Hierarchies relating Topology and Geometry

Walter G. Kropatsch, Yll Haxhimusa, Adrian Ion
PRIP (183/2)
Pattern Recognition and Image Processing
Institute of Computer Aided Automation
Vienna Univ. of Technology- Austria
Graphs in Image Analysis

- Dual Graph Contraction
- Graph Pyramids
CONTENTS

- The Problem
- Partitions and Adjacencies
- How to Reduce the Descriptions
- Topology Preserving Operations
- Combinatorial Maps
- Abstract Cellular Complexes (ACC)
- Conclusions, Discussion
The PROBLEM in 2D

Label Array $\mathcal{P}_0$

Regions $\mathcal{P}_1$

Pixel Adjacency $\mathcal{A}_0$

Region Adjacency $\mathcal{A}_1$

$\text{lab} : \mathcal{P}_0 \mapsto \mathcal{L}$

$|\mathcal{P}_0| > |\mathcal{L}|$

$(\mathcal{P}_0, \mathcal{A}_0, \mathcal{L})$-Properties

Partition

$\text{lab} : \mathcal{P}_1 \mapsto \mathcal{L}$

$|\mathcal{P}_0| > |\mathcal{P}_1| \geq |\mathcal{L}|$

$(\mathcal{P}_1, \mathcal{A}_1, \mathcal{L})$-Prop.

The PROBLEM in 3D
The PROBLEM in 3D

Label Array $\mathcal{P}_0$ $\rightarrow$ Regions $\mathcal{P}_1$

$\rightarrow$ HOLES

$\rightarrow$ TUNNELS

Properties/Relations between $(\mathcal{P}_0, A_0)$ and $(\mathcal{P}_1, A_1)$
From Pixels To Graphs

IMAGE PIXELS  NEIGHBORHOOD GRAPH
## Properties/Relations between $(\mathcal{P}_0, \mathcal{A}_0)$ and $(\mathcal{P}_1, \mathcal{A}_1)$

<table>
<thead>
<tr>
<th>$\mathcal{P}_0$</th>
<th>partition</th>
<th>1 Label/Cell</th>
<th>$\mathcal{P}_1$</th>
<th>partition (?)</th>
<th>1 Cell/Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c \in \mathcal{P}_0$</td>
<td>unit square</td>
<td>simply connected</td>
<td></td>
<td>arbitrary shape</td>
<td>(simply) connected</td>
</tr>
<tr>
<td></td>
<td>simplex</td>
<td>no inclusion</td>
<td></td>
<td>simplex ?</td>
<td>inclusion (tree)</td>
</tr>
<tr>
<td></td>
<td>1 Label/CC ?</td>
<td></td>
<td></td>
<td></td>
<td>$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>minimal</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\mathcal{A}_0$</th>
<th>4-connected ?</th>
<th>well composed</th>
<th>embedding</th>
<th>Genus(CC(label))</th>
<th>orientation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$(c_1, c_2) \in \mathcal{A}_0$</td>
<td>$lab(c_1) \neq lab(c_2)$</td>
<td>$lab(c_1) = lab(c_2)$</td>
<td>image edge</td>
<td>Jordan boundaries</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\mathcal{A}_1$</th>
<th>RAG (connected ?)</th>
<th>connectivity preserved</th>
<th>embedding</th>
<th>Genus(CC(label))</th>
<th>orientation</th>
<th>artefacts?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$(c_1, c_2) \in \mathcal{A}_1$</td>
<td>multiple $(lab_1, lab_2)$</td>
<td>$(lab_1, lab_1)$ (pseudo,self-loop)</td>
<td>connected segment</td>
<td>Jordan boundaries</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

How to perform $T : (\mathcal{P}_0, \mathcal{A}_0) \leftrightarrow (\mathcal{P}_1, \mathcal{A}_1)$ ?
How to perform $T : (\mathcal{P}_0, \mathcal{A}_0) \mapsto (\mathcal{P}_1, \mathcal{A}_1)$?

