Computer Vision I

Bjoern Andres, Holger Heidrich

Machine Learning for Computer Vision
TU Dresden

Winter Term 2022/2023

Definition 1. Let $n_0, n_1 \in \mathbb{N}$, let $V = [n_0] \times [n_1]$ and let $C \subseteq \mathbb{R}$. Given

- a metric $d_s : V \times V \to \mathbb{R}_0^+$ and a decreasing $w_s : \mathbb{R}_0^+ \to [0,1]$
- a metric $d_c : C \times C \to \mathbb{R}_0^+$ and a decreasing $w_c : \mathbb{R}_0^+ \to [0,1]$
- a $N : V \to 2^V$ that defines for every pixel $v \in V$ a set $N(v) \subseteq V$ called the **spatial neighborhood** of v
- the $\nu : C^V \to \mathbb{R}^V$, called **normalization**, such that for any digital image $f : V \to C$ and any pixel $v \in V$:

$$\nu(f)(v) = \sum_{v' \in N(v)} w_s(d_s(v,v'))\, w_c(d_c(f(v), f(v'))) \ , \tag{1}$$

the **bilateral filter** w.r.t. d_s, w_s, d_c, w_c and N is the $\beta : C^V \to (\mathbb{R}C)^V$ such that for any digital image $f : V \to C$ and any pixel $v \in V$:

$$\beta(f)(v) = \frac{1}{\nu(f)(v)} \sum_{v' \in N(v)} w_s(d_s(v,v'))\, w_c(d_c(f(v), f(v')))\, f(v') \tag{2}$$

Example.

- $d_s(v, v') = \|v - v'\|_2$ and, for a filter parameter $\sigma_s > 0$:

$$w_s(x) = \frac{1}{\sigma_s \sqrt{2\pi}} \exp\left(-\frac{x^2}{2\sigma_s^2}\right) \tag{3}$$

- $d_c(g, g') = |g - g'|$ and, for a filter parameter $\sigma_c > 0$:

$$w_c(x) = \frac{1}{1 + \frac{x^2}{\sigma_c^2}} \tag{4}$$

- for a filter parameter $n \in \mathbb{R}_0^+$:

$$N(v) = \{v' \in V \mid d_s(v, v') \leq n\} \tag{5}$$

Suggested self-study:

- Implement a bilateral filter for gray-scale images
- Apply your implementation to a digital image
- Try different metrics d_s, d_r and weighting functions w_s, w_r
- Try iterating bilateral filtering
- Share and discuss your implementations, outputs and findings via OPAL

Advanced self-study:

- Define, implement and apply bilateral filtering for color images
- Share and discuss your implementations, outputs and findings via OPAL

Definition 2. Let $n_0, n_1 \in \mathbb{N}$, let $V = [n_0] \times [n_1]$, let $C \subseteq \mathbb{R}$ and let $N : V \to 2^V$ define for every pixel $v \in V$ a set $N(v) \subseteq V$ called the **spatial neighborhood** of v. The **median operator** wrt. N is the function $M\colon C^V \to C^V$ such that for any $f\colon V \to C$ and any $v \in V$:

$$M(f)(v) = \text{median } f(N(v)) \tag{6}$$

Non-linear operators

Noisy image

f

Filtered image

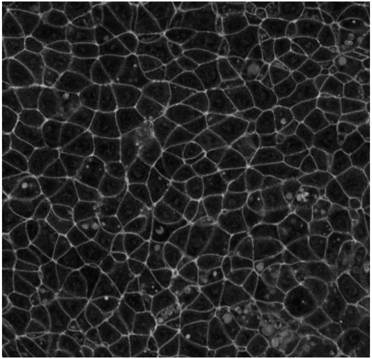
$M(f)$

Morphological operators

- We may identify any **binary** infinite digital image $f\colon \mathbb{Z}^2 \to \{0,1\}$ with its support set $f^{-1}(1) = \{v \in \mathbb{Z}^2 \mid f(v) = 1\}$.
- This allows us to apply operations from the field of **binary mathematical morphology** to binary infinite digital images.

