Inference of graph transformation rules for the design of geometric modeling operations

Romain Pascual

under the supervision of Pascale Le Gall, Hakim Belhaouari, and Agnès Arnould

November 29, 2022
Geometric modeling

▶ How to realize such a scene?
Geometric modeling

How to realize such a scene?

Creation of the objects, displacement mapping, normal mapping, 3D fractal texture, rendering, ...
Creating objects
Creating objects

Diversity of tools: Blender, AutoCAD, Catia, Houdini, Maya, Rhino 3D, SketchUp, SolidWorks, …
Designing modeling operations
Designing modeling operations

CGAL's sew operation

```cpp
template<unsigned int i>
void sew(Dart_descriptor adart1, Dart_descriptor adart2)
{
    CGAL_assertion( i<=dimension );
    CGAL_assertion( (is_sewable<i>(adart1,adart2)) );
    size_type amark=get_new_mark();
    CGAL::GMap_dart_iterator_basic_of_involution<Self, i> I1(*this, adart1, amark);
    CGAL::GMap_dart_iterator_basic_of_involution<Self, i> I2(*this, adart2, amark);
    for ( ; I1.cont(); ++I1, ++I2 )
    {
        Helper::template Foreach_enabled_attributes_except
            <CGAL::internal::GMap_group_attribute_functor<Self, i>, i>::
            run(*this, I1, I2);
    }
    negate_mark( amark );
    for ( I1.rewind(), I2.rewind(); I1.cont(); ++I1, ++I2 )
    {
        basic_link_alpha<i>(I1, I2);
    }
    negate_mark( amark );
    CGAL_assertion( is_whole_map_unmarked(amark) );
    free_mark(amark);
}
```
Inferring modeling operations

Standard Approach

```
int i = 0;
while (i < dimension) {
   if (is_enabled(dart, dart2)) {
      set_new_mark();
      iterator_basic_of_invocation<self, i>[
         (dart, dart2), mark];
      CGAL::CGAL.dart_iterator_basic_of_invocation<self, i>[
         (dart, dart2), mark];
      for (; i < cont(); ++i, ++k) {
         Helper::template forest::enabled_attribute_except<
            CGAL::internal::CGAL_group_attribute_function<self, i, k>, run<
               self, ii, i>, ii>;
         negate_mark( mark );
         for ( ii.rewind(); ii.rewind(); ii.cont(); ++ii, ++k ) {
            negate_mark( mark );
            CGAL::assertion< is_marched_map_unmarked(mark ) >;
            free_mark( mark );
         }
      }
   }
   ++i;
}
```
Inferring modeling operations

Standard Approach

Our Ambition
Inferring modeling operations

Standard Approach

Domain-Specific Language

Code Generation
Inferring modeling operations

Standard Approach

Automatic Inference?

Code Generation

\[
\begin{pmatrix}
  <0, 1> \\
  \text{n0}
\end{pmatrix}
\quad 3
\quad 3
\quad
\begin{pmatrix}
  <0, 1> \\
  \text{n1}
\end{pmatrix}
\quad
\begin{pmatrix}
  <0, 1> \\
  \text{n0}
\end{pmatrix}
\quad 3
\quad
\begin{pmatrix}
  <0, 1> \\
  \text{n1}
\end{pmatrix}
\]
**Introduction**

**Jerboa’s DSL**

**Strengths and weaknesses of Jerboa’s DSL**

- **Main characteristics:**
  - Dedicated to Gmaps\(^1\)
  - Syntax analyzer exploiting sufficient conditions\(^2\)

- **Successful applications:**
  - Plant growth
  - Architecture
  - Spring-mass
  - Geology

---

\(^1\)Poudret et al. 2008.  
\(^2\)Belhaouari et al. 2014.
A sneak peek at Jerboa’s language

```
<0, 1>
n0

<0, 1>
n1

<0, 1>
n0

<0, 1>
n1
```

R. Pascual

Ph.D. defense

November 29, 2022
A sneak peek at Jerboa’s language
A sneak peek at Jerboa’s language
A sneak peek at Jerboa’s language

Double-pushout (DPO) approach to graph transformations.\(^1\)

\(^1\)Rozenberg 1997; Ehrig et al. 2006; Heckel et al. 2020.
Running example: face triangulation
Plan

Standard Approach

Automatic Inference?

Code Generation

<0, 1> 3 3 <0, 1> 3 3 <0, 1>
Plan

1. Gmaps

2. Graph Transformations
## Plan

1. Gmaps

2. Graph Transformations

3. Inference
Geometric objects are represented with embedded generalized maps.
A graph $G = (V, E, s, t)$:

- a set of nodes $V$,
- a set of arcs $E$,
- a source function $s : E \rightarrow V$,
- a target arrow $t : E \rightarrow V$. 