<table>
<thead>
<tr>
<th>$(\mathcal{P}_0, \mathcal{A}_0)$</th>
<th>$(\mathcal{P}_1, \mathcal{A}_1)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pixelarray: $I(x, y)$</td>
<td>Combinatorial 2-Maps $(\mathcal{D}_1, \alpha_1, \sigma_1)$ + Incl. Tree</td>
</tr>
<tr>
<td>ACC</td>
<td>Abstract Cellular Complex-graph?</td>
</tr>
<tr>
<td>G-Maps + Incl. Tree</td>
<td>$(\mathcal{D}_1, \alpha_0, \alpha_1, \alpha_2)$</td>
</tr>
<tr>
<td>Dual Graphs: $((V_0, E_0), (F_0, \overline{E}_0))$</td>
<td>Dual Graphs $((V_1, E_1), (F_1, \overline{E}_1))$</td>
</tr>
<tr>
<td>2-Maps $(\mathcal{D}_0, \alpha_0, \sigma_0)$</td>
<td>$(\mathcal{D}_1, \alpha_1, \sigma_1)$</td>
</tr>
</tbody>
</table>
Possible Realizations of $T$:

1. **direct construction**
   
   (a) Connected Component Labeling
   (b) 2-Maps by Pixelscan
   (c) 2-Maps by Precode-Guided FUSION
   (d) G-Maps by Precode-Guided FUSION?

2. **DS conversion + repeated reductions**
   
   (a) 2-Maps by Precode-Guided (sequential) FUSION
   (b) Dual Graphs by Parallel Dual Conctractions
# 2D Removal and Contraction

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Primal Graph</th>
<th>Dual Graph</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A \parallel B$</td>
<td>Remove</td>
<td>Contract $\xrightarrow{A \rightarrow B}$</td>
</tr>
<tr>
<td>$A + B$</td>
<td>merges $A$ and $B$</td>
<td></td>
</tr>
</tbody>
</table>

**Common($A,B$) true** $\iff$ **Same Label** $\rightarrow$ **CCL**

- **Similar Color** $\rightarrow$ **Segmentation**
- ’belong together’ $\rightarrow$ **Grouping**
2 × 2 CCL

contract  contract
remove  contract
remove

Contraction Kernel, Simplification
Contraction Kernel, Simplification

contract more simplified

contraction kernel self-loop removed

Spanning tree \( \deg(f) < 3 \)
Irregular Graph Pyramid
Building Algorithm

Dual Graph Pyramid Algorithm

Input: Graphs $\langle G_0, \overline{G}_0 \rangle$

1: while further abstraction is possible do
2: select contraction kernels
3: perform graph contraction and simplification of dual graph (DGC [Kropatsch, 1995a])
4: apply reduction functions to compute new reduced level
5: end while

Output: Graph pyramid $\langle G_k, \overline{G}_k \rangle$, $0 \leq k \leq h$.

Graph pyramid is a stack of $\langle G_k, \overline{G}_k \rangle$, $0 \leq k \leq h$
Dual Graph Contraction

Label Array $\mathcal{P}_0$ → Regions $\mathcal{P}_1$

Contraction Kernels → (dual) Simplification Kernels

Example 1
What Remains...

Both Contraction Trees

Contraction Result

local characterization

1 cell/CC(label)
lab(c₁) ≠ lab(c₂)
≥ 3 regions meet
or background
removal/contraction

Example 2
Dual Graph Contraction

Example 2

Label Array $\mathcal{P}_0$ → Regions $\mathcal{P}_1$

Contraction Kernels → (dual) Simplification Kernels

What Remains...
What Remains...

