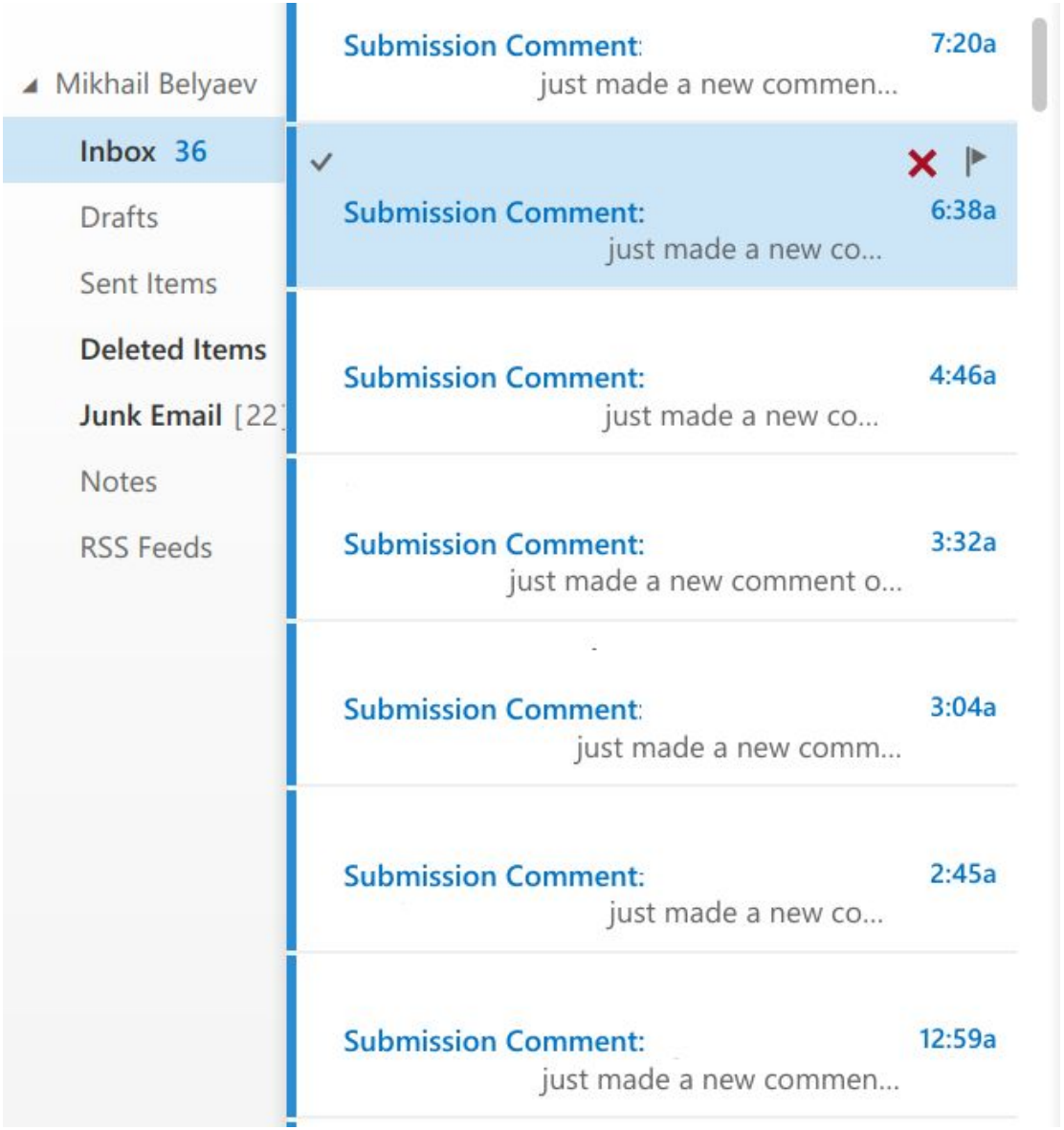
Introduction to Computer Vision

4. Hough recap.
Morphology. Image transforms.
In-class 1 analysis.

11.11.19

Mikhail Belyaev

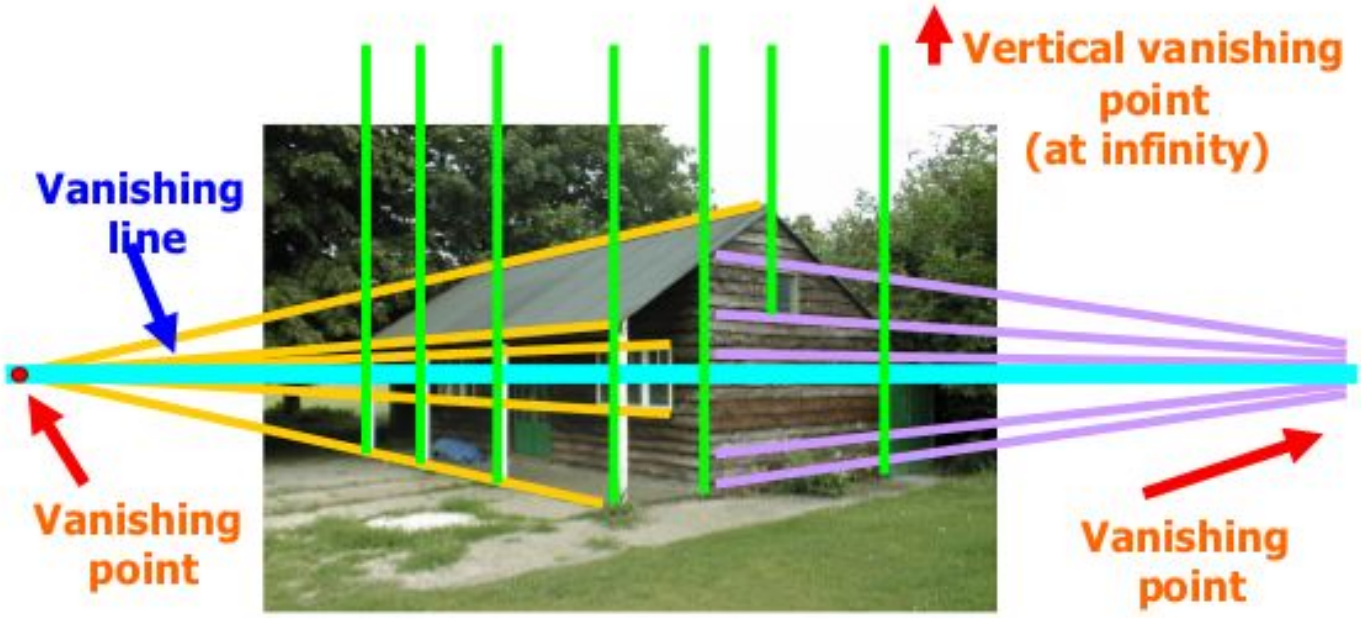
Skoltech
Skolkovo Institute of Science and Technology



Skoltech
Skolkovo Institute of Science and Technology

Hough transform

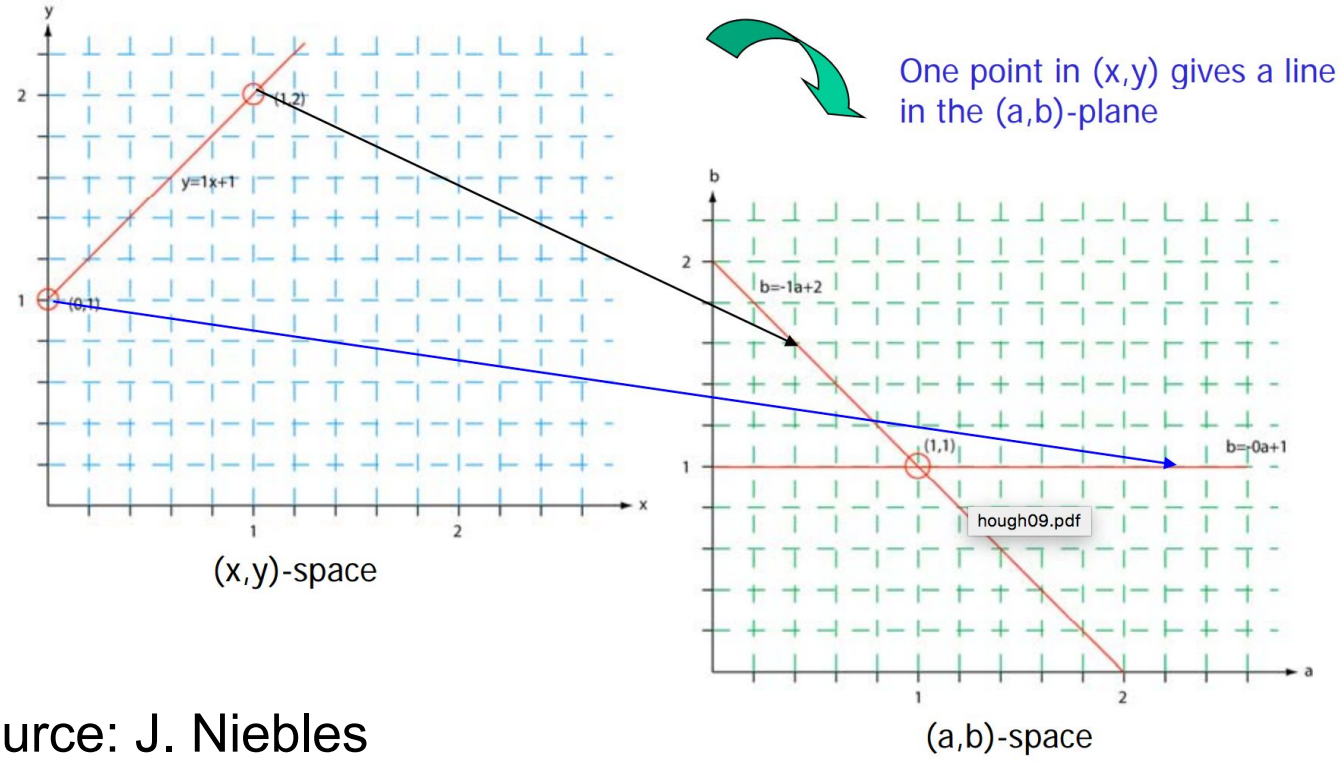
- Most popular usage - **lines detection in images**. Which gives us: key points, geometry structure, rotation angle, view point reconstruction
- Robust detection under noise conditions
- can detect other structures (not only lines) if their **parametric equation** is known



Hough transform

- Consider a point of known coordinates $(x_i; y_i)$
- Straight lines that pass that point: $y_i = a * x_i + b$

- Equation can be rewritten as follows: $b = - a * x_i + y_i$
- Point $(x_i; y_i)$ gives us line in **(a, b)-space**



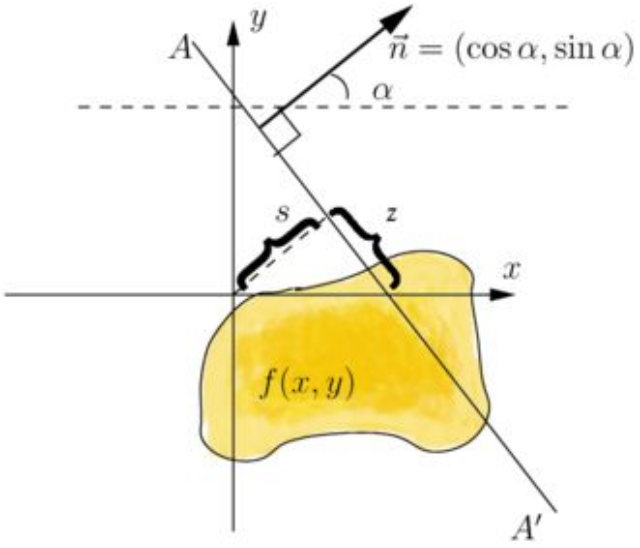
Source: J. Niebles

Hough transform parametrization

- Straight lines that pass that point: $y_i = a * x_i + b$
- Equation can be rewritten as follows: $b = - a * x_i + y_i$

- An alternative parametrization for a linear Hough transform:

$$x\,cos(\theta) + y\,sin(\theta) = \rho$$



Here we want to solve a primal (or direct) problem. Is it possible to solve an inverse one? Can it be useful?

Source: Wikipedia

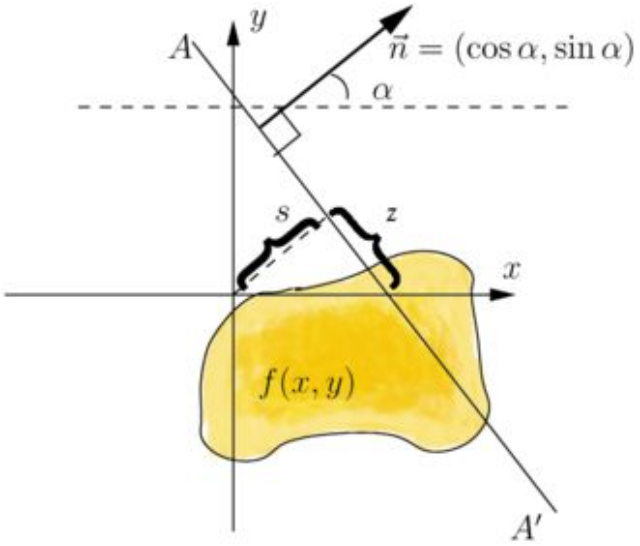
Hough transform parametrization

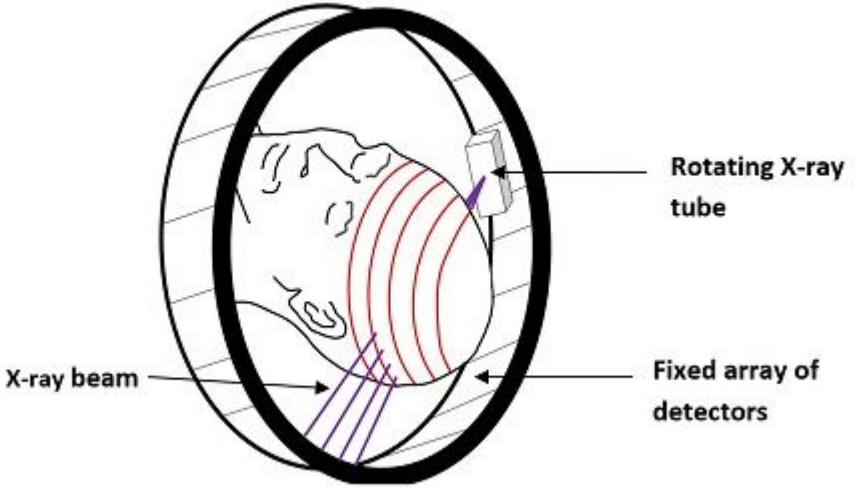
- An alternative parametrization for a linear Hough transform:

$$x\,cos(\theta) + y\,sin(\theta) = \rho$$

Is it possible to solve an inverse problem? Can it be useful?