The category of graphs
Generalized maps

The category of graphs

A graph $G = (V, E, s, t)$:

- a set of nodes $V$,
- a set of arcs $E$,
- a source function $s : E \rightarrow V$,
- a target arrow $t : E \rightarrow V$,

- Graphs can be decorated with labels, types, and attributes.

A morphism $G \rightarrow H$:

- a node function $V_G \rightarrow V_H$,
- an arc function $E_G \rightarrow E_H$, preserving structure.
Generalized maps\textsuperscript{1}

\textsuperscript{1}Damiand et al. 2014.
Generalized maps

Gmaps built as graphs:

Color legend: 0, 1, 2.

1Damian et al. 2014.
Generalized maps

Color legend: 0, 1, 2.

Gmaps built as graphs:
- topology: graph structure

\[^{1}\text{Damiand et al. 2014.}\]
Generalized maps

Gmaps built as graphs:
- topology: graph structure
- geometry: node attributes

Color legend: 0, 1, 2.

\(^1\)Damiand et al. 2014.
Orbits and topological cells

▶ Orbit (encode topological cell):
Graph induced by a subset \( \langle o \rangle \subseteq \{0, n\} \) of dimensions.

- positions on vertices (orbits \( \langle 1, 2 \rangle \)).

Color legend: 0, 1, 2.
Orbits and topological cells

Orbit (encode topological cell):
Graph induced by a subset $\langle o \rangle \subseteq [0, n]$ of dimensions.

- positions on vertices (orbits $\langle 1, 2 \rangle$).
- colors on faces (orbits $\langle 0, 1 \rangle$).

Color legend: $0$, $1$, $2$. 
Graph rewriting

- Operations on Gmaps are designed as graph rewriting rules.
Graph transformation rules

1Rozenberg 1997; Ehrig et al. 2006; Heckel et al. 2020.
Graph transformation rules\textsuperscript{1}

\begin{itemize}
\item $G \xrightarrow{m} L$
\item $D \xrightarrow{K} R$
\item $H \xrightarrow{R} L$
\end{itemize}

\textsuperscript{1}Rozenberg 1997; Ehrig et al. 2006; Heckel et al. 2020.
Rewriting Gmaps
Orbit rewriting

Graph rewriting  Topology and rule schemes
Orbit rewriting

\begin{equation}
\begin{aligned}
\langle 0, 1 \rangle & \rightarrow \langle 0, \_ \rangle & \rightarrow & \langle \_, 2 \rangle & \rightarrow & \langle 1, 2 \rangle \\
n_0 & & & n_1 & & n_2
\end{aligned}
\end{equation}

\begin{equation}
\begin{aligned}
\langle 0, 1 \rangle & \rightarrow \langle 0, \_ \rangle & \rightarrow & \langle \_, 2 \rangle & \rightarrow & \langle 1, 2 \rangle \\
n_0 & & & n_1 & & n_2
\end{aligned}
\end{equation}

\begin{equation}
\begin{aligned}
\langle 0, 1 \rangle & \rightarrow \langle 0, \_ \rangle & \rightarrow & \langle \_, 2 \rangle & \rightarrow & \langle 1, 2 \rangle \\
n_0 & & & n_1 & & n_2
\end{aligned}
\end{equation}

\begin{equation}
\begin{aligned}
\langle 0, 1 \rangle & \rightarrow \langle 0, \_ \rangle & \rightarrow & \langle \_, 2 \rangle & \rightarrow & \langle 1, 2 \rangle \\
n_0 & & & n_1 & & n_2
\end{aligned}
\end{equation}
Orbit rewriting

\[ \langle 0, 1 \rangle \xrightarrow{r} \langle 0, 2 \rangle \]

\[ \langle 0, 1 \rangle \xrightarrow{r} \langle 0, 2 \rangle \]

\[ \langle 0, 1 \rangle \xrightarrow{r} \langle 0, 2 \rangle \]

\[ \langle 0, 1 \rangle \xrightarrow{r} \langle 0, 2 \rangle \]

\[ \langle 0, 1 \rangle \xrightarrow{r} \langle 0, 2 \rangle \]
Orbit rewriting
Orbit rewriting
Orbit rewriting

<0, 1> n0

<0, _> n0

<_, 2> n1

<1, 2> n2

h a b
c d e
fg h

<0, _> n0

<1, 2> n2

r
Orbit rewriting

\[
\begin{align*}
&<0, 1> \\
&\rightarrow \\
&<0, _> \\
&\rightarrow \\
&<1, 2>
\end{align*}
\]

\[
\begin{align*}
&n0 \\
&\rightarrow \\
&n0 \\
&\rightarrow \\
&n2
\end{align*}
\]
Orbit rewriting
Orbit rewriting

\[ \langle 0, 1 \rangle \quad \rightarrow \quad \langle 0, - \rangle \quad \rightarrow \quad \langle -, 2 \rangle \quad \rightarrow \quad \langle 1, 2 \rangle \]

\[ \begin{array}{c}
\text{Before} \\
\text{After}
\end{array} \]

\[ \begin{array}{c}
\text{Graph 1} \\
\text{Graph 2}
\end{array} \]
Orbit rewriting
Graph products\textsuperscript{1}

