A Discrete, Continuous, and Self-Dual Representation of 2D Images

Thierry Géraud

theo@lrde.epita.fr

EPITA Research and Development Laboratory (LRDE)

February 2013

Outline

Outline

Outline

Outline

Outline

Outline

OUTLINE

Image Processing at LRDE

- young lab with a small image processing (IP) team
- (dev) Olena = IP platform, including Milena:
  - C++ image processing library
  - generic and efficient
  - easy prototyping / industrial code
  - many structures
- IP research topics
  - translation IP $\rightarrow$ generic code
  - algorithms
  - stuff about mathematical morphology / discrete topology $\leftarrow$ HERE
  - document image analysis / text extraction from natural images & videos

This Talk = a Story

Serendipity

Serendipity means a "happy accident" or "pleasant surprise"; specifically, the accident of finding something good or useful while not specifically searching for it. (Wikipedia)

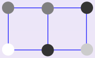
searching for an algorithm $\rightsquigarrow$ finding a representation of images

Some Representations of Image

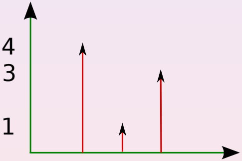
image = ?

2	2	1
4	1	3

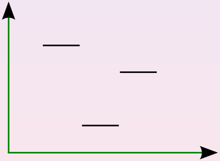
array



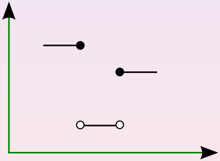
graph



distribution



function



function u.s.c.

This Talk is About...

Many subjects involved:

- image processing
- mathematical morphology
- mathematical analysis
- discrete topology
- and discrete geometry, computer graphics...

we are going to talk about pixel-level details...

...yet the devil is in the detail!

Context: Document Image Processing

Education

America's Hot Colleges

Yes, Harvard's on the list. But so are lesser-known schools. Here are our picks for the places creating buzz for 2005-06.

BY JAY MATHEWS

FOR STUDENTS LOOKING to attend an American university, a few names have always loomed large: the eight Ivies, a few small institutions like Amherst and some celebrated state schools like the University of California, Berkeley. But increasingly, today's students are widening their searches and discovering many schools that are just as good—and often just about as difficult to get into—as the famous ones. And it's sort of cool to find out that a hot college doesn't need to be one that Grandma and Grandpa have heard of.

With competition for spots in U.S. universities fiercer than ever, families are looking for lesser-known schools that make the grade, along with those icons that live up to their reputations. All the colleges on the Hot List for 2005-06 have one attribute in common: they're creating buzz among students, school officials and longtime observers of the admissions process. And each entry reflects a place that is preparing students well for a complex world. Herewith, eight of our picks for America's Hottest Colleges:

HOTTEST FOR REJECTING YOU

Harvard University, Cambridge, Massachusetts

Last year Yale edged out Harvard as the most selective Ivy. But after Harvard announced that families earning less than $40,000 wouldn't have to pay the usual parental contribution to tuition, applications jumped to a record 22,796, and the acceptance rate this spring dropped to a new low for the Ivies, only 9.1 percent. The aid initiative increased the number of low-income students—296 qualified. Bottom line: competition was tougher than ever. Harvard undergrads often mock themselves. "It's nice to know you're going to school with people who will control the world," says senior Simon Vozick-Levinson. But they also know how to take advantage. The student paper, the Crimson, is putting out a new book, "How They Got Into Harvard," which profiles successful applicants, along with a second book full of winning application essays.

HOTTEST FOR SCIENCE

University of California, San Diego, La Jolla

Science can be fun. UCSD undergraduates mark the end of the school year with the Watermelon Drop, a 40-year-old tradition that began when physics students tested velocity by dropping a melon from seven stories up. On a campus where a quarter of the $1.8 billion in revenue is federal research funds, and where there are eight Nobel laureates on the faculty, the science is also quite serious. UCSD chancellor Marye Anne Fox, an organic chemist, says welcoming undergrads into labs is a priority. The school, she says, is raising the quality of undergraduate education by offering new science majors like molecular synthesis and bioinformatics.

Its coastal location, too, is a plus. "Where else can you collect samples from the beach, the desert and the mountains all in one day—and still have time to run genetic tests on them that night?" says Meg Eckles, a biology doctoral student. Faculty and alumni have spun off nearly 200 companies, including about a third of the region's biotech firms.

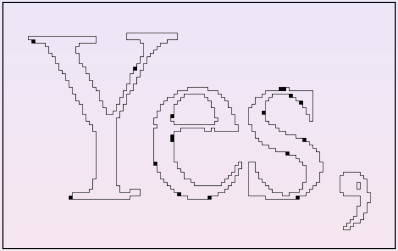
HOTTEST FOR LIBERAL ARTS

Macalester College, St. Paul, Minnesota

The 1,900-student campus in a vibrant metropolis has become a key recipient of the growing number of Harvard, Yale and Princeton applicants who are rejected for no other reason than that those schools don't have space

50

PHOTO ILLUSTRATION BY C. J. BURTON FOR NEWSWEEK



"$\Delta = 0$" lines

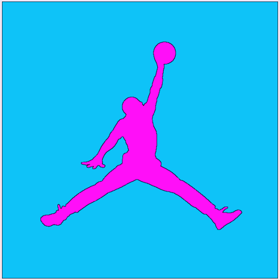
a **HUGE** magazine page image

yet with **very tiny and thin** objects!

Some Prerequisites 1/2

Jordan Curve Theorem

Every simple closed curve divides the plane into an ”interior” region and an “exterior” region.



Some Prerequisites 2/2

Excerpt from:

Digital topology and applications

by Jacques-Olivier Lachaud

Laboratoire de Mathématiques (UMR 5127), Université de Savoie

slides of "Séminaire de Géométrie, 4 avril 2008"

http://www.lama.univ-savoie.fr/~lachaud/People/LACHAUD-JO/Talks/chambery-geometry-2008-slides.pdf

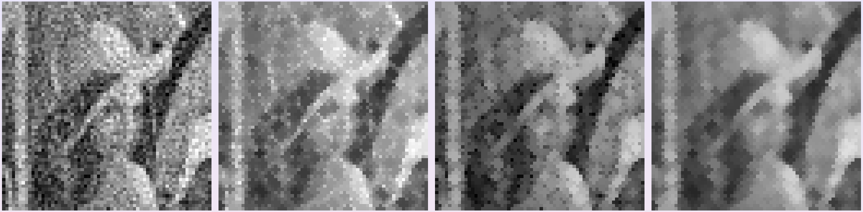
Outline

Morphology and Applications

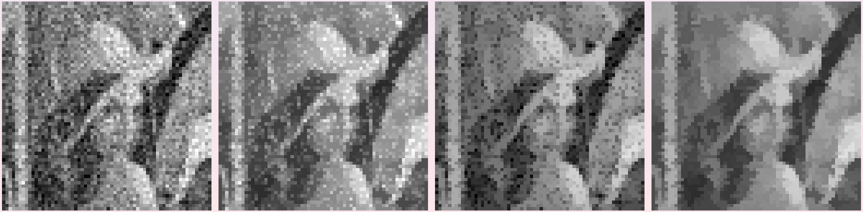
- document image analysis
  - objects with different scales (from tiny to very large)
  - different levels of contrast
  - contrast inversion $\leftarrow$ *self-duality is required*
  - gray levels / colors
- connected filters (v. "structuring element"-based morphology)
  - preserve contours
  - underlying tree representation $\leftarrow$ *tree manipulations are enabled*
  - many apps including:
    filtering / simplification / object recognition / indexing / segmentation...

Some Illustrations

"Structuring element"-based morphology:



Morphological connected filters:



From left to right: id, ϕ, γ, $\frac{\phi\gamma+\gamma\phi}{2}$.

REMARK 1: DUALITY V. SELF-DUALITY

- ϕ and γ are dual operators:
  - we have $\gamma = \complement\,\phi\,\complement$
  - ϕ filters dark objects over light background
  - γ does the opposite
- $\nu = \frac{\phi\gamma+\gamma\phi}{2}$ is self-dual:
  - it satisfies $\complement\,\nu = \nu\,\complement$
  - it makes no assumption about object/background contrast
- duality should be avoided when:
  - we <u>cannot</u> make an assumption about contrast
  - we <u>do not want</u> to make such an assumption

$\boxed{\text{notion of “object”} \neq \text{notion of “subject”}}$

Image and Function

in the following:

an image is a mapping $u : X \to Y$

we can have $X = \mathbb{Z}^2$ and $Y = \mathbb{Z}$ ...

