: Synth Heightfield
Status
Error Report
no errors

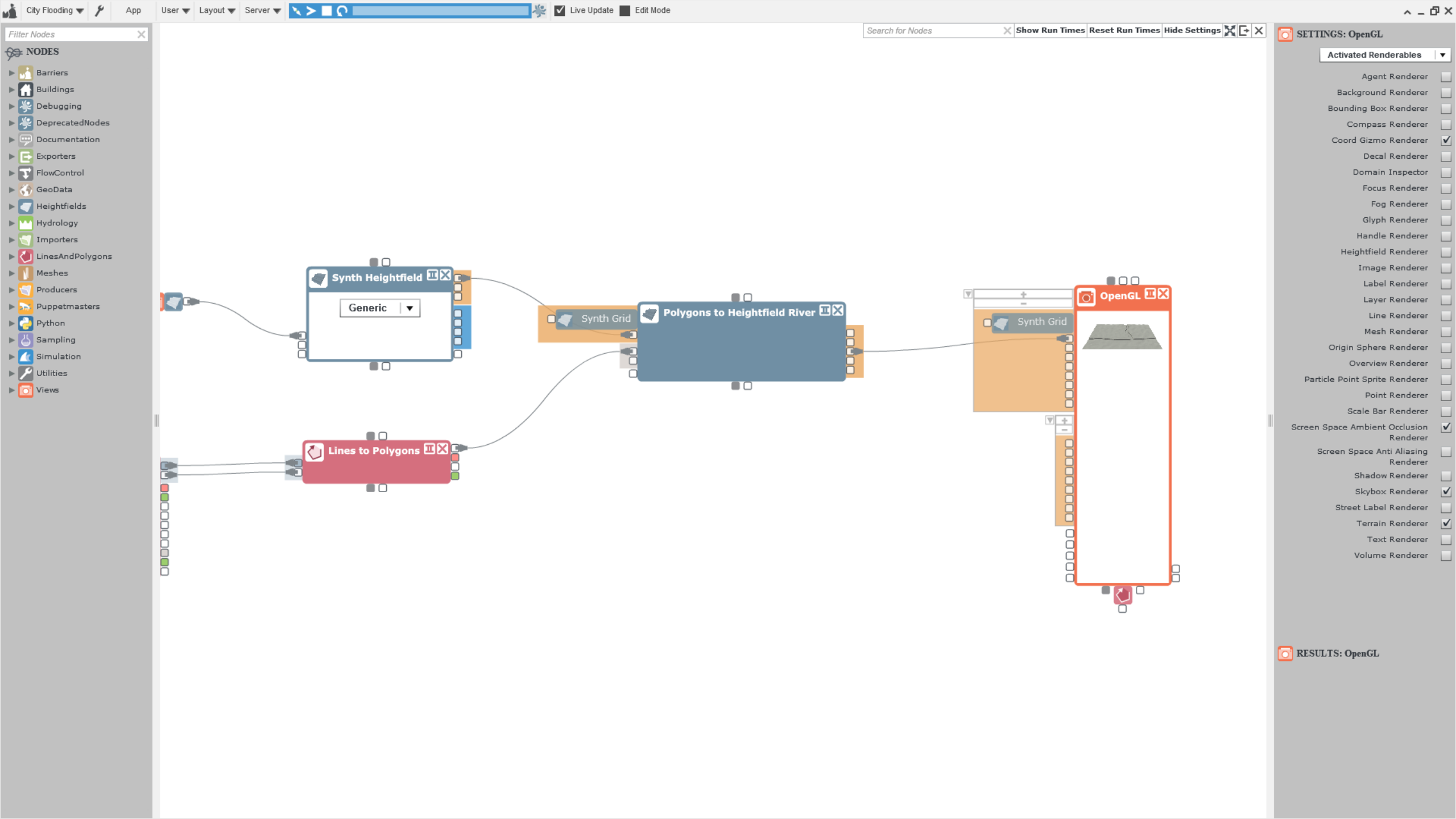

City Flooding
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
Line

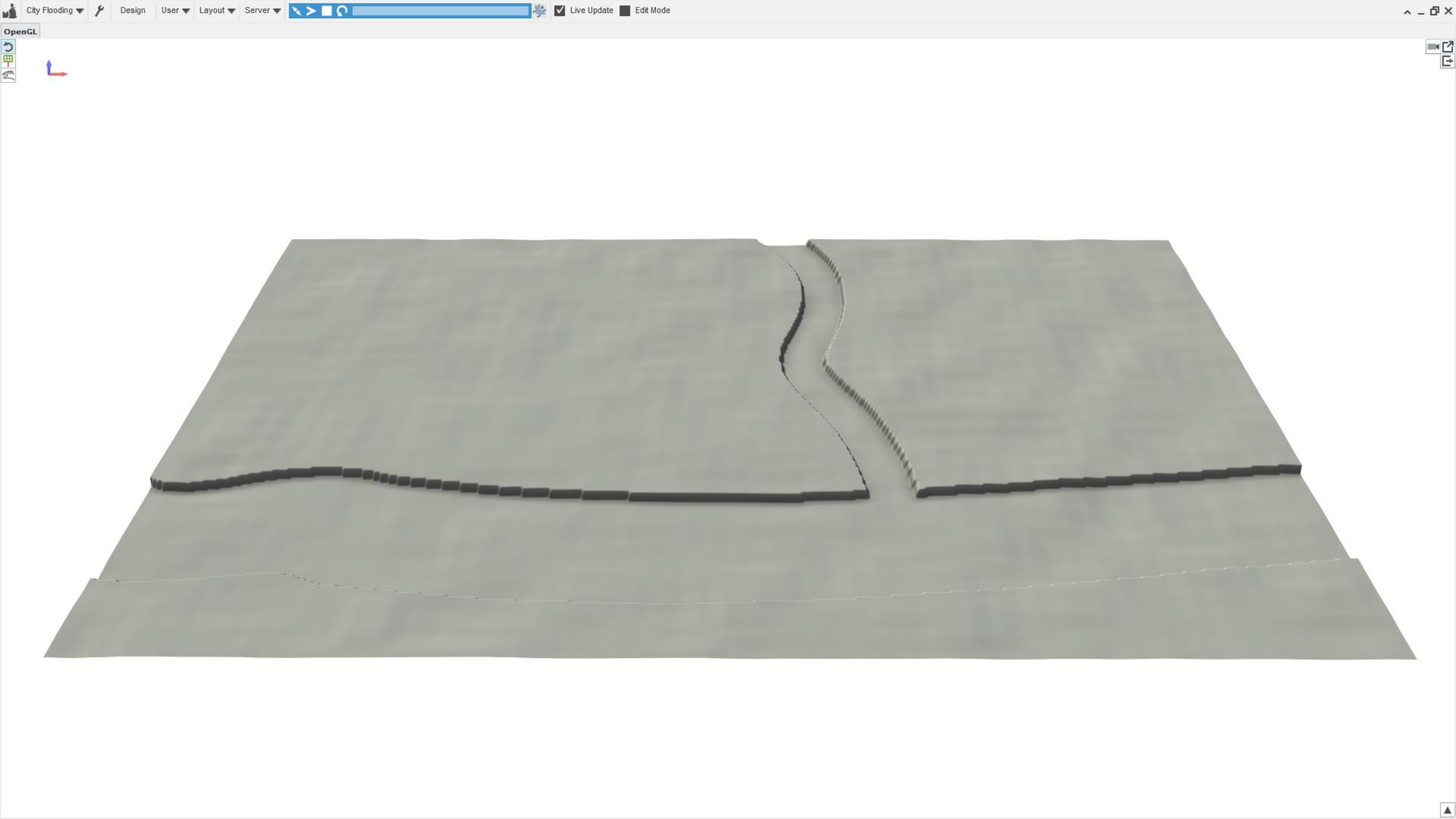

City Flooding
App
User
Layout
Server
Live Update
Edit Mode
Filter Nodes
NODES
Barriers
Buildings
Debugging
DeprecatedNodes
Documentation
Exporters
FlowControl
GeoData
Heightfields
Hydrology
Importers
LinesAndPolygons
Meshes
Producers
Puppetmasters
Python
Sampling
Simulation
Utilities
Views
Search for Nodes
Show Run Times
Reset Run Times
Hide Settings
2D
Synth Grid
Synth Heightfield
Generic
Lines
OpenGL : events
OpenGL
Synth Grid
Synth H...
SETTINGS: OpenGL
Decal Renderer
Effect settings
Default Settings
Advanced Mode
Effect
Default
Filling style
Solid
Filling color
Outline width [m or %]
1
Outline style
Solid
Outline mode
Normal
Outline color
Corner style
Miter
Miter angle limit [°]
90
Use fixed size
Clamp fixed size
Interpret sizes as pixels
Ignore trafo alpha
Ignore lowest trafo color
Drawing order
0
Show grid
Show decals from above
Show decals from below
Show overlay above decal
Validity visibility
Always
Is selectable
Receive shadows
Force fast update
Lines and Polygons
Quad-tree leaf capacity
8
Max. quad-tree depth
8
Points
Radius/Size [m]
1
Primitive segments
4
Primitive rotation [°]
0
Use trafo sizes
RESULTS: OpenGL