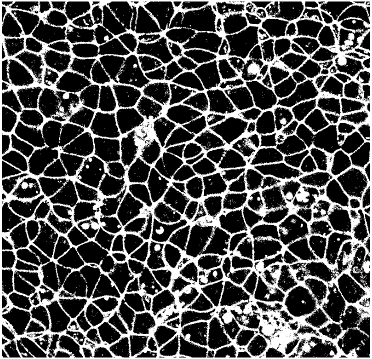
Non-linear operators

Image[1]



f

Binary image



$f \geq 45$

[1] By courtesy of Stephan Grill and his lab at the MPI of Molecular Cell Biology and Genetics.

Definition 3. For any $A, B \subseteq \mathbb{Z}^2$, we define

$$A \ominus B := \{v \in \mathbb{Z}^2 \mid B + v \subseteq A\} \tag{7}$$

$$A \oplus B := \{v \in \mathbb{Z}^2 \mid -B + v \cap A \neq \emptyset\} \tag{8}$$

and call these operations **erosion** and **dilation**. Moreover, we call the operations

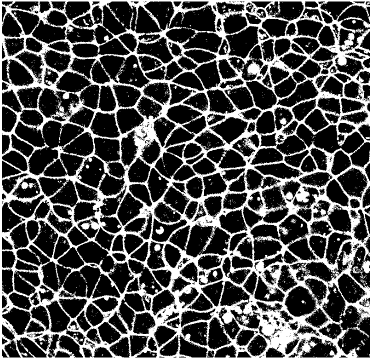
$$A \circ B := (A \ominus B) \oplus B \tag{9}$$

$$A \bullet B := (A \oplus B) \ominus B \tag{10}$$

opening and **closing**.

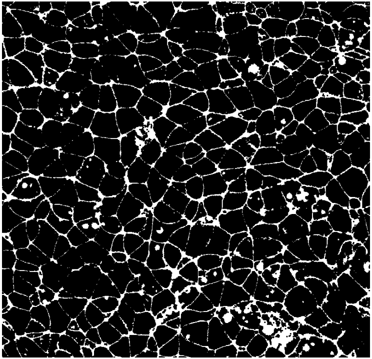
Definition 4. For any (typically small) support set B called a **structuring element** and any morphological operation $\otimes$, the operator $\varphi_{\otimes B} \colon \{0,1\}^{\mathbb{Z}\times\mathbb{Z}} \to \{0,1\}^{\mathbb{Z}\times\mathbb{Z}}$ such that for any (infinite binary digital image) $f \colon \mathbb{Z}^2 \to \{0,1\}$ and any (pixel) $v \in \mathbb{Z}^2$, we have $\varphi_{\otimes B}(f)(v) = 1$ if and only if $v \in f^{-1}(1) \otimes B$ is called the **morphological operator** wrt. $\otimes$ and B.

