Segmentation:
Watershed

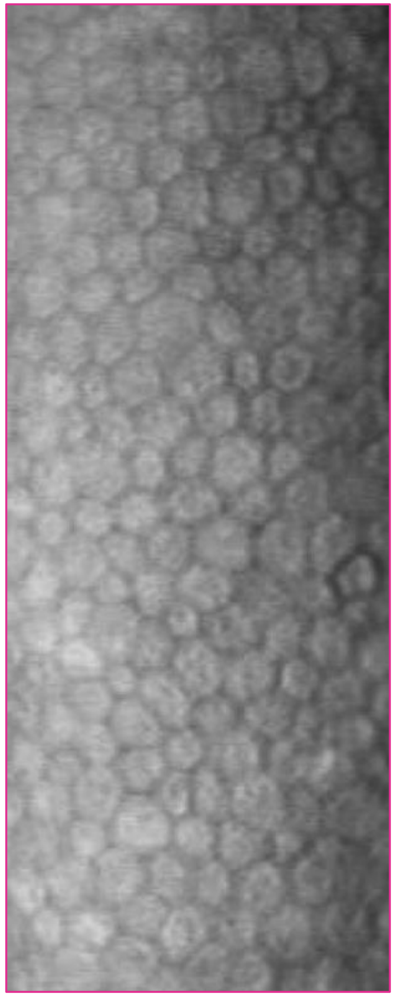
Dr. Tushar Sandhan

Introduction

- Number of segments?

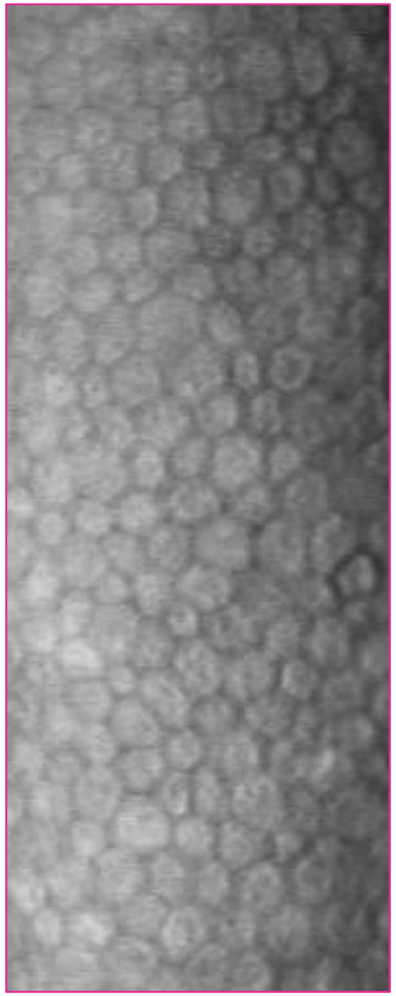
Introduction

- Number of segments?



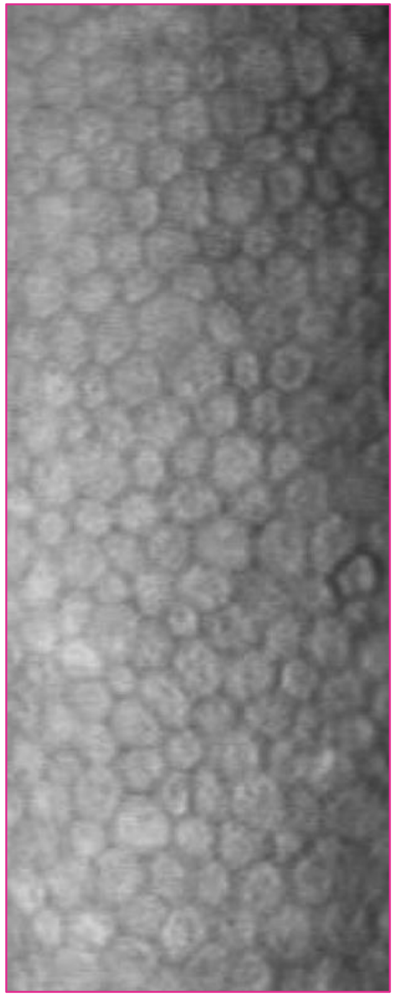
Introduction

- Number of segments?

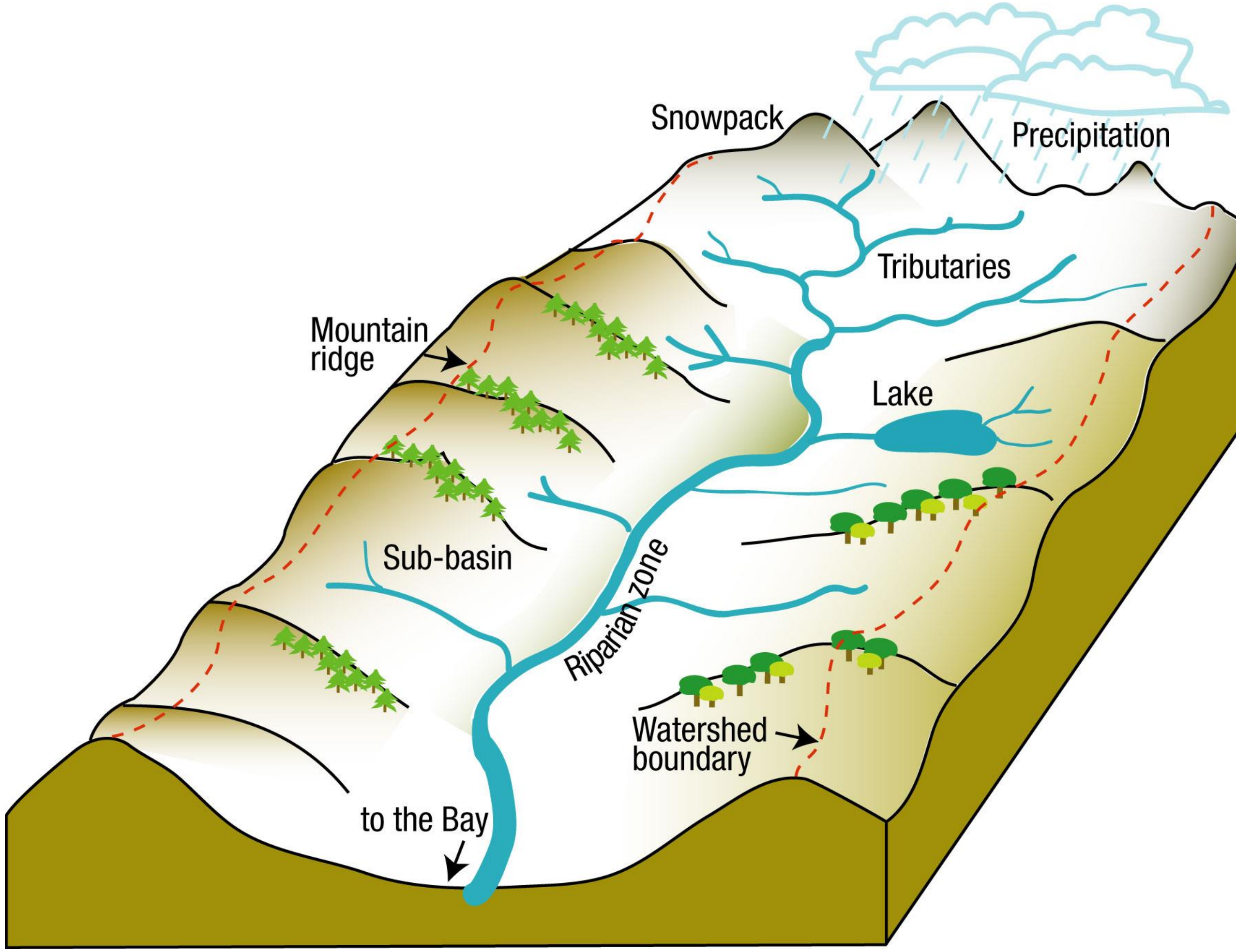


Introduction

- Number of segments?