Yes, in computed tomography!



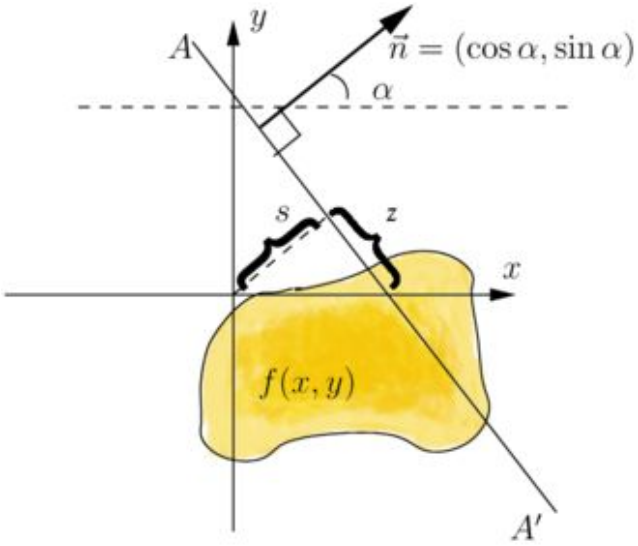


Source: Wikipedia

Hough transform parametrization

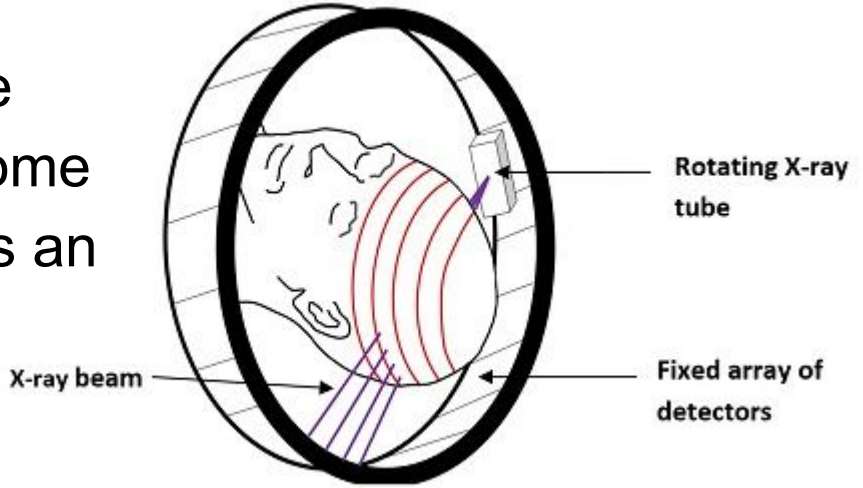
- An alternative parametrization for a linear Hough transform:

$$x\,cos(\theta) + y\,sin(\theta) = \rho$$



Is it possible to solve an inverse problem? Can it be useful?
Yes, in computed tomography!

The math behind this problem is the inverse Radon transformation. In some sense, Hough can be considered as an engineering approach to calculate discrete Radon transform.

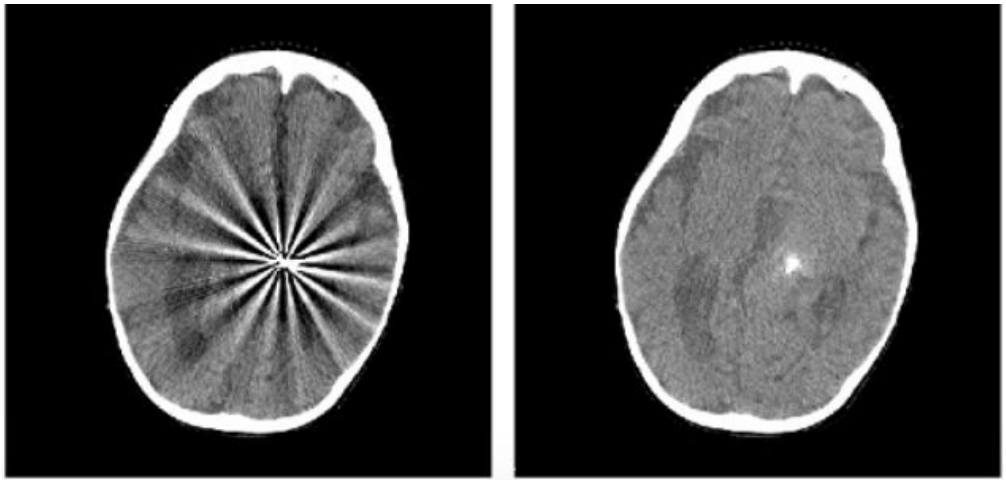


Source: Wikipedia

Radiotherapy projects: metal artifacts reduction

A slide from the first lecture

- ***Goals:*** *develop an algorithm to reduce metal artifacts, estimate the impact on dose distribution.*
- ***Partners:*** *Radiation therapy department at Burdenko Neurosurgery Institute.*
- ***Data:*** *~50 CT with metal artifacts, ~200 "normal" CT.*



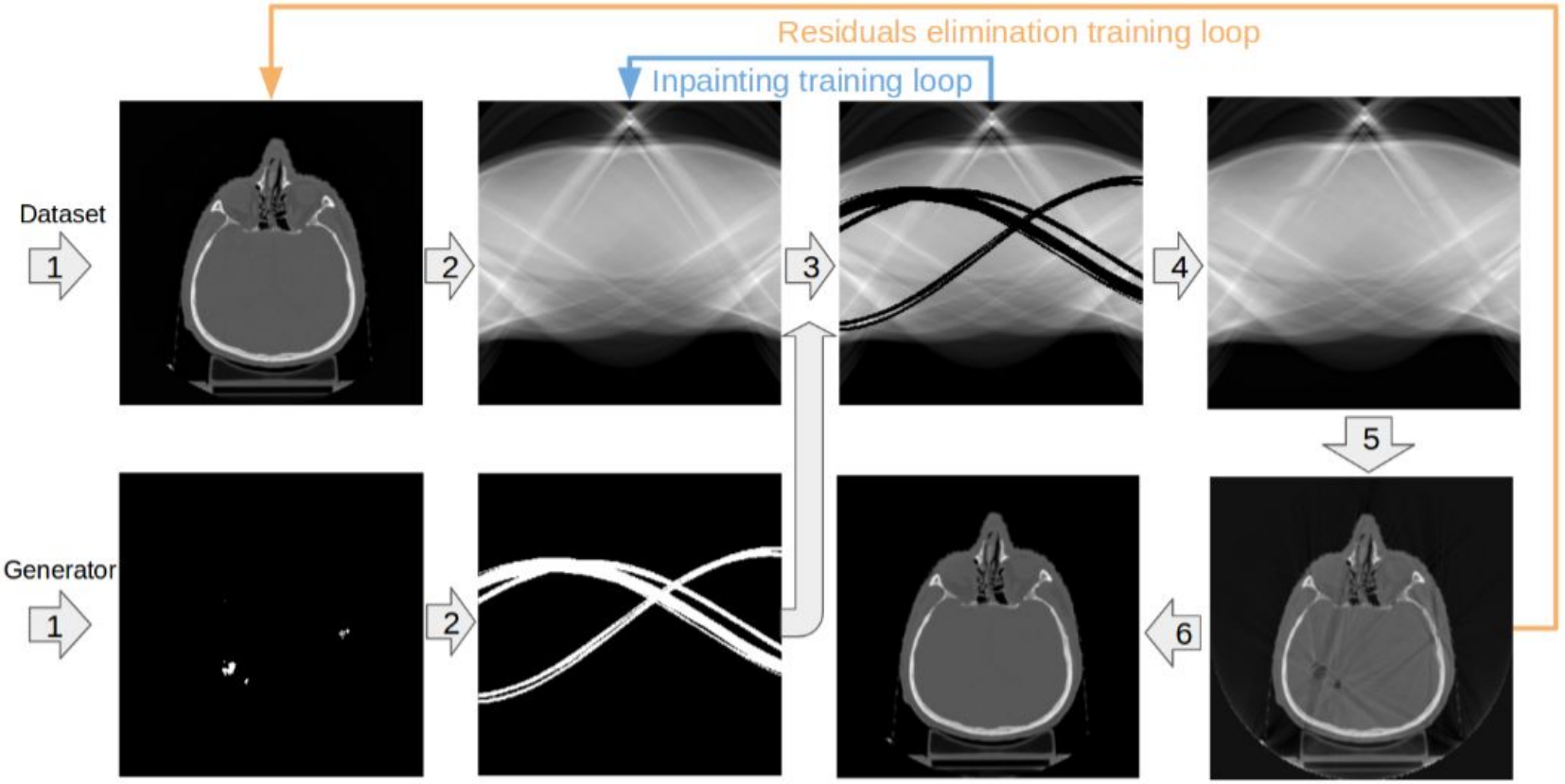
Algorithm	SSIM
Li-MAR	0.94
Inpainting only	0.94
Image-to-image only	0.97
Proposed algorithm	**0.99**

Multidomain partial convolutions CNN for metal artifacts reduction on brain CT. Pimkin et al. Submitted.

Skoltech
Skolkovo Institute of Science and Technology

Radiotherapy projects: metal artifacts reduction

Actually we worked in two domains and use Radon & Inverse Radon transformation to switch from one to another.



Multidomain partial convolutions CNN for metal artifacts reduction on brain CT. Pimkin et al. Submitted.

Skoltech
Skolkovo Institute of Science and Technology

Hough transform parametrization

- Straight lines that pass that point: $\mathbf{y_i = a * x_i + b}$
- Equation can be rewritten as follows: $\mathbf{b = - a * x_i + y_i}$

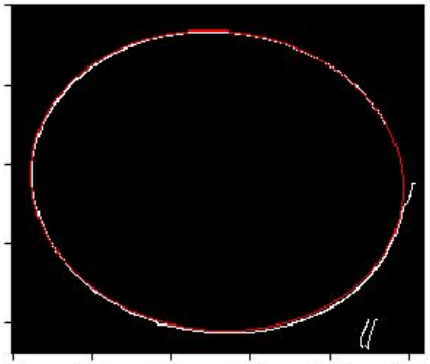
- An alternative parametrization for a linear Hough transform:

$$x\,cos(\theta) + y\,sin(\theta) = \rho$$

- A circular Hough transform:

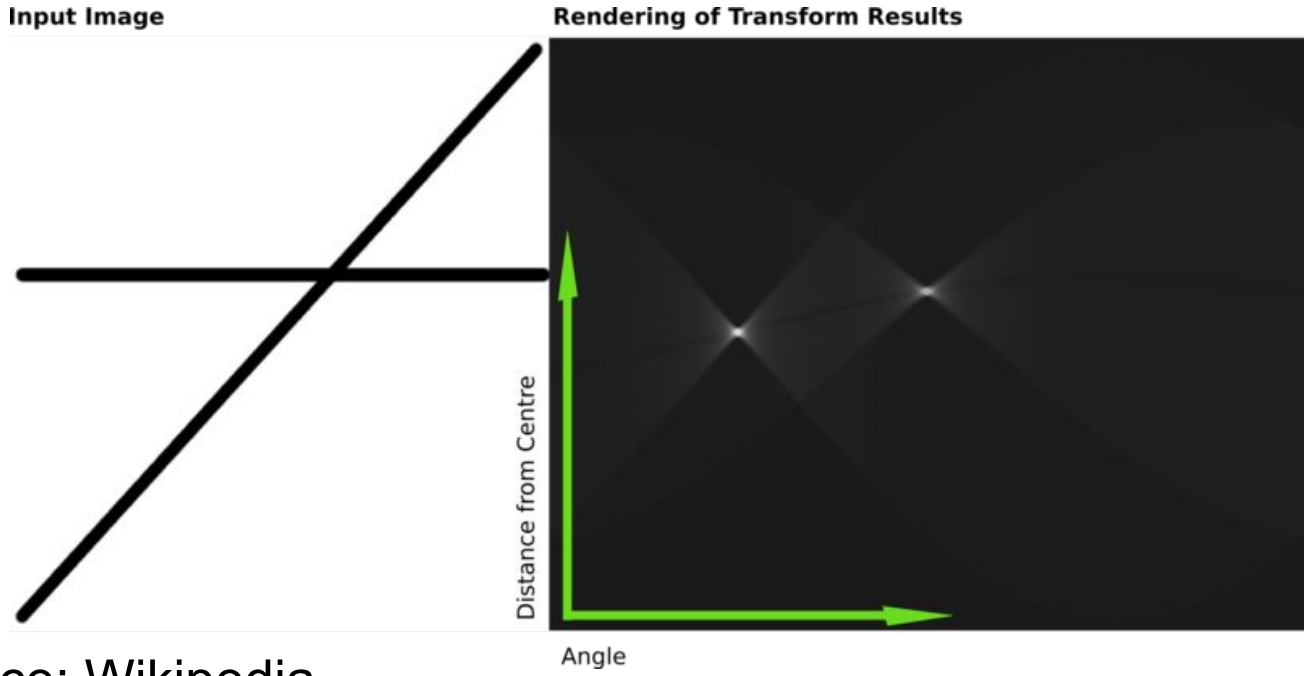
$$(x - x_0)^2 + (y - y_0)^2 = r^2$$

- It's also possible to look for ellipses using skimage



Hough transform

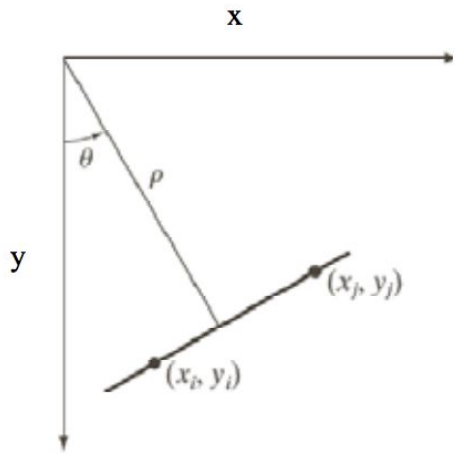
- Two points $(x_1; y_1)$ and $(x_2; y_2)$ define a line in the **(x, y)-space**
- These two points give rise to two different lines in **(a, b)-space**
- In **(a, b)-space** these lines will intersect in a point **(A, B)**
- All points on the line defined by $(x_1; y_1)$ and $(x_2; y_2)$ will parameterize lines that intersect in **(A, B)**