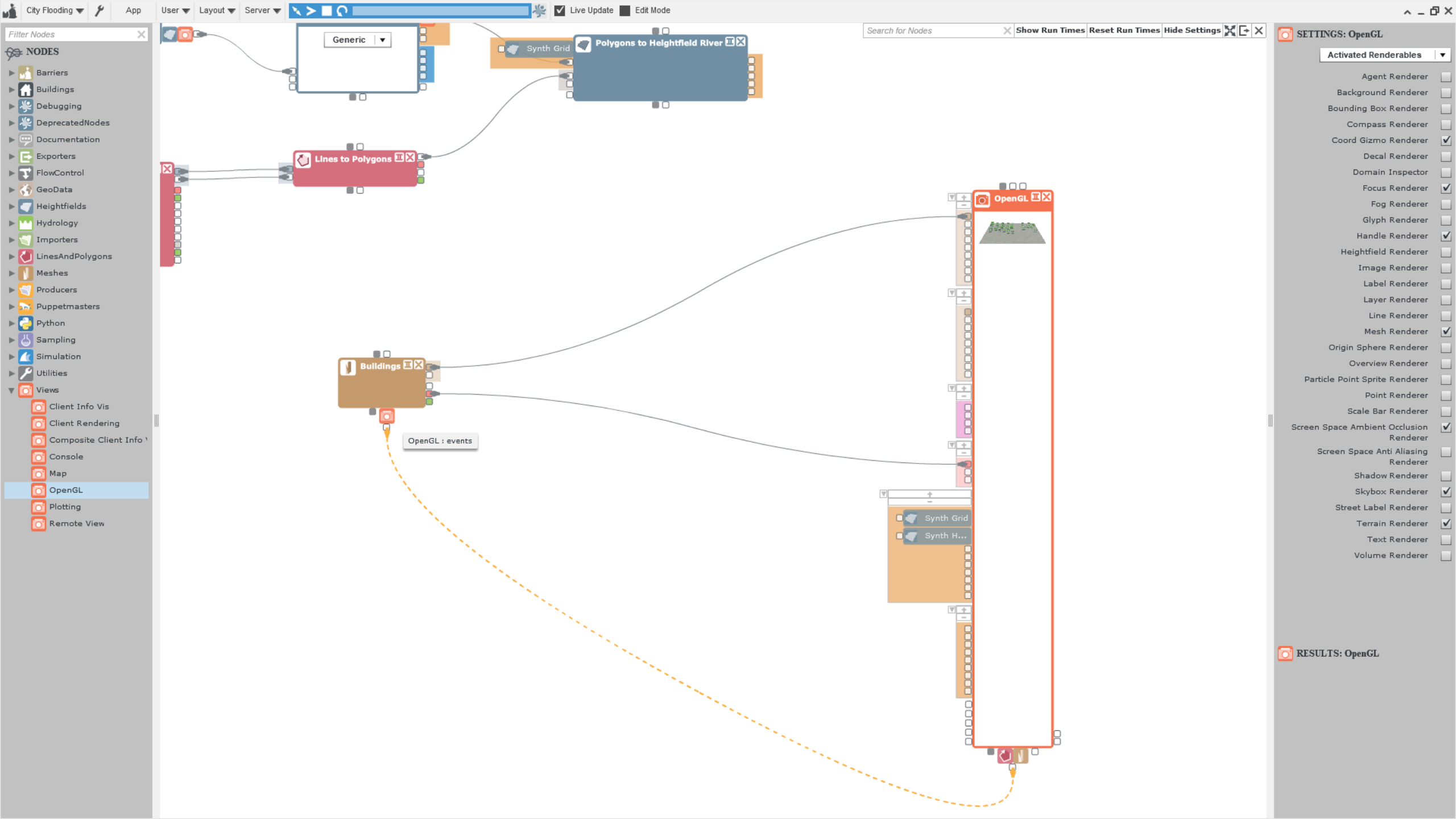

City Flooding
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
Line
A geometric line created by free hand drawing or multiple clicks
Delete All
Lines

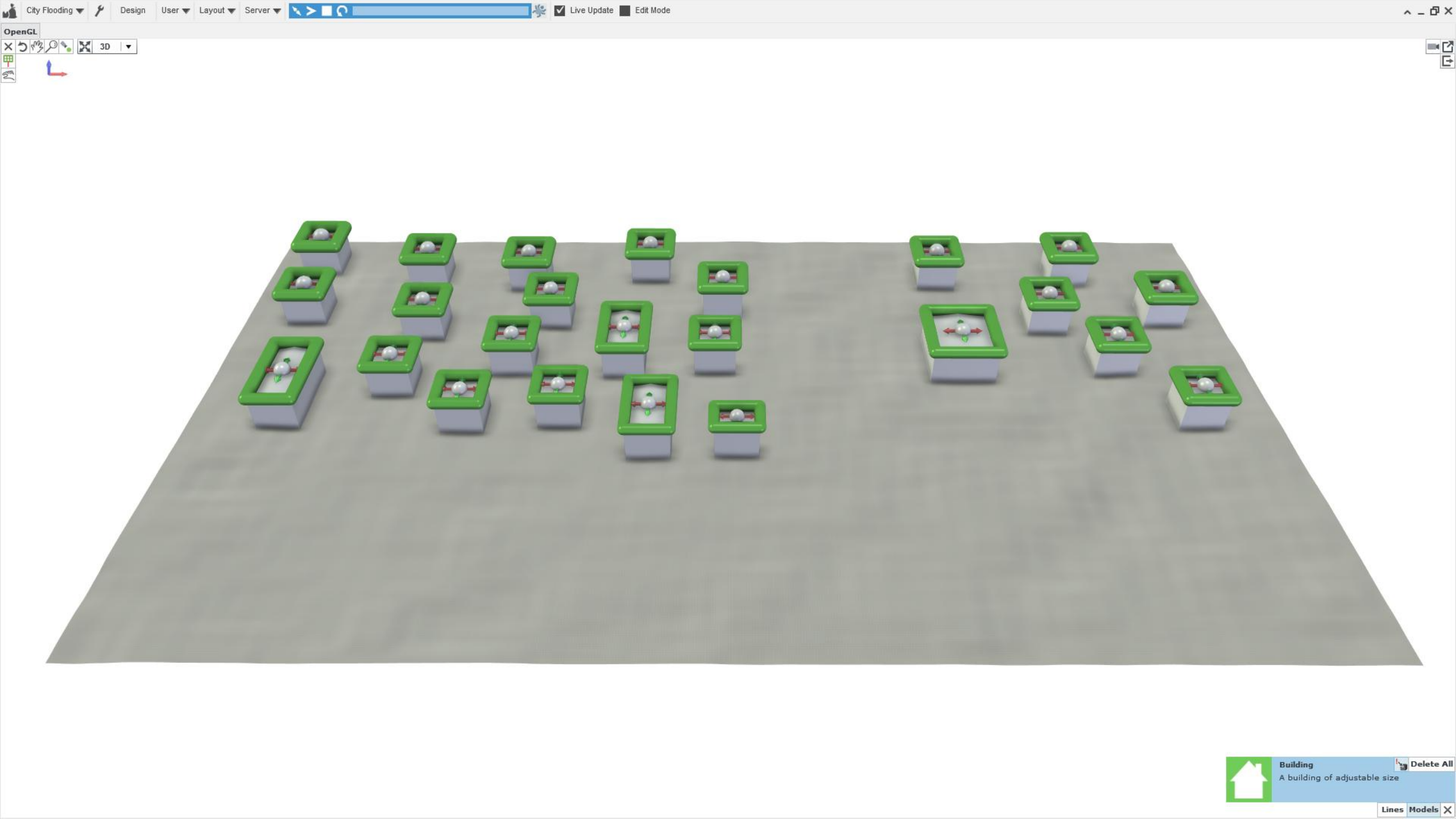

City Flooding
App
User
Layout
Server
Live Update
Edit Mode
Filter Nodes
NODES
Barriers
Buildings
Debugging
DeprecatedNodes
Documentation
Exporters
FlowControl
GeoData
Heightfields
Hydrology
Importers
LinesAndPolygons
Meshes
Producers
Puppetmasters
Python
Sampling
Simulation
Utilities
Views
Search for Nodes
Show Run Times
Reset Run Times
Hide Settings
2D
Synth Grid
Synth Heightfield
Generic
Lines
Lines to Polygons
OpenGL
Synth Grid
Synth H...
SETTINGS: OpenGL
Decal Renderer
Effect settings
Default Settings
Advanced Mode
Effect
Default
Filling style
Solid
Filling color
Outline width [m or %]
0.2
Outline style
Solid
Outline mode
Normal
Outline color
Corner style
Miter
Miter angle limit [°]
90
Use fixed size
Clamp fixed size
Interpret sizes as pixels
Ignore trafo alpha
Ignore lowest trafo color
Drawing order
0
Show grid
Show decals from above
Show decals from below
Show overlay above decal
Validity visibility
Always
Is selectable
Receive shadows
Force fast update
Lines and Polygons
Quad-tree leaf capacity
8
Max. quad-tree depth
8
Points
Radius/Size [m]
1
Primitive segments
4
Primitive rotation [°]
0
Use trafo sizes
RESULTS: OpenGL