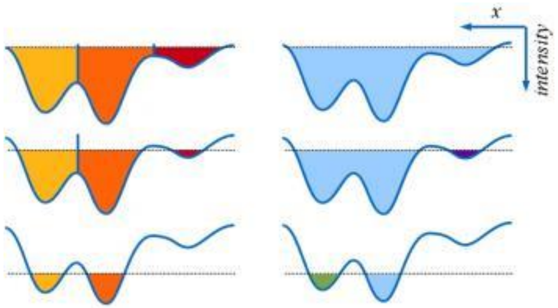
Watershed



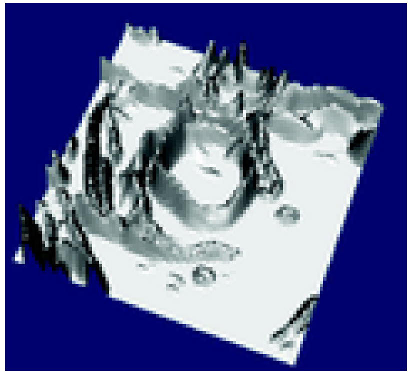
Snowpack
Precipitation
Tributaries
Mountain ridge
Lake
Sub-basin
Riparian zone
Watershed boundary
to the Bay

Watershed

- Watershed segmentation
  - considers an image as a topological surface
  - pixel intensities are heights (lowest level 0, highest peak 255)
  - segmenting overlapping objects
  - consider each waterbody as a separate object



waterbodies via watershed



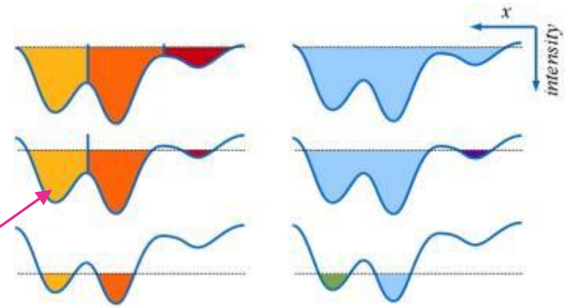
input

courtesy: wiki

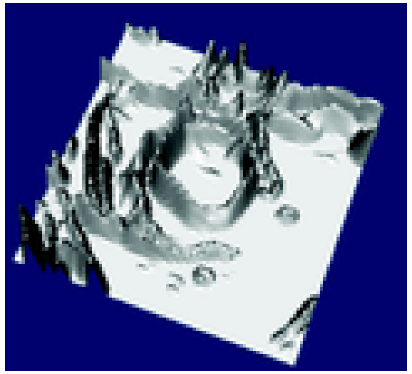
Watershed

- Watershed segmentation
  - considers an image as a topological surface
  - pixel intensities are heights (lowest level 0, highest peak 255)
  - segmenting overlapping objects
  - consider each waterbody as a separate object

catchment basins



waterbodies via watershed



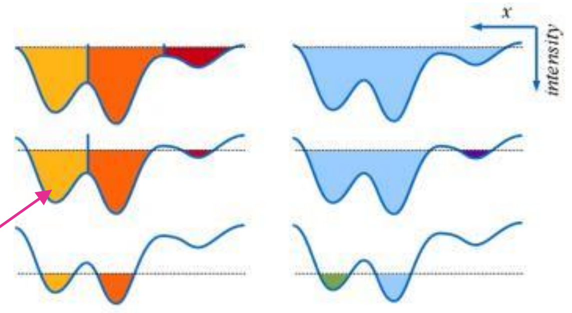
input

courtesy: wiki

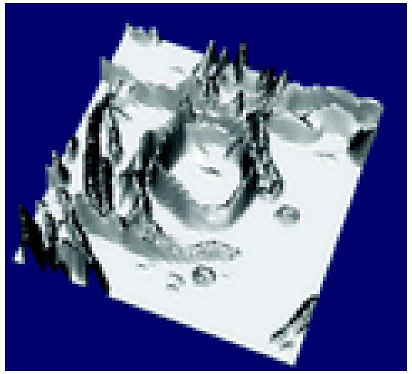
Watershed

- Watershed segmentation
  - considers an image as a topological surface
  - pixel intensities are heights (lowest level 0, highest peak 255)
  - segmenting overlapping objects
  - consider each waterbody as a separate object