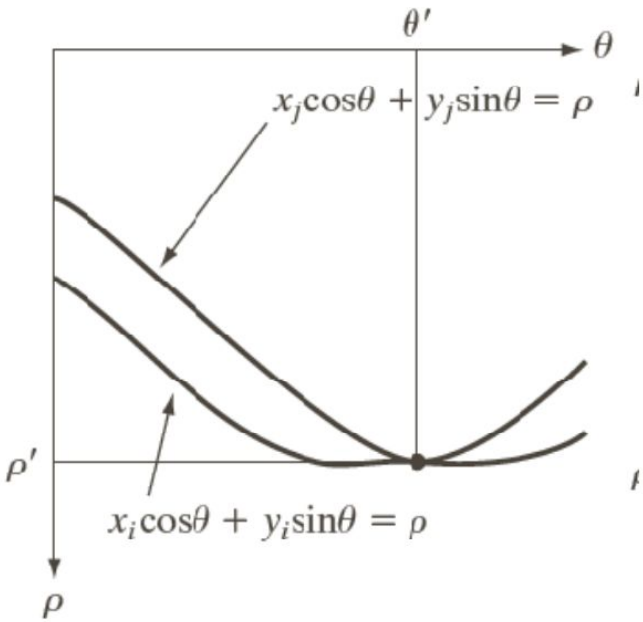


Source: Wikipedia

Hough transform

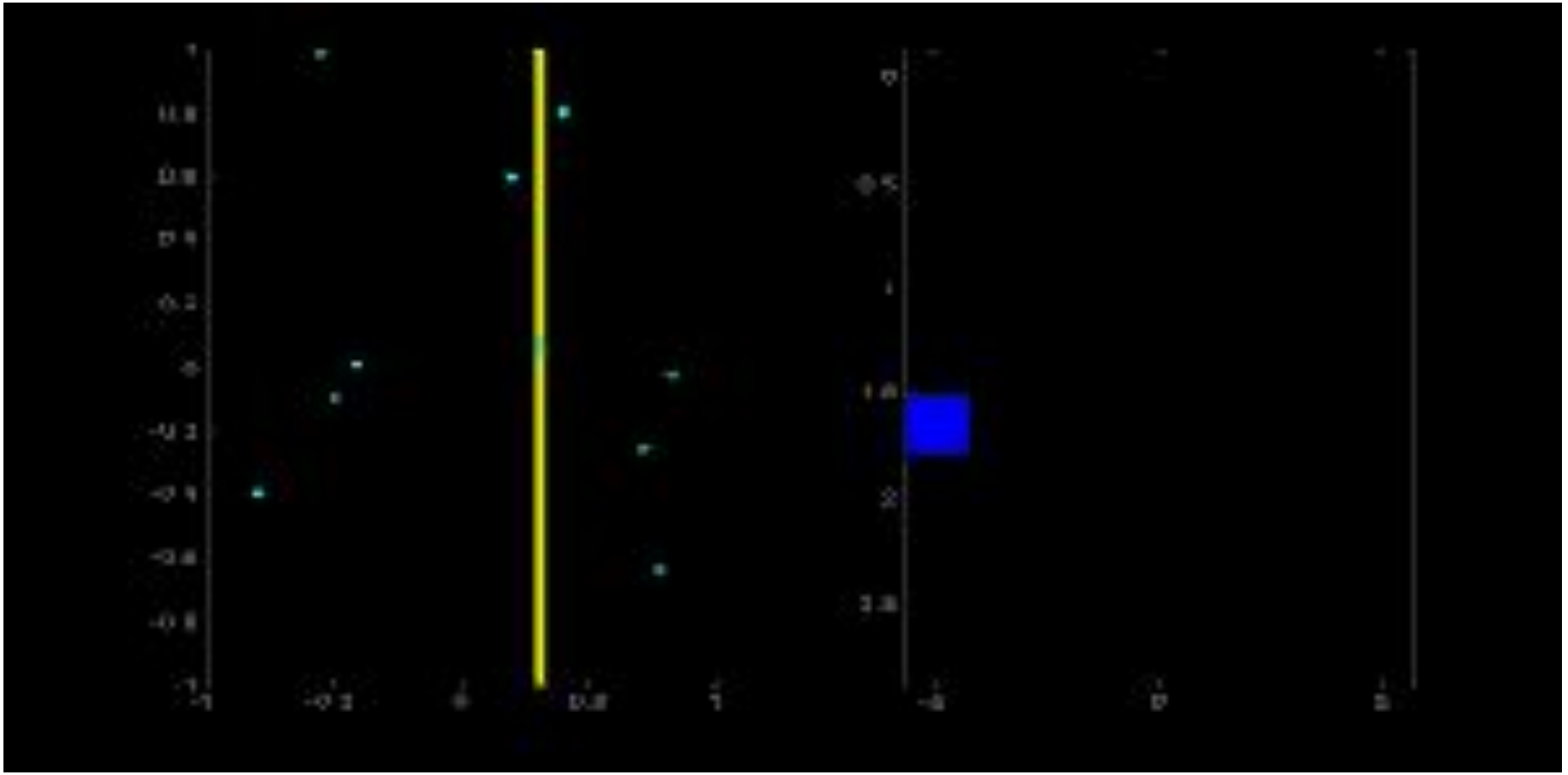
- Note that lines in (x y) space are not lines in (ρ θ) space, unlike (a b) space.
- A vertical line will have θ=0 and ρ equal to the intercept with the x-axis.
- A horizontal line will have θ=90 and ρ equal to the intercept with the y-axis.





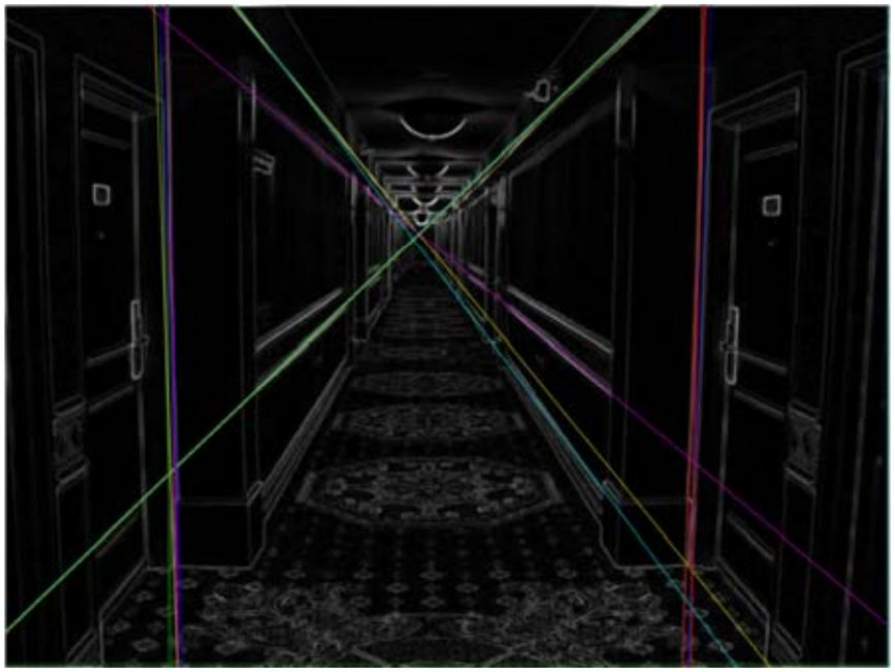
Source: J. Niebles

Hough transform



Source: Amos Storkey

Output of Hough transform



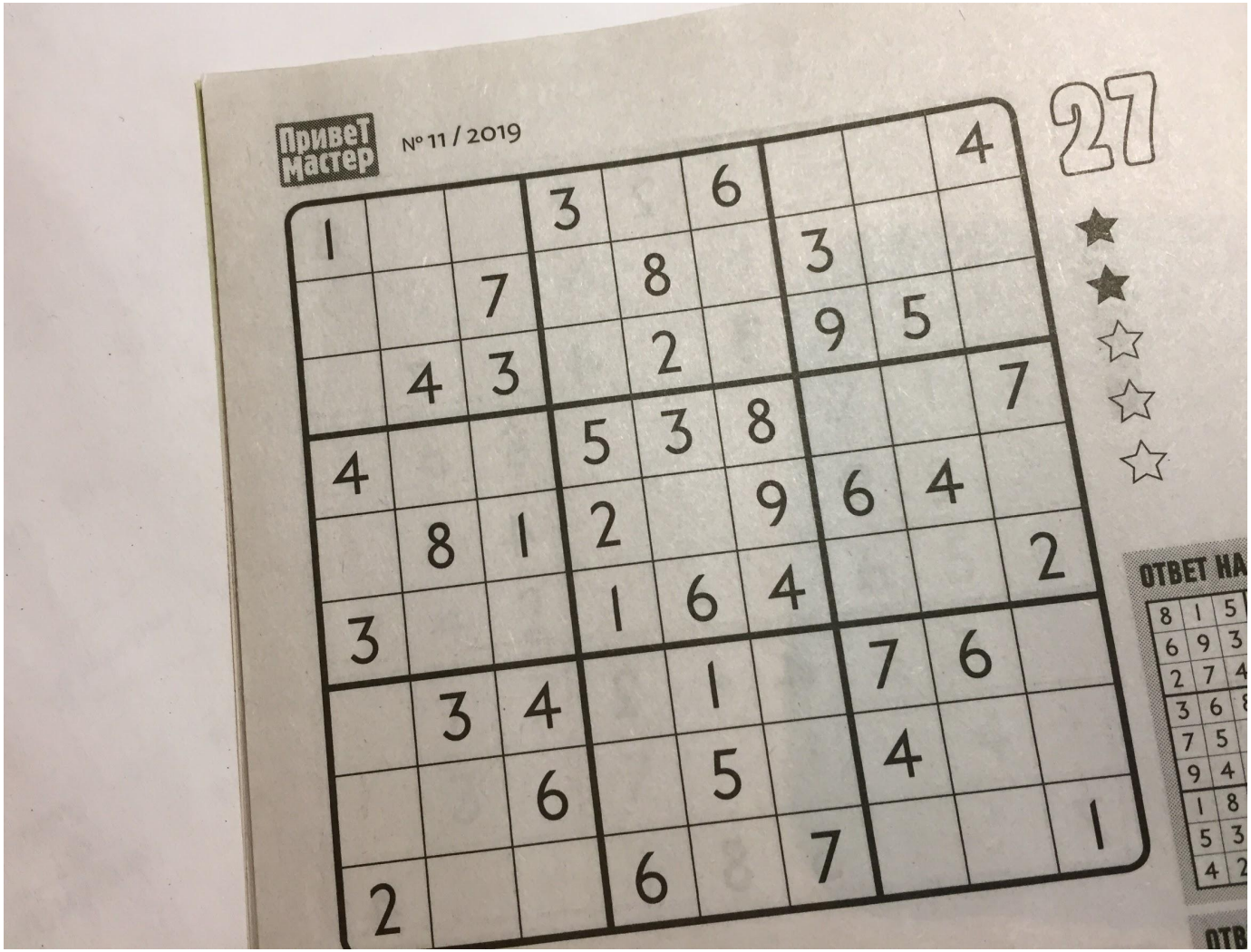
Source: J. Niebles

In-class 1 follow-up

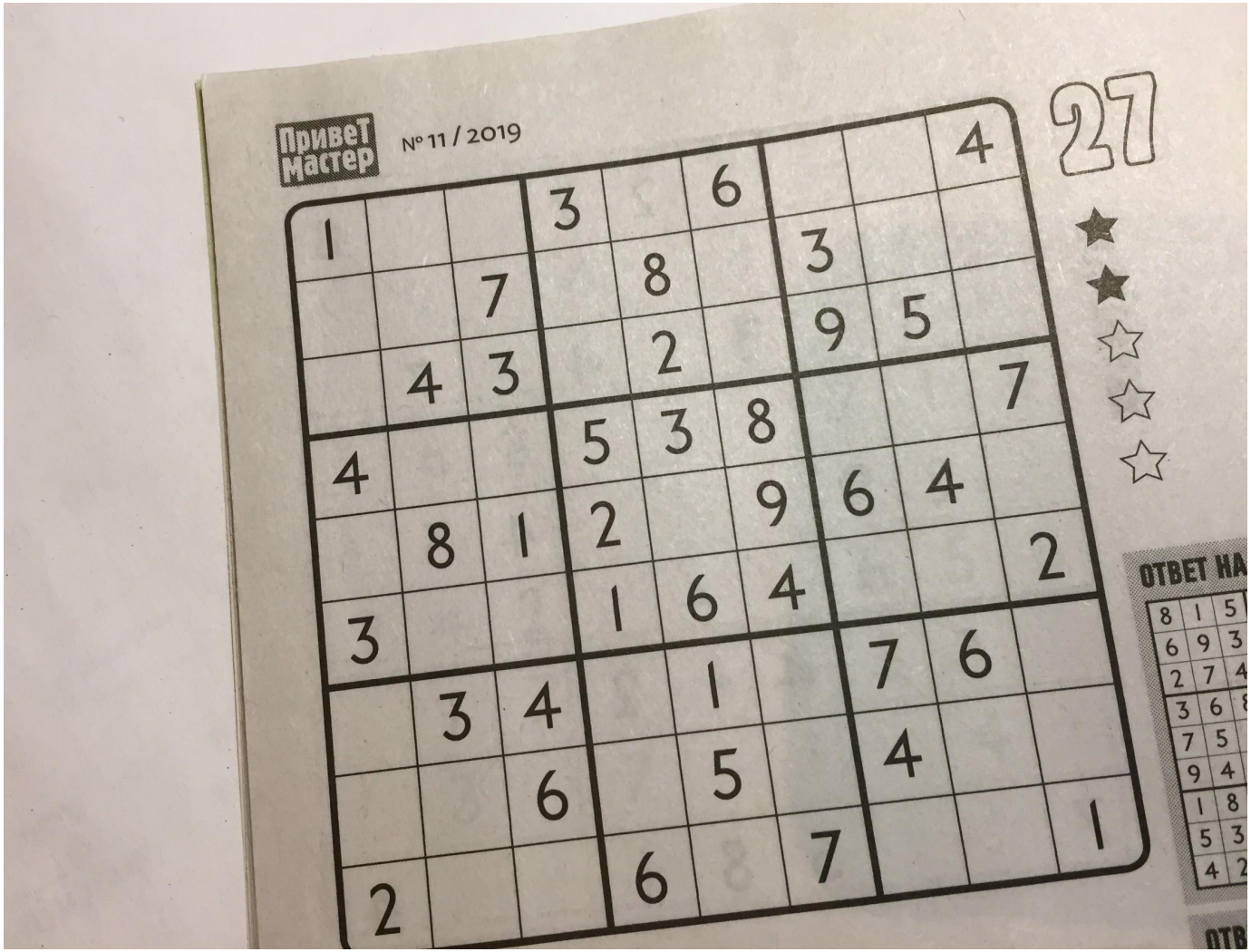
- Please names possible limitations of the developed Sudoku recognition algorithms (separately for two steps)