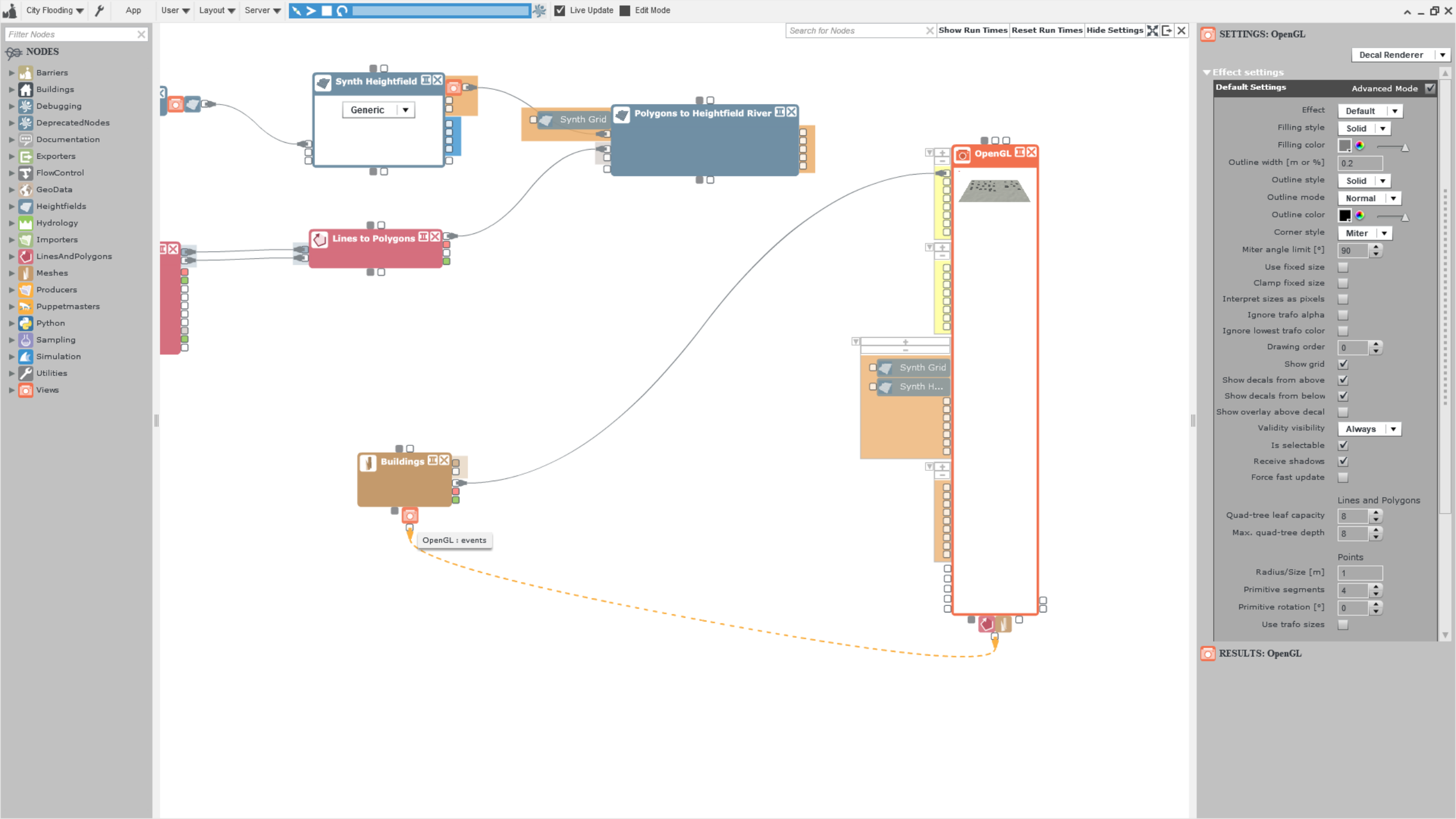

City Flooding
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL

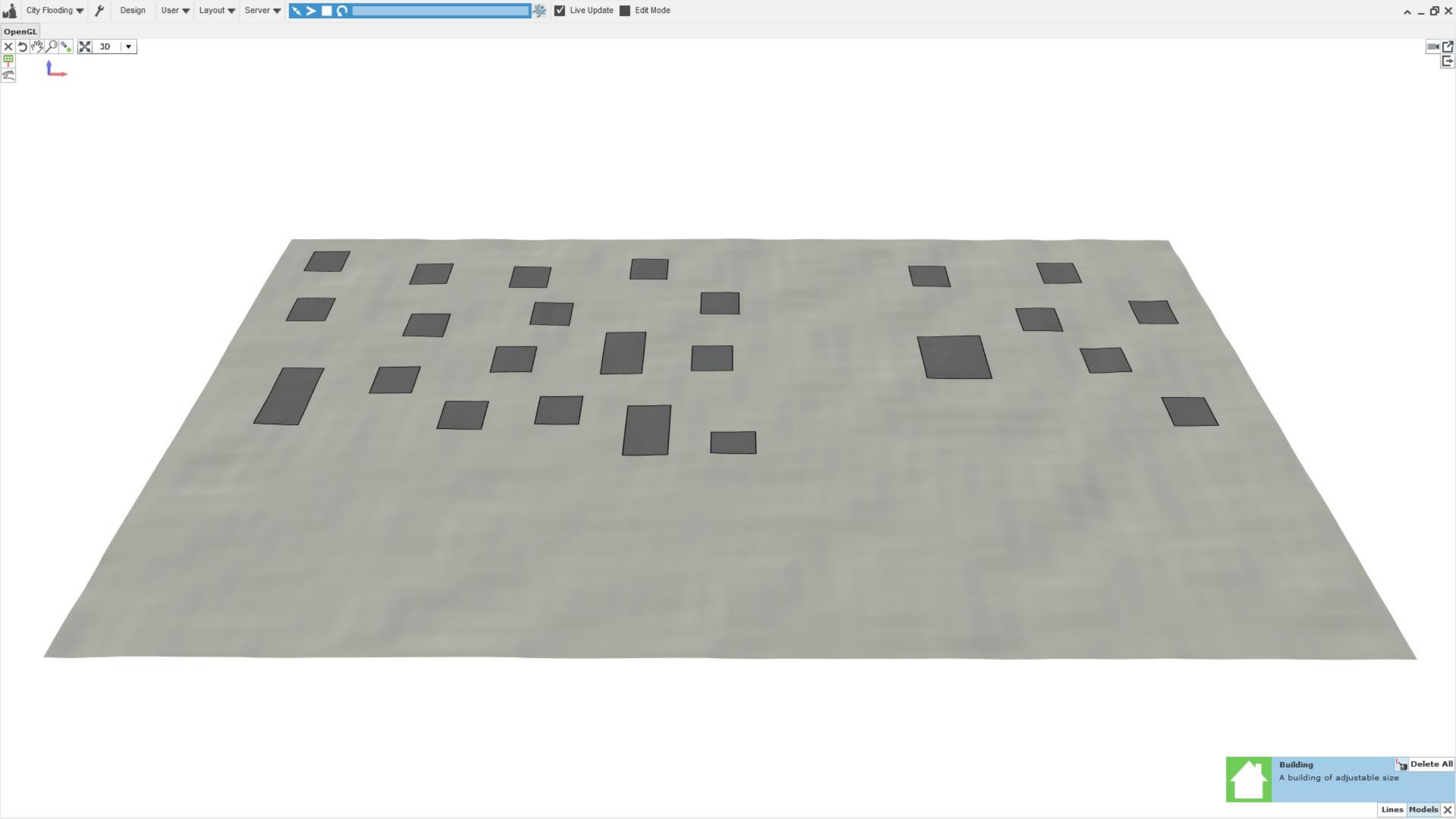

City Flooding
App
User
Layout
Server
Live Update
Edit Mode
Filter Nodes
NODES
Barriers
Buildings
Debugging
DeprecatedNodes
Documentation
Exporters
FlowControl
GeoData
Heightfields
Hydrology
Importers
LinesAndPolygons
Meshes
Producers
Puppetmasters
Python
Sampling
Simulation
Utilities
Views
Search for Nodes
Show Run Times
Reset Run Times
Hide Settings
Synth Heightfield
Generic
Synth Grid
Polygons to Heightfield River
Lines to Polygons
Synth Grid
OpenGL
SETTINGS: OpenGL
Activated Renderables
Agent Renderer
Background Renderer
Bounding Box Renderer
Compass Renderer
Coord Gizmo Renderer
Decal Renderer
Domain Inspector
Focus Renderer
Fog Renderer
Glyph Renderer
Handle Renderer
Heightfield Renderer
Image Renderer
Label Renderer
Layer Renderer
Line Renderer
Mesh Renderer
Origin Sphere Renderer
Overview Renderer
Particle Point Sprite Renderer
Point Renderer
Scale Bar Renderer
Screen Space Ambient Occlusion Renderer
Screen Space Anti Aliasing Renderer
Shadow Renderer
Skybox Renderer
Street Label Renderer
Terrain Renderer
Text Renderer
Volume Renderer
RESULTS: OpenGL