- A categorical construction of global relabeling

\textsuperscript{1}inspired from Bauderon 1995.
Graph products\textsuperscript{1}

- A categorical construction of global relabeling

\textsuperscript{1}inspired from Bauderon 1995.
Graph products\textsuperscript{1}

- A categorical construction of global relabeling

\textsuperscript{1}inspired from Bauderon 1995.
Graph products

▶ A categorical construction of global relabeling

\[ \imath(\Pi, P) \]

\[ P \]

- \( \imath(\Pi, P) \): instantiation.

\(^1\)inspired from Bauderon 1995.
Graph products\(^1\)

- A categorical construction of global relabeling

\[
\iota(\Pi, P)
\]

\[
P \xrightarrow{\mathbb{E}_\Sigma} \mathbb{E}_\Sigma(P)
\]

- \(\iota(\Pi, P)\): instantiation.
- \(\mathbb{E}_\Sigma\): embedding functor.

\(^1\)inspired from Bauderon 1995.
Graph products\(^1\)

- A categorical construction of global relabeling

\[ \iota(\Pi, P) \]

\[ \mathcal{E}_\Sigma(P) \times \Pi \rightarrow \Pi \]

\[ P \xrightarrow{\mathcal{E}_\Sigma} \mathcal{E}_\Sigma(P) \xrightarrow{!\mathcal{E}_\Sigma(P)} 1_{\Sigma^2} \]

- \(\iota(\Pi, P)\): instantiation.
- \(\mathcal{E}_\Sigma\): embedding functor.

\(^1\)inspired from Bauderon 1995.
Graph products\textsuperscript{1}

\begin{itemize}
  \item $\iota(\Pi, P)$: instantiation.
  \item $\mathcal{E}_\Sigma$: embedding functor.
  \item $\pi_\Sigma$: projecting functor.
\end{itemize}

\textsuperscript{1}inspired from Bauderon 1995.
Application of a rule scheme
Application of a rule scheme

\[\begin{align*}
1_{W \times D} &\quad 1_{E_D(P_v)} \\
!_{E_D(P_v)} &\quad !_L \\
!_K &\quad !_R \\
E_D(P_v) &\quad E_D(P_v) \times L \\
E_D &\quad E_D(P_v) \times K \\
P_v &\quad E_D(P_v) \times R \\
P(G) &\quad \iota(L, P_v) \\
\iota(K, P_v) &\quad \iota(R, P_v) \\
\mathcal{G} &\quad D \\
\mathcal{H} &\quad \text{Output: Gmap}
\end{align*}\]
Application of a rule scheme
Application of a rule scheme

Graph rewriting

Topology and rule schemes

'Standard' DPO step
Modifying geometric values\footnote{Bellet et al. 2017.}
Modifying geometric values\textsuperscript{1}

Embedding expressions modeled with algebraic data types:

- add
- middle
- scale
- …

\textsuperscript{1}Bellet et al. 2017.
Modifying geometric values¹

Embedding expressions are extended with topological operators:

- Neighbor operator:
  - \( a@0@1@0.\text{position} = f.\text{position} = C \)
  - \( a@1@0.\text{color} = c.\text{color} = \circ \)

¹Bellet et al. 2017.
Modifying geometric values

Embedding expressions are extended with topological operators:

- Neighbor operator:
- Collect operator:
  - $\textit{position}_{0,1}(a) = \{A, B, C, D\}$
  - $\textit{color}_{0,1}(a) = \{\bullet\}$

---

$^1$Bellet et al. 2017.
Extension to schemes

Graph rewriting
Geometry and graph attributes
Extension to schemes

Graph rewriting
Geometry and graph attributes

<0, 1> \rightarrow <0, _> \rightarrow <_, 2> \rightarrow <1, 2>

<0, _> \rightarrow <1, 2>
Extension to schemes
Extension to schemes

\[
\frac{1}{4}(A + B + C + D)
\]
Extension to schemes

\[ \frac{1}{4}(A + B + C + D) = \text{middle} \{ A, B, C, D \} \]
Extension to schemes

\[
\frac{1}{4}(A + B + C + D) = \text{middle}(\{A, B, C, D\})
\]

\[
= \text{middle}(\text{position}_{\langle 0,1 \rangle}(a))
\]
Extension to schemes

\[
\frac{1}{4} (A + B + C + D) = \text{middle}(\{A, B, C, D\}) \\
= \text{middle}(\text{position}_{\langle 0, 1 \rangle}(a)) \\
= \text{middle}(\text{position}_{\langle 0, 1 \rangle}(n0))
\]
Consistency preservation

- Modifications of a well-formed object should produce an equally well-formed object.