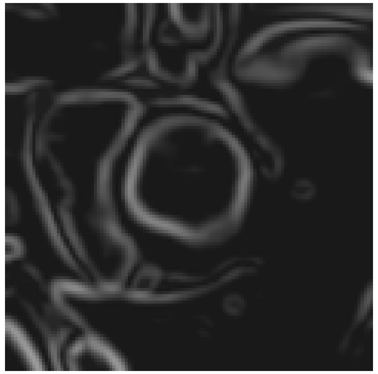
catchment basins



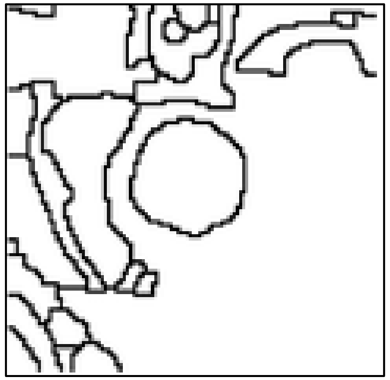
waterbodies via watershed



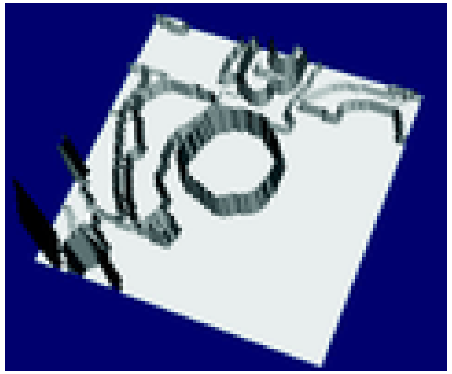
input



gradient



watershed 2D

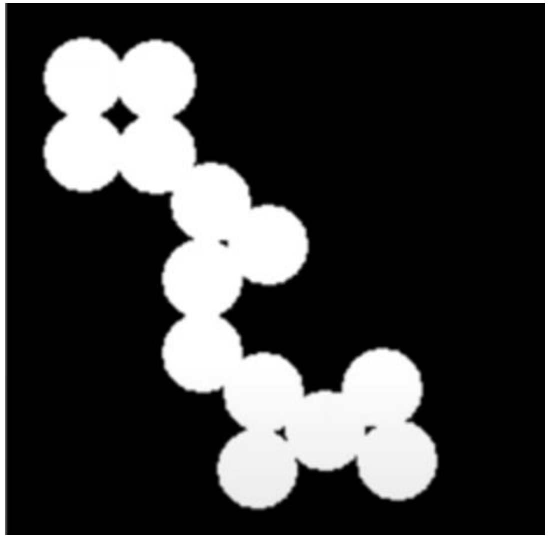


watershed 3D

courtesy: wiki

Watershed

- Watershed segmentation
  - overlapping objects



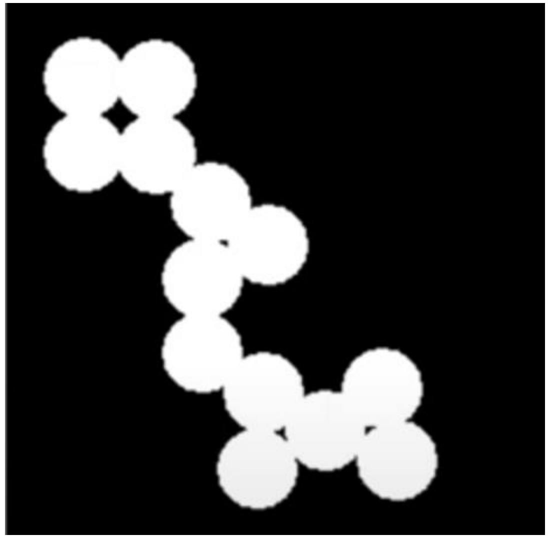
input

thresholding or clustering

courtesy: JW Tay

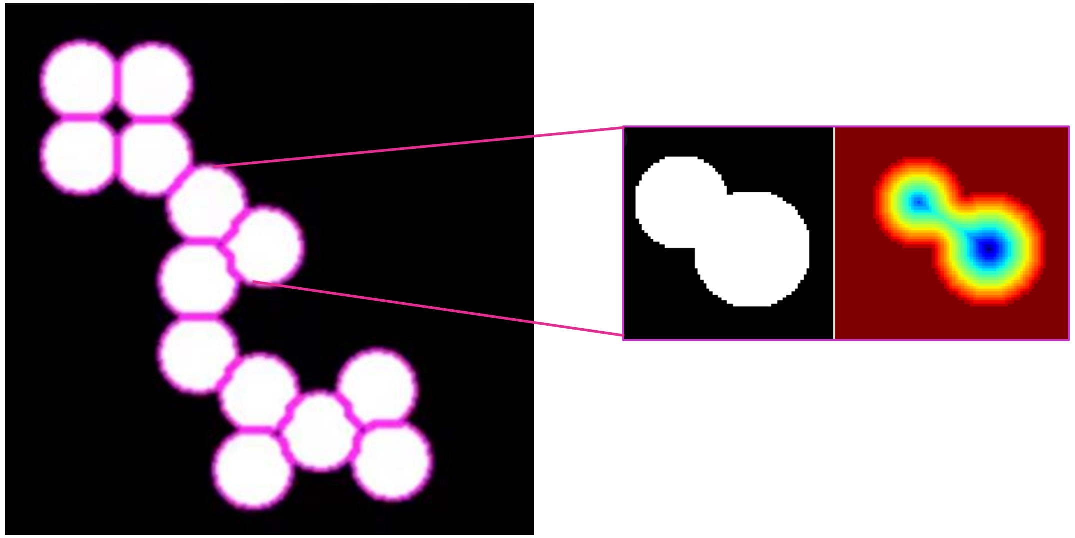
Watershed

- Watershed segmentation
  - overlapping objects



input

thresholding or clustering

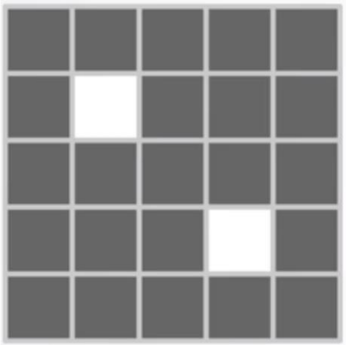


watershed

courtesy: JW Tay

Watershed

- Distance transform
  - computes the distance between each pix and nearest nonzero pix

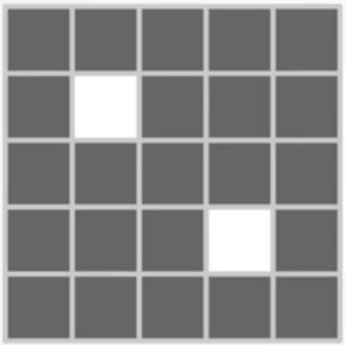


1.41	1	1.41	2.24	3.16
1	0	1	2	2.24
1.41	1	1.41	1	1.41
2.24	2	1	0	1
3.16	2.24	1.41	1	1.41

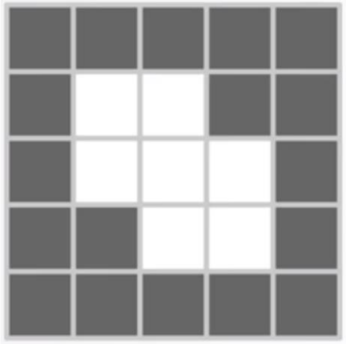
courtesy: JW Tay

Watershed

- Distance transform
  - computes the distance between each pix and nearest nonzero pix



1.41	1	1.41	2.24	3.16
1	0	1	2	2.24
1.41	1	1.41	1	1.41
2.24	2	1	0	1
3.16	2.24	1.41	1	1.41

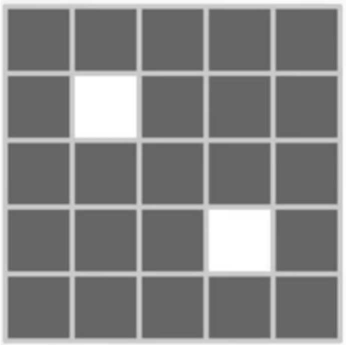


1.41	1	1	1.41	2.24
1	0	0	1	1.41
1	0	0	0	1
1.41	1	0	0	1
2.24	1.41	1	1	1.41

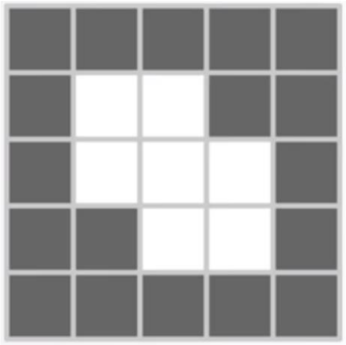
courtesy: JW Tay

Watershed

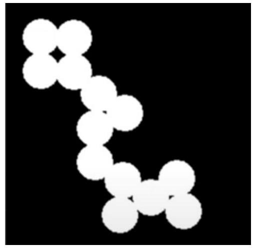
- Distance transform
  - computes the distance between each pix and nearest nonzero pix

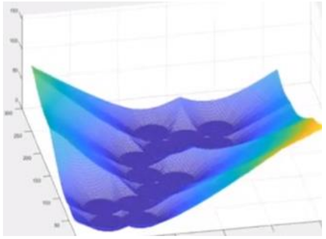


1.41	1	1.41	2.24	3.16
1	0	1	2	2.24
1.41	1	1.41	1	1.41
2.24	2	1	0	1
3.16	2.24	1.41	1	1.41



1.41	1	1	1.41	2.24
1	0	0	1	1.41
1	0	0	0	1
1.41	1	0	0	1
2.24	1.41	1	1	1.41

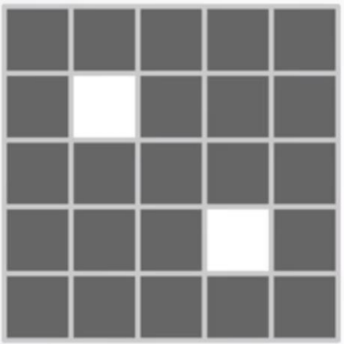




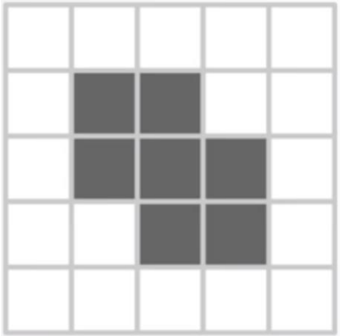
courtesy: JW Tay

Watershed

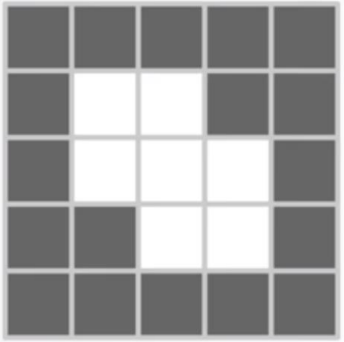
- Distance transform
  - computes the distance between each pix and nearest nonzero pix



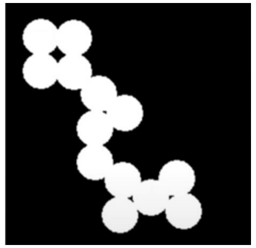
1.41	1	1.41	2.24	3.16
1	0	1	2	2.24
1.41	1	1.41	1	1.41
2.24	2	1	0	1
3.16	2.24	1.41	1	1.41

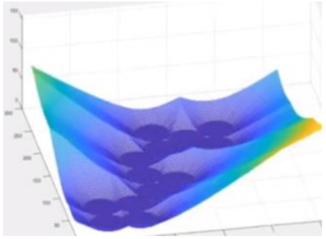


0	0	0	0	0
0	1	1	0	0
0	1	1.41	1	0
0	0	1	1	0
0	0	0	0	0



1.41	1	1	1.41	2.24
1	0	0	1	1.41
1	0	0	0	1
1.41	1	0	0	1
2.24	1.41	1	1	1.41

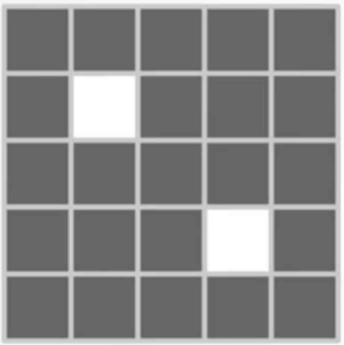




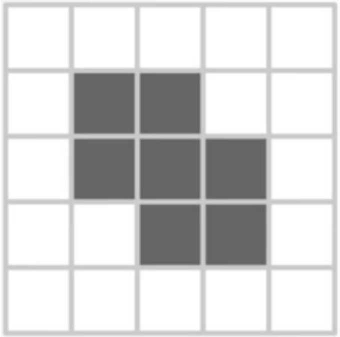
courtesy: JW Tay

Watershed

- Distance transform
  - computes the distance between each pix and nearest nonzero pix