- We'll try to address all these issues

Sudoku normalizing: issues

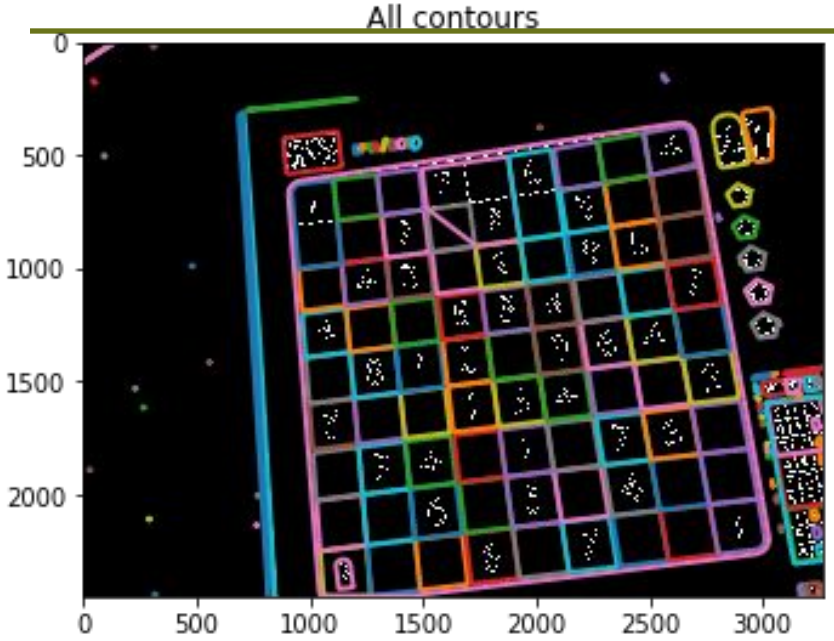


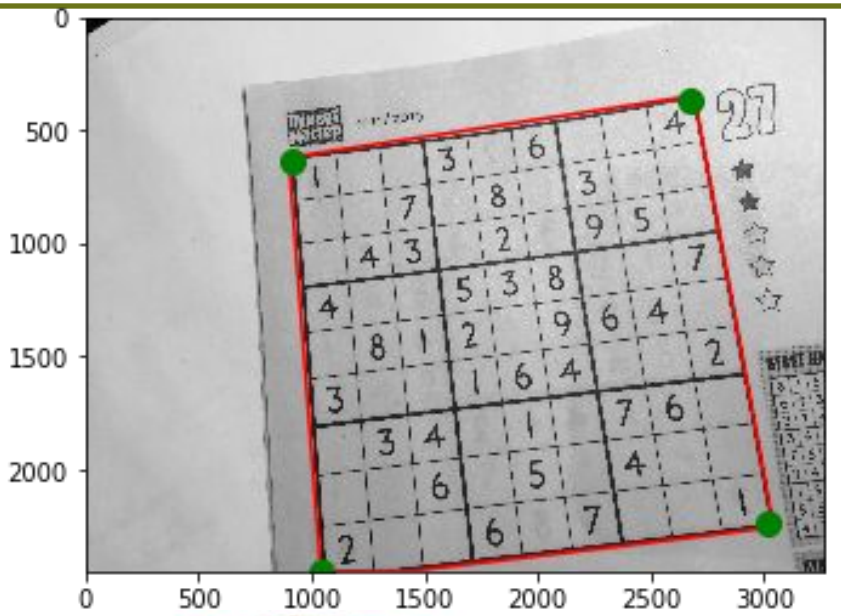
Sudoku normalizing: issues



Not all corners are present!

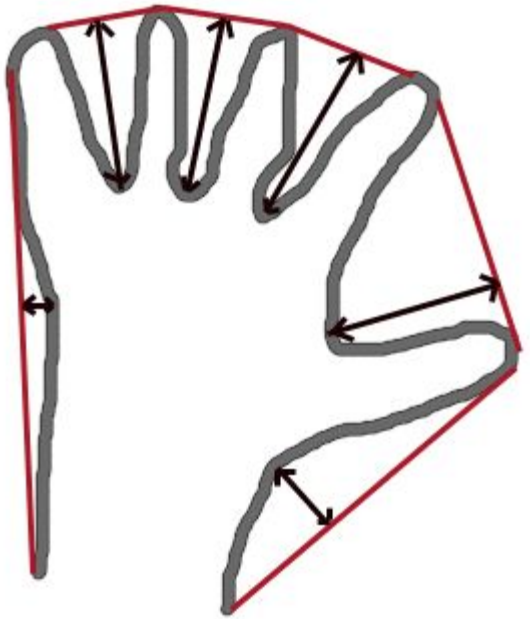
Sudoku normalizing: issues





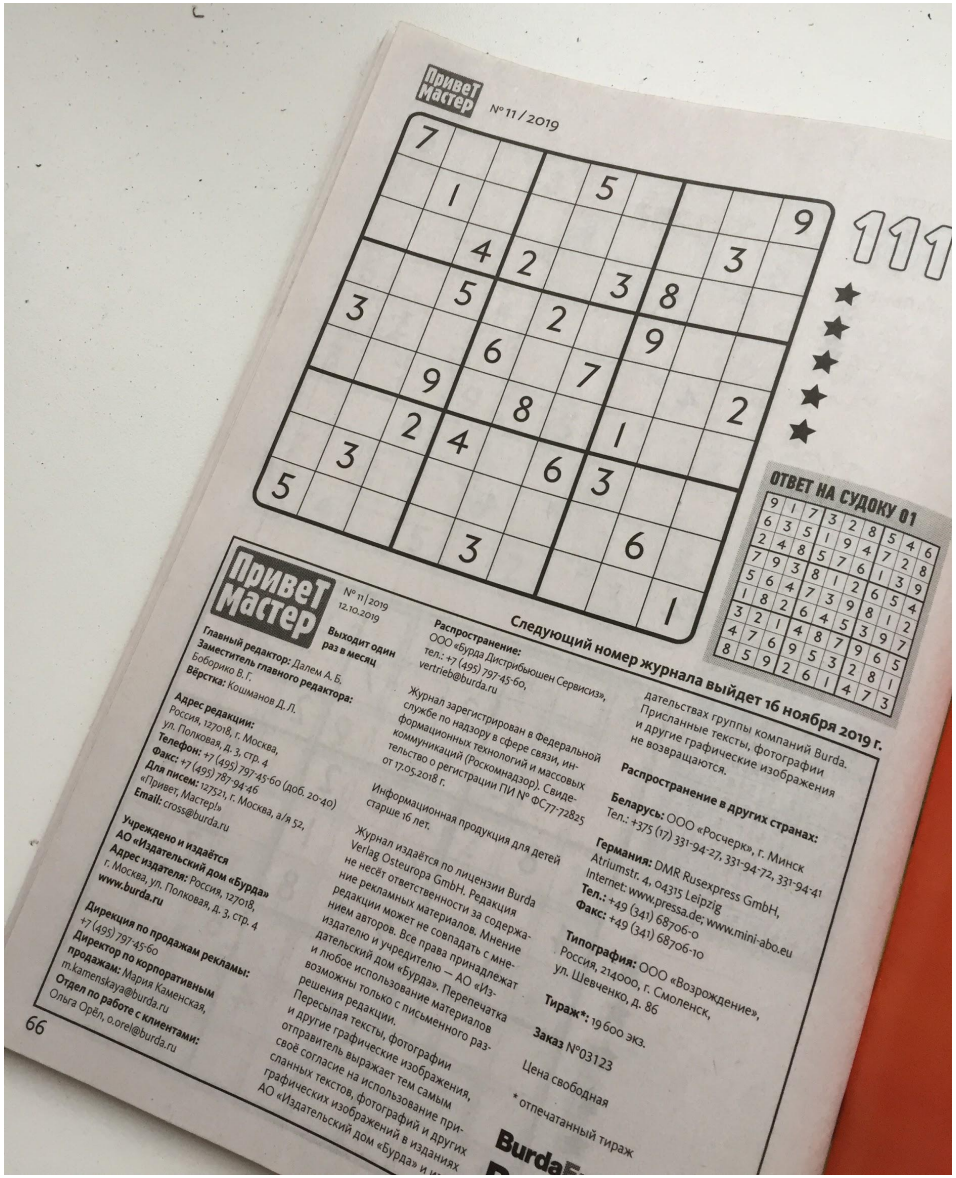
Issue: Not all corners are present!

Possible solution: calculate the convex hull for each contour



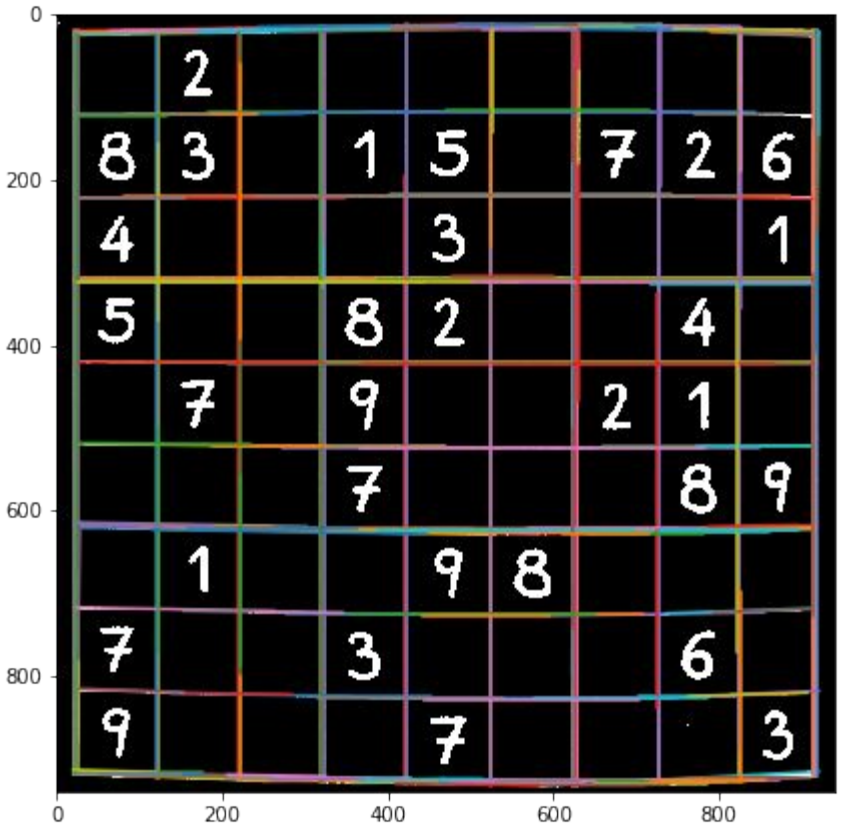
Skoltech
Skolkovo Institute of Science and Technology