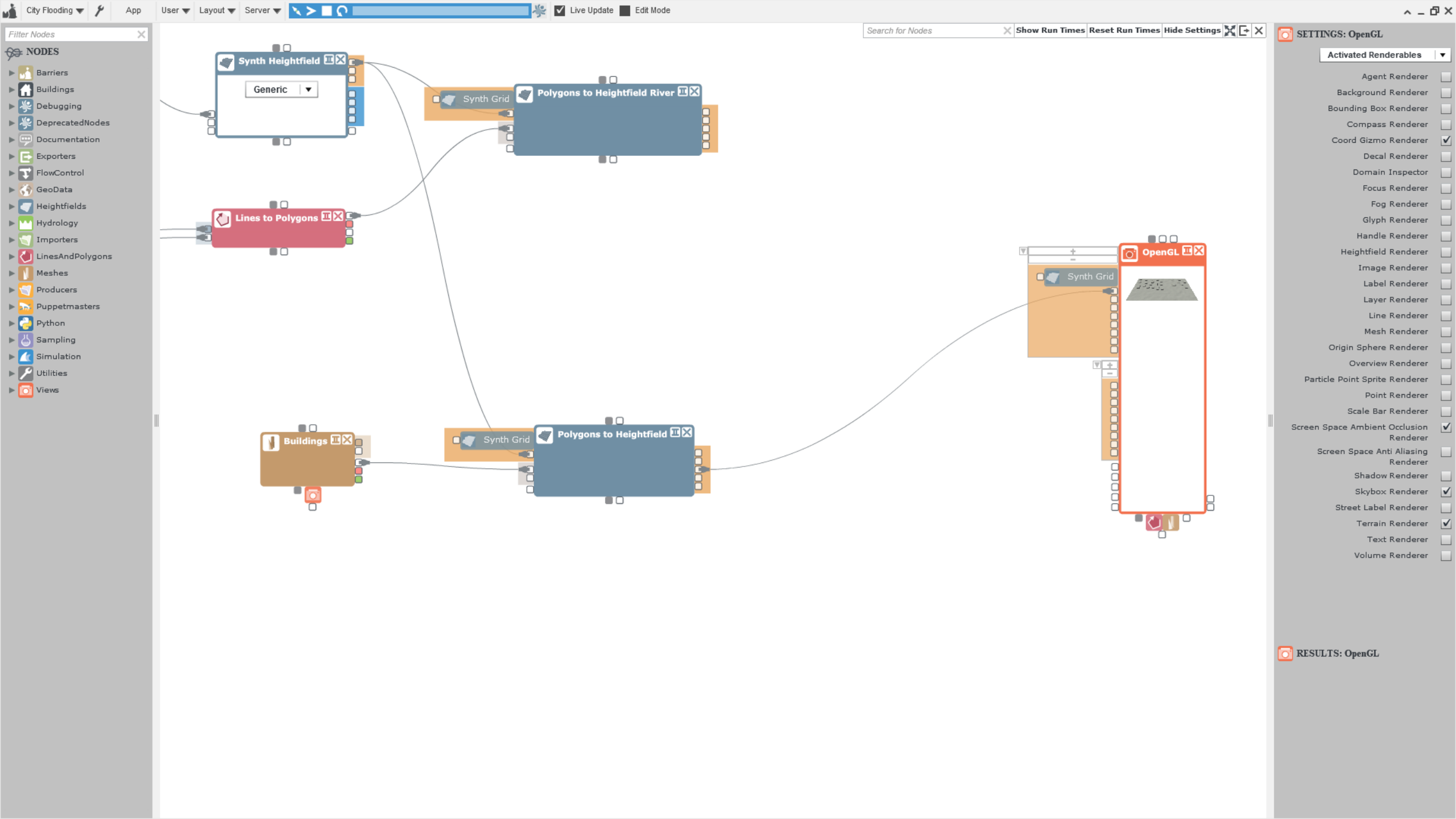

City Flooding
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL

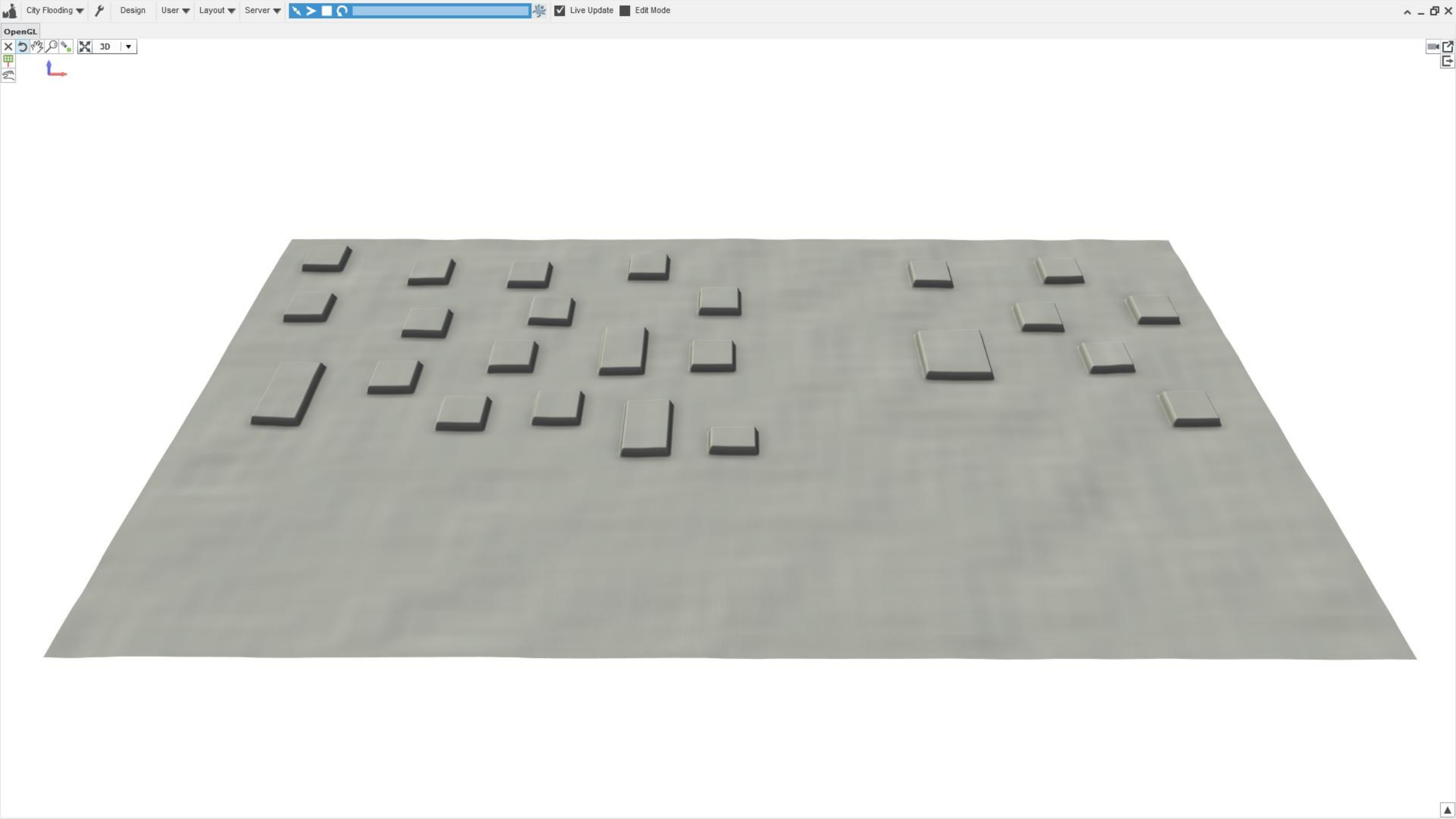

City Flooding
App
User
Layout
Server
Live Update
Edit Mode
Filter Nodes
NODES
Barriers
Buildings
Debugging
DeprecatedNodes
Documentation
Exporters
FlowControl
GeoData
Heightfields
Hydrology
Importers
LinesAndPolygons
Meshes
Producers
Puppetmasters
Python
Sampling
Simulation
Utilities
Views
Client Info Vis
Client Rendering
Composite Client Info
Console
Map
OpenGL
Plotting
Remote View
Search for Nodes
Show Run Times
Reset Run Times
Hide Settings
Generic
Synth Grid
Polygons to Heightfield River
Lines to Polygons
Buildings
OpenGL : events
OpenGL
Synth Grid
Synth H...
SETTINGS: OpenGL
Activated Renderables
Agent Renderer
Background Renderer
Bounding Box Renderer
Compass Renderer
Coord Gizmo Renderer
Decal Renderer
Domain Inspector
Focus Renderer
Fog Renderer
Glyph Renderer
Handle Renderer
Heightfield Renderer
Image Renderer
Label Renderer
Layer Renderer
Line Renderer
Mesh Renderer
Origin Sphere Renderer
Overview Renderer
Particle Point Sprite Renderer
Point Renderer
Scale Bar Renderer
Screen Space Ambient Occlusion Renderer
Screen Space Anti Aliasing Renderer
Shadow Renderer
Skybox Renderer
Street Label Renderer
Terrain Renderer
Text Renderer
Volume Renderer
RESULTS: OpenGL