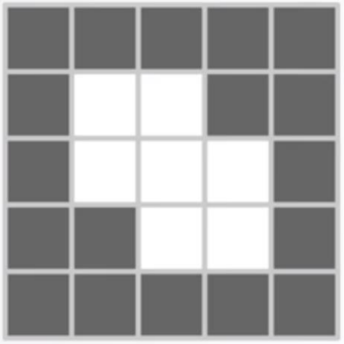


1.41	1	1.41	2.24	3.16
1	0	1	2	2.24
1.41	1	1.41	1	1.41
2.24	2	1	0	1
3.16	2.24	1.41	1	1.41

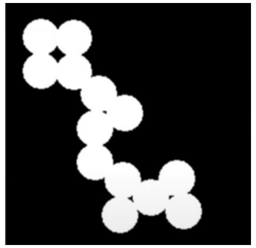


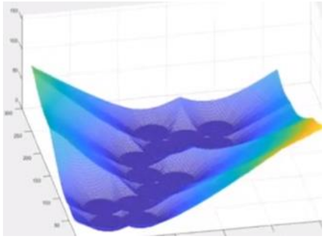
0	0	0	0	0
0	1	1	0	0
0	1	1.41	1	0
0	0	1	1	0
0	0	0	0	0

0	0	0	0	0
0	-1	-1	0	0
0	-1	-1.41	-1	0
0	0	-1	-1	0
0	0	0	0	0



1.41	1	1	1.41	2.24
1	0	0	1	1.41
1	0	0	0	1
1.41	1	0	0	1
2.24	1.41	1	1	1.41

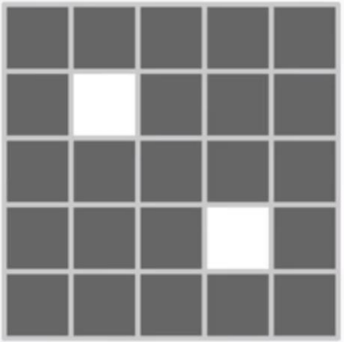




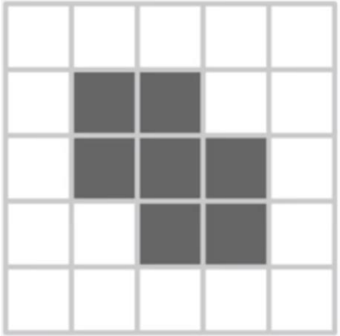
courtesy: JW Tay

Watershed

- Distance transform
  - computes the distance between each pix and nearest nonzero pix

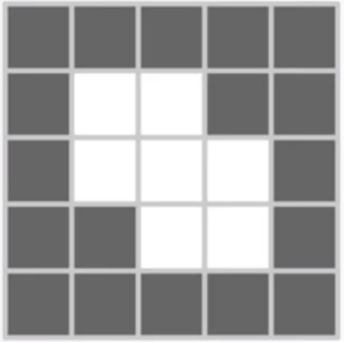


1.41	1	1.41	2.24	3.16
1	0	1	2	2.24
1.41	1	1.41	1	1.41
2.24	2	1	0	1
3.16	2.24	1.41	1	1.41

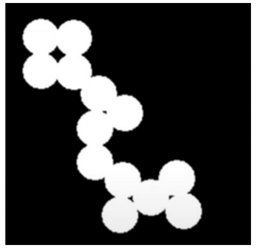


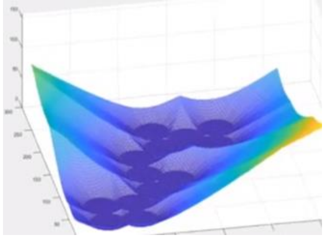
0	0	0	0	0
0	1	1	0	0
0	1	1.41	1	0
0	0	1	1	0
0	0	0	0	0

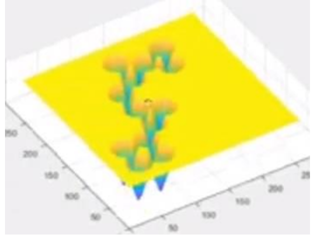
0	0	0	0	0
0	-1	-1	0	0
0	-1	-1.41	-1	0
0	0	-1	-1	0
0	0	0	0	0



1.41	1	1	1.41	2.24
1	0	0	1	1.41
1	0	0	0	1
1.41	1	0	0	1
2.24	1.41	1	1	1.41







courtesy: JW Tay

Watershed

- Algorithm: outline

- Requires:
  - each obj must be a basin, coinciding basin's bottom with approx obj center
- Input: image I
- convert I into inverted grey: $\hat{I} = L - grey(I)$
- compute the negative distance transform of $\hat{I}$
- non-max suppression over shallow minima
- fill basins & get watersheds
- update each segmented mask with watersheds

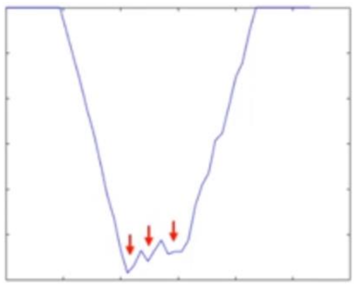
Watershed

- Algorithm: outline

- Requires:
  - each obj must be a basin, coinciding basin's bottom with approx obj center
- Input: image I
- convert I into inverted grey: $\hat{I} = L - grey(I)$
- compute the negative distance transform of $\hat{I}$
- non-max suppression over shallow minima
- fill basins & get watersheds
- update each segmented mask with watersheds

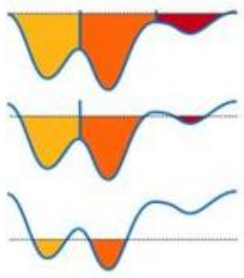
---

why non-max suppression?



Watershed

- Algorithm: getting watersheds



  - Initialize:

    - A $grey(I)$ having: $[\boldsymbol{h}_{min}\ \boldsymbol{h}_{max}]$
    - Minima points: $\boldsymbol{M}_1, \ldots, \boldsymbol{M}_R$
    - Thresholded set: $T_h = \{p \in I | I(p) \leq h\}$, where $\boldsymbol{p}$ is an pixel in $\boldsymbol{I}$ and $\boldsymbol{h}$ is some intensity level.

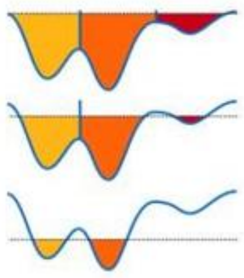
    - Let's define Influence set

$$C(M_i) = \text{cluster associated with } M_i$$

$$IZ_{h+1}(M_i) = \{p \in T_{h+1} | d(p, C(M_i)) < d(p, C(M_j))\} \quad \forall j,\ i \neq j$$

Watershed

- Algorithm: getting watersheds



- Initialize:
  - A $grey(I)$ having: $[\boldsymbol{h_{min}}\ \boldsymbol{h_{max}}]$
  - Minima points: $\boldsymbol{M_1, \ldots, M_R}$
  - Thresholded set: $T_h = \{p \in I | I(p) \leq h\}$, where $\boldsymbol{p}$ is an pixel in $\boldsymbol{I}$ and $\boldsymbol{h}$ is some intensity level.
  - Let's define Influence set

$$C(M_i) = \text{cluster associated with } M_i$$

$$IZ_{h+1}(M_i) = \{p \in T_{h+1} | d(p, C(M_i)) < d(p, C(M_j))\} \quad \forall j,\ i \neq j$$

- Run:
  - $\boldsymbol{h = h_{min}}$
  - Immersed set: $X_h = X_{h_{min}} = T_{h_{min}} = \{p \in I | I(p) \leq h_{min}\}$
  - **Loop** until $\boldsymbol{h_{max}}$

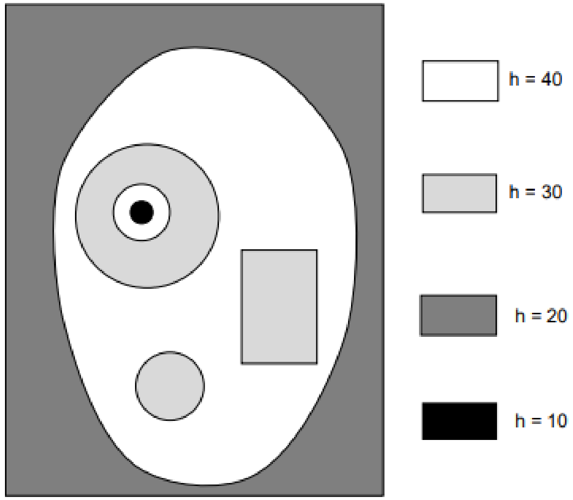
$$X_{h+1} = X_h \bigcup \underbrace{\boxed{IZ_{h+1}(M_1)}}_{\text{Influence set of minima } M_1 \text{ at level } h+1} \ldots \bigcup IZ_{h+1}(M_R)$$