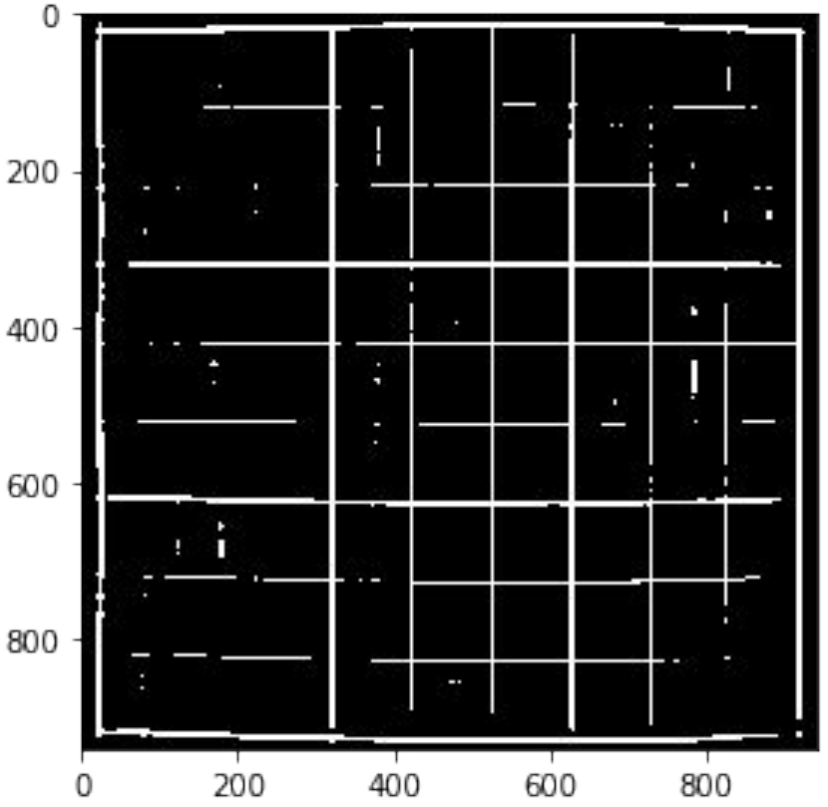
Sudoku normalizing: issues



Skoltech
Skolkovo Institute of Science and Technology

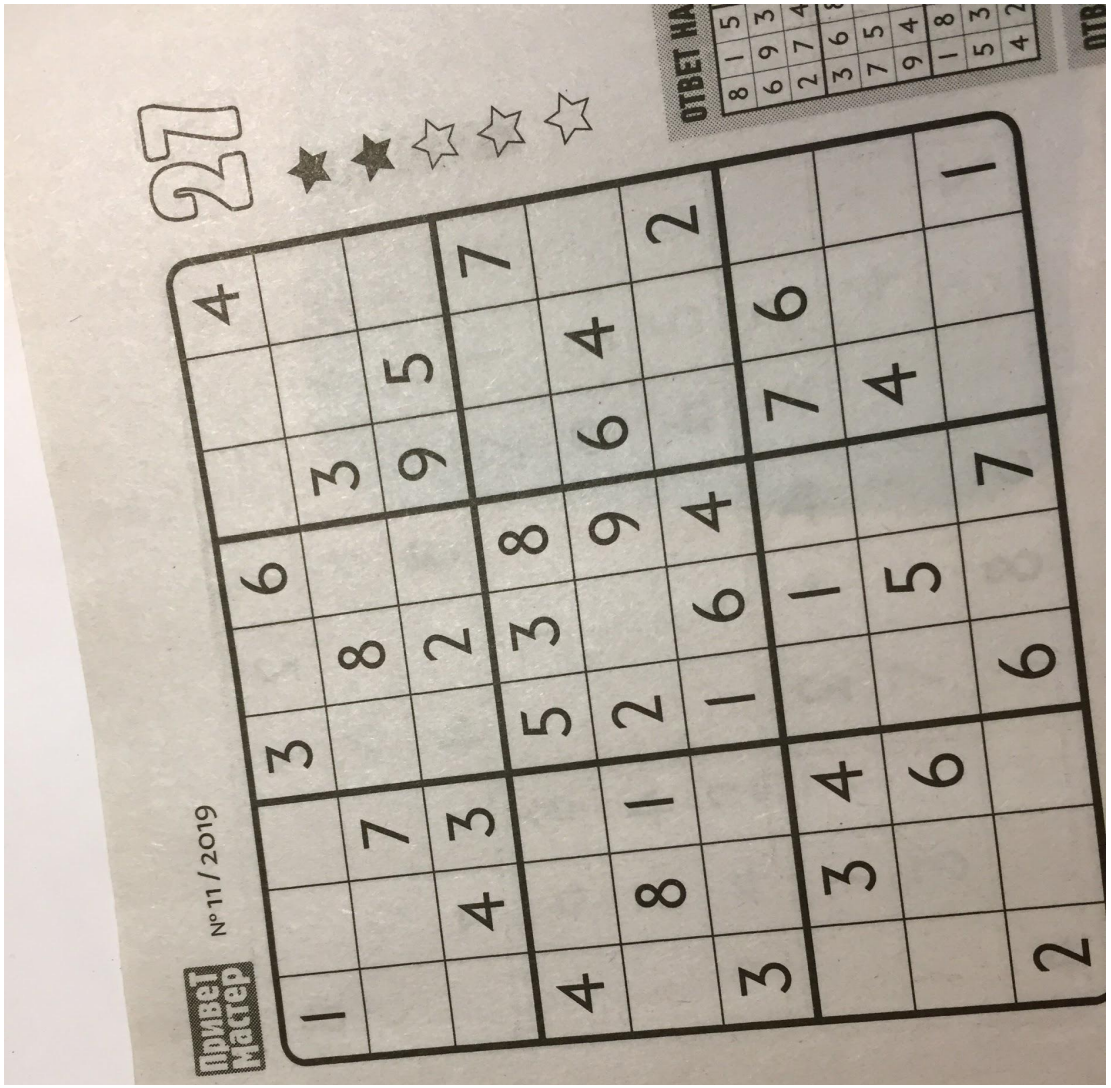
Sudoku normalizing: issues





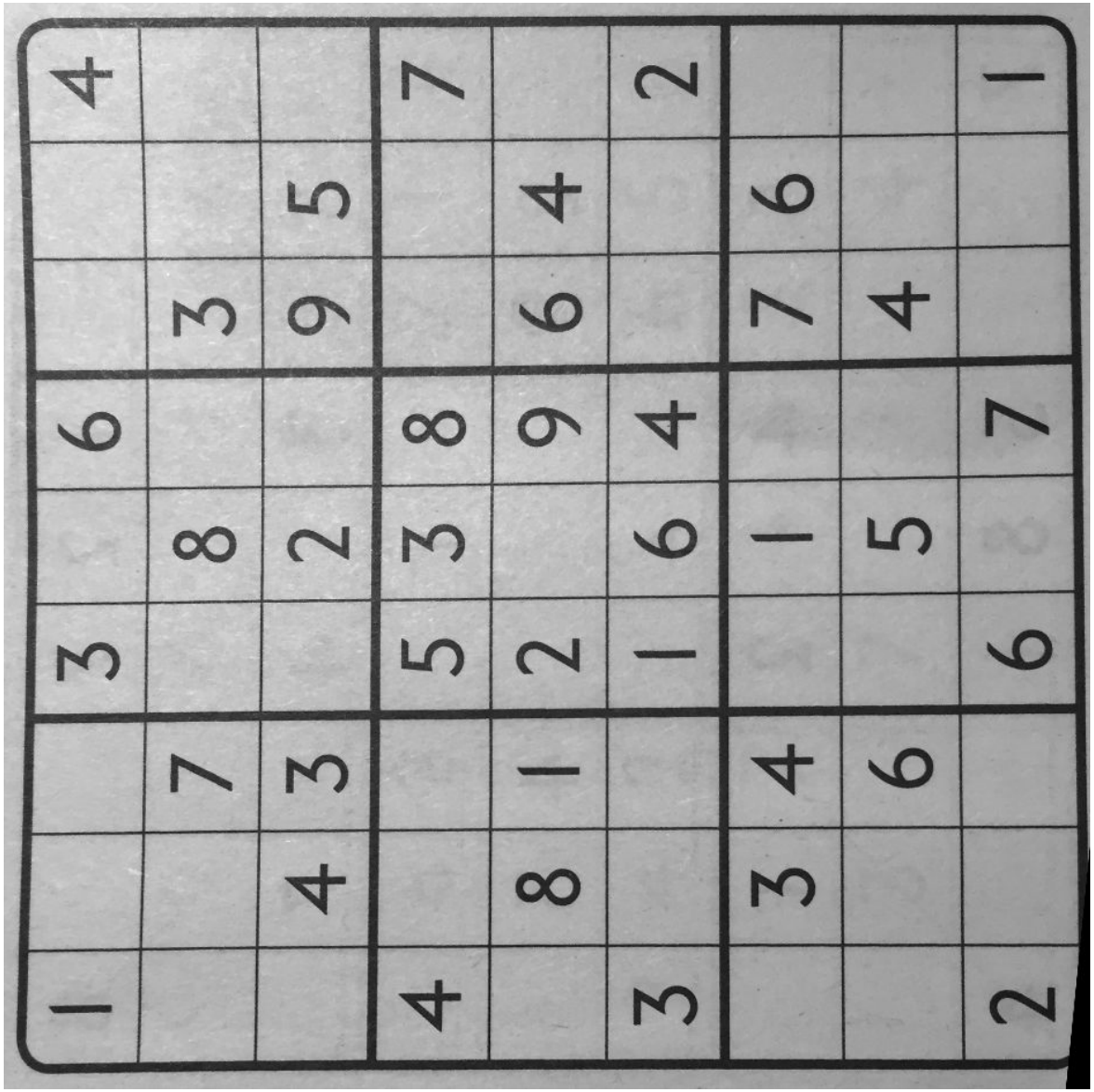
Probabilistic hough or template matching with large cross can be used to select a contour with sudoku

Sudoku normalizing: issues



Is it an issue with the first part of our pipeline?

Sudoku normalizing: issues



Is it an issue with the first part of our pipeline?

In-class 1 follow-up

- **Normalized Cross-Correlation**
- Template matching tips&tricks
- Image Binarization
- Morphological operations
- Nonlinear image transformation

Normalized Cross-Correlation

- Do we have issues with cross correlation?

$$\sum_{x,y} f(x,y)t(x-u,y-v))$$

-

Normalized Cross-Correlation

- Do we have issues with cross correlation?

$$\sum_{x,y} f(x,y)t(x-u,y-v))$$

- Yes!
  - If image intensity varies within the image, it will affect the absolute values of cross-correlation

$$\gamma = \frac{\sum_{x,y}(f(x,y) - \bar{f}_{u,v})(t(x-u,y-v) - \bar{t})}{\sqrt{\sum_{x,y}(f(x,y) - \bar{f}_{u,v})^2 \sum_{x,y}(t(x-u,y-v) - \bar{t})^2}}$$

In-class 1 follow-up

- Normalized Cross-Correlation
- **Template matching tips&tricks**
- Image Binarization
- Morphological operations
- Nonlinear image transformation

Template matching tips&tricks

- Topic to discuss interactively:
  - Binarization
  - Template crop
  - Padding

In-class 1 follow-up

- Normalized Cross-Correlation
- Template matching tips&tricks
- **Image Binarization**
- Morphological operations
- Nonlinear image transformation

Image binarization

- How to select a threshold for image binarization?
  - Manually select a reasonable value for 1 particular image
  - Estimate this value from an image. How can we do this?

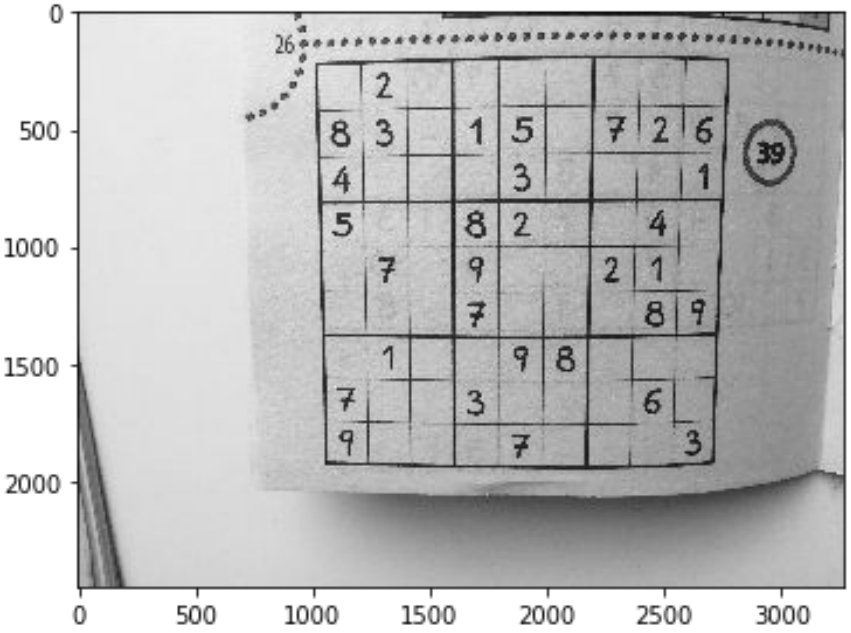
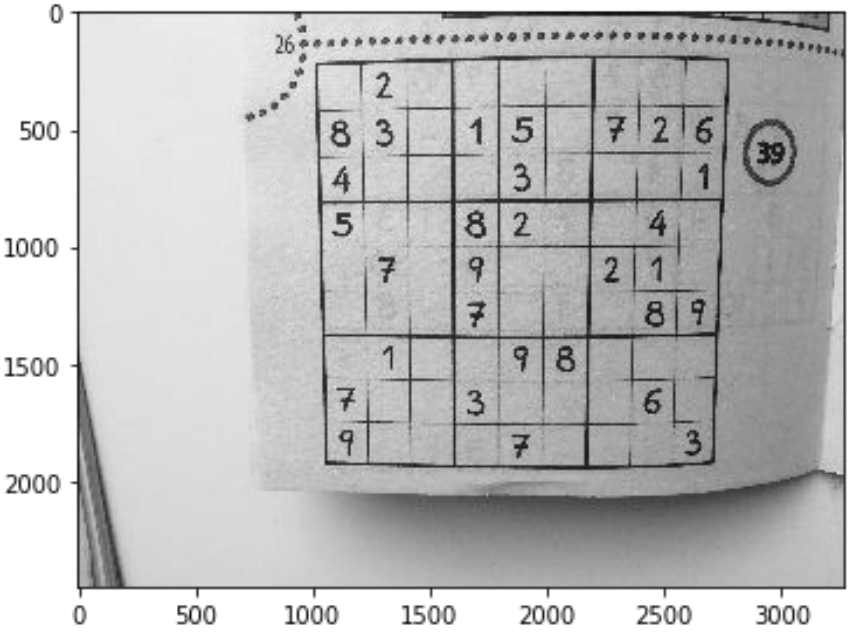


Image binarization

- How to select a threshold for image binarization?
  - Manually select a reasonable value for 1 particular image
  - Estimate this value from an image. How can we do this? Analyzing the intensities!



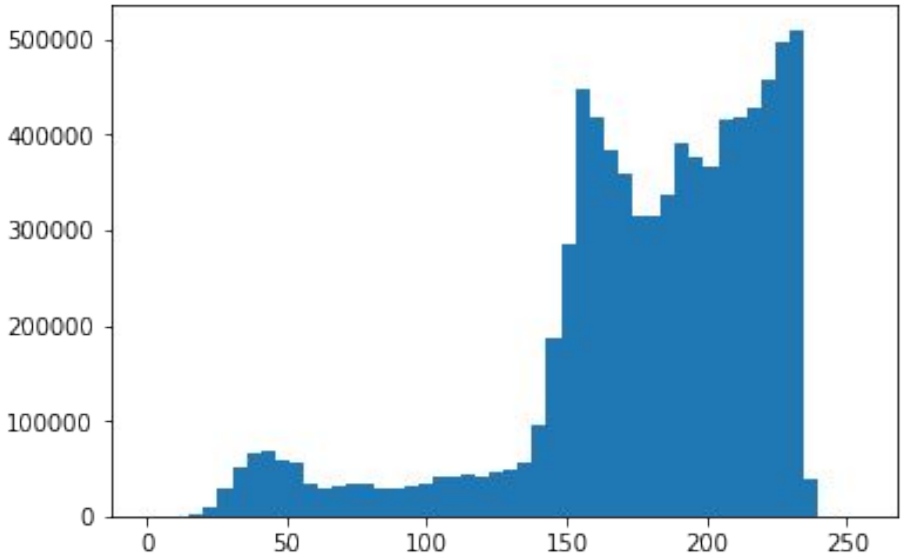
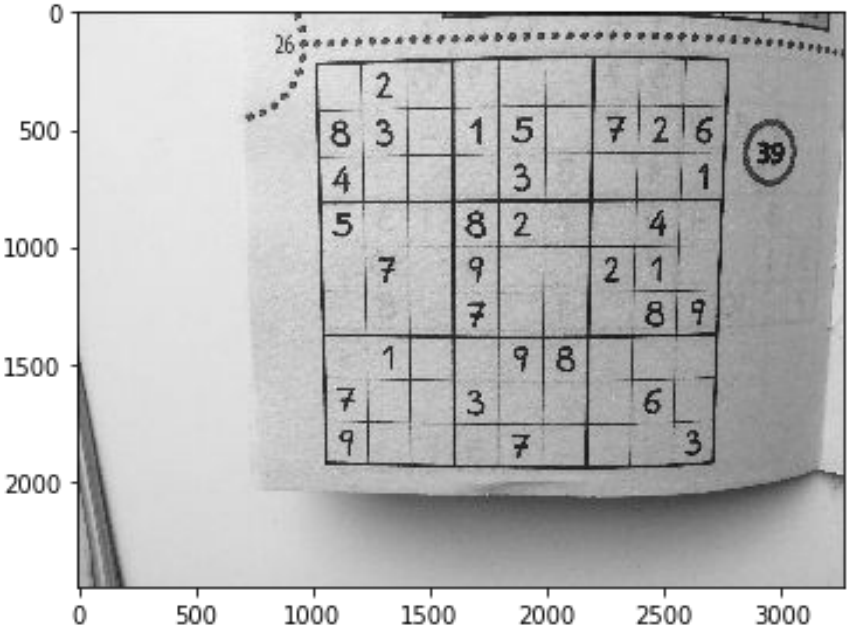


Image binarization

- How to select a threshold for image binarization?
  - Manually select a reasonable value for 1 particular image
  - Estimate this value from an image. How can we do this? Analyzing the intensities!