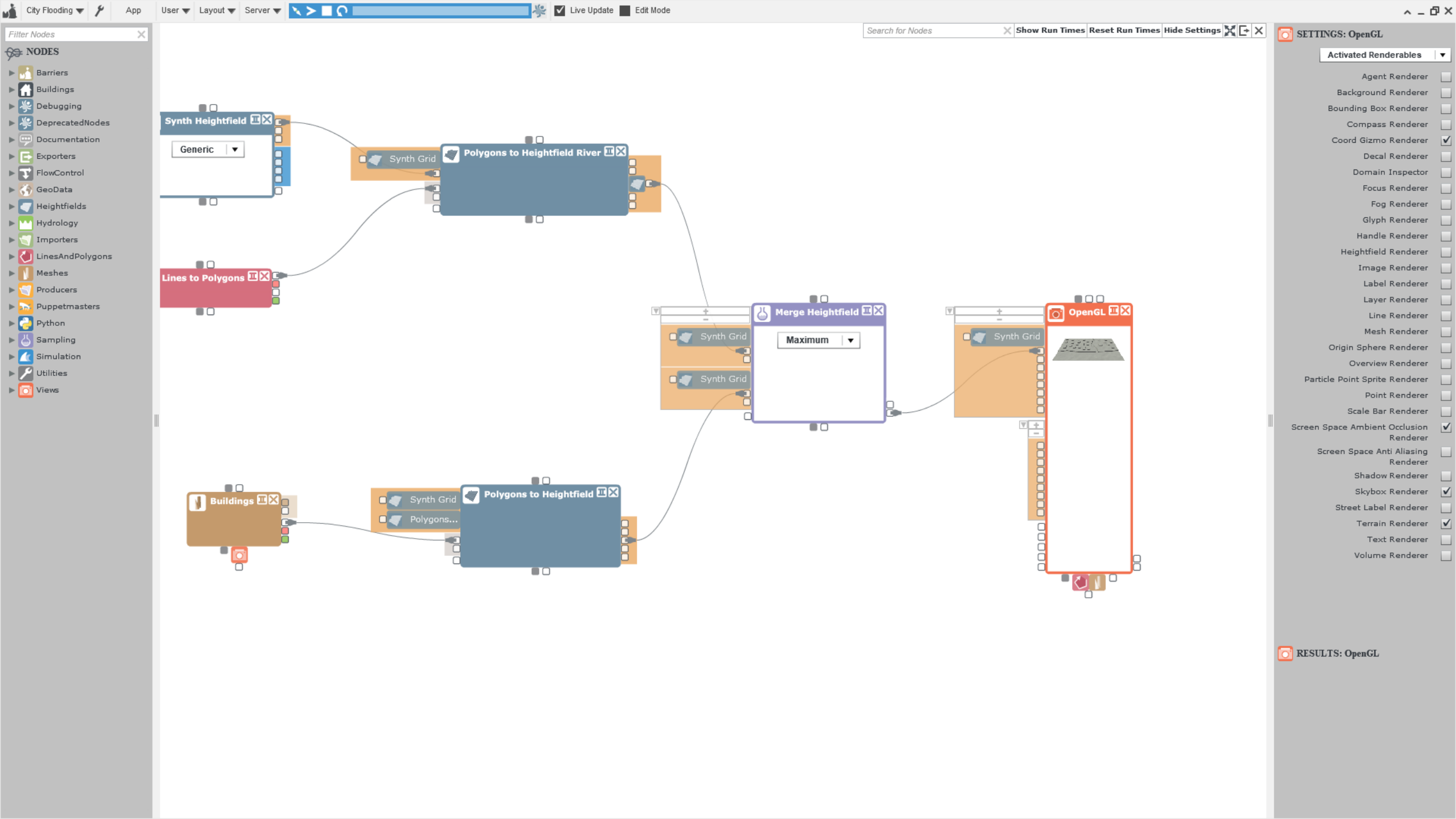

City Flooding
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
3D
Building
Delete All
A building of adjustable size
Lines
Models


City Flooding
App
User
Layout
Server
Live Update
Edit Mode
Filter Nodes
NODES
Barriers
Buildings
Debugging
DeprecatedNodes
Documentation
Exporters
FlowControl
GeoData
Heightfields
Hydrology
Importers
LinesAndPolygons
Meshes
Producers
Puppetmasters
Python
Sampling
Simulation
Utilities
Views
Search for Nodes
Show Run Times
Reset Run Times
Hide Settings
Synth Heightfield
Generic
Synth Grid
Polygons to Heightfield River
Lines to Polygons
Buildings
OpenGL : events
OpenGL
Synth Grid
Synth H...
SETTINGS: OpenGL
Decal Renderer
Effect settings
Default Settings
Advanced Mode
Effect
Default
Filling style
Solid
Filling color
Outline width [m or %]
0.2
Outline style
Solid
Outline mode
Normal
Outline color
Corner style
Miter
Miter angle limit [°]
90
Use fixed size
Clamp fixed size
Interpret sizes as pixels
Ignore trafo alpha
Ignore lowest trafo color
Drawing order
0
Show grid
Show decals from above
Show decals from below
Show overlay above decal
Validity visibility
Always
Is selectable
Receive shadows
Force fast update
Lines and Polygons
Quad-tree leaf capacity
8
Max. quad-tree depth
8
Points
Radius/Size [m]
1
Primitive segments
4
Primitive rotation [°]
0
Use trafo sizes
RESULTS: OpenGL

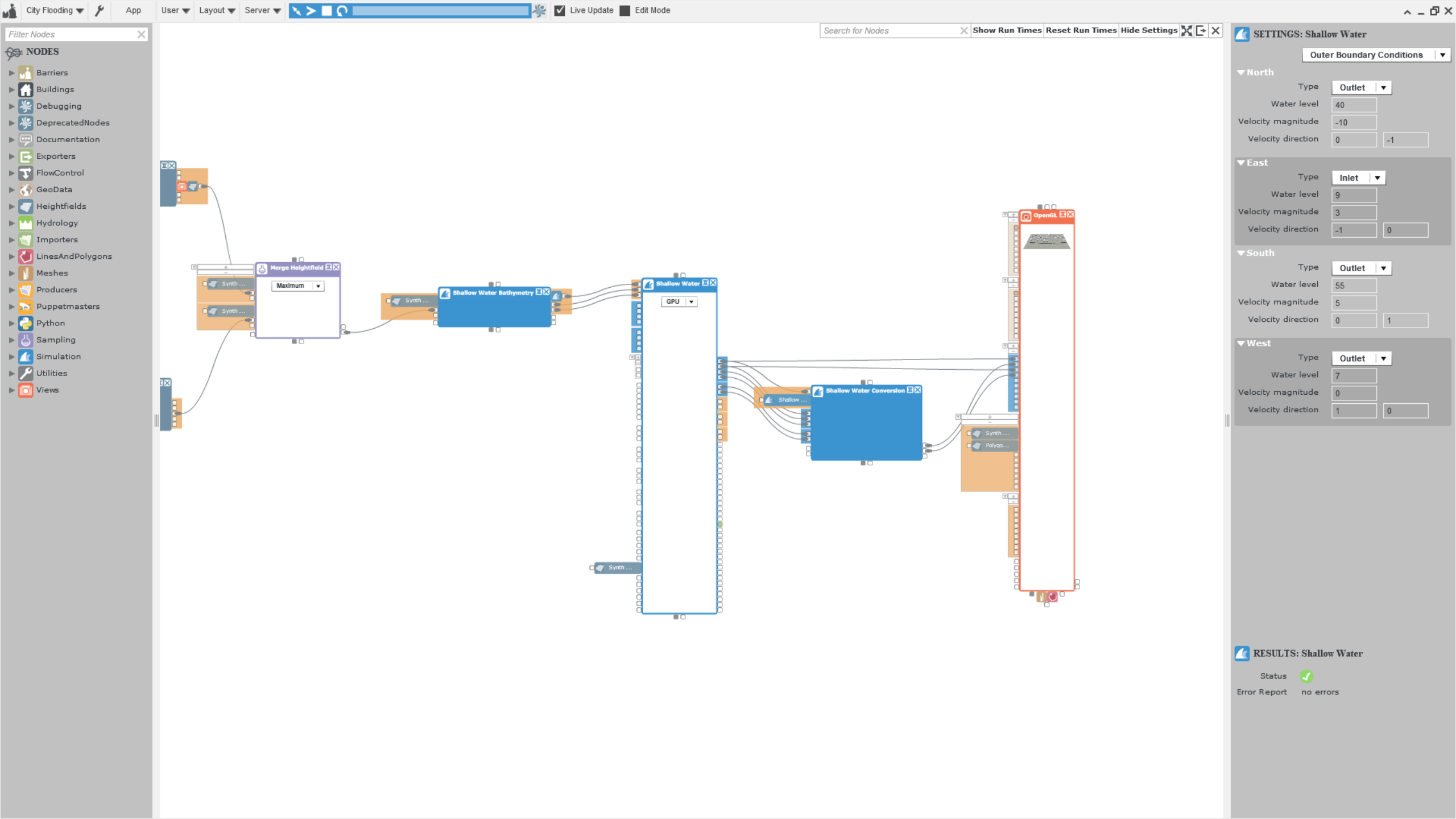

City Flooding
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
3D
Building
Delete All
A building of adjustable size
Lines
Models