**Requirement:** Provide feedback to the rule designer.
Consistency preservation

- Modifications of a well-formed object should produce an equally well-formed object.

**Requirement:** Provide feedback to the rule designer.

- **Topological inconsistency**

- **Geometric inconsistency**
Breaking the topological consistency

Constraint: 0202 paths should be cycles.

Diagram showing a graph before and after transformation, indicating broken topological consistency.
Breaking the topological consistency

**Constraint:** 0202 paths should be cycles.
Breaking the topological consistency

**Constraint:** 0202 paths should be cycles.

\[
\begin{align*}
\langle 0, 1 \rangle & \quad \rightarrow \quad \langle 0, \_ \rangle \\
\langle \_, 2 \rangle & \quad \rightarrow \quad \langle 2, 1 \rangle
\end{align*}
\]
Breaking the geometric consistency

Constraint: nodes in a \( \langle 0, 1 \rangle \)-orbit should have the same color.

\[
\text{mix(a.color, b.color)}
\]
Breaking the geometric consistency

Constraint: nodes in a $\langle 0, 1 \rangle$-orbit should have the same color.

$$\text{mix}(a.\text{color}, b.\text{color})$$
Breaking the geometric consistency

**Constraint:** nodes in a $\langle 0, 1 \rangle$-orbit should have the same color.

$\text{mix}(a.\text{color}, b.\text{color})$

Rule completion
Main results

- Expressing Jerboa’s DSL\(^1\) with categorical constructions:
  - Graph products (topology)
  - Rule completion (geometry)

- Weaker consistency conditions:
  - Necessary and sufficient conditions on DPO rules
  - Reduce false negatives in the analyzer (safeguard for inference)

- Unified framework to study generalized and oriented maps.

\(^1\)Poudret 2009; Bellet 2012.
Main results

- Expressing Jerboa’s DSL\(^1\) with categorical constructions:
  - Graph products (topology)
  - Rule completion (geometry)

- Weaker consistency conditions:
  - Necessary and sufficient conditions on DPO rules
  - Reduce false negatives in the analyzer (safeguard for inference)

- Unified framework to study generalized and oriented maps.

\(^1\)Poudret 2009; Bellet 2012.
Main results

- Expressing Jerboa’s DSL\textsuperscript{1} with categorical constructions:
  - Graph products (topology)
  - Rule completion (geometry)

- Weaker consistency conditions:
  - Necessary and sufficient conditions on DPO rules
  - Reduce false negatives in the analyzer (safeguard for inference)

- Unified framework to study generalized and oriented maps.

\textsuperscript{1}Poudret 2009; Bellet 2012.
Inferring geometric modeling operations

- Retrieving the operation described by an example.
Reversing the instantiation process

\[ \langle 0, 1 \rangle_{\text{n0}} \rightarrow \langle 0, 1 \rangle_{\text{n1}} \rightarrow \langle 0, 1 \rangle_{\text{n0}} \rightarrow \langle 0, 1 \rangle_{\text{n1}} \]

3 3 3
Inferring geometric modeling operations

Reversing the instantiation process
Inference workflow

- **Input:** A graph $G$ encoding the preservation relation between two partial Gmaps, an orbit type $\langle o \rangle$ and a dart $a$ of $G$.

- **Output:** A graph $S$ that encodes the Jerboa rule with the variable $\langle o \rangle$, given that the operation is applied to the dart $a$. 
Folding a joint representation of the rule
Folding a joint representation of the rule

Besides the two Gmaps and the preservation links, we chose a dart in the initial Gmap and an orbit type. ▶ Graph traversal algorithm. Iteratively applying two foldings:
• Folding of a node.
• Folding of the arcs. ▶ Illustration on face triangulation with the orbit type $\langle 0, 1 \rangle$ and the dart $a_0$. 
Folding a joint representation of the rule

Color legend: 0, 1, 2, $\kappa$. 
Folding a joint representation of the rule

Besides the two Gmaps and the preservation links, we chose a dart in the initial Gmap and an orbit type.

- Graph traversal algorithm.
  Iteratively applying two foldings:
  - Folding of a node.
  - Folding of the arcs.

- Illustration on face triangulation with the orbit type $\langle 0, 1 \rangle$ and the dart $a0$. 

Color legend: 0, 1, 2, $\kappa$. 
Execution

Creation of the hook (orbit $\langle 0, 1 \rangle$).

Color legend: 0, 1, 2, $\kappa$. 
Execution

Folding of the arcs.

Color legend: 0, 1, 2, κ.
Execution

Color legend: 0, 1, 2, κ.

Folding of a node.
Execution

The algorithm terminates.

Color legend: 0, 1, 2, \( \kappa \).
Execution

Color legend: 0, 1, 2, $\kappa$.

Splitting the joint representation.
Results

- **Correctness:** The algorithm returns a topological folding of the rule if it exists and halts otherwise.