Remark 2: Cuts

Cuts

$$\text{lower cuts:} \quad [\, u \leq \lambda \,] = \{\, x \in X \mid u(x) \leq \lambda \,\}$$
$$\text{upper cuts:} \quad [\, u \geq \lambda \,] = \{\, x \in X \mid u(x) \geq \lambda \,\}$$

Remark 2: Cuts

Cuts

$$\text{lower cuts:} \quad [\, u \le \lambda \,] = \{\, x \in X \mid u(x) \le \lambda \,\}$$
$$\text{upper cuts:} \quad [\, u \ge \lambda \,] = \{\, x \in X \mid u(x) \ge \lambda \,\}$$

- clue of the extension of morphology on sets to functions
- used to define some connected filters
  - algebraic openings and closings
  - some levelings

Remark 2: Cuts

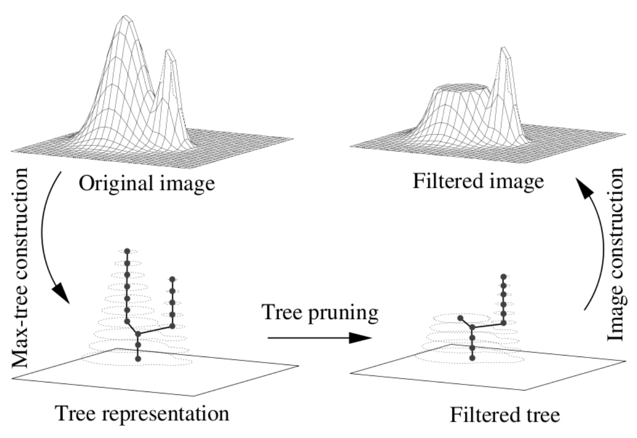
Cuts

$$\begin{aligned} \text{lower cuts:} \quad [\, u \leq \lambda \,] &= \{\, x \in X \mid u(x) \leq \lambda \,\} \\ \text{upper cuts:} \quad [\, u \geq \lambda \,] &= \{\, x \in X \mid u(x) \geq \lambda \,\} \end{aligned}$$

$\Rightarrow$ a couple of dual trees

- lower cuts $\rightsquigarrow$ min-tree $\rightsquigarrow$ filtering dark objects (e.g., ϕ)
- upper cuts $\rightsquigarrow$ max-tree $\rightsquigarrow$ filtering light objects (e.g., γ)
- filtering = tree pruning

Remark 2: Cuts



how to run γ

Remark 2: Cuts

we can try to be self-dual with two trees...

yet we get some info redundancy between trees

and

we have to juggle with 2 structures

Remark 3: A Direct Self-Dual Approach

- the best case is to “directly run a filter”

$\xrightarrow{\text{tree + pruning}}$

- grain filters κ:
  - connected filters (preserve some level lines $\partial[\, u \leq \lambda \,]$)
  - actually $\frac{\phi\gamma + \gamma\phi}{2} \;\approx\; \kappa$
  - based on a single and self-dual tree
- we need this *tree of shapes* (ToS)!

Remark 3: A Direct Self-Dual Approach

- the best case is to “directly run a filter”

$\xrightarrow{\text{tree + pruning}}$

- grain filters κ:
  - connected filters (preserve some level lines $\partial[\, u \leq \lambda \,]$)
  - actually $\frac{\phi\gamma+\gamma\phi}{2} \approx \kappa$
  - based on a single and self-dual tree
- we need this *tree of shapes* (ToS)!

Remark 3: A Direct Self-Dual Approach

- the best case is to "directly run a filter"

$\xrightarrow{\text{tree + pruning}}$

- grain filters κ:
  - connected filters (preserve some level lines $\partial[\, u \leq \lambda \,]$)
  - actually $\frac{\phi\gamma+\gamma\phi}{2} \approx \kappa$
  - based on a single and self-dual tree

- we need this *tree of shapes* (ToS)!

REMARK 3: A DIRECT SELF-DUAL APPROACH

- the best case is to “directly run a filter”

$\xrightarrow{\text{tree + pruning}}$

- grain filters κ:
  - connected filters (preserve some level lines $\partial[\, u \leq \lambda \,]$)
  - actually $\frac{\phi\gamma+\gamma\phi}{2} \approx \kappa$
  - based on a single and self-dual tree

- we need this *tree of shapes* (ToS)!

Remark 3: A Direct Self-Dual Approach

- the best case is to "directly run a filter"

$\xrightarrow{\text{tree + pruning}}$

- grain filters κ:
  - connected filters (preserve some level lines $\partial[\, u \leq \lambda \,]$)
  - actually $\frac{\phi\gamma + \gamma\phi}{2} \approx \kappa$
  - based on a <u>single and self-dual</u> tree

- we need this *tree of shapes* (ToS)!

Remark 3: A Direct Self-Dual Approach

- the best case is to “directly run a filter”

$\xrightarrow{\text{tree + pruning}}$

- grain filters κ:
  - connected filters (preserve some level lines $\partial[\, u \leq \lambda \,]$)
  - actually $\frac{\phi\gamma+\gamma\phi}{2} \approx \kappa$
  - based on a single and self-dual tree

- we need this *tree of shapes* (ToS)!

WHAT'S WRONG

state of the art $=$ 3 different algorithms to compute the ToS:

- yet with $\mathcal{O}(N^2)$ complexity...
- rather hard to implement...
- unusable for nD images...

(just unthinkable for a computer scientist!)

What's Wrong

state of the art = 3 different algorithms to compute the ToS:

- yet with $\mathcal{O}(N^2)$ complexity...
- rather hard to implement...
- unusable for nD images...

(just unthinkable for a computer scientist!)

moreover:

- some topological inconsistencies...
- and only a "quasi-self-dual" ToS...

(so we want to fix those issues...)

Outline

Two Ways of Cutting

Dual Cuts

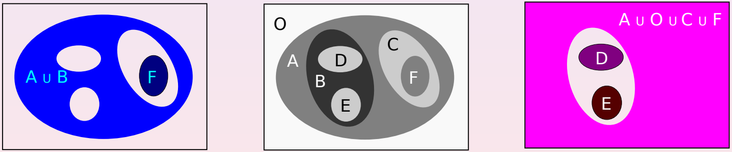
$$\text{lower cuts:} \quad [\, u \leq \lambda \,] = \{\, x \in X \mid u(x) \leq \lambda \,\}$$
$$\text{upper cuts:} \quad [\, u \geq \lambda \,] = \{\, x \in X \mid u(x) \geq \lambda \,\}$$

Two Ways of Cutting

Dual Cuts

$$\text{lower cuts:} \quad [\, u \leq \lambda \,] = \{\, x \in X \mid u(x) \leq \lambda \,\}$$
$$\text{upper cuts:} \quad [\, u \geq \lambda \,] = \{\, x \in X \mid u(x) \geq \lambda \,\}$$

with λ being dark gray:



$[\, u \leq \lambda \,]$ u $[\, u \geq \lambda \,]$

DUAL TREES

Given a function u, consider the set of components of every upper cuts:

$$\mathcal{T}_{\geq}(u) \;=\; \{\, \Gamma \in \mathcal{CC}([\, u \geq \lambda\,])\,\}_{\lambda}$$

Dual Trees

Given a function u, consider the set of components of every upper cuts:

$$\mathcal{T}_{\geq}(u) \;=\; \{\, \Gamma \in \mathcal{CC}([\, u \geq \lambda\,])\,\}_{\lambda}$$

we have:

- a couple of components are *either* disjoint *or* included one in another,
- so the components of $\mathcal{T}_{\geq}(u)$ form a tree, the max-tree of u.

Dual Trees

Given a function u, consider the set of components of every upper cuts:

$$\mathcal{T}_{\geq}(u) \;=\; \{\, \Gamma \in \mathcal{CC}([\, u \geq \lambda\,])\,\}_{\lambda}$$

we have:

- a couple of components are *either* disjoint *or* included one in another,
- so the components of $\mathcal{T}_{\geq}(u)$ form a tree, the max-tree of u.

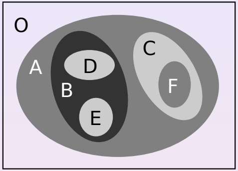
and with the lower cuts' components:

$$\mathcal{T}_{\leq}(u) \;=\; \{\, \Gamma \in \mathcal{CC}([\, u \leq \lambda\,])\,\}_{\lambda}$$

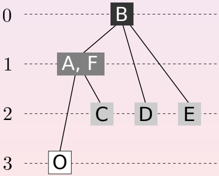
we have the min-tree of u.

A Schematic Example

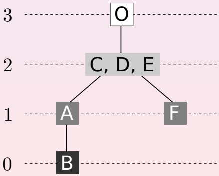
image



max-tree



min-tree



Tree of Shapes

Consider the saturation (fill holes) set operator Sat:

$$\mathcal{T}(u) = \{ \, \mathrm{Sat}(\Gamma), \; \Gamma \in \mathcal{T}_{\geq} \cup \mathcal{T}_{<} \, \}$$

Tree of Shapes

Consider the saturation (fill holes) set operator Sat:

$$\mathcal{T}(u) = \{ \operatorname{Sat}(\Gamma), \; \Gamma \in \mathcal{T}_{\geq} \cup \mathcal{T}_{<} \}$$

we have:

- an element of $\mathcal{T}$ is called a *shape*
- a couple of shapes are either disjoint or included one in another
- so the elements of $\mathcal{T}(u)$ form a tree, the *tree of shapes* of u

Tree of Shapes

Consider the saturation (fill holes) set operator Sat:

$$\mathcal{T}(u) \;=\; \{\, \mathrm{Sat}(\Gamma),\; \Gamma \in \mathcal{T}_{\geq} \cup \mathcal{T}_{<} \,\}$$

we have:

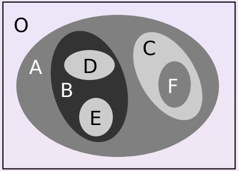
- an element of $\mathcal{T}$ is called a *shape*
- a couple of shapes are either disjoint or included one in another
- so the elements of $\mathcal{T}(u)$ form a tree, the *tree of shapes* of u

actually

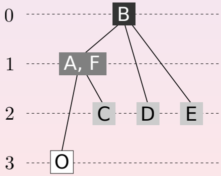
- the shapes are the holes of cut components

A Schematic Example

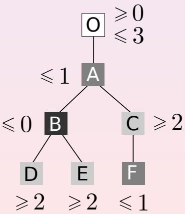
image



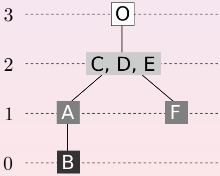
max-tree



tree of shapes



min-tree



A TWO-PASS ALGORITHM

A two-pass algorithm is known to compute the max-tree or min-tree:

1. sort the pixels in the *descending tree order*
2. following the *reverse order*,
   distort the Union-Find algorithm to compute the tree.

A Two-Pass Algorithm

A two-pass algorithm is known to compute the max-tree or min-tree:

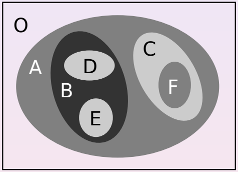
1. sort the pixels in the *descending tree order*
2. following the *reverse order*,
   distort the Union-Find algorithm to compute the tree.