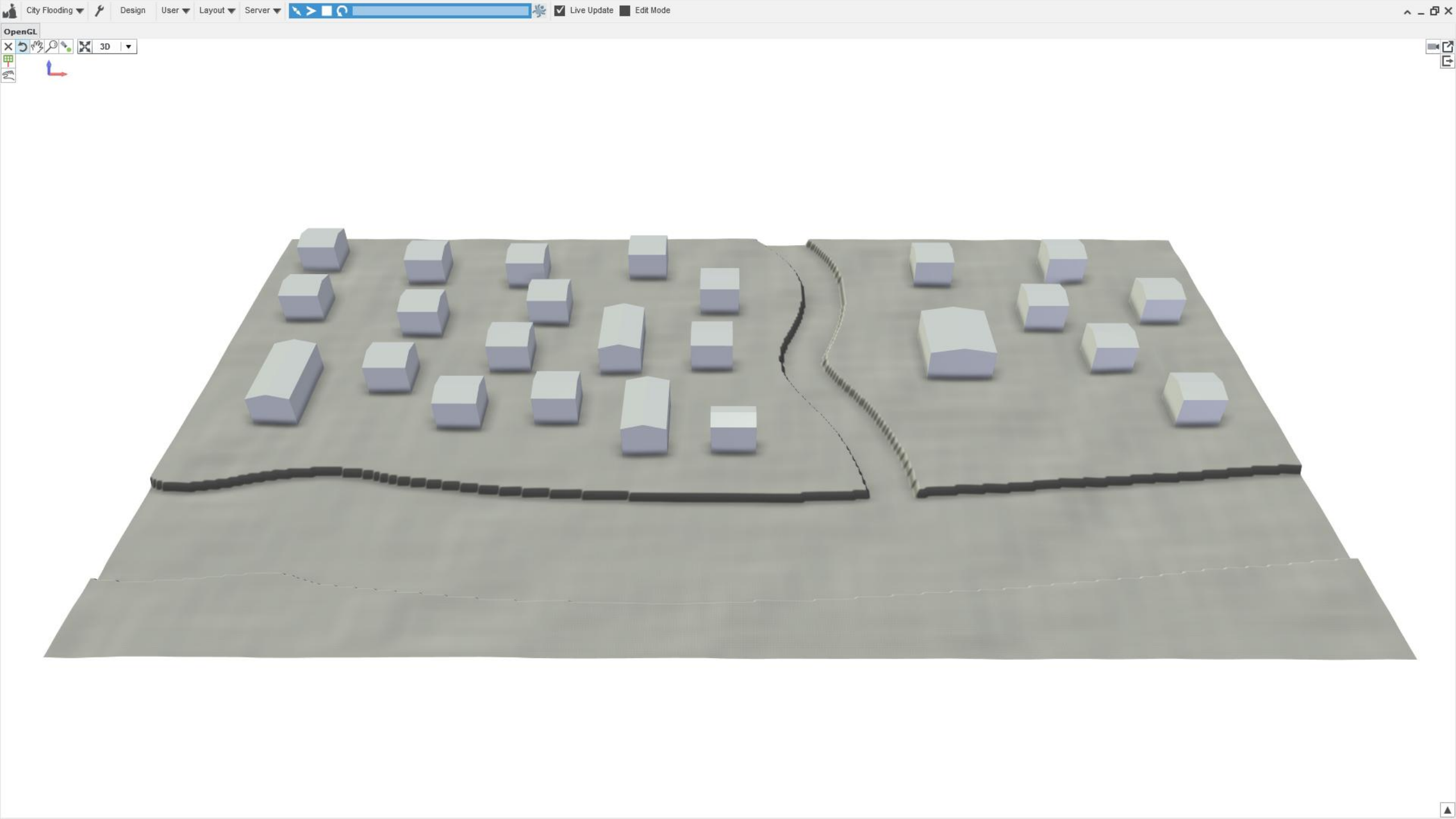

City Flooding
App
User
Layout
Server
Live Update
Edit Mode
Filter Nodes
NODES
Barriers
Buildings
Debugging
DeprecatedNodes
Documentation
Exporters
FlowControl
GeoData
Heightfields
Hydrology
Importers
LinesAndPolygons
Meshes
Producers
Puppetmasters
Python
Sampling
Simulation
Utilities
Views
Search for Nodes
Show Run Times
Reset Run Times
Hide Settings
Synth Heightfield
Generic
Synth Grid
Polygons to Heightfield River
Lines to Polygons
Buildings
Synth Grid
Polygons to Heightfield
Synth Grid
OpenGL
SETTINGS: OpenGL
Activated Renderables
Agent Renderer
Background Renderer
Bounding Box Renderer
Compass Renderer
Coord Gizmo Renderer
Decal Renderer
Domain Inspector
Focus Renderer
Fog Renderer
Glyph Renderer
Handle Renderer
Heightfield Renderer
Image Renderer
Label Renderer
Layer Renderer
Line Renderer
Mesh Renderer
Origin Sphere Renderer
Overview Renderer
Particle Point Sprite Renderer
Point Renderer
Scale Bar Renderer
Screen Space Ambient Occlusion Renderer
Screen Space Anti Aliasing Renderer
Shadow Renderer
Skybox Renderer
Street Label Renderer
Terrain Renderer
Text Renderer
Volume Renderer
RESULTS: OpenGL