- What about cases where we cannot fold the rule? Example with the orbit $\langle 0, 1, 2 \rangle$. 

![Diagram showing topological folding of a rule]
Results

- **Correctness:** The algorithm returns a topological folding of the rule if it exists and halts otherwise.

- What about cases where we cannot fold the rule? Example with the orbit $\langle 0, 1, 2 \rangle$. 

![Diagram showing the folding process from left to right.]
Results

- **Correctness:** The algorithm returns a topological folding of the rule if it exists and halts otherwise.

- What about cases where we cannot fold the rule? Example with the orbit $\langle 0, 1, 2 \rangle$. 

![Diagram showing topological folding](image)
Objective

The rule is missing its embedding expressions.
Method (inference of positions)

▶ Hypothesis: The vertex positions of the target object $C$ are obtained as affine combinations of vertex positions in the initial object $O$. 

\[ p = \sum w_i p_i + t \]

- $p$: target position (known)
- $p_i$: position of the initial vertex $i$ (known)
- $w_i$: weight (unknown)
- $t$: translation (unknown)
Method (inference of positions)

▶ Hypothesis: The vertex positions of the target object $C$ are obtained as affine combinations of vertex positions in the initial object $O$.

For each vertex in $C$, we want a position $p$ expressed as:

$$ p = \sum_{i=0}^{k} w_i p_i + t $$

where:

- $p$: target position (known)
Hypothesis: The vertex positions of the target object $C$ are obtained as affine combinations of vertex positions in the initial object $O$.

For each vertex in $C$, we want a position $p$ expressed as:

$$p = \sum_{i=0}^{k} w_i p_i + t$$

where:

- $p$: target position (known)
- $p_i$: position of the initial vertex $i$ (known)
Method (inference of positions)

▶ Hypothesis: The vertex positions of the target object \( C \) are obtained as affine combinations of vertex positions in the initial object \( O \).

For each vertex in \( C \), we want a position \( p \) expressed as:

\[
p = \sum_{i=0}^{k} w_i p_i + t
\]

where:

- \( p \): target position (known)
- \( p_i \): position of the initial vertex \( i \) (known)
- \( w_i \): weight (unknown)
- \( t \): translation (unknown)
Hypothesis: The vertex positions of the target object $C$ are obtained as affine combinations of vertex positions in the initial object $O$.

For each vertex in $C$, we want a position $p$ expressed as:

$$p = \sum_{i=0}^{k} w_i p_i + t$$

where:

- $p$: target position (known)
- $p_i$: position of the initial vertex $i$ (known)
- $w_i$: weight (unknown)
- $t$: translation (unknown)
Need for abstraction on schemes

We want \((w_i)_{0 \leq i \leq k}\) such that:

\[ p = \sum_{i=0}^{k} w_i p_i + t \]
We want \((w_i)_{0 \leq i \leq k}\) such that:
\[ p = \sum_{i=0}^{k} w_i p_i + t \]

**Issue:** darts in the Gmap will share the same expression.
- Because rule schemes abstract topological cells.
Need for abstraction on schemes

We want \((w_i)_{0 \leq i \leq k}\) such that:

\[
p = \sum_{i=0}^{k} w_i p_i + t
\]

**Issue:** darts in the Gmap will share the same expression.

- **Because** rule schemes abstract topological cells.

**Solution:** Exploit the topology.

- **Use** points of interest that share the same expression.
Points of interest

- $p_v$: vertex
- $p_e$: edge midpoint
- $p_f$: face barycenter
- $p_s$: volume barycenter
- $p_{cc}$: CC barycenter
Points of interest

- \( p_v \): vertex
- \( p_e \): edge midpoint
- \( p_f \): face barycenter
- \( p_s \): volume barycenter
- \( p_cc \): CC barycenter
Points of interest

with

- $p_v$ : vertex

\[
p_v = \text{middle}(\text{position}_{\langle 1,2,3 \rangle}(d))
\]
Inferring geometric modeling operations

Geometric inference

Points of interest

with

- $p_v$: vertex
- $p_e$: edge midpoint

$$p_e = \text{middle}(\text{position}_{\langle 0,2,3 \rangle}(d))$$
Points of interest

with

- \( p_v \): vertex
- \( p_e \): edge midpoint
- \( p_f \): face barycenter

\[
p_f = \text{middle}(\text{position}_{0,1,3}(d))
\]
Points of interest

with

- $p_v$: vertex
- $p_e$: edge midpoint
- $p_f$: face barycenter
- $p_s$: volume barycenter

$$p_s = middle(position_{\langle 0,1,2 \rangle}(d))$$
Points of interest

with

• $p_v$ : vertex
• $p_e$ : edge midpoint
• $p_f$ : face barycenter
• $p_s$ : volume barycenter
• $p_{cc}$ : CC barycenter

$$p_{cc} = \text{middle}(\text{position}_{\langle 0,1,2,3 \rangle}(d))$$
Points of interest

with

- $p_v$: vertex
- $p_e$: edge midpoint
- $p_f$: face barycenter
- $p_s$: volume barycenter
- $p_{cc}$: CC barycenter