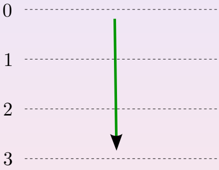
When pixel values have a low quantization (less than 16 bit):

- sorting is of linear complexity (distributed sort),
- so we get a <u>quasi-linear algorithm</u> (complexity of the Union-Find step).

Computing the Max-Tree

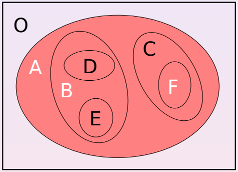
sort : B — A, F — C — D — E — O →

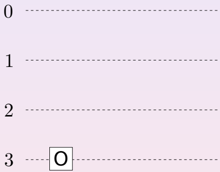




Computing the Max-Tree

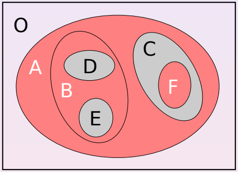
sort : B A, F C D E O

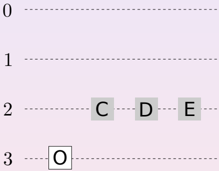




Computing the Max-Tree

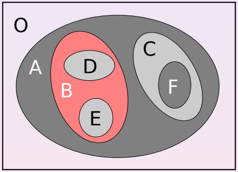
sort : B A, F C D E O

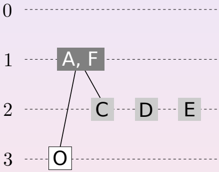




Computing the Max-Tree

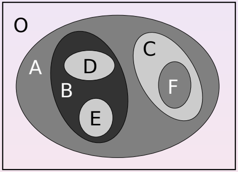
sort : B A, F C D E O

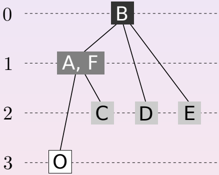




Computing the Max-Tree

sort : B A, F C D E O





What If…

A First Key Idea

if we succeed in sorting the pixels such as descending the tree of shapes, then we have a simple and efficient algorithm.

What If...

A First Key Idea

if we succeed in sorting the pixels such as descending the tree of shapes,
then we have a simple and efficient algorithm.

A Second Key Idea

sorting the pixels means progress “continuously”
both in *image space*[1] and in *value space*[2]

What If...

A First Key Idea

if we succeed in sorting the pixels such as descending the tree of shapes, then we have a simple and efficient algorithm.

A Second Key Idea

sorting the pixels means progress "continuously"
both in *image space*[1] and in *value space*[2]

[1] i.e., through a spatially consistent growing (thanks to a propagation front)

What If...

A First Key Idea

if we succeed in sorting the pixels such as descending the tree of shapes, then we have a simple and efficient algorithm.

A Second Key Idea

sorting the pixels means progress "continuously"
both in *image space*[1] and in *value space*[2]

[1] i.e., through a spatially consistent growing (thanks to a propagation front)

[2] i.e., jumping from a gray level to the "next" one (either upper or lower)

What If...

A First Key Idea

if we succeed in sorting the pixels such as descending the tree of shapes, then we have a simple and efficient algorithm.

A Second Key Idea

sorting the pixels means progress "continuously"
both in *image space*[1] and in *value space*[2]

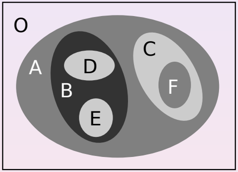
[1] i.e., through a spatially consistent growing (thanks to a propagation front)

[2] i.e., jumping from a gray level to the "next" one (either upper or lower)

⇒ we can use a *hierarchical* queue!

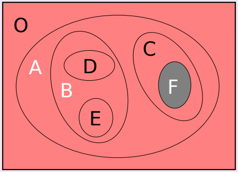
Computing the Tree of Shapes

sort : 



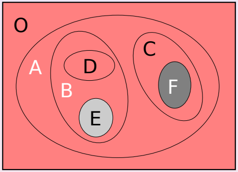
Computing the Tree of Shapes

sort : O A B C D E F



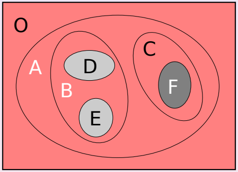
Computing the Tree of Shapes

sort : O A B C D E F



Computing the Tree of Shapes

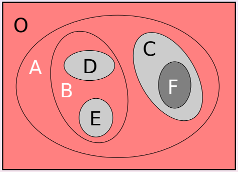
sort : O A B C D E F



D E F

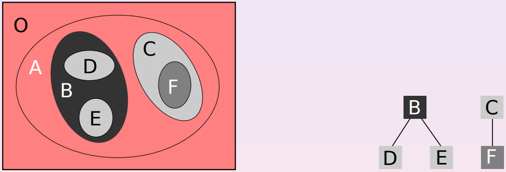
Computing the Tree of Shapes

sort : O A B C D E F



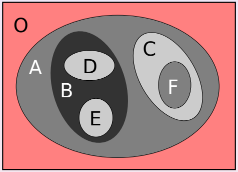
Computing the Tree of Shapes

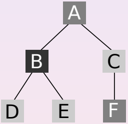
sort : O A B C D E F



Computing the Tree of Shapes

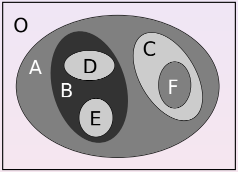
sort : O A B C D E F

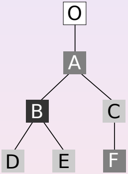




Computing the Tree of Shapes

sort : O A B C D E F





done!   (done? no, we first have to sort...)

Intermediate Conclusion

The Need

we need a <u>discrete</u> image representation...

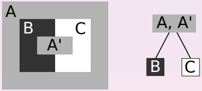
...that has some appropriate <u>continuous</u> properties!

Intermediate Conclusion

The Need

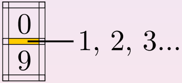
we need a <u>discrete</u> image representation...
...that has some appropriate <u>continuous</u> properties!

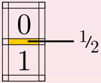
Catching two ideas:



1	1	1	1	1	1
1	0	0	2	2	1
1	0	1	1	2	1
1	0	0	2	2	1
1	1	1	1	1	1

we need to pass <u>between pixels</u>...

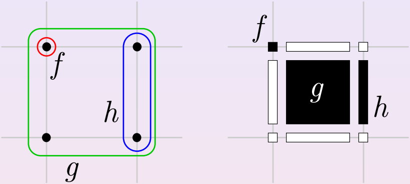


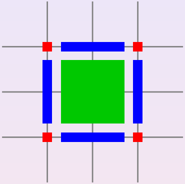


...and with many values

Outline

Cubical Complexes v. Khalimsky's Grid





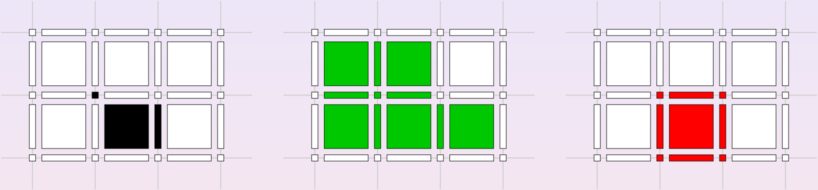
Two representations of a set of faces...

...and Khalimsky's grid.

$: -)$

WHAT IS NICE

we have some topological operators:

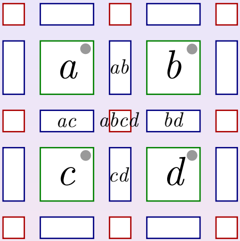


$E = \{f, g, h\}$ star: $E^{\uparrow}$ closure: $E^{\downarrow}$

... and an easy and effective structure to work on

Image (Basic) Immersion 1/3

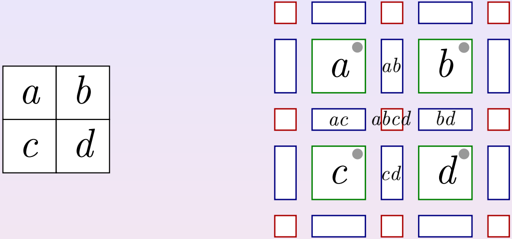
a	b
c	d



Here, where Op is an operator over a set of values:

- we have $ab = \mathrm{Op}(\{\, a, b \,\})$, $abcd = \mathrm{Op}(\{\, a, b, c, d \,\})$, etc.
- a discrete function u on domain $\mathcal{D}$ becomes $u_{\mathcal{K}}^{\mathrm{Op}} = \mathcal{I}_{\mathcal{K}}^{\mathrm{Op}}(u)$ on domain $\mathcal{K}$
- and the gray dots indicate where the *primary pixel* values are assigned.

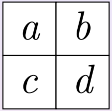
Image (Basic) Immersion 1/3

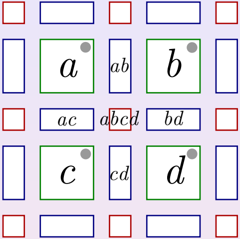


Here, where Op is an operator over a set of values:

- we have $ab = \mathrm{Op}(\{\, a, b \,\})$, $abcd = \mathrm{Op}(\{\, a, b, c, d \,\})$, etc.
- a discrete function u on domain $\mathcal{D}$ becomes $u_{\mathcal{K}}^{\mathrm{Op}} = \mathcal{I}_{\mathcal{K}}^{\mathrm{Op}}(u)$ on domain $\mathcal{K}$
- and the gray dots indicate where the *primary pixel* values are assigned.