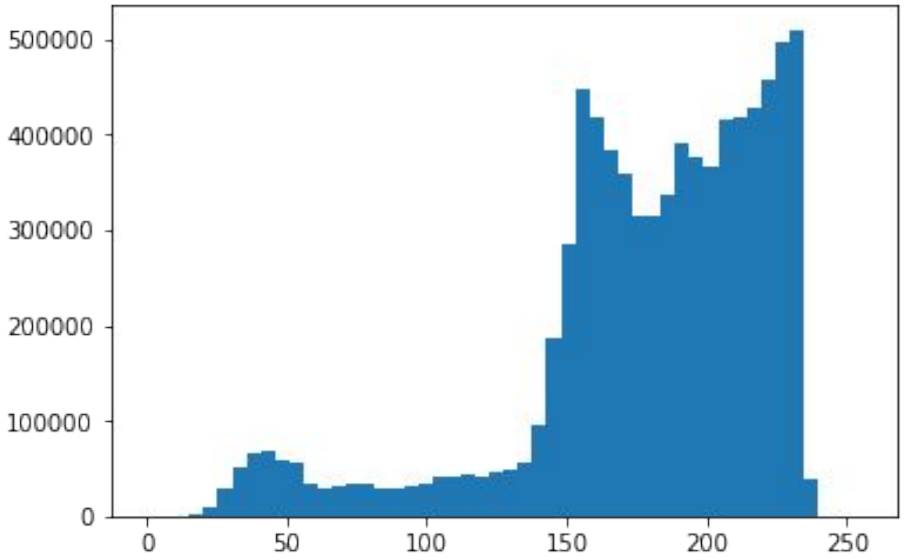


Image binarization - Otsu

- The idea is to find the threshold maximizing intraclass variance for “black” and “white” clases of pixels

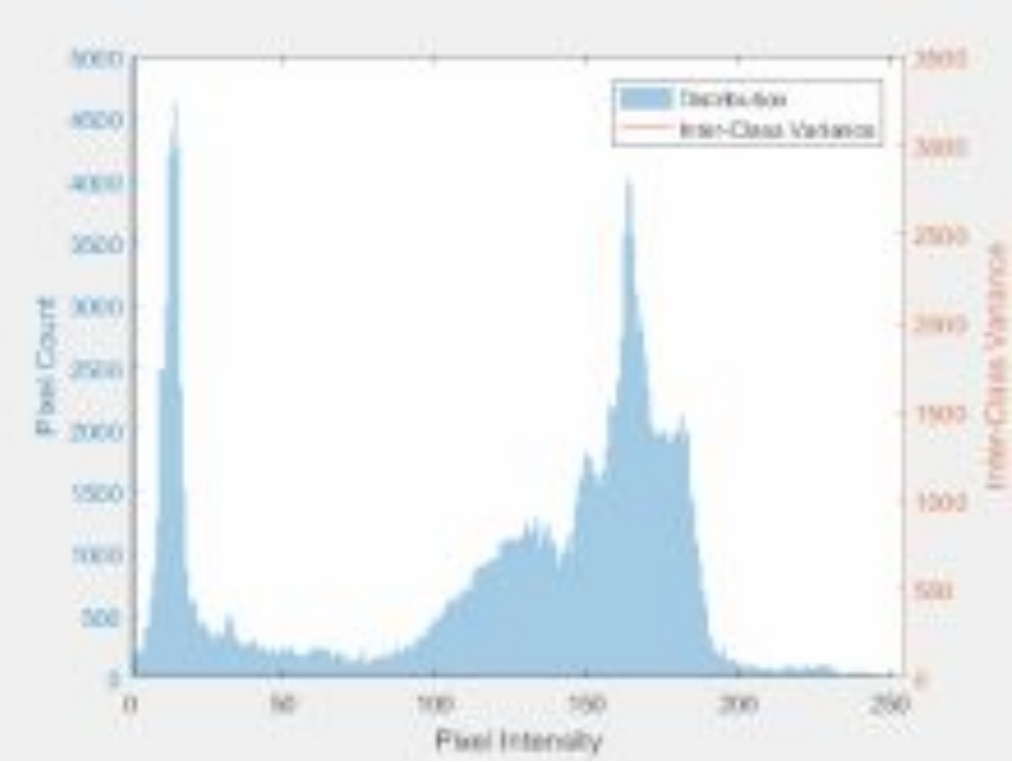
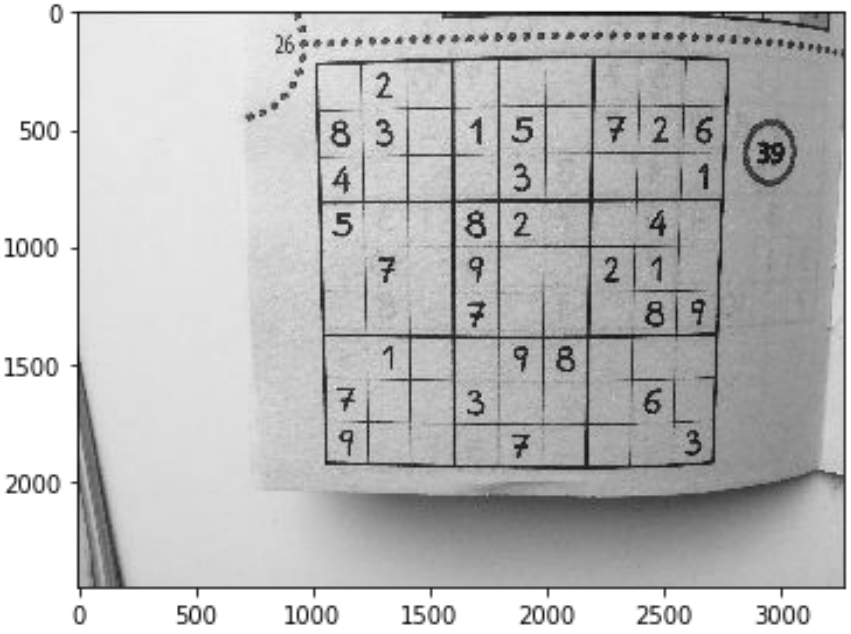


Image binarization

- How to select a threshold for image binarization?
  - Manually select a reasonable value for 1 particular image
  - Estimate this value from an image. How can we do this? Analyzing the intensities!



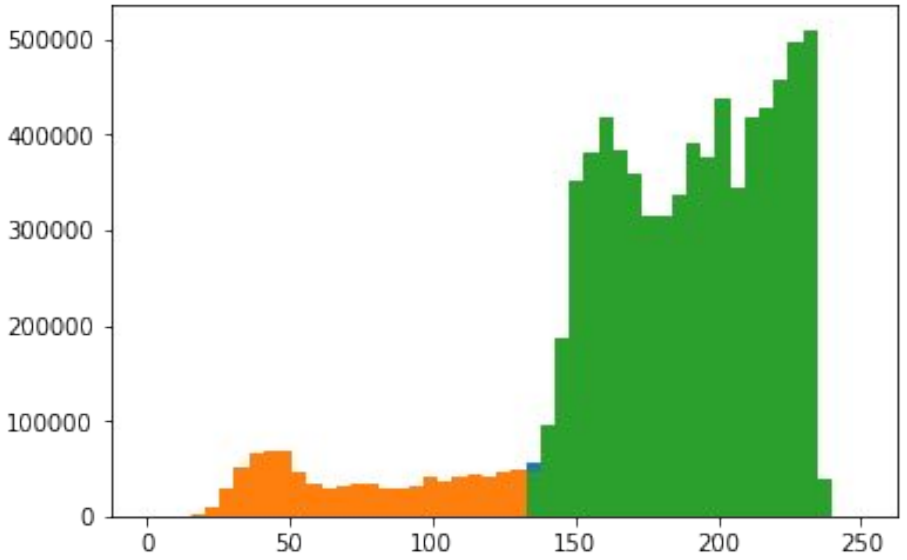
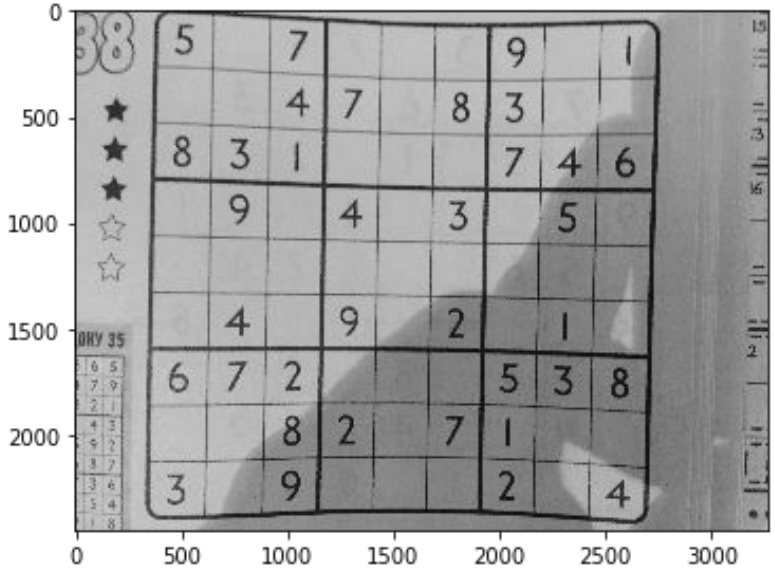


Image binarization

- What are the drawbacks of this approach?
- Exercise: binarize yet another sudoku image using Otsu (skimage.filters.threshold_otsu)



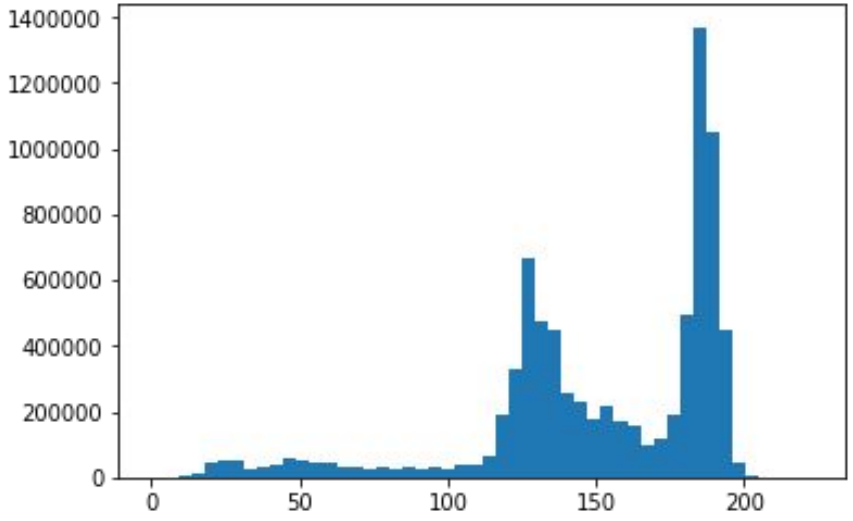
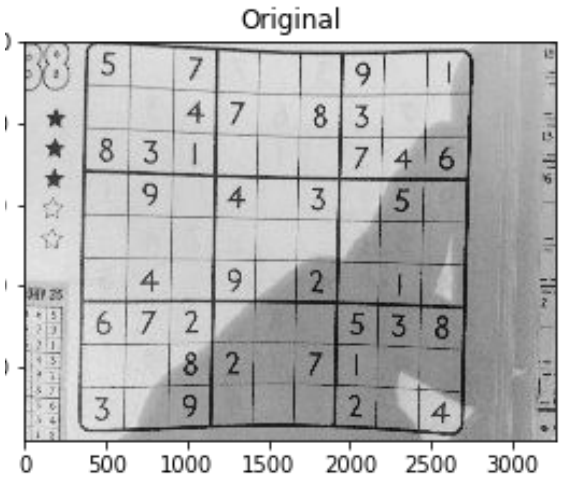
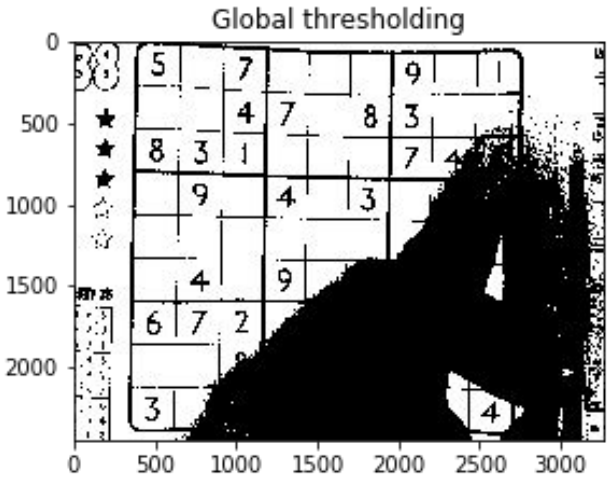
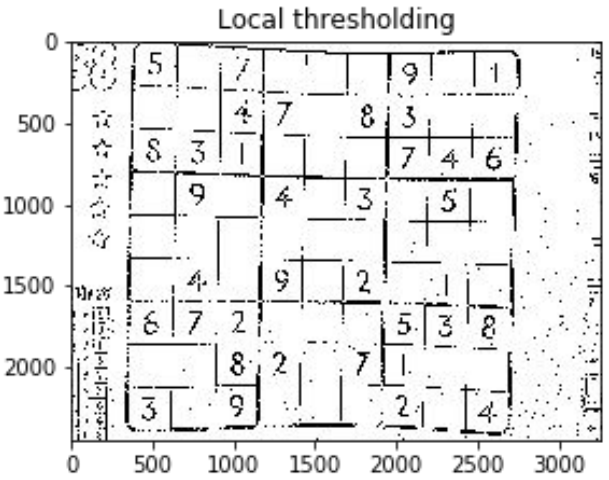


Image binarization

- What are the drawbacks of this approach?
- Exercise: binarize yet another sudoku image using Otsu (skimage.filters.threshold_otsu)
- Try skimage.filters.threshold_local!