  - $Watershed(\boldsymbol{I})$ = Set of all pixels in $\boldsymbol{I} \setminus \boldsymbol{X_{hmax}}$

Watershed

- Algorithm: another implementation
  - values graph: (V, E, f)

input image

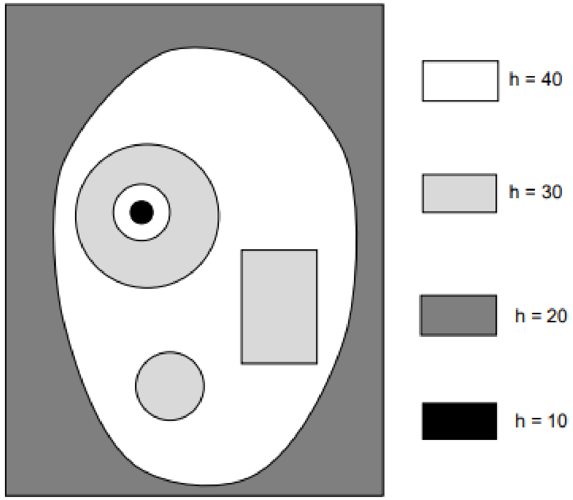


Watershed

- Algorithm: another implementation
  - values graph: (V, E, f)

$f(p)$ denotes the grey value

input image



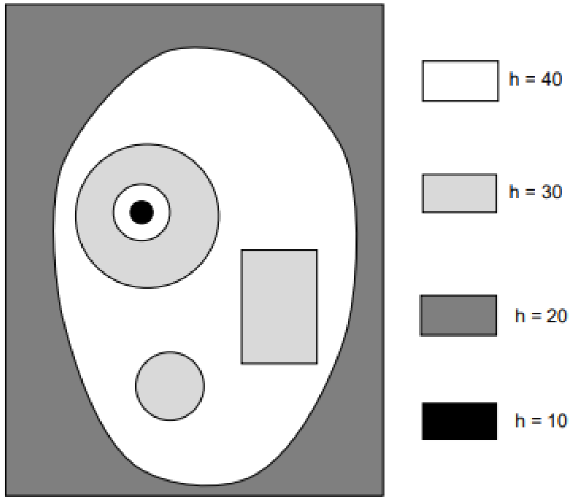
Watershed

- Algorithm: another implementation
  - values graph: (V, E, f)

$f(p)$ denotes the grey value

pixel p, $p \in V$

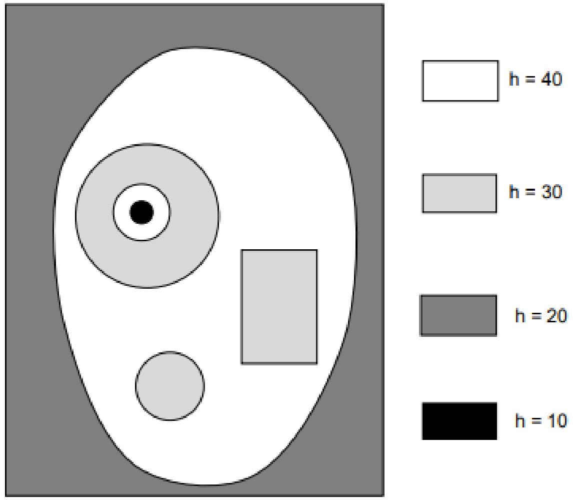
input image



Watershed

- Algorithm: another implementation
  - values graph: (V, E, f)

input image



$f(p)$ denotes the grey value

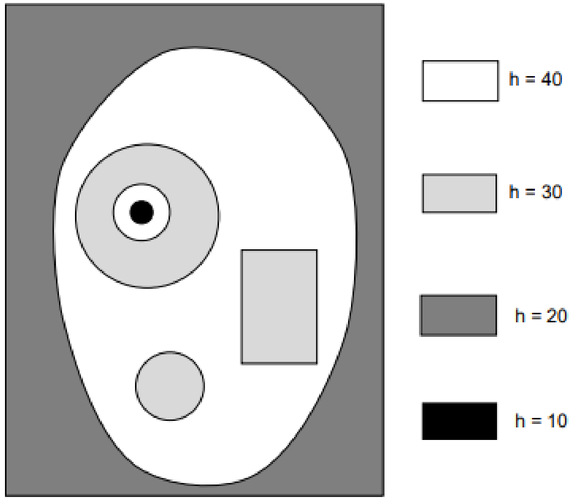
pixel p, $p \in V$

level component C_h at level h

Watershed

- Algorithm: another implementation
  - values graph: (V, E, f)

input image



$f(p)$ denotes the grey value

pixel p, $p \in V$

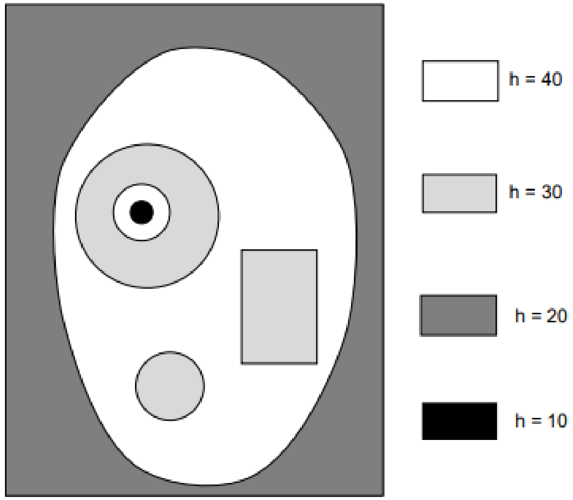
level component C_h at level h

are represented by a single node $v \in V$

Watershed

- Algorithm: another implementation
  - values graph: (V, E, f)

input image



$f(p)$ denotes the grey value

pixel p, $p \in V$

level component C_h at level h

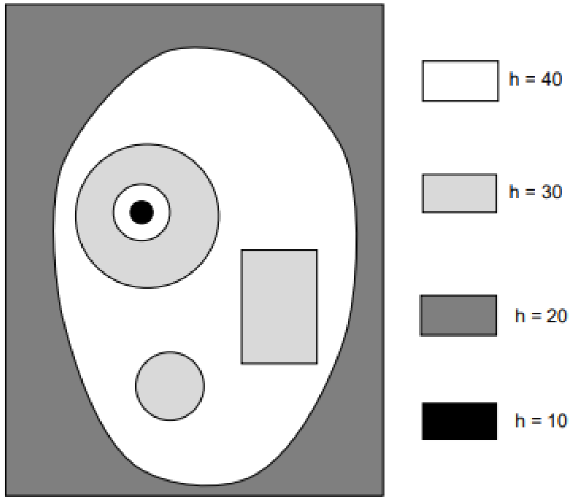
are represented by a single node $v \in V$

$$v = \{p \in V | p \in C_h\}$$

Watershed

- Algorithm: another implementation
  - values graph: (V, E, f)

input image



$f(p)$ denotes the grey value

pixel p, $p \in V$

level component C_h at level h

are represented by a single node $v \in V$

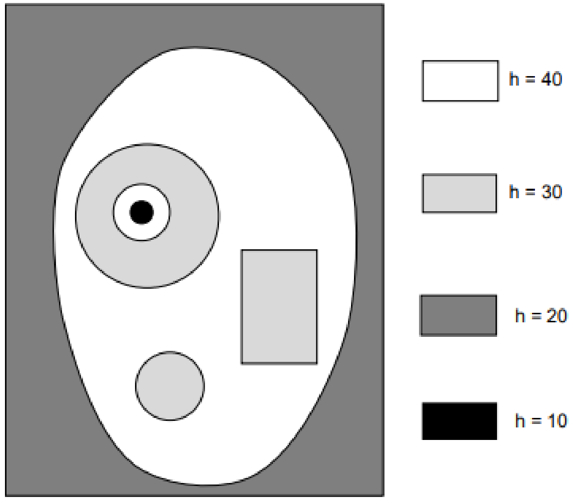
$$v = \{p \in V | p \in C_h\}$$

pair (v, w) of level components is an element of

Watershed

- Algorithm: another implementation
  - values graph: (V, E, f)