Image (Basic) Immersion 1/3





Here, where Op is an operator over a set of values:

- we have $ab = \text{Op}(\{a, b\})$, $abcd = \text{Op}(\{a, b, c, d\})$, etc.
- a discrete function u on domain $\mathcal{D}$ becomes $u_{\mathcal{K}}^{\text{Op}} = \mathcal{I}_{\mathcal{K}}^{\text{Op}}(u)$ on domain $\mathcal{K}$
- and the gray dots indicate where the *primary pixel* values are assigned.

Image (Basic) Immersion 2/3

with Op = max and $\lambda = 3$:

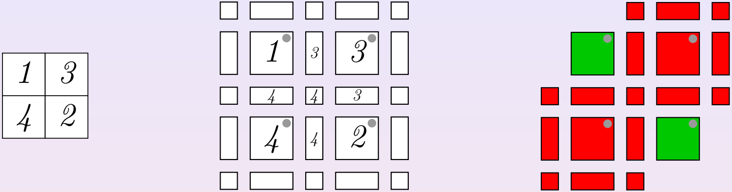
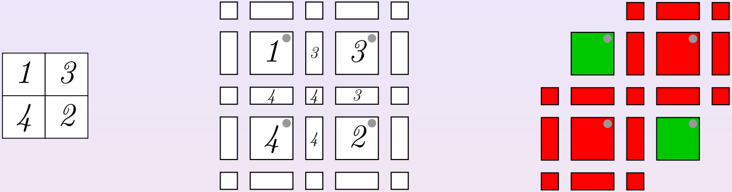


Image (Basic) Immersion 2/3

with Op = max and $\lambda = 3$:

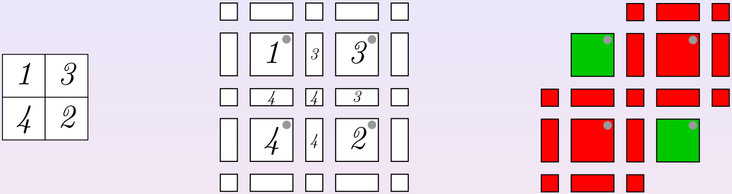


we have:

- any $\Gamma \in \mathcal{CC}([\, u_{\mathcal{K}}^{\max} \geq \lambda \,])$ is a closed set
- $\Gamma \cap \mathcal{D} \in \mathcal{CC}_{c8}([\, u \geq \lambda \,])$ $\quad \rightsquigarrow \quad \mathcal{T}_{\geq}^{\mathcal{K}}(u_{\mathcal{K}}^{\max})\,|_{\mathcal{D}} = \mathcal{T}_{\geq_{c8}}^{\mathcal{D}}(u)$

Image (Basic) Immersion 2/3

with $\text{Op} = \max$ and $\lambda = 3$:



we have:

- any $\Gamma \in \mathcal{CC}([\, u^{\max}_{\mathcal{K}} \geq \lambda \,])$ is a closed set
- $\Gamma \cap \mathcal{D} \in \mathcal{CC}_{c8}([\, u \geq \lambda \,])$ $\quad \rightsquigarrow \quad \mathcal{T}^{\mathcal{K}}_{\geq}(u^{\max}_{\mathcal{K}})\,|_{\mathcal{D}} = \mathcal{T}^{\mathcal{D}}_{\geq_{c8}}(u)$

and

- any $\Gamma' \in \mathcal{CC}([\, u^{\max}_{\mathcal{K}} < \lambda \,])$ is an open set
- $\Gamma' \cap \mathcal{D} \in \mathcal{CC}_{c4}([\, u < \lambda \,])$ $\quad \rightsquigarrow \quad \mathcal{T}^{\mathcal{K}}_{<}(u^{\max}_{\mathcal{K}})\,|_{\mathcal{D}} = \mathcal{T}^{\mathcal{D}}_{<_{c4}}(u)$

Image (Basic) Immersion 3/3

we have:

- the set of (upper and lower) cuts $\mathcal{T}_{\geq}^{\mathcal{K}}(u_{\mathcal{K}}^{\max}) \cup \mathcal{T}_{<}^{\mathcal{K}}(u_{\mathcal{K}}^{\max})$ gives a tree of shapes
- whose restriction over $\mathcal{D}$ is “the” state-of-the-art tree of shapes:

$$\mathcal{T}_{(\geq/<)}^{\mathcal{K}}(u_{\mathcal{K}}^{\max})\,|_{\mathcal{D}} \;=\; \mathcal{T}_{(\geq_{c8}/<_{c4})}^{\mathcal{D}}(u).$$

Image (Basic) Immersion 3/3

we have:

- the set of (upper and lower) cuts $\mathcal{T}_{\geq}^{\mathcal{K}}(u_{\mathcal{K}}^{\max}) \cup \mathcal{T}_{<}^{\mathcal{K}}(u_{\mathcal{K}}^{\max})$ gives a tree of shapes
- whose restriction over $\mathcal{D}$ is "the" state-of-the-art tree of shapes:

$$\mathcal{T}_{(\geq/<)}^{\mathcal{K}}(u_{\mathcal{K}}^{\max})\,|_{\mathcal{D}} \;=\; \mathcal{T}_{(\geq_{c8}/<_{c4})}^{\mathcal{D}}(u).$$

what's nice:

- $u_{\mathcal{K}}^{\max}$ is an u.s.c. function so cuts over $\mathcal{D}$ rely on the duality $\geq_{c8}$ / $<_{c4}$
- there is no topological problem
- over $\mathcal{K}$ upper and lower cuts have the same connectivity ($c4$)

IMAGE (BASIC) IMMERSION 3/3

we have:

- the set of (upper and lower) cuts $\mathcal{T}_{\geq}^{\mathcal{K}}(u_{\mathcal{K}}^{\max}) \cup \mathcal{T}_{<}^{\mathcal{K}}(u_{\mathcal{K}}^{\max})$ gives a tree of shapes
- whose restriction over $\mathcal{D}$ is "the" state-of-the-art tree of shapes:

$$\mathcal{T}_{(\geq/<)}^{\mathcal{K}}(u_{\mathcal{K}}^{\max})\,|_{\mathcal{D}} \;=\; \mathcal{T}_{(\geq_{c8}/<_{c4})}^{\mathcal{D}}(u).$$

what's nice:

- $u_{\mathcal{K}}^{\max}$ is an u.s.c. function so cuts over $\mathcal{D}$ rely on the duality $\geq_{c8}$ / $<_{c4}$
- there is no topological problem
- over $\mathcal{K}$ upper and lower cuts have the same connectivity ($c4$)

<u>yet we cannot compute the ToS with that BASIC immersion...</u>

SELF-DUALITY FLAW AND ABNORMALITIES (1/3)

the tree of shapes is <u>not purely self-dual</u>:

$$\mathcal{T}^{\mathcal{D}}_{(\geq_{c8}/<_{c4})}(u) \;=\; \complement\, \mathcal{T}^{\mathcal{D}}_{(>_{c4}/\leq_{c8})}(\complement\, u)$$

Self-Duality Flaw and Abnormalities (1/3)

the tree of shapes is <u>not purely self-dual</u>:

$$\mathcal{T}^{\mathcal{D}}_{(\geq_{c8}/<_{c4})}(u) \;=\; \complement\, \mathcal{T}^{\mathcal{D}}_{(>_{c4}/\leq_{c8})}(\complement\, u)$$

that starts with two immersions that are not self-dual:

$$\complement\, \mathcal{I}^{\max}_{\mathcal{K}}(u) \;=\; \mathcal{I}^{\min}_{\mathcal{K}}(\complement\, u)$$

SELF-DUALITY FLAW AND ABNORMALITIES (1/3)

the tree of shapes is not purely self-dual:

$$\mathcal{T}^{\mathcal{D}}_{(\geq_{c8}/<_{c4})}(u) \;=\; \complement\, \mathcal{T}^{\mathcal{D}}_{(>_{c4}/\leq_{c8})}(\complement\, u)$$

that starts with two immersions that are not self-dual:

$$\complement\, \mathcal{I}^{\max}_{\mathcal{K}}(u) \;=\; \mathcal{I}^{\min}_{\mathcal{K}}(\complement\, u)$$

$\leadsto$ definitely such immersions are not so good image representations...

Self-Duality Flaw and Abnormalities (2/3)

consider these examples:

1	*0*
0	*1*

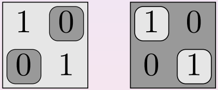
ToS?

1	1	1	1	1	1	1	1	1
1	0	0	0	1	2	2	2	1
1	0	1	0	1	2	1	2	1
1	0	0	1	1	1	2	2	1
1	1	1	1	1	1	1	1	1

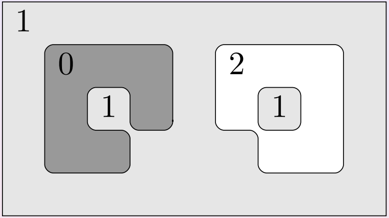
ToS?

SELF-DUALITY FLAW AND ABNORMALITIES (2/3)

consider these examples:



two possible trees!



a <u>non</u> symmetrical tree!

Self-Duality Flaw and Abnormalities (3/3)

furthermore, from a topological point of view:

- some shapes are closed, the other ones are open...
- some shapes contain their level lines, the other ones do not...
- there is an arbitrary choice between $(\geq_{c8} \,/\, <_{c4})$ and $(>_{c4} \,/\, \leq_{c8})$...