Thanks to the points of interest, the system is rewritten as:

$$p = w_v p_v + w_e p_e + w_f p_f + w_s p_s + w_{cc} p_{cc} + t$$
The position expression of $n_2$ only depends on $n_0$. 
The position expression of $n2$ only depends on $n0$.

\[ n2.\text{position} = w_v n0.p_v + w_e n0.p_e + w_f n0.p_f + w_s n0.p_s + w_{cc} n0.p_{cc} + t \]
The position expression of \( n_2 \) only depends on \( n_0 \).

- One equation per dart (8 darts).

\[
n_2.\text{position} = w_v n_0.\text{p}_v + w_e n_0.\text{p}_e + w_f n_0.\text{p}_f + w_s n_0.\text{p}_s + w_{cc} n_0.\text{p}_{cc} + t
\]
The position expression of $n_2$ only depends on $n_0$.

- One equation per dart (8 darts).
- Split per coordinate (on $x$, $y$, $z$).

$$n_2.\text{position} = w_v n_0.p_v + w_e n_0.p_e + w_f n_0.p_f + w_s n_0.p_s + w_{cc} n_0.p_{cc} + t$$
The position expression of $n_2$ only depends on $n_0$.

- One equation per dart (8 darts).
- Split per coordinate (on $x$, $y$, $z$).
- 24 equations and 8 variables.

$n_2\cdot\text{position} = w_v n_0 \cdot p_v + w_e n_0 \cdot p_e + w_f n_0 \cdot p_f + w_s n_0 \cdot p_s + w_{cc} n_0 \cdot p_{cc} + t$

- vertex
- edge
- face
- volume
- cc
The position expression of $n2$ only depends on $n0$.

- One equation per dart (8 darts).
- Split per coordinate (on $x$, $y$, $z$).
- 24 equations and 8 variables.

\[
n2.\text{position} = w_v n0.\text{p}_v + w_e n0.\text{p}_e + w_f n0.\text{p}_f + w_s n0.\text{p}_s + w_{cc} n0.\text{p}_{cc} + t
\]

- Solved as a CSP. Solvers used: OR-Tools (Google), Z3 (Microsoft)
Solving the barycentric triangulation

Global equation:

\[ n2.\text{position} = w_v n0.p_v + w_e n0.p_e + w_f n0.p_f + w_s n0.p_s + w_cc n0.p_cc + t \]
Solving the barycentric triangulation

► Global equation:

\[ n2.\text{position} = w_v n0.\text{p}_v + w_e n0.\text{p}_e + w_f n0.\text{p}_f + w_s n0.\text{p}_s + w_{cc} n0.\text{p}_{cc} + t \]

► Generated system (only on x and y)

\[
\begin{align*}
(0.5; 0.5) &= w_v \ast (0; 0) + w_e \ast (0.5; 0) + w_f \ast (0.5; 0.5) + w_s \ast (0.5; 0.5) + w_{cc} \ast (0.5; 0.5) + (tx; ty) \\
(0.5; 0.5) &= w_v \ast (1; 0) + w_e \ast (0.5; 0) + w_f \ast (0.5; 0.5) + w_s \ast (0.5; 0.5) + w_{cc} \ast (0.5; 0.5) + (tx; ty) \\
(0.5; 0.5) &= w_v \ast (1; 0) + w_e \ast (1; 0.5) + w_f \ast (0.5; 0.5) + w_s \ast (0.5; 0.5) + w_{cc} \ast (0.5; 0.5) + (tx; ty) \\
(0.5; 0.5) &= w_v \ast (1; 1) + w_e \ast (1; 0.5) + w_f \ast (0.5; 0.5) + w_s \ast (0.5; 0.5) + w_{cc} \ast (0.5; 0.5) + (tx; ty)
\end{align*}
\]
Solving the barycentric triangulation

► Global equation:
\[
n2\text{.position} = w_v n0.p_v + w_e n0.p_e + w_f n0.p_f + w_s n0.p_s + w_{cc} n0.p_{cc} + t
\]

► Generated system (only on \(x\) and \(y\))
\[
\begin{align*}
(0.5; 0.5) &= w_v * (0; 0) + w_e * (0.5; 0) + w_f * (0.5; 0.5) + w_s * (0.5; 0.5) + w_{cc} * (0.5; 0.5) + (tx; ty) \\
(0.5; 0.5) &= w_v * (1; 0) + w_e * (0.5; 0) + w_f * (0.5; 0.5) + w_s * (0.5; 0.5) + w_{cc} * (0.5; 0.5) + (tx; ty) \\
(0.5; 0.5) &= w_v * (1; 0) + w_e * (1; 0.5) + w_f * (0.5; 0.5) + w_s * (0.5; 0.5) + w_{cc} * (0.5; 0.5) + (tx; ty) \\
(0.5; 0.5) &= w_v * (1; 1) + w_e * (1; 0.5) + w_f * (0.5; 0.5) + w_s * (0.5; 0.5) + w_{cc} * (0.5; 0.5) + (tx; ty)
\end{align*}
\]