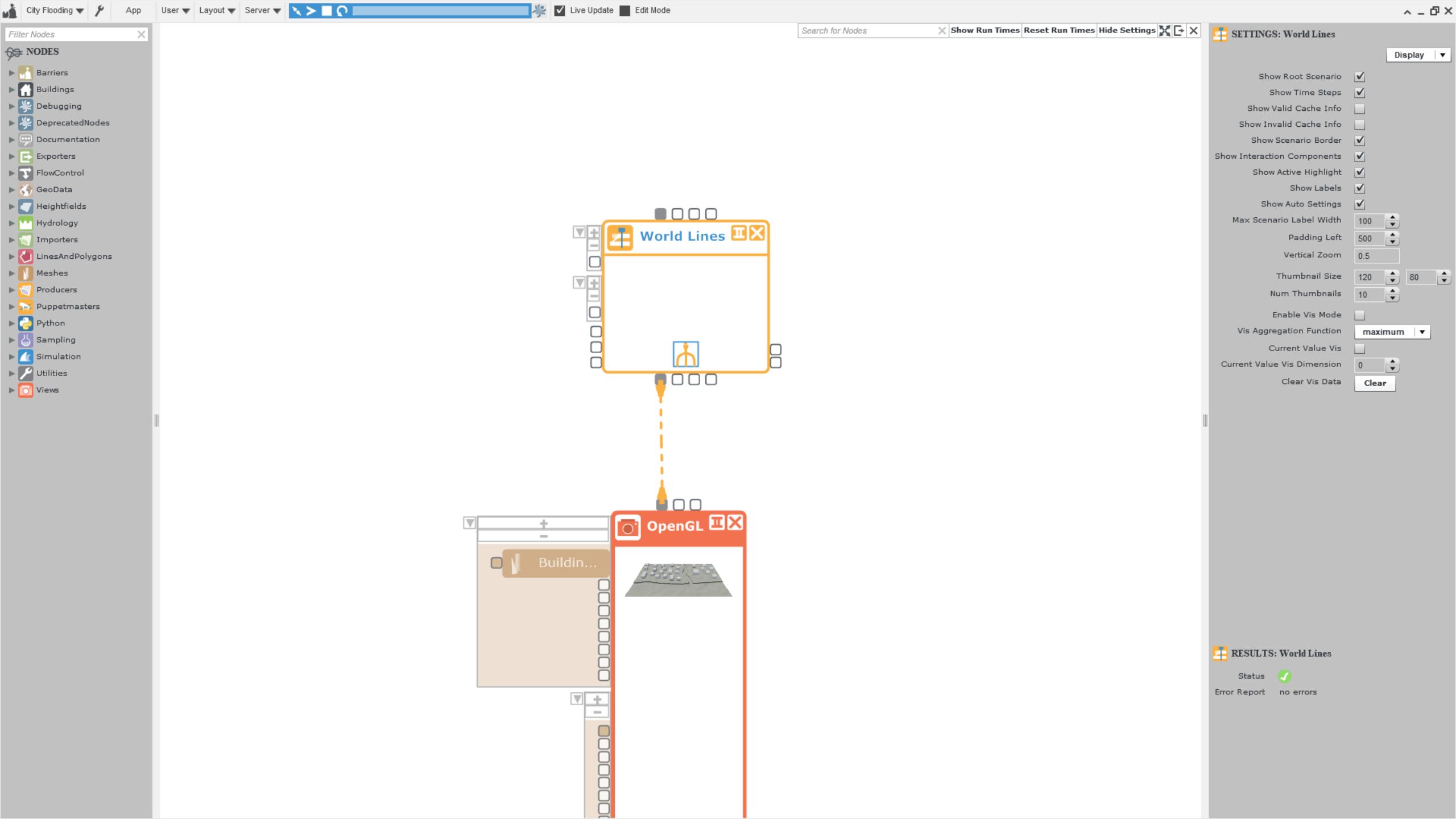

City Flooding
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
3D

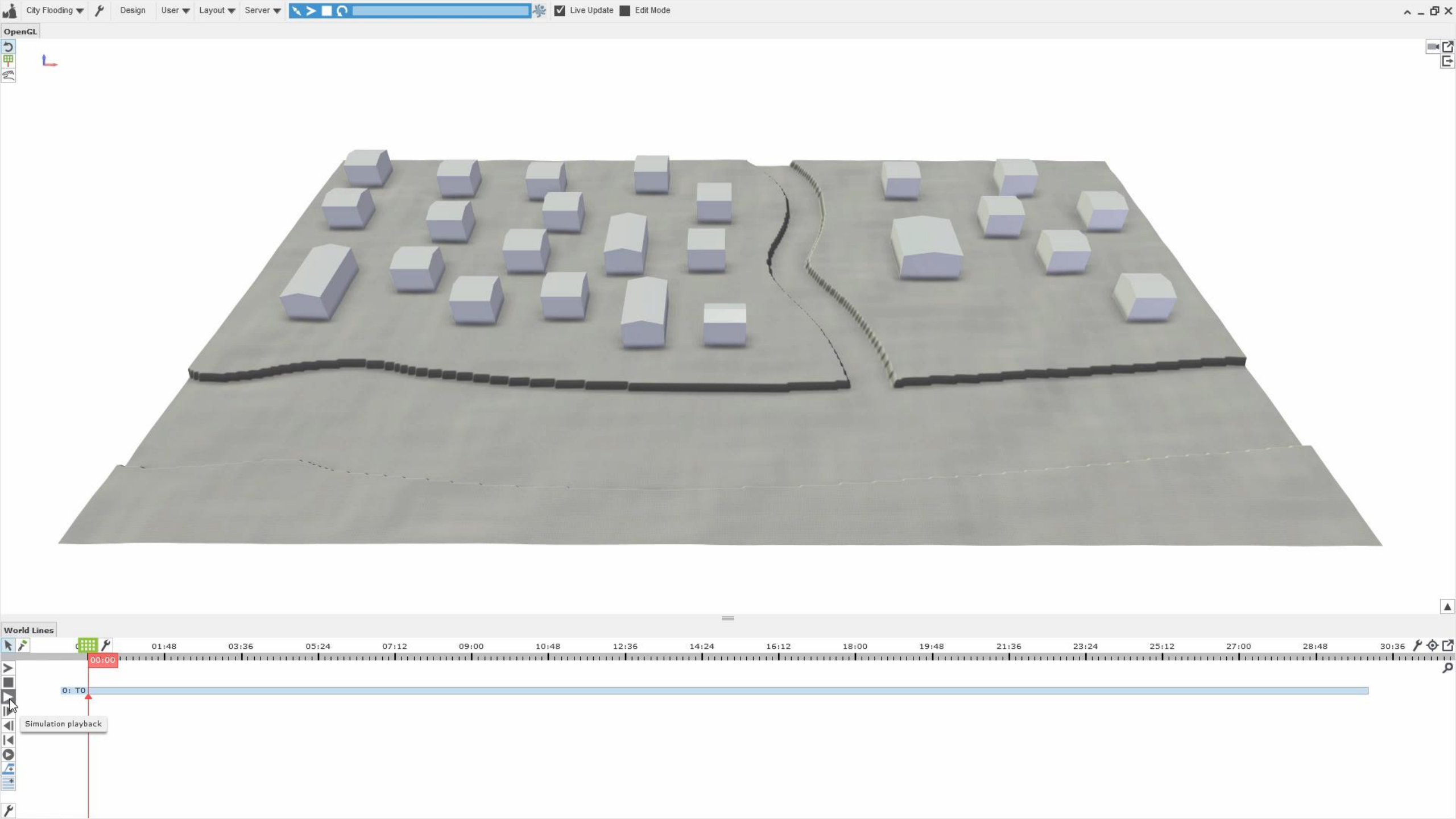

City Flooding
App
User
Layout
Server
Live Update
Edit Mode
Filter Nodes
NODES
Barriers
Buildings
Debugging
DeprecatedNodes
Documentation
Exporters
FlowControl
GeoData
Heightfields
Hydrology
Importers
LinesAndPolygons
Meshes
Producers
Puppetmasters
Python
Sampling
Simulation
Utilities
Views
Search for Nodes
Show Run Times
Reset Run Times
Hide Settings
Synth Heightfield
Generic
Synth Grid
Polygons to Heightfield River
Lines to Polygons
Merge Heightfield
Maximum
Synth Grid
Synth Grid
Synth Grid
OpenGL
Buildings
Synth Grid
Polygons...
Polygons to Heightfield
SETTINGS: OpenGL
Activated Renderables
Agent Renderer
Background Renderer
Bounding Box Renderer
Compass Renderer
Coord Gizmo Renderer
Decal Renderer
Domain Inspector
Focus Renderer
Fog Renderer
Glyph Renderer
Handle Renderer
Heightfield Renderer
Image Renderer
Label Renderer
Layer Renderer
Line Renderer
Mesh Renderer
Origin Sphere Renderer
Overview Renderer
Particle Point Sprite Renderer
Point Renderer
Scale Bar Renderer
Screen Space Ambient Occlusion Renderer
Screen Space Anti Aliasing Renderer
Shadow Renderer
Skybox Renderer
Street Label Renderer
Terrain Renderer
Text Renderer
Volume Renderer
RESULTS: OpenGL