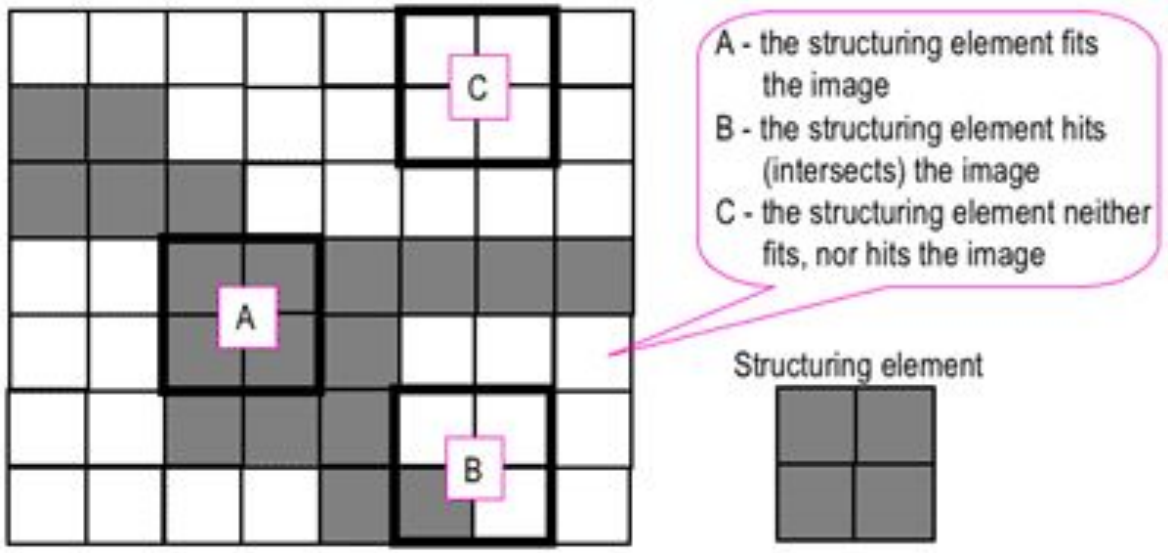
In-class 1 follow-up

- Normalized Cross-Correlation
- Template matching tips&tricks
- Image Binarization
- **Morphological operations**
- Nonlinear image transformation

Morphology

Besides convolution, there are other useful windowed operations. The pipeline is simple

1. Convolve your image with a structuring elements (a simple predefined kernel such as 3x3 star or square)
2. Calculate min / max value for each window.

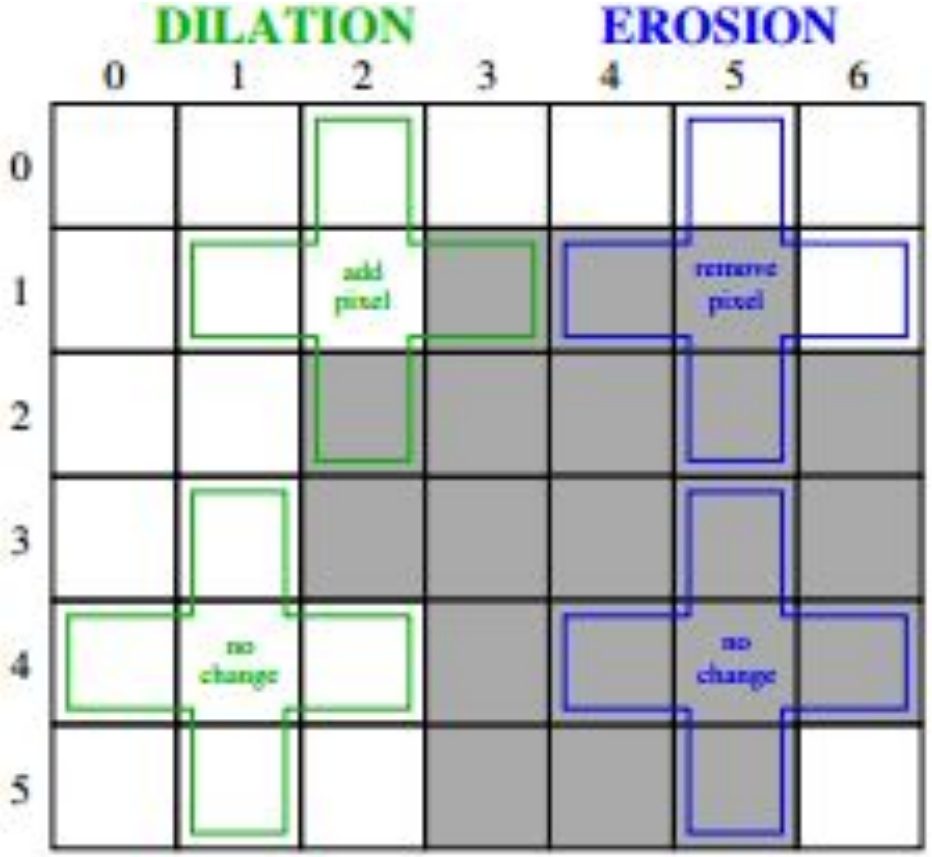


Source

Morphology

Binary dilation - taking the maximal value in a window.

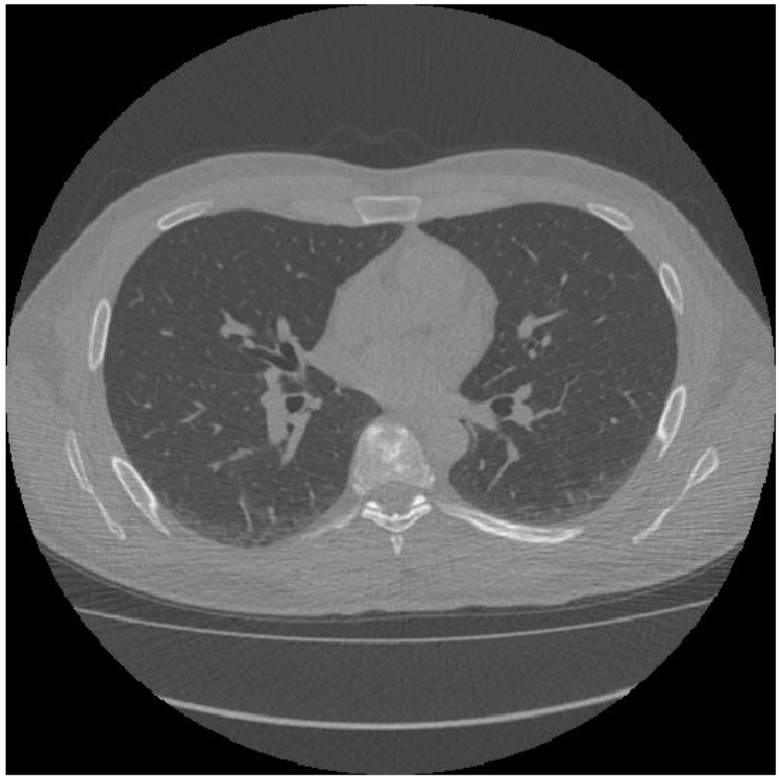
Binary erosion - taking the minimal value in a window.



Morphology

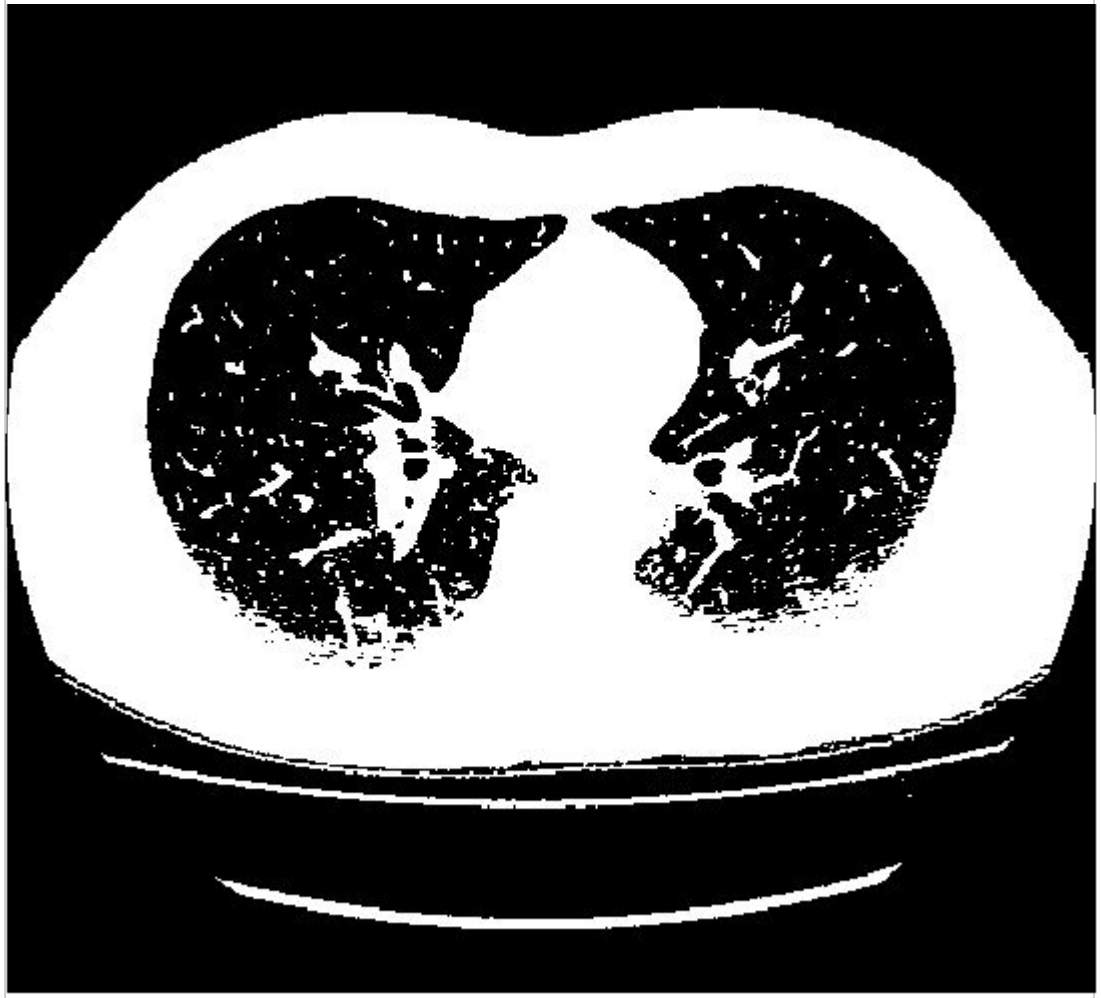
Hands-on

1. Binarize the image using Otsu thresholding.
2. Apply dilation to remove trachea (skimage.morphology.dilation).
3. Apply erosion ro remove the table.
4. How can we keep the table thin but fill lungs completely?