► Solution found:
- \(w_v = 0.0\)
- \(w_e = 0.0\)
- \(w_f = 1.0\)
- \(w_s = 0.0\)
- \(w_{cc} = 0.0\)
- \(t = (0.0, 0.0)\)
JerboaStudio and applications

▶ Implementation of the inference mechanism in Jerboa.
JerboaStudio: inferring the quad subdivision
Folding the quad subdivision

- 768 possible schemes
- 48 schemes tried (marking).
- 14 schemes built (removal of isomorphic rules).
Example inspired from geology

Before

After

Operation

Inference time: $\sim 3$ ms
Example inspired from geology

Before
Example inspired from geology

After
We infer interpolations both for the positions and the colors.
Example inspired from geology (part 2)

Operation

Inference time: ~ 26 ms for the topology, 
~ 549 ms for the embedding expressions
Example inspired from geology (part 2)

Before
Example inspired from geology (part 2)

After
Doo-Sabin subdivision

▶ Rule scheme used and inferred:

\[<0, 1, 2>\]
\[<0, 1, _>\]
\[<_, _, 0>\]
\[<0, _, _>\]
\[<_, 1, 0>\]

\[\text{iteration:} 1\]

Doo et al. 1978.
Doo-Sabin subdivision\textsuperscript{1}

- Rule scheme used and inferred:

\begin{itemize}
  \item \text{<0, 1, 2>}
  \item \text{<0, 1, _>}
  \item \text{<_, _, 0>}
  \item \text{<0, _, _>}
  \item \text{<_, 1, 0>}
\end{itemize}

\begin{itemize}
  \item 2nd iteration:
\end{itemize}

\textsuperscript{1}Doo et al. 1978.
Doo-Sabin subdivision$^1$

Rule scheme used and inferred:

$<0, 1, 2>$
$n_0$

$<0, 1, _>$
$n_0$

$<_, _, 0>$
$n_2$

$<0, _, _>$
$n_1$

$<_, 1, 0>$
$n_3$

$3^{rd}$ iteration:

$^1$Doo et al. 1978.
Menger \((2, 2, 2)\)\(^1\)

\(^1\)Richaume et al. 2019.
Menger $(2, 2, 2)^1$

\[ \begin{align*}
\text{n0} & \rightarrow \text{n6} \\
\text{n71} & \rightarrow \text{n76} \\
\text{n27} & \rightarrow \text{n25} \\
\text{n48} & \rightarrow \text{n46} \\
\text{n39} & \rightarrow \text{n40} \\
\text{n58} & \rightarrow \text{n52} \\
\text{n33} & \rightarrow \text{n34} \\
\text{n55} & \rightarrow \text{n56} \\
\text{n59} & \rightarrow \text{n61} \\
\text{n36} & \rightarrow \text{n38} \\
\text{n14} & \rightarrow \text{n15} \\
\text{n7} & \rightarrow \text{n9} \\
\text{n54} & \rightarrow \text{n57} \\
\text{n62} & \rightarrow \text{n63} \\
\text{n64} & \rightarrow \text{n65} \\
\text{n13} & \rightarrow \text{n16} \\
\text{n74} & \rightarrow \text{n78} \\
\text{n60} & \rightarrow \text{n79} \\
\text{n12} & \rightarrow \text{n11} \\
\text{n70} & \rightarrow \text{n72} \\
\text{n56} & \rightarrow \text{n66} \\
\text{n1} & \rightarrow \text{n3} \\
\text{n77} & \rightarrow \text{n80} \\
\text{n44} & \rightarrow \text{n43} \\
\text{n49} & \rightarrow \text{n47} \\
\text{n21} & \rightarrow \text{n20} \\
\text{n6} & \rightarrow \text{n4} \\
\text{n73} & \rightarrow \text{n17} \\
\end{align*} \]

\[ \begin{align*}
\text{n10} & \rightarrow \text{n5} \\
\text{n17} & \rightarrow \text{n1} \\
\text{n57} & \rightarrow \text{n3} \\
\text{n41} & \rightarrow \text{n42} \\
\text{n76} & \rightarrow \text{n75} \\
\text{n66} & \rightarrow \text{n71} \\
\text{n18} & \rightarrow \text{n19} \\
\text{n8} & \rightarrow \text{n28} \\
\end{align*} \]