Both Contraction Trees

\[ \rightarrow \]

Contraction Result

local characterization

-emplate/CC(label)\nlab(c_1) \neq lab(c_2)

except

self-loop, multi-edge with inclusion

\geq 3 \text{ regions meet} \n\text{or background}

\text{removal/contraction}

Topology Preserving Operations in 2D
### Topology Preserving Operations in 2D

<table>
<thead>
<tr>
<th>Operation</th>
<th>Points</th>
<th>Lines</th>
<th>Faces</th>
<th>Cond.</th>
<th>PRE-CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Euler</td>
<td># P</td>
<td>-# L</td>
<td>+# F</td>
<td>= const.</td>
<td>to preserve:</td>
</tr>
<tr>
<td>Incre.</td>
<td>ΔP</td>
<td>-ΔL</td>
<td>+ΔF</td>
<td>= 0</td>
<td>Euler; Orientation</td>
</tr>
<tr>
<td>Contract(l, p₀)</td>
<td>-1</td>
<td>-1</td>
<td></td>
<td>(p₁, l, p₀)</td>
<td>p₁ ≠ p₀;</td>
</tr>
<tr>
<td>Remove(l, f₀)</td>
<td></td>
<td></td>
<td>-1</td>
<td>(fₓ, l, f₀)</td>
<td>fₓ ≠ f₀; deg(f₀) ≤ 2</td>
</tr>
<tr>
<td>Any Incre.</td>
<td>-a</td>
<td>-b</td>
<td>-c</td>
<td>×a</td>
<td>b = a + c;</td>
</tr>
<tr>
<td>by a contr.</td>
<td>-1</td>
<td>-1</td>
<td></td>
<td>×a</td>
<td></td>
</tr>
<tr>
<td>by c remov.</td>
<td>-1</td>
<td>-1</td>
<td></td>
<td>×c</td>
<td></td>
</tr>
</tbody>
</table>
A Few Comments

1. $p_0 \neq p_1 \Rightarrow$ no contraction of self-loop.

2. $(f_x, l, f_0)$: self-loop in dual = bridge in primal, Removal $\Rightarrow$ Disconnection.

3. removal: $l$ not bridge, since $(f_x, l, f_0 \neq f_x)$ not self-loop
   $\Rightarrow$ removal does NOT disconnect.

4. $\deg(l_0) > 2 : (f_x, l, f_0, l_i), i = 1, 2, \ldots$: order of $(f_x, l_i)$ may change.

5. Contraction and removal are the ONLY operations needed.

6. Any other topology-preserving operation can be achieved by appropriate combinations of contraction and removals.

7. Note that negative contractions and negative removals are possible as the inverse operations under certain pre-conditions.

8. Pre-conditions for individual operations can be extended to sets of operations: FOREST requirement (no cycle) for contraction.

9. CC: additional pre-condition: $lab(p_1) \neq lab(p_2)$. 
What remains after repetition?

Contraction inhibited

BY CONTENT
(e.g. diff. LABELs)

BY TOPOLOGY
(e.g. DISCONNECT)

HOLEs
(Self-loops, multi-edge)
Betti number: \( \# \) holes = \( \# \) fictive edges

FICTION Edges
not UNIQUE!
unified data structure

Removal + Inclusion tree
well defined
separate data structure

Topology Preserving Operations in 3D
Dual Graphs vs. Maps

Dual Graphs CANNOT distinguish,

Reason: self-loops lose local orientation:

\{(1, 1; \Omega, \infty), (1, 5; \Omega, \Omega), (5, 5; A, \Omega), (5, 2; A, A), (5, 5; C, \Omega), (5, 4; C, C), (5, 5; B, \Omega), (5, 3; B, B)\}

MAPS can!