Binary image



f

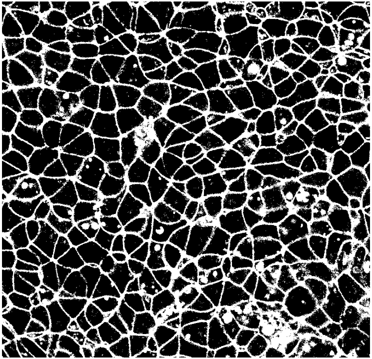
Erosion



$\varphi_{\ominus B}(f)$

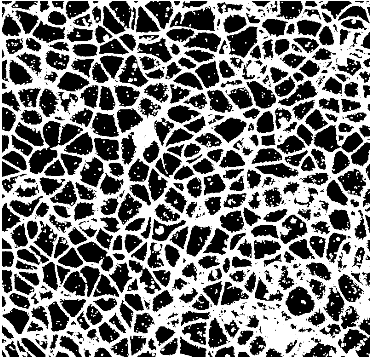
Non-linear operators

Binary image



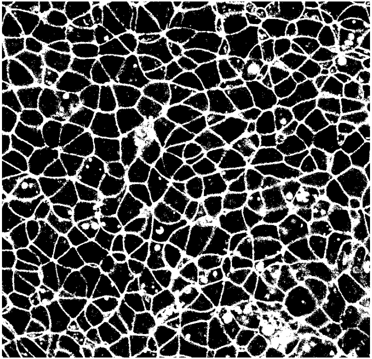
f

Dilation



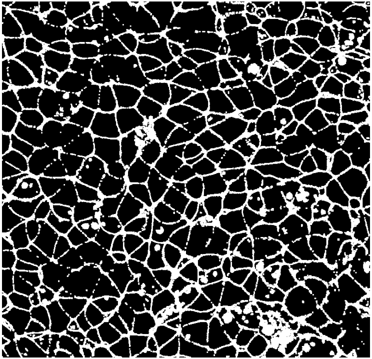
$\varphi_{\oplus B}(f)$

Binary image



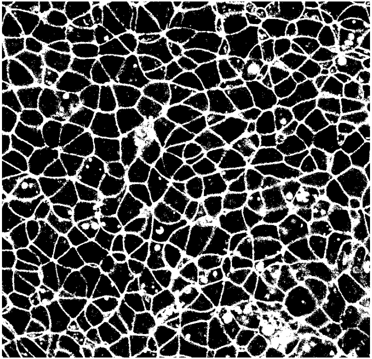
f

Opening



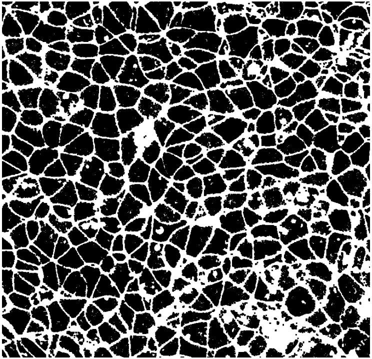
$\varphi_{\circ B}(f)$

Binary image



f

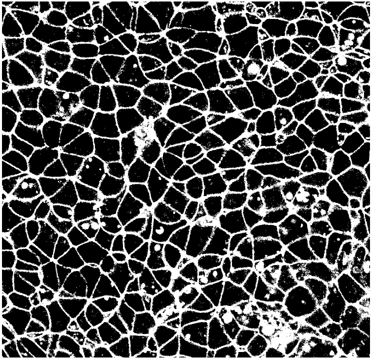
Closing



$\varphi_{\bullet B}(f)$

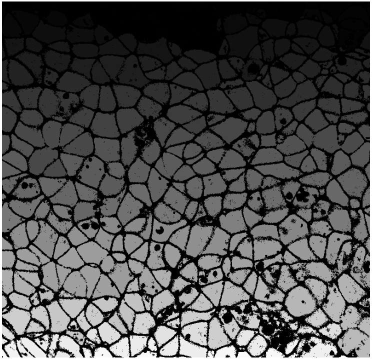
Definition 5. For any $n_0, n_1 \in \mathbb{N}$, the set $V = [n_0] \times [n_1]$ and the pixel grid graph $G = (V, E)$, an operator $\varphi\colon \mathbb{N}_0^V \to \mathbb{N}_0^V$ is called a **(connected) components operator** if for any digital image $f\colon V \to \mathbb{N}_0$ and any pixels $v, w \in V$, we have $\varphi(f)(v) = \varphi(f)(w)$ iff there exists a vw-path in G along which all pixels have the color zero.