input image



$f(p)$ denotes the grey value

pixel p, $p \in V$

level component C_h at level h

are represented by a single node $v \in V$

$$v = \{p \in V | p \in C_h\}$$

pair (v, w) of level components is an element of

E if and only if $\exists(p \in v, q \in w : (p, q) \in E \wedge f(p) < f(q))$

Watershed

$f(p)$ denotes the grey value

pixel p, $p \in V$

level component C_h at level h

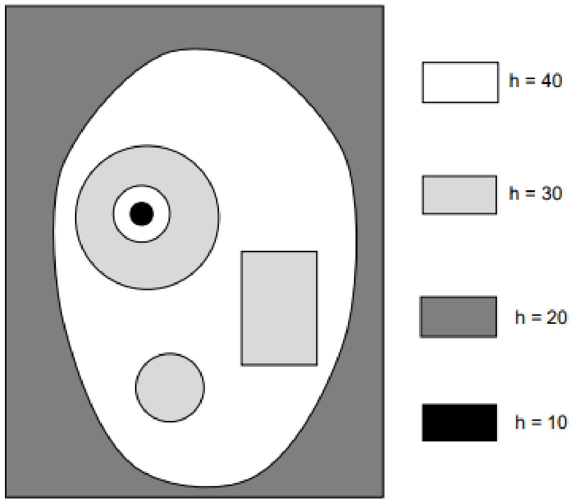
are represented by a single node $v \in V$

$$v = \{p \in V | p \in C_h\}$$

$$E \text{ if and only if } \exists (p \in v, q \in w : (p, q) \in E \wedge f(p) < f(q))$$

- Algorithm: another implementation
  - compute the watershed transform of directed graph
  - Transform the labelled graph back to the image
  - Dijkstra shortest path

input image



Watershed

$f(p)$ denotes the grey value

pixel p, $p \in V$

level component C_h at level h

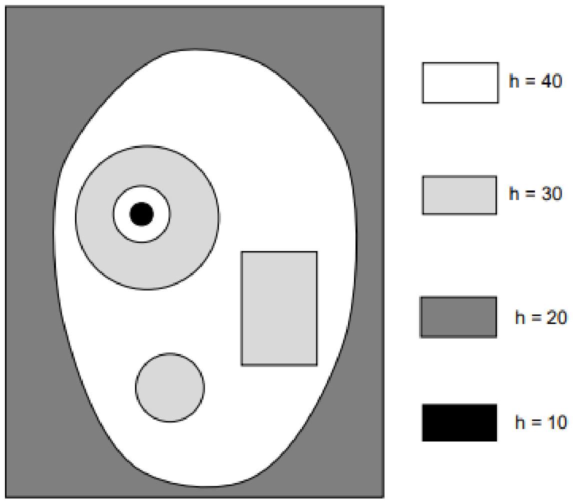
are represented by a single node $v \in V$

$$v = \{p \in V | p \in C_h\}$$

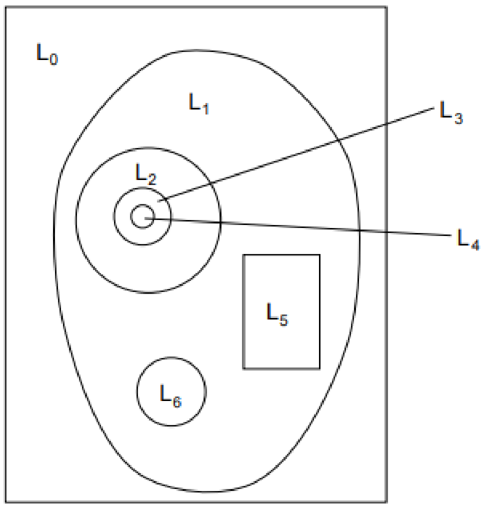
E if and only if $\exists (p \in v, q \in w : (p,q) \in E \wedge f(p) < f(q))$

- Algorithm: another implementation
  - compute the watershed transform of directed graph
  - Transform the labelled graph back to the image
  - Dijkstra shortest path

input image



level components



Watershed

$f(p)$ denotes the grey value

pixel p, $p \in V$

level component C_h at level h

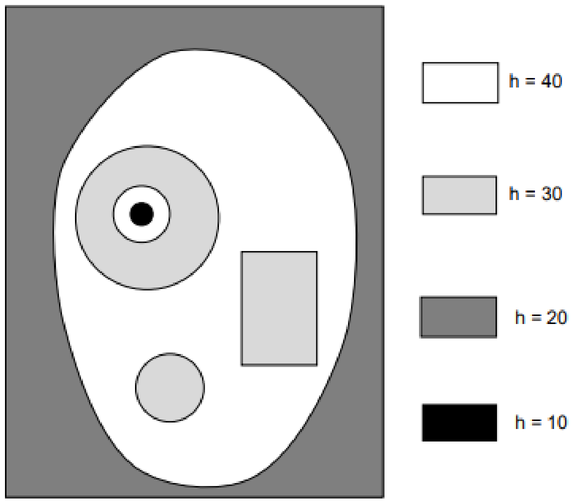
are represented by a single node $v \in V$

$$v = \{p \in V | p \in C_h\}$$

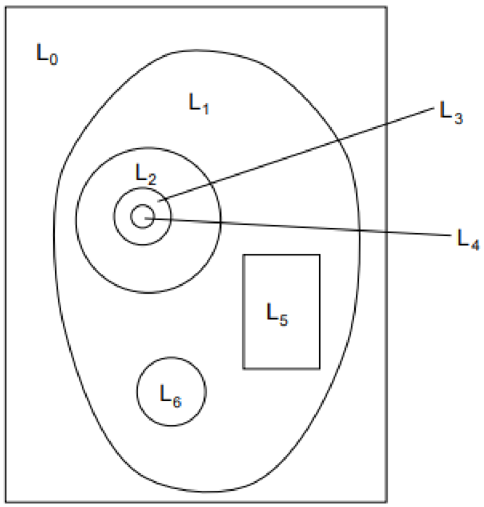
E if and only if $\exists (p \in v, q \in w : (p,q) \in E \wedge f(p) < f(q))$

- Algorithm: another implementation
  - compute the watershed transform of directed graph
  - Transform the labelled graph back to the image
  - Dijkstra shortest path