Dual Graph Contraction

Example 1
Envelope: Graph

Adjacency Matrix:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Envelope: Combinatorial Map \((\mathcal{D}, \sigma, \alpha)\)
Envelope: Combinatorial Map \((\mathcal{D}, \sigma, \alpha)\)
Envelope: Combinatorial Map \((\mathcal{D}, \beta_1, \beta_2)\)

\[
\begin{array}{cccccccccc}
\mathcal{D} & 12 & 13 & 21 & 24 & 25 & 31 & 34 & 42 & 43 & 45 & 52 & 54 \\
\beta_2 & 21 & 31 & 12 & 42 & 52 & 13 & 43 & 24 & 34 & 54 & 25 & 45 \\
\beta_1 & 13 & 12 & 25 & 21 & 24 & 34 & 31 & 45 & 42 & 43 & 54 & 52 \\
\end{array}
\]
# Topology Preserving Operations in 3D

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Euler</td>
<td>$#P$</td>
<td>$-#L$</td>
<td>$+#F$</td>
<td>$-#V$</td>
<td>$= \text{const.}$</td>
<td>pres.Euler</td>
</tr>
<tr>
<td>Incr.</td>
<td>$\Delta P$</td>
<td>$-\Delta L$</td>
<td>$+\Delta F$</td>
<td>$-\Delta V$</td>
<td>$= 0$</td>
<td>$v_1 \neq v_2? \ (\text{deg } f = 2)$</td>
</tr>
<tr>
<td>V-Fusion</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>$(v_1, f, v_2)$</td>
<td>$f_1 \neq f_2? \ \text{deg } l \leq 2$</td>
<td></td>
</tr>
<tr>
<td>F-Fusion</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>$(f_1, l, f_2)$</td>
<td>$l_1 \neq l_2? \ \text{deg } p \leq 2$</td>
<td></td>
</tr>
<tr>
<td>L-Fusion</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>$(l_1, p, l_2)$</td>
<td>pres.Euler</td>
<td>see below</td>
</tr>
<tr>
<td>Any Incr.</td>
<td>$(-a, -b, -c, -d)$</td>
<td>pres.Euler</td>
<td>see below</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>by V-Fus.</td>
<td>$(-1, -1)$</td>
<td>$\times d$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>by F-Fus.</td>
<td>$(-1, -1)$</td>
<td>$\times (c - d)$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>by L-Fus.</td>
<td>$(-1, -1)$</td>
<td>$\times a$</td>
<td>with $b + c = a + d$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
A Few Comments

1. V-Fusion, F-Fusion and L-Fusion are the ONLY operations needed.
2. Any other topology-preserving operation can be achieved by appropriate combinations.
3. Pre-conditions for individual operations can be extended to sets of operations: FOREST requirement (NOTHING INSIDE) and ??.
4. Note that negative Fusions are possible under certain pre-conditions as the inverse operations.
5. Pre-conditions 2 in 3D are not trivial except for faces: a line may delimit more than 2 faces, and a point may be the intersection of more than 2 lines.
6. Note that the L-fusion in 3D eliminates a point along a line sequence whereas the contraction in 2D eliminates a line between two points. As result the 2D contraction produces a point while 3D l-fusion produces a line.
7. CC: additional pre-condition: \( lab(v_1) \neq lab(v_2) \), no further pre-condition for F-Fusion and L-Fusion.
What remains after repetition?

**Contraction inhibited**

- **BY CONTENT**
  (e.g. diff. LABELs)

- **BY TOPOLOGY**
  (e.g. DISCONNECT)

- **TUNNELS**

- **HOLES**

- **FICTIVE Elements**

- Removal + Inclusion tree

The Olympic Solution?
Some useful properties:

1. we need multi-edge and self-loop
2. combine operations (like ECK)
3. repeated contraction, termination criteria
4. pseudo/fictive elements characterize topological relations
   pseudo edge $\leftrightarrow$ hole, pseudo face $\leftrightarrow$ tunnel, ...
5. independence of operations allows for:
   - parallelism
   - optimized scan
   - divide and conquer

Open problems:
Open problems:

1. 3D, 4D
2. interlaced thori
3. Re-insertion of removed edges/darts (a la wavelet)
4. pre-condition (single OP) $\implies$ pre-condition (set of OPs)
5. repeated contraction:
   - different selection criteria
   - different termination criteria
   - different attributes
   - different reduction functions
6. Betti numbers? homology groups? generators?
7. appropriate applications?

Thank you