Binary image



f

Connected component labeling



$\varphi(f)$

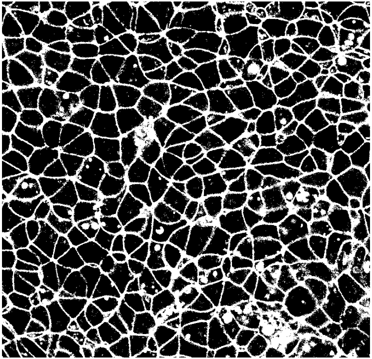
Non-linear operators

```cpp
size_t componentsImage(
    Marray<size_t> const & image,
    Marray<size_t> & components
) {
    components.resize({image.shape(0), image.shape(1)});
    PixelGridGraph pixelGridGraph({image.shape(0), image.shape(1)});
    size_t component = 0;
    stack<size_t> stack;
    for(size_t v = 0; v < pixelGridGraph.numberOfVertices(); ++v) {
        Pixel pixel = pixelGridGraph.coordinate(v);
        if(image(pixel[0], pixel[1]) == 0
        && components(pixel[0], pixel[1]) == 0) {
            ++component;
            components(pixel[0], pixel[1]) = component;
            stack.push(v);
            while(!stack.empty()) {
                size_t const v = stack.top();
                stack.pop();
                for(auto it = pixelGridGraph.verticesFromVertexBegin(v);
                it != pixelGridGraph.verticesFromVertexEnd(v); ++it) {
                    Pixel pixel = it.coordinate();
                    if(image(pixel[0], pixel[1]) == 0
                    && components(pixel[0], pixel[1]) == 0) {
                        components(pixel[0], pixel[1]) = component;
                        stack.push(*it);
                    }
                }
            }
        }
    }
    return component; // number of components
}
```

Definition 6. For any $n_0, n_1 \in \mathbb{N}$, the set $V = [n_0] \times [n_1]$ and the pixel grid graph $G = (V, E)$, the **distance operator** $\varphi \colon \mathbb{N}_0^V \to \mathbb{N}_0^V$ is such that for any digital image $f \colon V \to \mathbb{N}_0$ and any pixel $v \in V$, the number $\varphi(f)(v)$ is the minimum distance in the pixel grid graph from v to a pixel w with $f(w) = 1$.

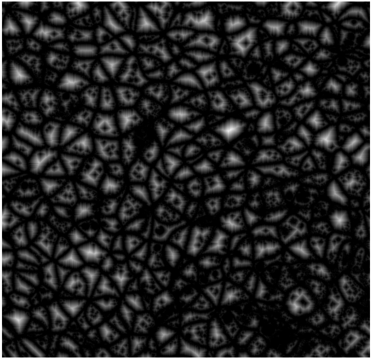
Non-linear operators

Binary image



f

Distance image



$\varphi(f)$

```
size_t distanceImage(
    Marray<size_t> const & image,
    Marray<size_t> & distances
) {
    distances.resize({image.shape(0), image.shape(1)}, 0);
    GridGraph pixelGridGraph({image.shape(0), image.shape(1)});
    size_t distance = 0;
    array<stack<size_t>, 2> stacks;
    for(size_t v = 0; v < pixelGridGraph.numberOfVertices(); ++v) {
        Pixel pixel = pixelGridGraph.coordinates(v);
        if(image(pixel[0], pixel[1]) != 0)
            stacks[0].push(v);
    }
    ++distance;
    for(;;) {
        auto & stack = stacks[(distance - 1) % 2];
        if(stack.empty())
            return distance - 1; // maximal distance
        while(!stack.empty()) {
            size_t const v = stack.top();
            stack.pop();
            for(auto it = pixelGridGraph.verticesFromVertexBegin(v);
            it != pixelGridGraph.verticesFromVertexEnd(v); ++it) {
                Pixel pixel = it.coordinate();
                if(image(pixel[0], pixel[1]) == 0
                && distances(pixel[0], pixel[1]) == 0) {
                    distances(pixel[0], pixel[1]) = distance;
                    stacks[distance % 2].push(*it);
                }
            }
        }
        ++distance;
    }
}
```

For any set V of pixels and neighborhood function $N\colon V \to 2^V$, **non-maximum suppression** is the operator $\varphi_{\mathrm{NMS}}\colon \mathbb{R}^V \to \mathbb{R}^V$ such that for each digital image $f\colon V \to \mathbb{R}$ and all pixels $v \in V$:

$$\varphi_{\mathrm{NMS}}(f)(v) = \begin{cases} f(v) & \text{if } f(v) = \max f(N(v)) \\ 0 & \text{otherwise} \end{cases} \tag{11}$$