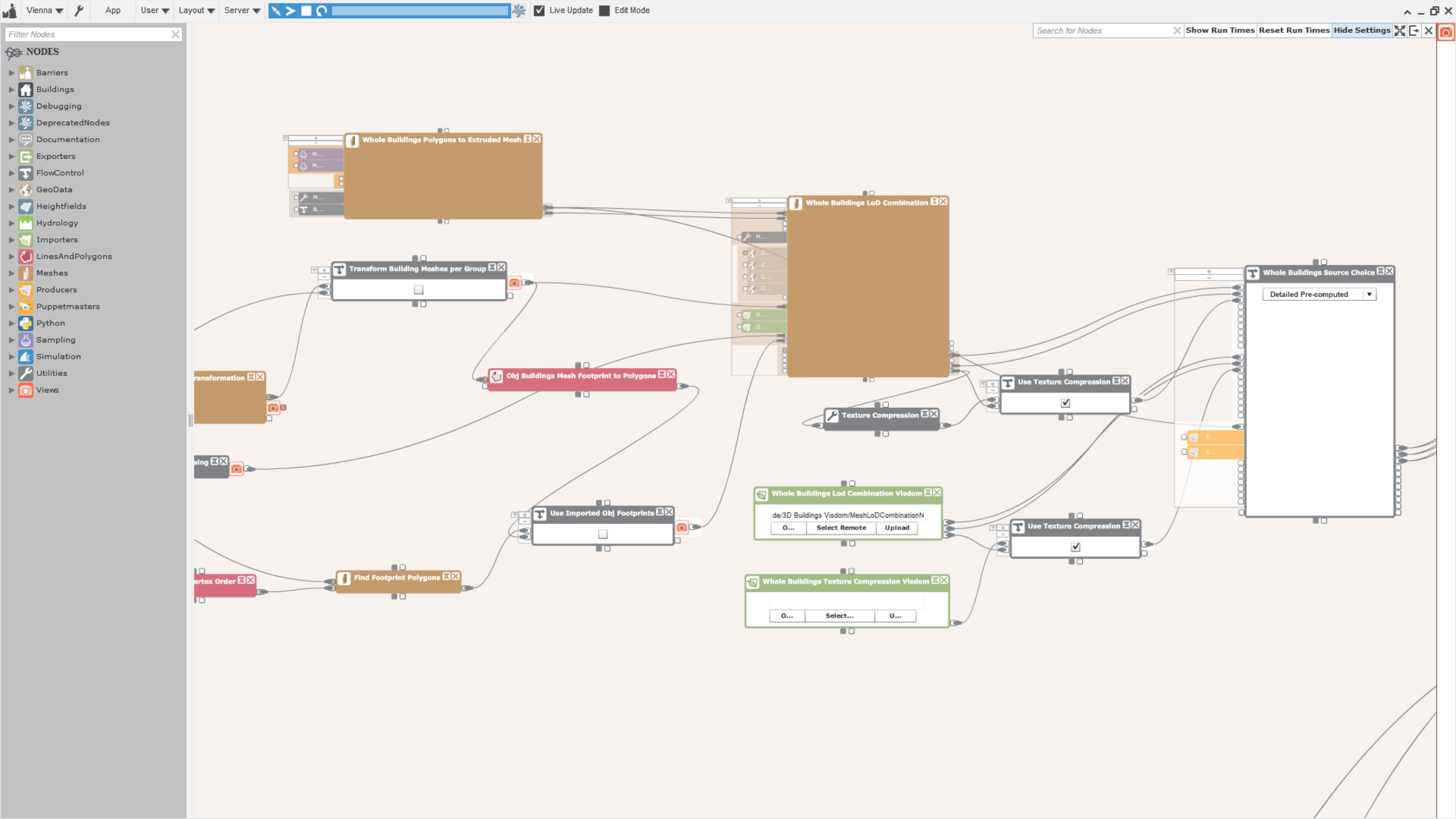

City Flooding
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
3D

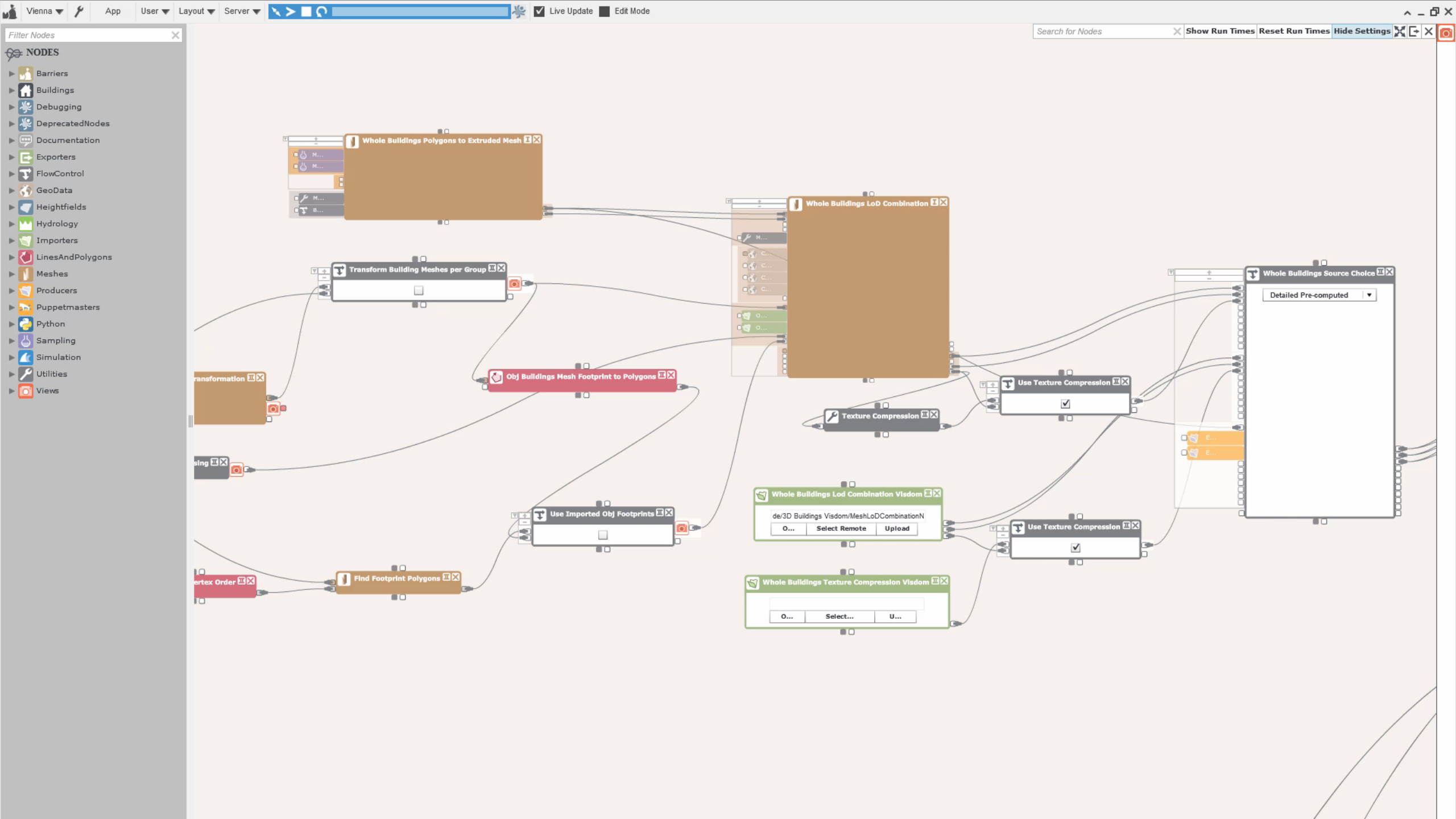

City Flooding
App
User
Layout
Server
Live Update
Edit Mode
Filter Nodes
NODES
Barriers
Buildings
Debugging
DeprecatedNodes
Documentation
Exporters
FlowControl
GeoData
Heightfields
Hydrology
Importers
LinesAndPolygons
Meshes
Producers
Puppetmasters
Python
Sampling
Simulation
Utilities
Views
Search for Nodes
Show Run Times
Reset Run Times
Hide Settings
Merge Heightfield
Maximum
Shallow Water Bathymetry
Shallow Water
GPU
Shallow Water Conversion
OpenGL
SETTINGS: Shallow Water
Outer Boundary Conditions
North
Type
Outlet
Water level
40
Velocity magnitude
-10
Velocity direction
0
-1
East
Type
Inlet
Water level
9
Velocity magnitude
3
Velocity direction
-1
0
South
Type
Outlet
Water level
55
Velocity magnitude
5
Velocity direction
0
1
West
Type
Outlet
Water level
7
Velocity magnitude
0
Velocity direction
1
0
RESULTS: Shallow Water
Status
Error Report
no errors


City Flooding
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
3D


City Flooding
App
User
Layout
Server
Live Update
Edit Mode
Filter Nodes
NODES
Barriers
Buildings
Debugging
DeprecatedNodes
Documentation
Exporters
FlowControl
GeoData
Heightfields
Hydrology
Importers
LinesAndPolygons
Meshes
Producers
Puppetmasters
Python
Sampling
Simulation
Utilities
Views
Search for Nodes
Show Run Times
Reset Run Times
Hide Settings
World Lines
OpenGL
Buildin...
SETTINGS: World Lines
Display
Show Root Scenario
Show Time Steps
Show Valid Cache Info
Show Invalid Cache Info
Show Scenario Border
Show Interaction Components
Show Active Highlight
Show Labels
Show Auto Settings
Max Scenario Label Width
100
Padding Left
500
Vertical Zoom
0.5
Thumbnail Size
120
80
Num Thumbnails
10
Enable Vis Mode
Vis Aggregation Function
maximum
Current Value Vis
Current Value Vis Dimension
0
Clear Vis Data
Clear
RESULTS: World Lines
Status
Error Report
no errors


City Flooding
Design
User
Layout
Server
Live Update
Edit Mode
OpenGL
World Lines
01:48
03:36
05:24
07:12
09:00
10:48
12:36
14:24
16:12
18:00
19:48
21:36
23:24
25:12
27:00
28:48
30:36
00:00
0: T0
Simulation playback


Vienna
App
User
Layout
Server
Live Update
Edit Mode
Filter Nodes
NODES
Barriers
Buildings
Debugging
DeprecatedNodes
Documentation
Exporters
FlowControl
GeoData
Heightfields
Hydrology
Importers
LinesAndPolygons
Meshes
Producers
Puppetmasters
Python
Sampling
Simulation
Utilities
Views
Search for Nodes
Show Run Times
Reset Run Times
Hide Settings
Whole Buildings Polygons to Extruded Mesh
Whole Buildings LoD Combination
Transform Building Meshes per Group
Obj Buildings Mesh Footprint to Polygons
Use Texture Compression
Texture Compression
Whole Buildings Source Choice
Detailed Pre-computed
Use Imported Obj Footprints
Whole Buildings Lod Combination Visdom
de/3D Buildings Visdom/MeshLoDCombinationN
O...
Select Remote
Upload
Use Texture Compression
Find Footprint Polygons
Whole Buildings Texture Compression Visdom
O...
Select...
U...


Vienna
App
User
Layout
Server
Live Update
Edit Mode
Filter Nodes
NODES
Barriers
Buildings
Debugging
DeprecatedNodes
Documentation
Exporters
FlowControl
GeoData
Heightfields
Hydrology
Importers
LinesAndPolygons
Meshes
Producers
Puppetmasters
Python
Sampling
Simulation
Utilities
Views
Search for Nodes
Show Run Times
Reset Run Times
Hide Settings
Whole Buildings Polygons to Extruded Mesh
Whole Buildings LoD Combination
Transform Building Meshes per Group
Obj Buildings Mesh Footprint to Polygons
Use Texture Compression
Texture Compression
Whole Buildings Source Choice
Detailed Pre-computed
Use Imported Obj Footprints
Whole Buildings Lod Combination Visdom
de/3D Buildings Visdom/MeshLoDCombinationN
Select Remote
Upload
Use Texture Compression
Find Footprint Polygons
Whole Buildings Texture Compression Visdom
Select...

```
      </output>
    </metaConnectors>
  </node>
  <node name="Merge Heightfield" category="Sampling" plugin="Sampling" type="MergeFieldNode" id="MergeHeightfieldNode_Mon-Feb-24-2014-05-30-06-PM-694" active="true" activeInNormalMode="true" activeInEditMode="true">
    <settings xsi:type="GenericSettings_t">
      <content>
        <operatorType>Maximum</operatorType>
        <defaultValue>
          <x>-1</x>
          <y>0</y>
          <z>0</z>
          <w>0</w>
        </defaultValue>
        <respectTargetDomainValidity>false</respectTargetDomainValidity>
        <useExtrapolation>false</useExtrapolation>
        <extrapolationNumShells>1</extrapolationNumShells>
        <metaLabel/>
        <metaUnit/>
        <metaId>-1</metaId>
        <overrideMetaData>false</overrideMetaData>
        <minOperatorUseAnyValid>false</minOperatorUseAnyValid>
        <useGPU>false</useGPU>
        <numCellsThresholdGPU>1000000</numCellsThresholdGPU>
      </content>
    </settings>
    <connectors>
      <input>
        <group name="field" numConnectors="2" type="SEQUENCE" collapsed="false">
          <group name="field" styleName="heightfieldGroup" type="ASSOCIATIVE">
            <connector name="domain"/>
            <connector name="field" default="true"/>
            <connector name="order"/>
          </group>
        </group>
        <connector name="targetDomain"/>
      </input>
      <output>
        <group type="CATEGORICAL">
          <connector name="domain"/>
          <connector name="field"/>
        </group>
      </output>
    </connectors>
    <metaConnectors>
      <input>
        <connector name="events"/>
      </input>
      <output>
        <connector name="events"/>
      </output>
    </metaConnectors>
  </node>
  <node name="Buildings" category="Meshes" plugin="Meshes" type="MeshDesignerNode" id="ModelDesignerNode_Tue-Feb-25-2014-01-21-43-AM-629" active="true" activeInNormalMode="true" activeInEditMode="true">
    <settings xsi:type="GenericSettings_t">
      <content>
```

Daniel Cornel – cornel@vrvis.at