Set-Valued Maps

a set-valued map $\text{U} : X \rightsquigarrow Y$ is characterized by its graph $\text{Gra}(\text{U})$:

$$\text{Gra}(\text{U}) \;=\; \{\, (x,y) \in X \times Y \mid y \in \text{U}(x) \,\}$$

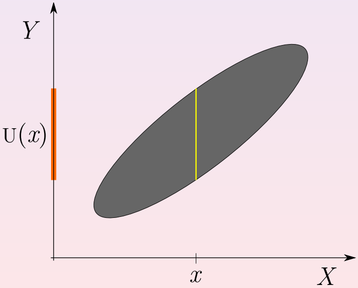
actually we have $\text{U} : X \to \mathcal{P}(Y)$

Set-Valued Maps

a set-valued map $\mathrm{U} : X \rightsquigarrow Y$ is characterized by its graph $\mathrm{Gra}(\mathrm{U})$:

$$\mathrm{Gra}(\mathrm{U}) = \{ (x, y) \in X \times Y \mid y \in \mathrm{U}(x) \}$$

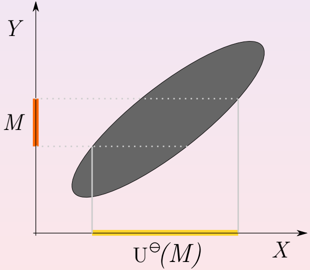
actually we have $\mathrm{U} : X \to \mathcal{P}(Y)$



INVERSE BY U OF A SUBSET M

two ways:

- the *inverse image* of $M \subset Y$ by U is
$$\mathrm{U}^{-1}(M) \;=\; \{\, x \in X \mid \mathrm{U}(x) \cap M \neq \varnothing \,\}$$
-

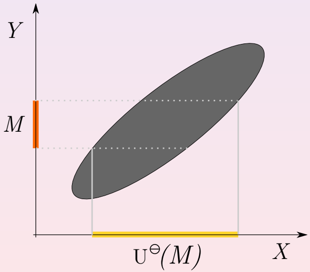


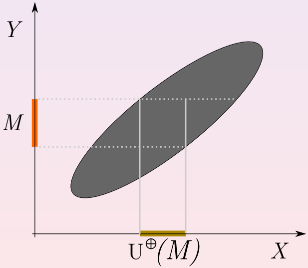
INVERSE BY U OF A SUBSET M

two ways:

- the *inverse image* of $M \subset Y$ by U is
$$\mathrm{U}^{-1}(M) \;=\; \{\, x \in X \mid \mathrm{U}(x) \cap M \neq \varnothing \,\}$$

- the *core* of $M \subset Y$ by U is
$$\mathrm{U}^{+1}(M) \;=\; \{\, x \in X \mid \mathrm{U}(x) \subset M \,\}$$





MATHS!

- we have some nice properties:

$$\text{e.g.,} \quad X \setminus \mathrm{U}^{-1}(M) = \mathrm{U}^{+1}(Y \setminus M)$$

- we have some continuity:
  - when $\mathrm{U}(x)$ is compact, U is Upper Semi-Continuous (U.S.C.) at x if

    $$\forall \varepsilon > 0,\ \exists \eta > 0 \text{ such that } \forall x' \in B_X(x, \eta),\ \mathrm{U}(x') \subset B_Y(\mathrm{U}(x), \varepsilon).$$

  - this is the "natural" extension of the continuity of a single-valued function

- some characterization of U.S.C. maps:

  U is U.S.C. if and only if the core of any open subset is open

MATHS!

- we have some nice properties:

$$\text{e.g.,}\quad X \setminus \mathrm{U}^{-1}(M) = \mathrm{U}^{+1}(Y \setminus M)$$

- we have some continuity:
  - when $\mathrm{U}(x)$ is compact, U is Upper Semi-Continuous (U.S.C.) at x if

$$\forall \varepsilon > 0,\ \exists \eta > 0 \text{ such that } \forall x' \in B_X(x,\eta),\ \mathrm{U}(x') \subset B_Y(\mathrm{U}(x),\varepsilon).$$

  - this is the "natural" extension of the continuity of a single-valued function

- some characterization of U.S.C. maps:

U is U.S.C. if and only if the core of any open subset is open

MATHS!

- we have some nice properties:

$$\text{e.g.,}\quad X \setminus \mathrm{U}^{-1}(M) = \mathrm{U}^{+1}(Y \setminus M)$$

- we have some continuity:
  - when $\mathrm{U}(x)$ is compact, U is Upper Semi-Continuous (U.S.C.) at x if

$$\forall \varepsilon > 0,\ \exists\, \eta > 0 \text{ such that } \forall\, x' \in B_X(x,\eta),\ \mathrm{U}(x') \subset B_Y(\mathrm{U}(x),\varepsilon).$$

  - this is the "natural" extension of the continuity of a single-valued function

- some characterization of U.S.C. maps:

U is U.S.C. if and only if the core of any open subset is open

MATHS!

- we have some nice properties:
  $$\text{e.g.,}\quad X \setminus \mathrm{U}^{-1}(M) = \mathrm{U}^{+1}(Y \setminus M)$$

- we have some continuity:
  - when $\mathrm{U}(x)$ is compact, U is Upper Semi-Continuous (U.S.C.) at x if
    $$\forall \varepsilon > 0,\ \exists\, \eta > 0 \text{ such that } \forall\, x' \in B_X(x,\eta),\ \mathrm{U}(x') \subset B_Y(\mathrm{U}(x),\varepsilon).$$
  - this is the "natural" extension of the continuity of a single-valued function

- some characterization of U.S.C. maps:

  U is U.S.C. if and only if the core of any open subset is open

Maths!

- we have some nice properties:

$$\text{e.g.,}\quad X \setminus \mathrm{U}^{-1}(M) = \mathrm{U}^{+1}(Y \setminus M)$$

- we have some continuity:
  - when $\mathrm{U}(x)$ is compact, U is Upper Semi-Continuous (U.S.C.) at x if

$$\forall \varepsilon > 0,\ \exists\, \eta > 0 \text{ such that } \forall\, x' \in B_X(x,\eta),\ \mathrm{U}(x') \subset B_Y(\mathrm{U}(x),\varepsilon).$$

  - this is the <u>"natural" extension of the continuity of a single-valued function</u>

- some characterization of U.S.C. maps:

U is U.S.C. if and only if <u>the core of any open subset is open</u>

...

What do we have to do

remember:

- we can compute the ToS if we can adequately sort pixels
- for that, we need
  - to pass between pixels $\rightsquigarrow$ Khalimsky's grid
  - to deal with many values between pixels $\rightsquigarrow$ set-valued maps

- morphology on functions and the ToS is based on cuts.

so we have to:

- define cuts of set-valued maps
  (note that $[\, \text{U} \overset{?}{\geq} \lambda \,]$ implies an *external* relation since $\text{U}(x) \in \mathcal{P}(Y)$ whereas $\lambda \in Y$)
- define some proper ways to represent an image on Khalimsky's grid

What do we have to do

remember:

- we can compute the ToS if we can adequately sort pixels
- for that, we need
  - to pass between pixels $\rightsquigarrow$ Khalimsky's grid
  - to deal with many values between pixels $\rightsquigarrow$ set-valued maps

- morphology on functions and the ToS is based on cuts.

so we have to:

- define cuts of set-valued maps

  (note that $[\, u \overset{?}{\geq} \lambda \,]$ implies an *external* relation since $u(x) \in \mathcal{P}(Y)$ whereas $\lambda \in Y$)

- define some proper ways to represent an image on Khalimsky's grid

What do we have to do

remember:

- we can compute the ToS if we can adequately sort pixels
- for that, we need
  - to pass between pixels $\rightsquigarrow$ Khalimsky's grid
  - to deal with many values between pixels $\rightsquigarrow$ set-valued maps

- morphology on functions and the ToS is based on cuts.

so we have to:

- define cuts of set-valued maps

  (note that $[\, \text{U} \overset{?}{\geq} \lambda \,]$ implies an *external* relation since $\text{U}(x) \in \mathcal{P}(Y)$ whereas $\lambda \in Y$)

- define some proper ways to represent an image on Khalimsky's grid

What do we have to do

remember:

- we can compute the ToS if we can adequately sort pixels
- for that, we need
  - to pass between pixels $\rightsquigarrow$ Khalimsky's grid
  - to deal with many values between pixels $\rightsquigarrow$ set-valued maps

- morphology on functions and the ToS is based on cuts.

so we have to:

- define cuts of set-valued maps
  (note that $[\,\mathrm{U} \overset{?}{\geq} \lambda\,]$ implies an *external* relation since $\mathrm{U}(x) \in \mathcal{P}(Y)$ whereas $\lambda \in Y$)
- define some proper ways to represent an image on Khalimsky's grid

What do we have to do

remember:

- we can compute the ToS if we can adequately sort pixels
- for that, we need
  - to pass between pixels $\rightsquigarrow$ Khalimsky's grid
  - to deal with many values between pixels $\rightsquigarrow$ set-valued maps

- morphology on functions and the ToS is based on cuts.

so we have to:

- define cuts of set-valued maps
  (note that $[\, u \overset{?}{\geq} \lambda \,]$ implies an *external* relation since $u(x) \in \mathcal{P}(Y)$ whereas $\lambda \in Y$)
- define some proper ways to represent an image on Khalimsky's grid

What do we have to do

remember:

- we can compute the ToS if we can adequately sort pixels
- for that, we need
  - to pass between pixels $\rightsquigarrow$ Khalimsky's grid
  - to deal with many values between pixels $\rightsquigarrow$ set-valued maps

- morphology on functions and the ToS is based on cuts.

so we have to:

- define cuts of set-valued maps

  (note that $[\, \mathrm{U} \overset{?}{\geq} \lambda \,]$ implies an *external* relation since $\mathrm{U}(x) \in \mathcal{P}(Y)$ whereas $\lambda \in Y$)
- define some proper ways to represent an image on Khalimsky's grid

What do we have to do

remember:

- we can compute the ToS if we can adequately sort pixels
- for that, we need
  - to pass between pixels $\rightsquigarrow$ Khalimsky's grid
  - to deal with many values between pixels $\rightsquigarrow$ set-valued maps

- morphology on functions and the ToS is based on cuts.

so we have to:

