Blender for Architects 2013

Propositions for Procedural Modelling and Drawing in Blender
Architectural Projections at Stadtfest Baden 2012
Architectural Projections at ETH Campus 2011 & 2010
Master graduation in 2011 at ETH Zurich
websites (PHP/MySQL), Processing/Java, Python
Simulated Architecture
Lukas Treyer - Stadtfest Baden

Public Projections in Baden 2012, Special Thanks to Peter Heusler, Eliane Zgraggen, Anna Hüveler, Kaspar Stöbe, Kristel Guzman, Martin Fröhlich
normal & shifted camera
Why Blender?

- parametrization / relations by:
  parent-child, constraints, bones, modifiers
- customize relations with drivers
--> drivers are amazing! let's extend the idea of
  extendable parametrization to architectural workflows!
2D Drawing

Modifier Nodes

New Modelling Nodes

Control Lines

Self-Made Parametrization, Template Patterns

Apply Mask Modifier

Better Hooks

Procedural Drawing

Better Team Workflow for Drawings

How do architects work?

Level of Detail

Curves

Parametric Drawing

Parametric Modelling

Better API
Better Hooks

---> hook vertex to vertex
---> hook objects and vertices to hooks
    = all hooks with the same name hook to each other
---> problem: what should hook to what?
    --> the object in edit mode has the lead
    --> in edit mode you don't see the original position anymore, only once the hook modifier is hidden or gets deleted
2D Drawing

2D drawing hooked to 3D geometry

---

Better

--> hook vertex to vertex
--> hook objects and vertices
= all hooks with the same
--> problem: what should happen
--> the object in edit mode
--> in edit mode you don’t
anymore, only once
hidden or gets deleted
Parametric Modelling

**Modifier Nodes**
- Today's Modifiers = procedural
- Node network = data flow, more flexible
- Example: Grasshopper
  - every operator is a modifier, every modifier is a node
  - every modeling step is captured with a new node in the node editor
  - IF the user wants it, i.e., by pressing Caps Lock

**Better Hooks**
- hook vertex to vertex
- hook objects and vertices to hooks
- all hooks with the same name hook to each other
- warning: what should hook to what
- the object in edit mode has the lead
- in edit mode you don't see the original position anymore, only once the hook modifier is hidden or gets deleted

**New Modelling Nodes**
- Subdivision Modifier Node, Distribution Node, Duplication Node

**Control Lines**
- basically the same idea as curbs:
  - high level editing
  - but with extended / different meaning

**Self-Made Parametrization, Template Patterns**
Modifier Nodes

Today's Modifiers = procedural
Node network = data flow, more flexible
example: Grasshopper
every operator is a modifier, every modifier is a node
→ every modeling step is captured with a new node in
  the node editor
→ IF the user wants it, i.e. by pressing Caps Lock

New Modelling Nodes
Subdivision Modifier Node, Distribution Node, Duplication Node
Control Lines

- basically the same idea as nurbs:
  high level editing
- but with extended / different meaning
New Modelling Nodes
Subdivision Modifier Node, Distribution Node, Duplication Node
Better Hooks

- back vertex to vertex
- hooks objects and vertices to hooks
- all hooks with the same name hook to each other
- problem: what should hook to what?
- the short in old node has the default
- in edit mode you don't see the original position anymore, only once the hook modifier is hideout or get deleted

New Modelling Nodes

Subdivision Modifier Node, Distribution Node, Duplication Node

Self-Made Parametrization, Template Patterns

Apply Mask Modifier

- override rule based, parametric model manually
- there are always exceptions
- make the masked area editable
- link it with hooks to the parametric geometry

Better Team Workflow for Drawings

How do architects work?
Apply Mask Modifier

- override rule based, parametric model manually
- there are always exceptions
- make the masked area editable
- link it with hooks to the parametric geometry
How do architects work?

read my paper :-)
Modifier Nodes

- Today's Modifiers: procedural
- Node network: fast, flexible, more flexible
- Examples: Grasshopper
- Every operator is a modifier, every modifier is a node
- Every modeling step is captured with a new node in the node editor
- If the user wants it, we can press Caps Lock

New Modelling Nodes

- Subdivision Modifier Node, Distribution Tool, Duplication Tool

Control Lines

- Based on the same idea as nursery
- High level editing
- But with extended / different meaning

Looks

Self-Made Parametrization, Template Patterns

Apply Mask Modifier

- Override rule-based, parametric model manually
- There are always exceptions
- Make the masked area editable
- Link it with hooks to the parametric geometry

Better API

- Operators take objects/vertices as parameters
- Make us (the artists) code new modifier nodes (with Python)
- Maybe we can at some point connect our geometry to a simulation node, created by somebody who researches on fluid dynamics on GPUs
Better API

- operators take objects/vertices as parameters
- make us (the artists) code new modifier nodes (with python)
--> maybe we can at some point connect our geometry to a simulation node, created by somebody who researches on fluid dynamics on GPUs
Thank you for your attention!