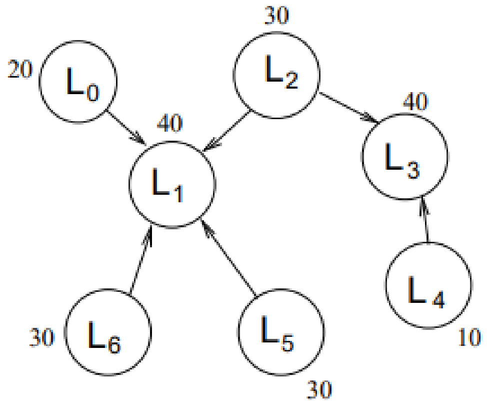
input image



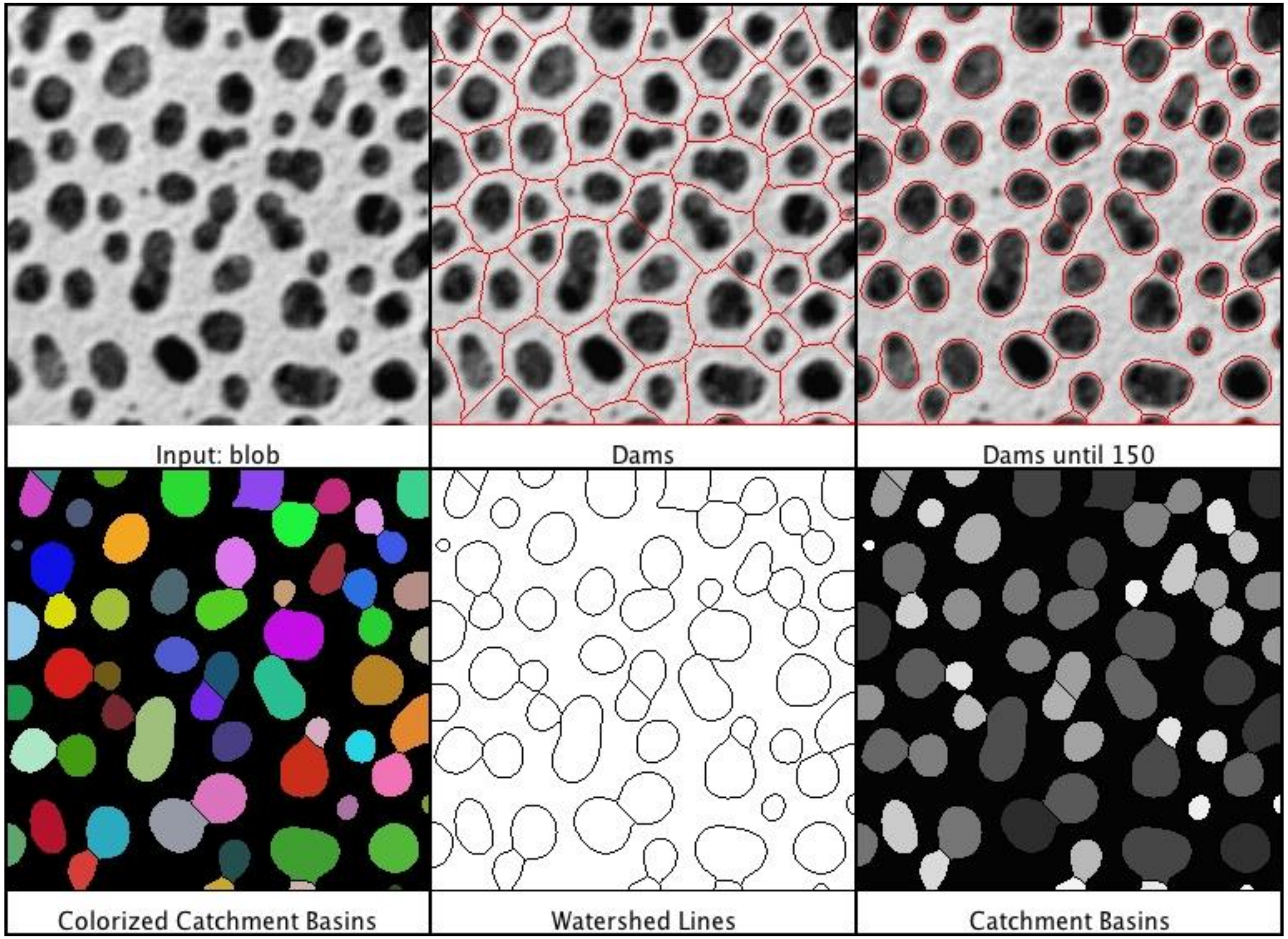
level components



directed graph

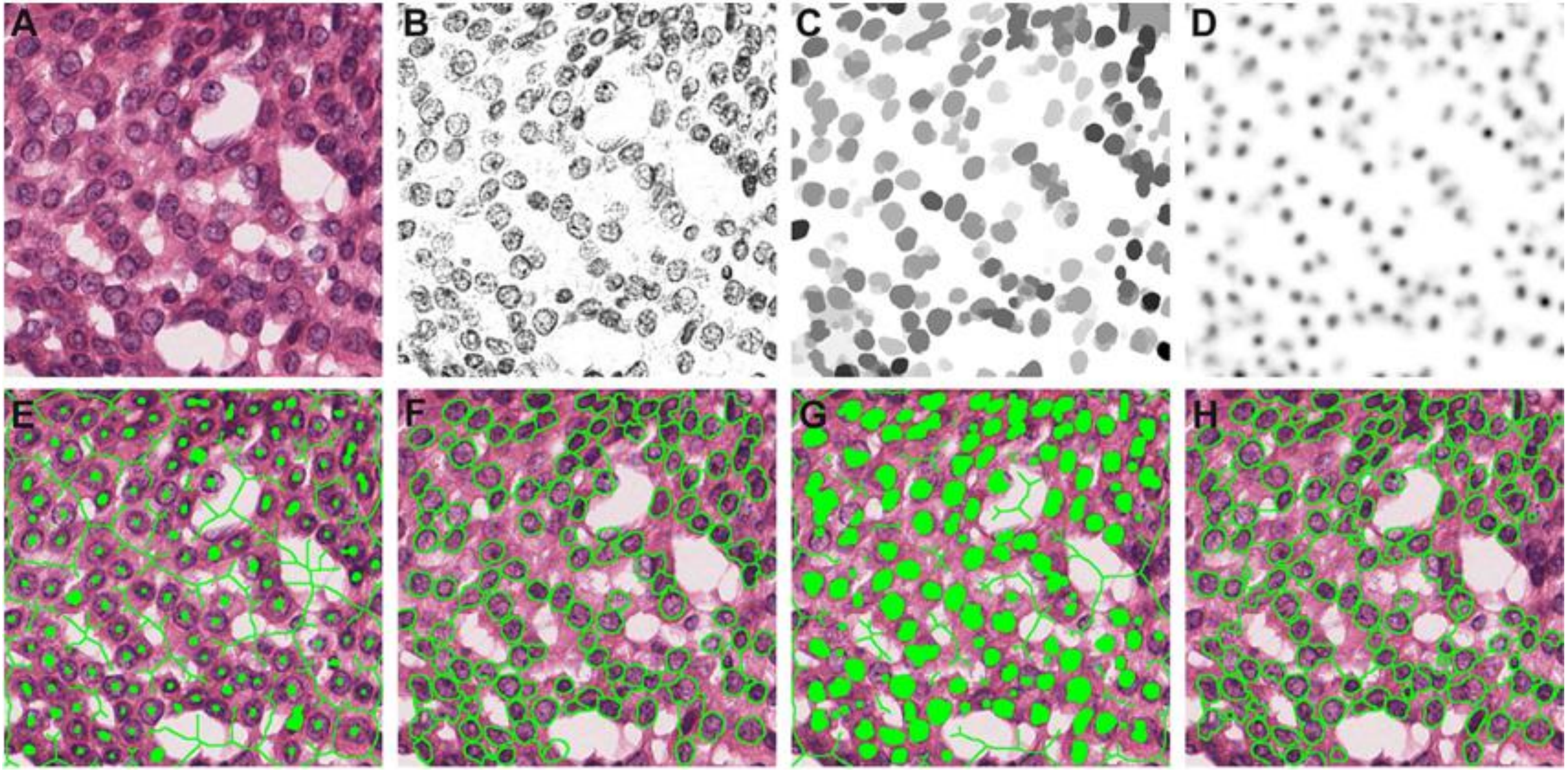


Watershed



courtesy: EPFL

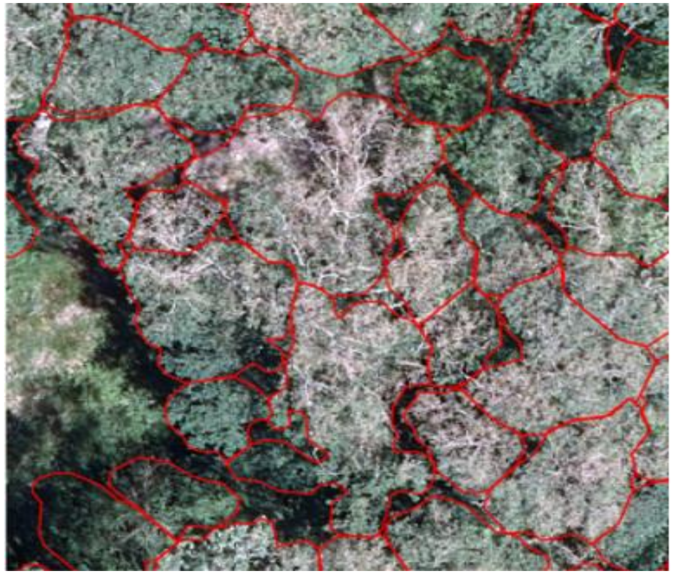
Watershed



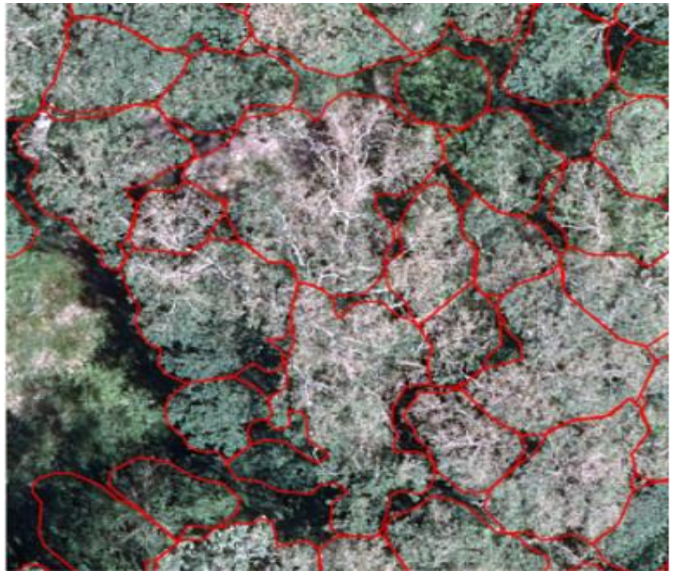
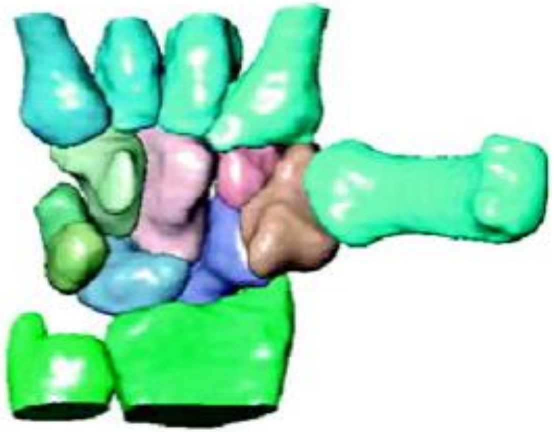
courtesy: M Veta