- define cuts of set-valued maps

  (note that $[\, \mathrm{U} \overset{?}{\geq} \lambda \,]$ implies an *external* relation since $\mathrm{U}(x) \in \mathcal{P}(Y)$ whereas $\lambda \in Y$)
- define some proper ways to represent an image on Khalimsky's grid

OUTLINE

Cuts of Set-Valued Maps NEW!

definition of large cuts:

$$
\begin{aligned}
[\mathrm{U} \trianglelefteq \lambda] &= \{\, x \in X \mid \exists \mu \in \mathrm{U}(x),\ \mu \leq \lambda \,\} \\
[\mathrm{U} \trianglerighteq \lambda] &= \{\, x \in X \mid \exists \mu \in \mathrm{U}(x),\ \mu \geq \lambda \,\}
\end{aligned}
$$

Cuts of Set-Valued Maps NEW!

definition of large cuts:

$$\begin{aligned}
[\mathrm{U} \trianglelefteq \lambda] &= \{\, x \in X \mid \exists \mu \in \mathrm{U}(x),\ \mu \leq \lambda \,\} \\
[\mathrm{U} \trianglerighteq \lambda] &= \{\, x \in X \mid \exists \mu \in \mathrm{U}(x),\ \mu \geq \lambda \,\}
\end{aligned}$$

by extension we define:

$$\begin{aligned}
[\mathrm{U} \triangleleft \lambda] &= X \setminus [\mathrm{U} \trianglerighteq \lambda] \\
[\mathrm{U} \triangleright \lambda] &= X \setminus [\mathrm{U} \trianglelefteq \lambda] \\
[\mathrm{U} \,\square\, \lambda] &= [\mathrm{U} \trianglelefteq \lambda] \cap [\mathrm{U} \trianglerighteq \lambda] \\
[\mathrm{U} \boxslash \lambda] &= X \setminus [\mathrm{U} \,\square\, \lambda]
\end{aligned}$$

Cuts of Set-Valued Maps NEW!

definition of large cuts:

$$
\begin{aligned}
[\mathrm{U} \trianglelefteq \lambda] &= \{ x \in X \mid \exists \mu \in \mathrm{U}(x),\ \mu \leq \lambda \} \\
[\mathrm{U} \trianglerighteq \lambda] &= \{ x \in X \mid \exists \mu \in \mathrm{U}(x),\ \mu \geq \lambda \}
\end{aligned}
$$

by extension we define:

$$
\begin{aligned}
[\mathrm{U} \triangleleft \lambda] &= X \setminus [\mathrm{U} \trianglerighteq \lambda] \\
[\mathrm{U} \triangleright \lambda] &= X \setminus [\mathrm{U} \trianglelefteq \lambda] \\
[\mathrm{U} \,\square\, \lambda] &= [\mathrm{U} \trianglelefteq \lambda] \cap [\mathrm{U} \trianglerighteq \lambda] \\
[\mathrm{U} \boxslash \lambda] &= X \setminus [\mathrm{U} \,\square\, \lambda]
\end{aligned}
$$

so we have:

$$
\begin{aligned}
[\mathrm{U} \triangleleft \lambda] &= \{ x \in X \mid \forall \mu \in \mathrm{U}(x),\ \mu < \lambda \} \\
[\mathrm{U} \triangleright \lambda] &= \{ x \in X \mid \forall \mu \in \mathrm{U}(x),\ \mu > \lambda \} \\
[\mathrm{U} \,\square\, \lambda] &= \{ x \in X \mid \lambda \in \mathrm{U}(x) \} \\
[\mathrm{U} \boxslash \lambda] &= \{ x \in X \mid \lambda \notin \mathrm{U}(x) \}
\end{aligned}
$$

Cuts Properties

we have some inclusions:

$$\lambda_1 < \lambda_2 \quad \Rightarrow \quad [\mathrm{U} \trianglelefteq \lambda_1] \subseteq [\mathrm{U} \trianglelefteq \lambda_2] \qquad \longrightarrow \text{ min-tree } \mathcal{T}_{\trianglelefteq}$$

$$\lambda_1 < \lambda_2 \quad \Rightarrow \quad [\mathrm{U} \trianglerighteq \lambda_2] \subseteq [\mathrm{U} \trianglerighteq \lambda_1] \qquad \longrightarrow \text{ max-tree } \mathcal{T}_{\trianglerighteq}$$

Cuts Properties

we have some inclusions:

$$\lambda_1 < \lambda_2 \quad \Rightarrow \quad [\mathrm{U} \trianglelefteq \lambda_1] \subseteq [\mathrm{U} \trianglelefteq \lambda_2] \qquad \longrightarrow \text{ min-tree } \mathcal{T}_{\trianglelefteq}$$

$$\lambda_1 < \lambda_2 \quad \Rightarrow \quad [\mathrm{U} \trianglerighteq \lambda_2] \subseteq [\mathrm{U} \trianglerighteq \lambda_1] \qquad \longrightarrow \text{ max-tree } \mathcal{T}_{\trianglerighteq}$$

some separations with strict cuts:

$$\lambda_1 \leq \lambda_2 \quad \Rightarrow \quad [\mathrm{U} \triangleleft \lambda_1] \cap [\mathrm{U} \triangleright \lambda_2] = \varnothing$$

Cuts Properties

we have some inclusions:

$$\lambda_1 < \lambda_2 \quad \Rightarrow \quad [\mathrm{U} \trianglelefteq \lambda_1] \subseteq [\mathrm{U} \trianglelefteq \lambda_2] \qquad \longrightarrow \text{ min-tree } \mathcal{T}_{\trianglelefteq}$$

$$\lambda_1 < \lambda_2 \quad \Rightarrow \quad [\mathrm{U} \trianglerighteq \lambda_2] \subseteq [\mathrm{U} \trianglerighteq \lambda_1] \qquad \longrightarrow \text{ max-tree } \mathcal{T}_{\trianglerighteq}$$

some separations with strict cuts:

$$\lambda_1 \leq \lambda_2 \quad \Rightarrow \quad [\mathrm{U} \triangleleft \lambda_1] \cap [\mathrm{U} \triangleright \lambda_2] = \varnothing$$

!!! but also an *oddity* with large cuts:

$$\lambda_1 < \lambda_2 \quad \not\Rightarrow \quad [\mathrm{U} \trianglelefteq \lambda_1] \cap [\mathrm{U} \trianglerighteq \lambda_2] = \varnothing$$

Cuts Properties

we have some inclusions:

$$\lambda_1 < \lambda_2 \quad \Rightarrow \quad [\,\mathrm{U} \trianglelefteq \lambda_1\,] \subseteq [\,\mathrm{U} \trianglelefteq \lambda_2\,] \qquad \longrightarrow \text{ min-tree } \mathcal{T}_{\trianglelefteq}$$

$$\lambda_1 < \lambda_2 \quad \Rightarrow \quad [\,\mathrm{U} \trianglerighteq \lambda_2\,] \subseteq [\,\mathrm{U} \trianglerighteq \lambda_1\,] \qquad \longrightarrow \text{ max-tree } \mathcal{T}_{\trianglerighteq}$$

some separations with strict cuts:

$$\lambda_1 \leq \lambda_2 \quad \Rightarrow \quad [\,\mathrm{U} \triangleleft \lambda_1\,] \cap [\,\mathrm{U} \triangleright \lambda_2\,] = \varnothing$$

!!! but also an *oddity* with large cuts:

$$\lambda_1 < \lambda_2 \quad \not\Rightarrow \quad [\,\mathrm{U} \trianglelefteq \lambda_1\,] \cap [\,\mathrm{U} \trianglerighteq \lambda_2\,] = \varnothing$$

e.g., with $\mathrm{U}(x) = [\![1,2]\!]$, we have both $x \in [\,\mathrm{U} \trianglelefteq 1\,]$ and $x \in [\,\mathrm{U} \trianglerighteq 2\,]$...

Propositions (1/3)

given an image $u : \mathbb{Z}^2 \to \mathbb{Z}$, we want to define $\mathrm{U}_{\mathcal{K}} : X \rightsquigarrow Y$

PROPOSITIONS (1/3)

given an image $u : \mathbb{Z}^2 \to \mathbb{Z}$, we want to define $\mathrm{U}_{\mathcal{K}} : X \rightsquigarrow Y$

- we want $\mathrm{U}_{\mathcal{K}}$ to be reconstructible from its component tree

  $\Rightarrow$ values of U have to be <u>intervals</u>

PROPOSITIONS (1/3)

given an image $u : \mathbb{Z}^2 \to \mathbb{Z}$, we want to define $\mathrm{U}_{\mathcal{K}} : X \rightsquigarrow Y$

- we want $\mathrm{U}_{\mathcal{K}}$ to be reconstructible from its component tree
  $\Rightarrow$ values of U have to be <u>intervals</u>
- we want $\mathrm{U}_{\mathcal{K}}$ to be continuous à-la U.S.C.
  $\Leftrightarrow$ values on 0-faces and 1-faces are the <u>span</u>
  of their resp. 1-faces and 2-faces neighbors values

Propositions (1/3)

given an image $u : \mathbb{Z}^2 \to \mathbb{Z}$, we want to define $\mathrm{U}_{\mathcal{K}} : X \rightsquigarrow Y$

- we want $\mathrm{U}_{\mathcal{K}}$ to be reconstructible from its component tree
  $\Rightarrow$ values of U have to be intervals
- we want $\mathrm{U}_{\mathcal{K}}$ to be continuous à-la U.S.C.
  $\Leftrightarrow$ values on 0-faces and 1-faces are the span
  of their resp. 1-faces and 2-faces neighbors values
- we want shapes of $\mathrm{U}_{\mathcal{K}}$ to get a chance to form a ToS
  $\Rightarrow$ shapes are obtained with strict cuts only