\[ \begin{align*}
\text{n45} & \rightarrow \text{n46} \\
\text{n57} & \rightarrow \text{n58} \\
\text{n43} & \rightarrow \text{n44} \\
\text{n59} & \rightarrow \text{n60} \\
\text{n22} & \rightarrow \text{n21} \\
\text{n8} & \rightarrow \text{n78} \\
\text{n23} & \rightarrow \text{n24} \\
\text{n29} & \rightarrow \text{n25} \\
\text{n32} & \rightarrow \text{n31} \\
\text{n34} & \rightarrow \text{n33} \\
\text{n50} & \rightarrow \text{n51} \\
\text{n16} & \rightarrow \text{n17} \\
\text{n53} & \rightarrow \text{n52} \\
\text{n61} & \rightarrow \text{n62} \\
\text{n52} & \rightarrow \text{n53} \\
\text{n9} & \rightarrow \text{n10} \\
\text{n37} & \rightarrow \text{n38} \\
\text{n65} & \rightarrow \text{n64} \\
\text{n26} & \rightarrow \text{n27} \\
\text{n67} & \rightarrow \text{n68} \\
\text{n68} & \rightarrow \text{n69} \\
\text{n55} & \rightarrow \text{n54} \\
\text{n47} & \rightarrow \text{n48} \\
\end{align*} \]

1Richaume et al. 2019.
Menger \((2, 2, 2)^1\)

\[ \text{Richaume et al. 2019.} \]
Edge cases

- Von Koch’s snowflake generated with L-systems

- Inferred:
Edge cases

- **Von Koch’s snowflake generated with L-systems**

- **Inferred:**
JerboaStudio’s architecture

Editor
- Object specifications
  - Dimensions and embeddings
- Creation of rules
  - Quad subdivision, face triangulation, ...
- Static analysis

Embedding Libraries
- 3D Coordinates, RGB Colors, ...

Bridge to view

Jerboa Kernel
- Rule application engine

Generated Modeler Kernel
- Quad subdivision, face triangulation, ...

Generic Viewer
- Load
- Save
- Apply Operations

Automated User input

Generic

Automated
JerboaStudio’s architecture

- Editor
  - Object specifications
  - Dimensions and embeddings
  - Creation of rules
    - Quad subdivision, face triangulation, ...
  - Static analysis

- Embedding Libraries
  - 3D Coordinates, RGB Colors, ...

- Jerboa Kernel
  - Rule application engine
  - Quad subdivision, face triangulation, ...

- Generic Viewer
  - Load
  - Save
  - Apply Operations

- Inference Module

- Generated Modeler Kernel

R. Pascual
Ph.D. defense
November 29, 2022
Conclusion

- Related works, main contributions, and future works.
Other lines of research on inference

- **Inferring the generation of an object:**
  - Inverse procedural modeling: retrieving parameters.\(^1\)
  - L-systems: retrieving formal rules.\(^2\) Illustration from (Guo et al. 2020).
  - Constructive solid geometry: retrieving sequences of operations.\(^3\)

\(^1\)Wu et al. 2014; Emilien et al. 2015.
\(^3\)Sharma et al. 2018; Kania et al. 2020; Xu et al. 2021.
Other lines of research on inference

- Inferring the generation of an object
- Pure geometry

- Retrieve non-linear weights of a Loop-based subdivision scheme for mesh refinement. Illustration from (Liu et al. 2020).
Other lines of research on inference

▶ Inferring the generation of an object
▶ Pure geometry
▶ Graph transformations
  • Domain-based inference mechanism retrieving or exploiting graph transformations.\(^1\) Illustration from (Dinella et al. 2020).

Main contributions

**Inference of modeling operations:**
- Topological folding algorithm
- Values of interest and CSP

▶ JerboaStudio.

**Graph transformations for geometric modeling:**
- Graph products
- Rule completion

▶ Unified framework to study generalized and oriented maps.
Future works

- **Automatic mapping**
  - Cumbersome step in the inference workflow.
Future works

- Automatic mapping

- Other hypotheses for the geometric inference
  - Most subdivision schemes rely on other computations: the Catmull-Clark subdivision.\(^1\)

\[\text{Catmull et al. 1978.}\]

```
// From Catmull and Clark 1978, Recursively generated
// B-spline surfaces on arbitrary topological meshes

// nl#position
// midpoint of the incident face
Point3 face1Mid = Point3::middle(<0,1>.position(n0));
// midpoint of the adjacent face
Point3 face2Mid = Point3::middle(<0,1>.position(n0@2));
// average of the face points
Point3 faceMid = Point3::middle(face1Mid, face2Mid);
// midpoint of the edge
Point3 edgeMid = Point3::middle(<0>.position(n0));
// average of the edge and face points
return Point3::middle(faceMid, edgeMid);
```

Out of scope
Future works

- Automatic mapping
- Other hypotheses for the geometric inference
- Inference in graph transformations
  - Formalize the inference mechanism with categorical constructions.
Thank you for listening
Appendix

References


References III


References IV


References VI