Watershed

Watershed

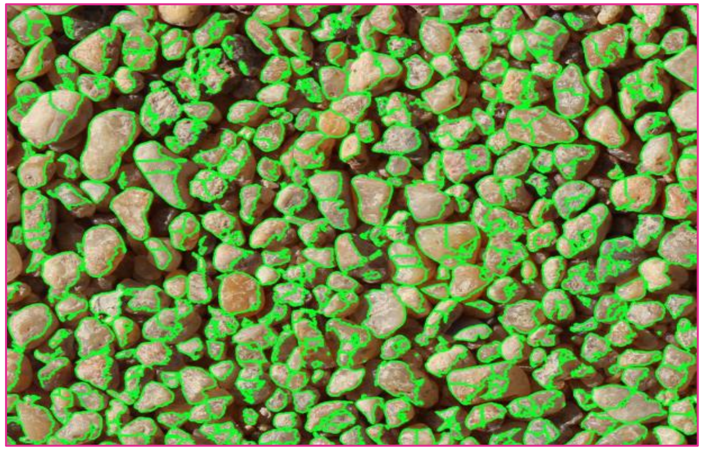


Watershed

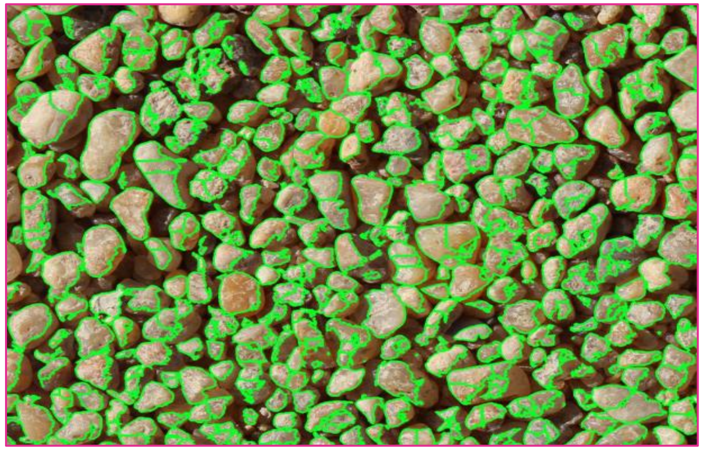




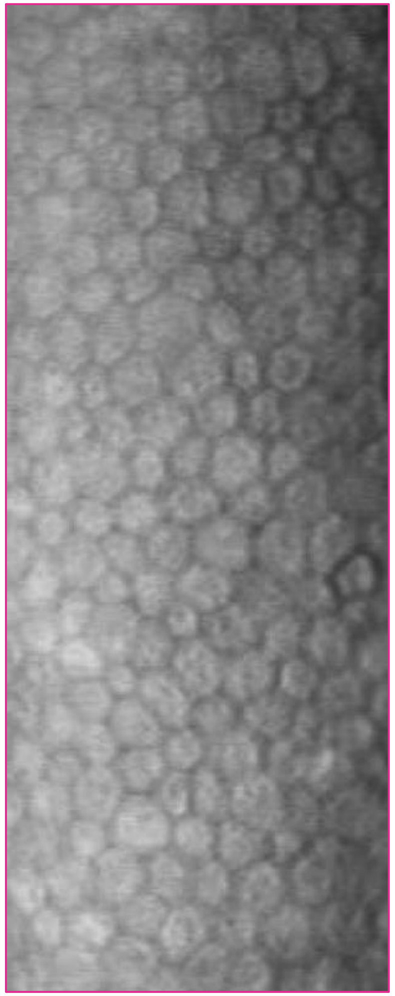
Watershed

Watershed

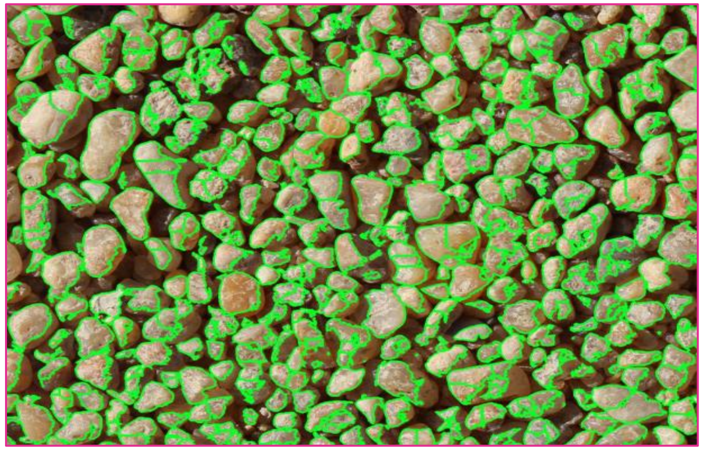


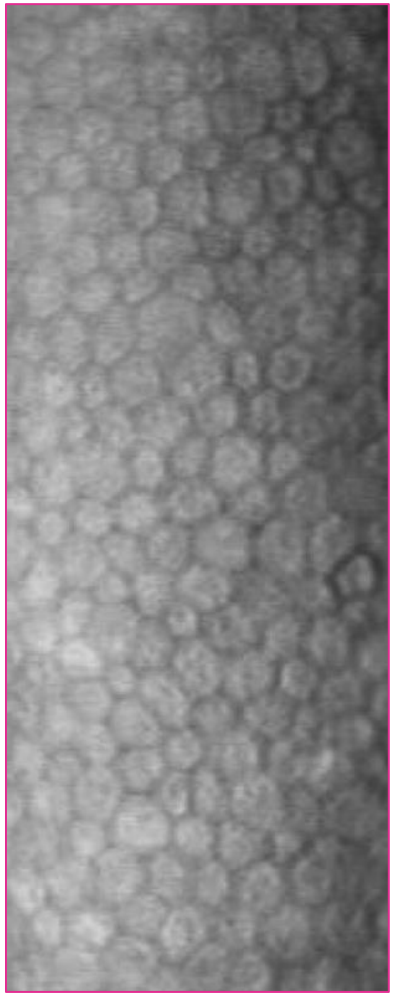
Watershed





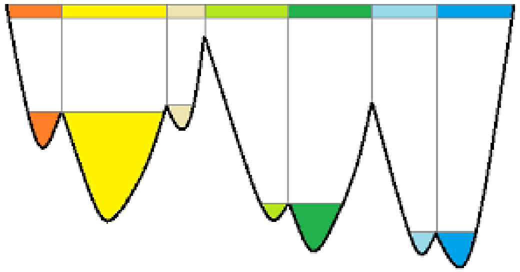
Watershed





Conclusion

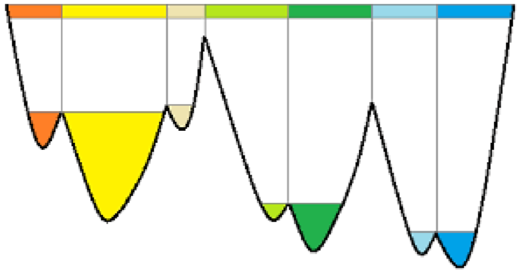
- Watershed



Conclusion

- Watershed

- Watershed
  - Precise boundaries even for overlapping similar objects
  - Images treated as topological surfaces



Conclusion

- Watershed

❑ Watershed

- Precise boundaries even for overlapping similar objects
- Images treated as topological surfaces