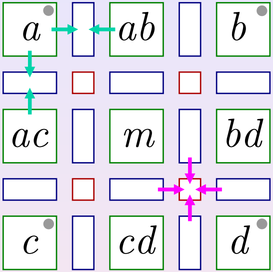
Propositions (1/3)

given an image $u : \mathbb{Z}^2 \to \mathbb{Z}$, we want to define $\mathrm{U}_{\mathcal{K}} : X \rightsquigarrow Y$

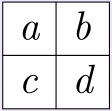
- we want $\mathrm{U}_{\mathcal{K}}$ to be reconstructible from its component tree
  $\Rightarrow$ values of U have to be <u>intervals</u>
- we want $\mathrm{U}_{\mathcal{K}}$ to be continuous à-la U.S.C.
  $\Leftrightarrow$ values on 0-faces and 1-faces are the <u>span</u>
  of their resp. 1-faces and 2-faces neighbors values
- we want shapes of $\mathrm{U}_{\mathcal{K}}$ to get a chance to form a ToS
  $\Rightarrow$ shapes are obtained with <u>strict cuts</u> only
- we want $\mathrm{U}_{\mathcal{K}}$ to preserve extrema of u
  $\Rightarrow$ values on non-primary 2-faces are *intermediate* values

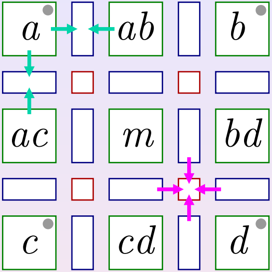
Proposed Image Immersion

a	b
c	d



Proposed Image Immersion



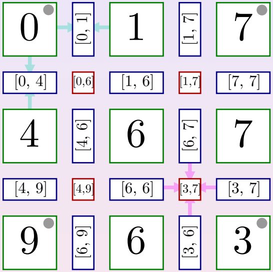


with

- all denoted values being degenerated,
- on the border non-primary 2-faces (with “two-letter” values): $\min(a, b) \leq ab \leq \max(a, b)$, ...
- on the center non-primary 2-face:
$$\max(\,\min(ab, cd), \min(ac, bd)\,) \leq m \leq \min(\,\max(ab, cd), \max(ac, bd)\,)$$
- on 1-faces: the span of 2-faces neighbors (turquoise arrows)
- on 0-faces: the span of 1-faces neighbors (pink arrows)

AN EXAMPLE

0	7
9	3



from u to a correct $\mathrm{U}_{\mathcal{K}}$

WITH AN OPERATOR

- we can rely again on an operator, Op, to construct $\mathrm{U}_{\mathcal{K}}^{\mathrm{Op}}$:
  - $ab = \mathrm{Op}(\{\, a, b \,\})$
  - $ac = \mathrm{Op}(\{\, a, c \,\})$
  - ...
  - $m = \mathrm{Op}(\{\, a, b, c, d \,\})$
- except that it *now* operates on 2-faces
- 0- and 1-faces are *now* here to ensure continuity

With an Operator

- we can rely again on an operator, Op, to construct $\mathrm{U}_{\mathcal{K}}^{\mathrm{Op}}$:
    - $ab \;=\; \mathrm{Op}(\{\, a,\, b\,\})$
    - $ac \;=\; \mathrm{Op}(\{\, a,\, c\,\})$
    - ...
    - $m \;=\; \mathrm{Op}(\{\, a,\, b,\, c,\, d\,\})$
- except that it *now* operates on 2-faces
- 0- and 1-faces are *now* here to ensure continuity

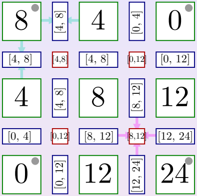
about common operators:

- namely they are min, mean, median, max
- the mean operator is commonly used for subdivision / subsampling...

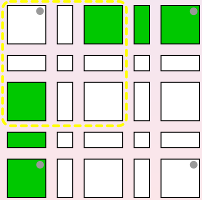
ARITHMETICAL MEAN

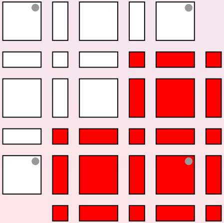
from u to $\mathrm{U}_{\mathcal{K}}^{\text{mean}}$:

8	0
0	24



sample cuts: $[\mathrm{U}_{\mathcal{K}}^{\text{mean}} \lhd 5]$ $[\mathrm{U}_{\mathcal{K}}^{\text{mean}} \unrhd 10]$





Propositions (2/3)

- strict cuts verify $[\mathrm{u}_{\mathcal{K}} \triangleright \lambda] = \mathrm{u}_{\mathcal{K}}^{+1}(\,]\lambda, +\infty]\,)$ and $[\mathrm{u}_{\mathcal{K}} \triangleleft \lambda] = \mathrm{u}_{\mathcal{K}}^{+1}(\,[-\infty, \lambda[\,)$ and they are open sets

PROPOSITIONS (2/3)

- strict cuts verify $[\,\mathrm{U}_{\mathcal{K}} \rhd \lambda\,] = \mathrm{U}_{\mathcal{K}}^{+1}(\,]\lambda, +\infty]\,)$ and $[\,\mathrm{U}_{\mathcal{K}} \lhd \lambda\,] = \mathrm{U}_{\mathcal{K}}^{+1}(\,[-\infty, \lambda[\,)$ and they are open sets
- large cuts verify $[\,\mathrm{U}_{\mathcal{K}} \unlhd \lambda\,] = \mathrm{U}_{\mathcal{K}}^{-1}(\,[-\infty, \lambda]\,)$ and $[\,\mathrm{U}_{\mathcal{K}} \unrhd \lambda\,] = \mathrm{U}_{\mathcal{K}}^{-1}(\,[\lambda, +\infty]\,)$ and they are closed sets

Propositions (2/3)

- strict cuts verify $[\mathrm{u}_{\mathcal{K}} \triangleright \lambda] = \mathrm{u}_{\mathcal{K}}^{+1}(\,]\lambda, +\infty]\,)$ and $[\mathrm{u}_{\mathcal{K}} \triangleleft \lambda] = \mathrm{u}_{\mathcal{K}}^{+1}(\,[-\infty, \lambda[\,)$ and they are open sets

- large cuts verify $[\mathrm{u}_{\mathcal{K}} \trianglelefteq \lambda] = \mathrm{u}_{\mathcal{K}}^{-1}(\,[-\infty, \lambda]\,)$ and $[\mathrm{u}_{\mathcal{K}} \trianglerighteq \lambda] = \mathrm{u}_{\mathcal{K}}^{-1}(\,[\lambda, +\infty]\,)$ and they are closed sets

- the set of components

$$\mathcal{S}(\mathrm{u}_{\mathcal{K}}) = \{\, \mathrm{Sat}(\Gamma),\ \Gamma \in \mathcal{T}_{\triangleleft}^{\mathcal{K}}(\mathrm{u}_{\mathcal{K}}) \cup \mathcal{T}_{\triangleright}^{\mathcal{K}}(\mathrm{u}_{\mathcal{K}}) \,\}$$

forms a lattice w.r.t. component inclusion

so *a priori* $\mathcal{S}(\mathrm{u}_{\mathcal{K}})$ does not form a tree...

Propositions (2/3)

- strict cuts verify $[\,\mathrm{U}_\mathcal{K} \rhd \lambda\,] = \mathrm{U}_\mathcal{K}^{+1}(\,]\lambda, +\infty]\,)$ and $[\,\mathrm{U}_\mathcal{K} \lhd \lambda\,] = \mathrm{U}_\mathcal{K}^{+1}(\,[-\infty, \lambda[\,)$ and they are open sets

- large cuts verify $[\,\mathrm{U}_\mathcal{K} \unlhd \lambda\,] = \mathrm{U}_\mathcal{K}^{-1}(\,[-\infty, \lambda]\,)$ and $[\,\mathrm{U}_\mathcal{K} \unrhd \lambda\,] = \mathrm{U}_\mathcal{K}^{-1}(\,[\lambda, +\infty]\,)$ and they are closed sets

- the set of components

$$\mathcal{S}(\mathrm{U}_\mathcal{K}) = \{\, \mathrm{Sat}(\Gamma),\ \Gamma \in \mathcal{T}_\lhd^\mathcal{K}(\mathrm{U}_\mathcal{K}) \cup \mathcal{T}_\rhd^\mathcal{K}(\mathrm{U}_\mathcal{K}) \,\}$$

forms a lattice w.r.t. component inclusion

so *a priori* $\mathcal{S}(\mathrm{U}_\mathcal{K})$ does not form a tree...

- we have the classical couple of trees of (quasi-self-dual) shapes

$$\mathcal{S}(\mathrm{U}_\mathcal{K}^{\max})|_\mathcal{D} = \mathcal{T}^\mathcal{D}_{(\geq_{c8}/<_{c4})}(u) \quad \text{and} \quad \mathcal{S}(\mathrm{U}_\mathcal{K}^{\min})|_\mathcal{D} = \mathcal{T}^\mathcal{D}_{(>_{c4}/\leq_{c8})}(u)$$

Propositions (3/3)

- with the median operator*, we have

$$\complement\ \mathcal{I}_{\mathcal{K}}^{\mathrm{med}}(u) \;=\; \mathcal{I}_{\mathcal{K}}^{\mathrm{med}}(\complement\ u)$$

* the one with $\mathrm{med}(\{a,b\}) = (a+b)/2$

PROPOSITIONS (3/3)

- with the median operator*, we have

$$\complement\ \mathcal{I}_{\mathcal{K}}^{\text{med}}(u) \;=\; \mathcal{I}_{\mathcal{K}}^{\text{med}}(\complement\ u)$$

  * the one with $\text{med}(\{a, b\}) = (a + b)/2$

- $\mathcal{S}(\text{U}_{\mathcal{K}}^{\text{median}})$ forms a tree of shapes

Propositions (3/3)

- with the median operator*, we have

$$\complement\ \mathcal{I}_{\mathcal{K}}^{\text{med}}(u) \;=\; \mathcal{I}_{\mathcal{K}}^{\text{med}}(\complement\ u)$$

  * the one with $\text{med}(\{a,b\}) = (a+b)/2$

- $\mathcal{S}(\text{U}_{\mathcal{K}}^{\text{median}})$ forms a tree of shapes

- the <u>only</u> operator to get a pure self-dual ToS is the <u>median</u>

Propositions (3/3)

- with the median operator*, we have

$$\complement\ \mathcal{I}_{\mathcal{K}}^{\text{med}}(u) = \mathcal{I}_{\mathcal{K}}^{\text{med}}(\complement\ u)$$

  * the one with $\text{med}(\{a, b\}) = (a + b)/2$

- $\mathcal{S}(\text{U}_{\mathcal{K}}^{\text{median}})$ forms a tree of shapes

- the <u>only</u> operator to get a pure self-dual ToS is the <u>median</u>

- we can compute the three ToS with quasi-linear time complexity

Propositions (3/3)

- with the median operator*, we have

$$\complement\ \mathcal{I}_{\mathcal{K}}^{\text{med}}(u) \;=\; \mathcal{I}_{\mathcal{K}}^{\text{med}}(\complement\ u)$$

* the one with $\text{med}(\{a, b\}) = (a + b)/2$

- $\mathcal{S}(\text{U}_{\mathcal{K}}^{\text{median}})$ forms a tree of shapes
- the <u>only</u> operator to get a pure self-dual ToS is the <u>median</u>
- we can compute the three ToS with quasi-linear time complexity

two key points:

- for any set in $\mathcal{S}(\text{U}_{\mathcal{K}}^{\text{median}})$ the saturation op. *commutes* with the closure op.

PROPOSITIONS (3/3)

- with the median operator*, we have

$$\complement\ \mathcal{I}_{\mathcal{K}}^{\text{med}}(u) = \mathcal{I}_{\mathcal{K}}^{\text{med}}(\complement\ u)$$

* the one with $\text{med}(\{a, b\}) = (a + b)/2$

- $\mathcal{S}(\text{U}_{\mathcal{K}}^{\text{median}})$ forms a tree of shapes

- the <u>only</u> operator to get a pure self-dual ToS is the <u>median</u>

- we can compute the three ToS with quasi-linear time complexity

two key points:

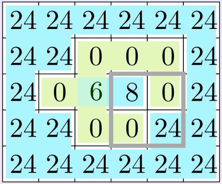
- for any set in $\mathcal{S}(\text{U}_{\mathcal{K}}^{\text{median}})$ the saturation op. *commutes* with the closure op.

- $\text{U}_{\mathcal{K}}^{\text{median}}$ is a *well-composed* image w.r.t. strict cuts

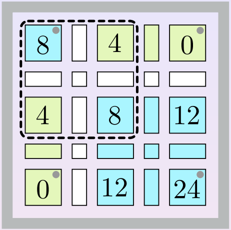
FORGET ARITHMETICAL MEAN

24	24	24	24	24	24
24	24	0	0	0	24
24	0	6	8	0	24
24	24	0	0	24	24
24	24	24	24	24	24

a sample image u



two sample cuts

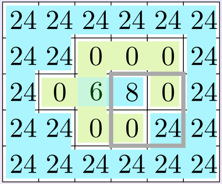


zoom

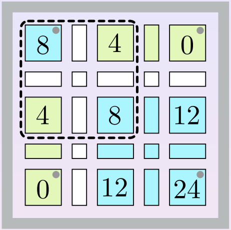
Forget Arithmetical Mean

24	24	24	24	24	24
24	24	0	0	0	24
24	0	6	8	0	24
24	24	0	0	24	24
24	24	24	24	24	24

a sample image u



two sample cuts



zoom

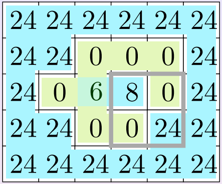
we have

- the cut $[\mathrm{U}^{\text{mean}}_{\mathcal{K}} \lhd 7]$ (light green) intersects the cut $[\mathrm{U}^{\text{mean}}_{\mathcal{K}} \rhd 5]$ at '6'
- saturation is a no-op on the "6 8" component and on the "6 & 0s" component
- the resulting shapes are neither disjoint nor included one in another!

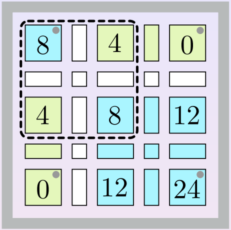
Forget Arithmetical Mean

24	24	24	24	24	24
24	24	0	0	0	24
24	0	6	8	0	24
24	24	0	0	24	24
24	24	24	24	24	24

a sample image u



two sample cuts



zoom

we have

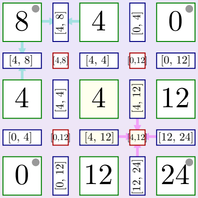
- the cut $[\mathrm{u}_{\mathcal{K}}^{\mathrm{mean}} \lhd 7]$ (light green) intersects the cut $[\mathrm{u}_{\mathcal{K}}^{\mathrm{mean}} \rhd 5]$ at '6'
- saturation is a no-op on the "6 8" component and on the "6 & 0s" component
- the resulting shapes are neither disjoint nor included one in another!

so we do <u>not</u> have a tree of shapes for $\mathrm{u}_{\mathcal{K}}^{\mathrm{mean}}$...

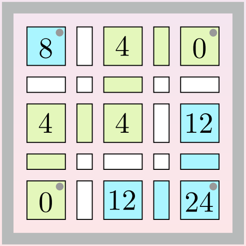
MEDIAN

from u to $\text{U}_{\mathcal{K}}^{\text{median}}$:

8	0
0	24



with sample cuts $[\text{U}_{\mathcal{K}}^{\text{median}} \lhd 7]$ (light green) and $[\text{U}_{\mathcal{K}}^{\text{median}} \rhd 5]$ (light blue):



Outline

Recap

- we have an *algorithmic scheme*
  - generic
- we have coined a *new representation* based on
  - cubical complex / Khalimsky's grid
  - multi-valued maps
  - subdivision with the *median* operator
- we have defined *cuts* for multi-valued maps
- we have proven some topological properties
  - including well-composedness

What Is Interesting

- the algorithm in itself
  - is incredibly simple
  - has a good (quasi-linear) time complexity
  - gives a tree even on a huge inclusion lattice
- the proposed image representation
  - fixes a lot of issues
  - is theoretically sexy
  - is very useful in practice

What We Did Not Talk About

- defining p_∞ for the saturation operator
- defining what is Y
- characterizing what we have in $X \times Y$
- relating Jordan's theorem to the "lattice v. tree"
- extending this work to partial orderings on Y

and also

- making a hierarchical queue deal with intervals
- reducing the space complexity of the algorithm
- parallelizing the algorithm

Related Works

about filters:

- *Connected Operator*
  P. Salembier and M. Wilkinson, IEEE Signal Processing Magazine, 2009.
- *Grain Filters*
  V. Caselles and P. Monasse, JMIV, 2002.

about algorithms:

- *Fast Computation of a Contrast Invariant Image Representation*
  P. Monasse and F. Guichard, IP, 2000.
- *A Topdown Algorithm for Computation of Level Line Trees*
  Y. Song, IP, 2007.
- *Constructing the Tree of Shapes of an Image by Fusion...*
  V. Caselles, E. Meinhardt and P. Monasse, Positivity, 2008.
- *Geometric Description of Images as Topographic Maps*
  V. Caselles and P. Monasse, LNCS Vol. 1984, Springer, 2009.

RELATED WORKS

about digital topology:

- *Regular Open or Closed Sets*
  C. Ronse, WD59, Philips, 1990.
- *Topology on Digital Images*
  L. Mazo, N. Passat, M. Couprie, and C. Ronse, JMIV, 2011.
- *Topological Equivalence between a 3D object and the Reconstruction of Its Digital Image*
  P. Stelldinger, L. Latecki, and M. Siqueira, PAMI, 2007.
- *Digitally Continuous Multivalued Functions*
  C. Escribano, A. Giraldo, and M.A. Sastre, DGCI, 2007.

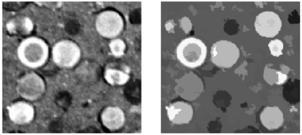
about applications:

- *Morphological Filtering in Shape Spaces: Applications using Tree-Based Image Representations*
  Y. Xu, T. Géraud, and L. Najman, ICPR, 2012.
- *Fast Object Segmentation on the Tree of Shapes using a Quasi-Local Energy Functional*
  Y. Xu, T. Géraud, and L. Najman, ICPR, 2012.

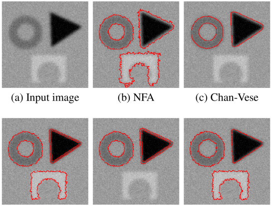
PERSPECTIVES

- working out the 3D case
- dealing with nD
- having a tree of "shapes" for color images
- transfering results towards
  - computer graphics
  - Morse theory
- exploring the many use cases of the ToS...

SOME NICE IMAGES

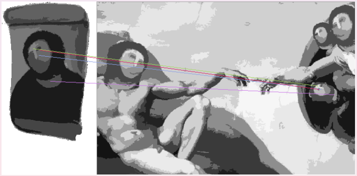


(a) Input image. (b) Shaping 1.



(a) Input image (b) NFA (c) Chan-Vese

(d) Ballester, $\lambda = 2k$ (e) Ballester, $\lambda = 3k$ (f) Our method



SOME NICE IMAGES

CONTRIBUTORS

- Laurent Najman (as a guide)
- Edwin Carlinet (for algorithm issues)
- Sébastien Crozet (for exploration and parallelization)
- Yongchao Xu (for applications)

and also

- Olena's contributors ⟶ see `http://olena.lrde.epita.fr` thanks for your attention

any questions?