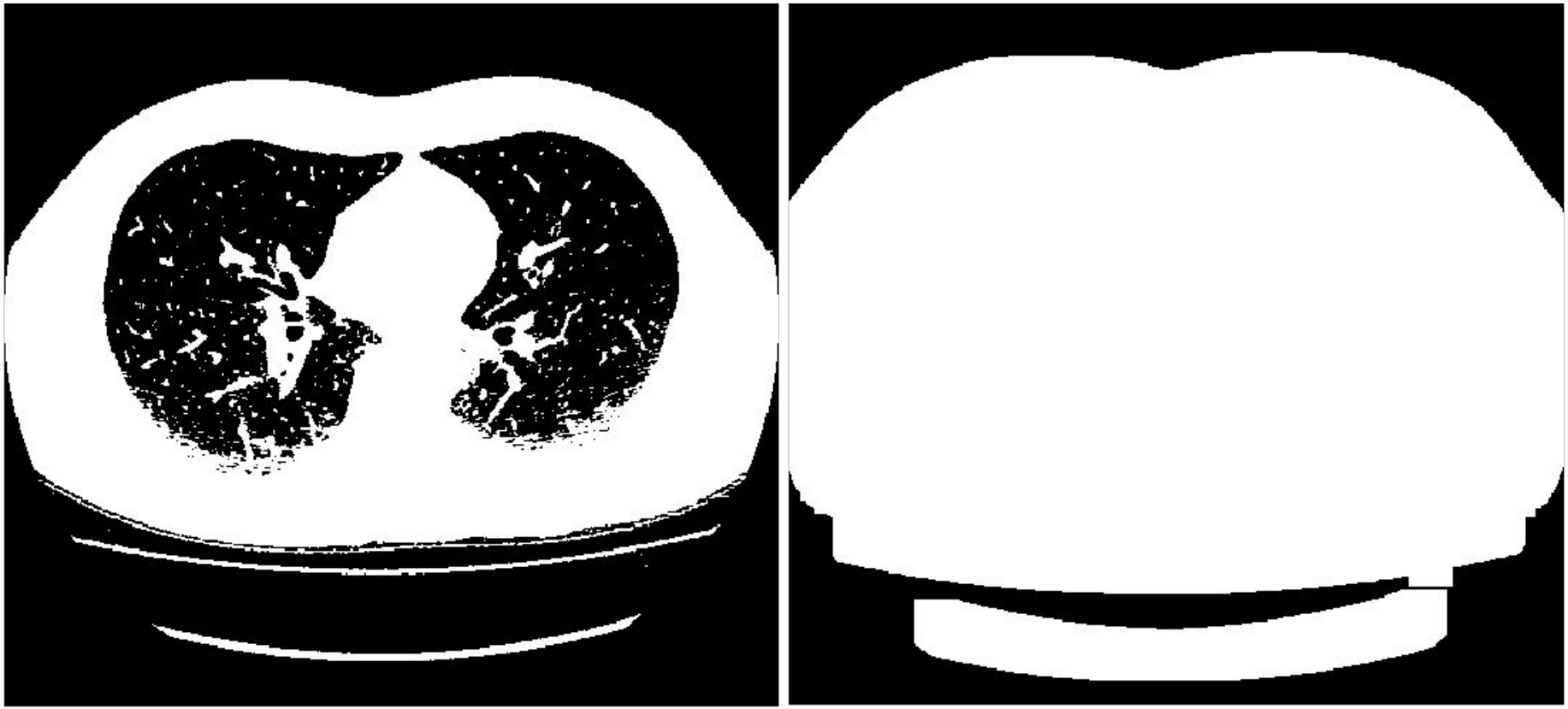
Morphology

Black pixels - air; White ones - body, table



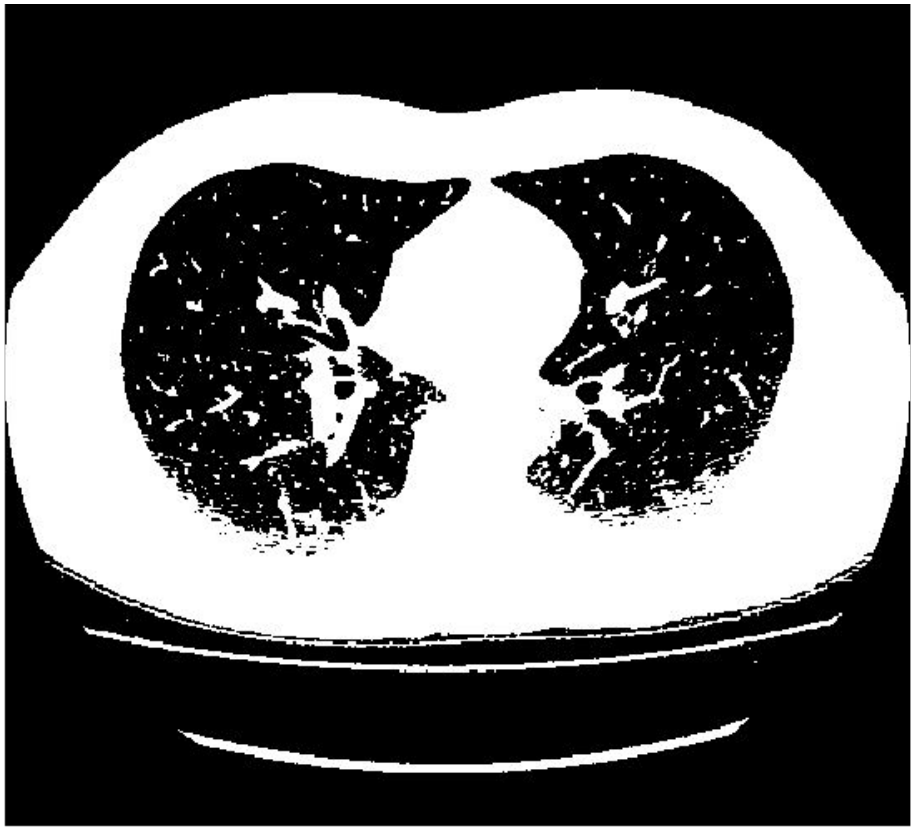
Morphology: dilation

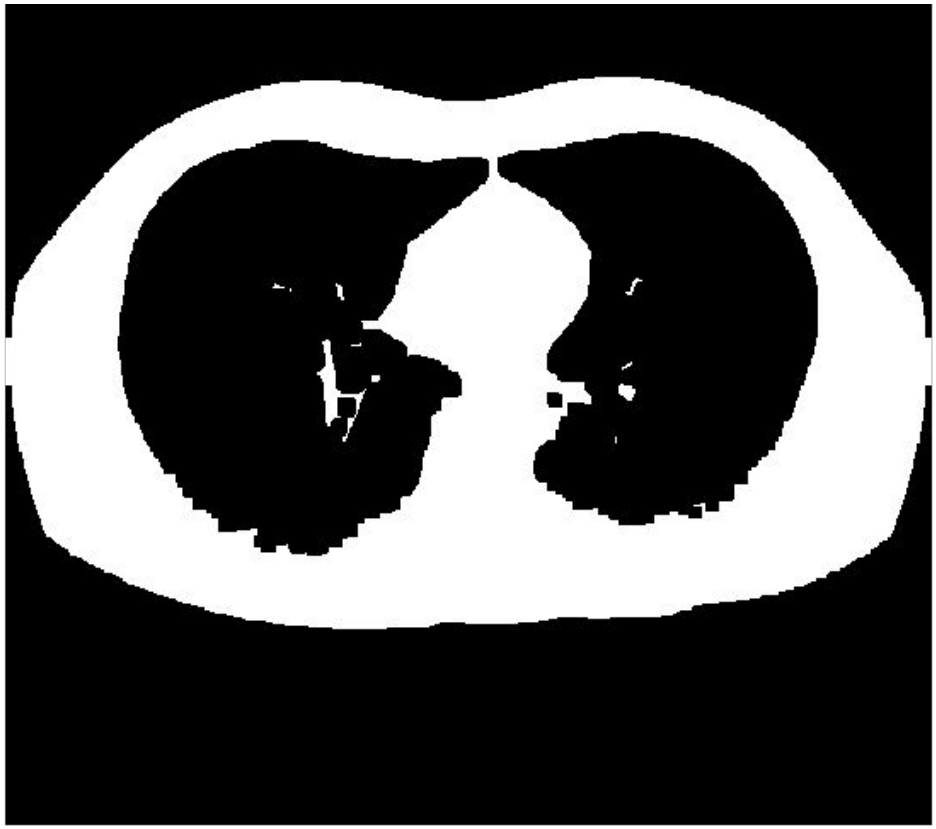
Taking maximum in a window effectively makes all pixels in a neighbourhood of a white pixel also white.



Morphology: erosion

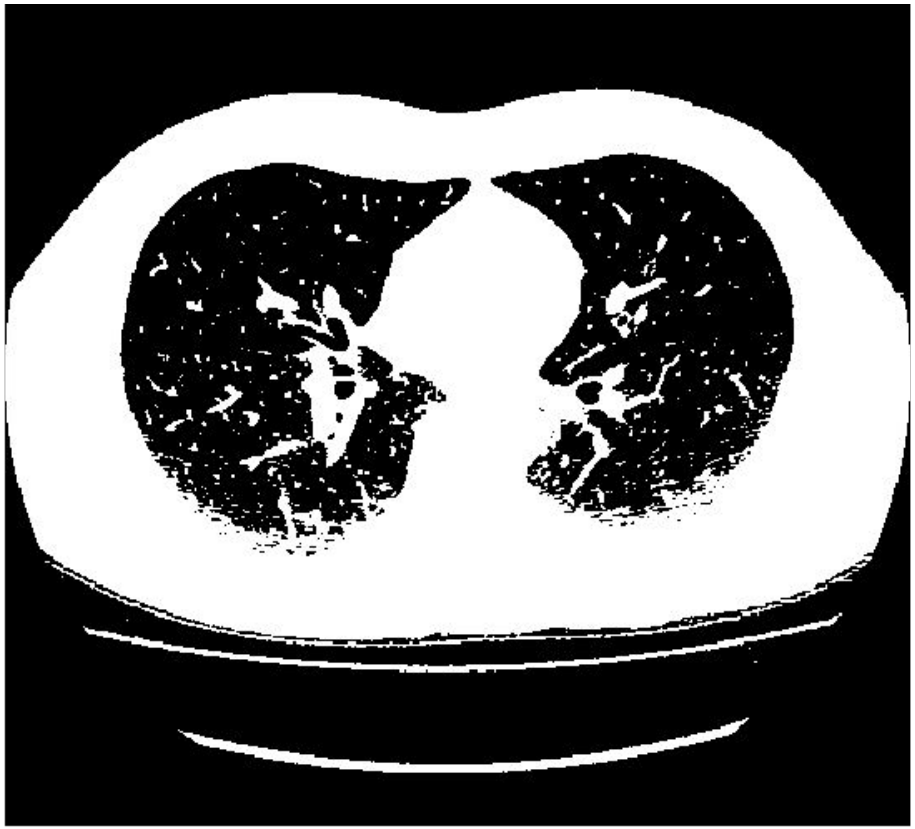
Taking minimum in a window will do the opposite.

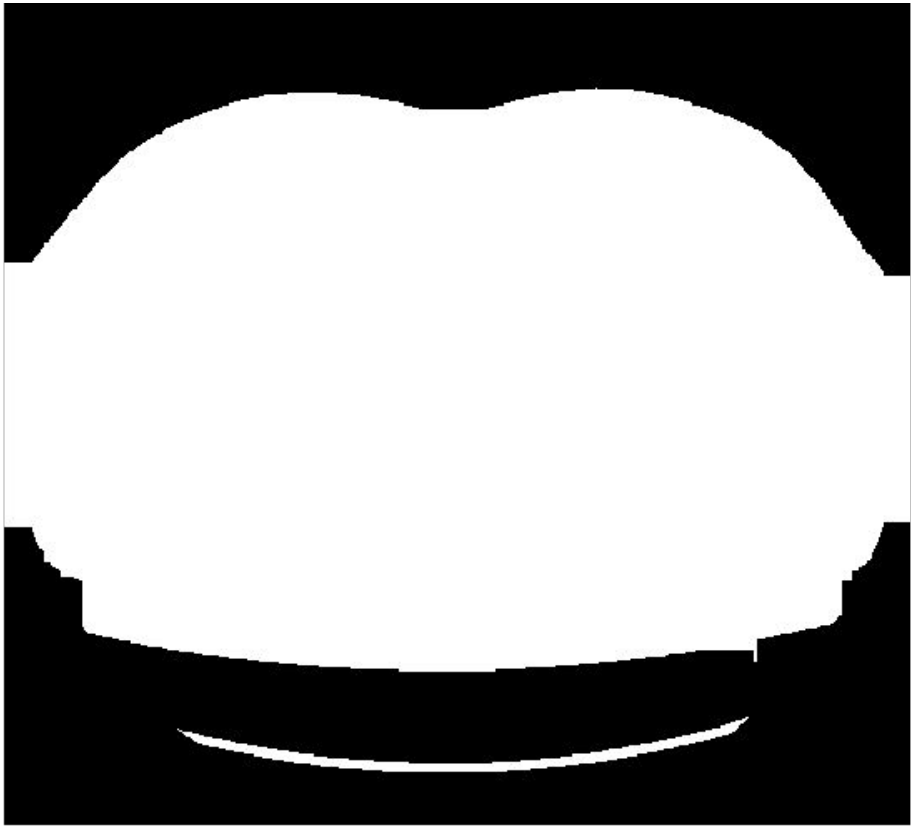




Morphology: closing

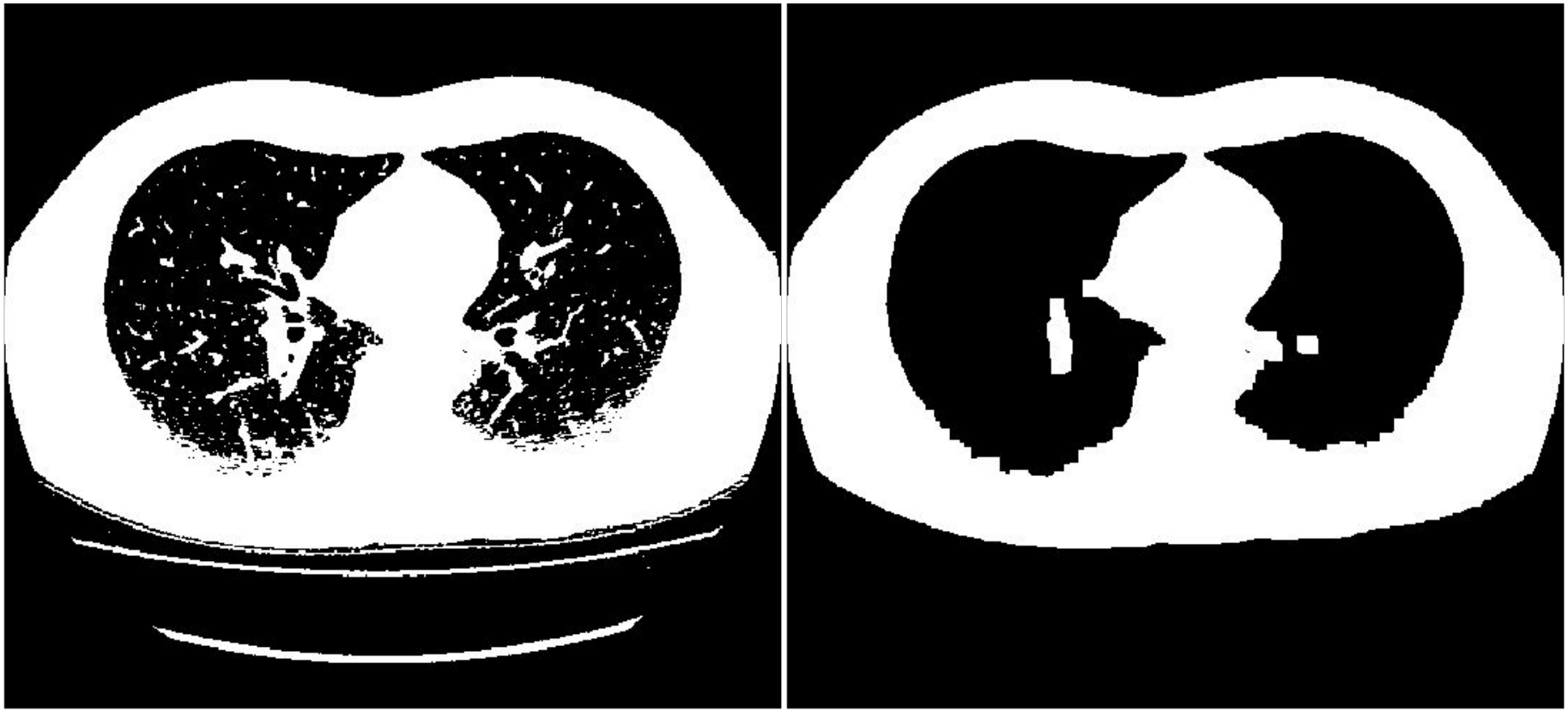
Applying dilation and erosion will close small holes.





Morphology: opening

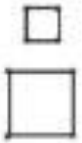
Applying erosion and dilation will remove small components.



In-class 1 follow-up

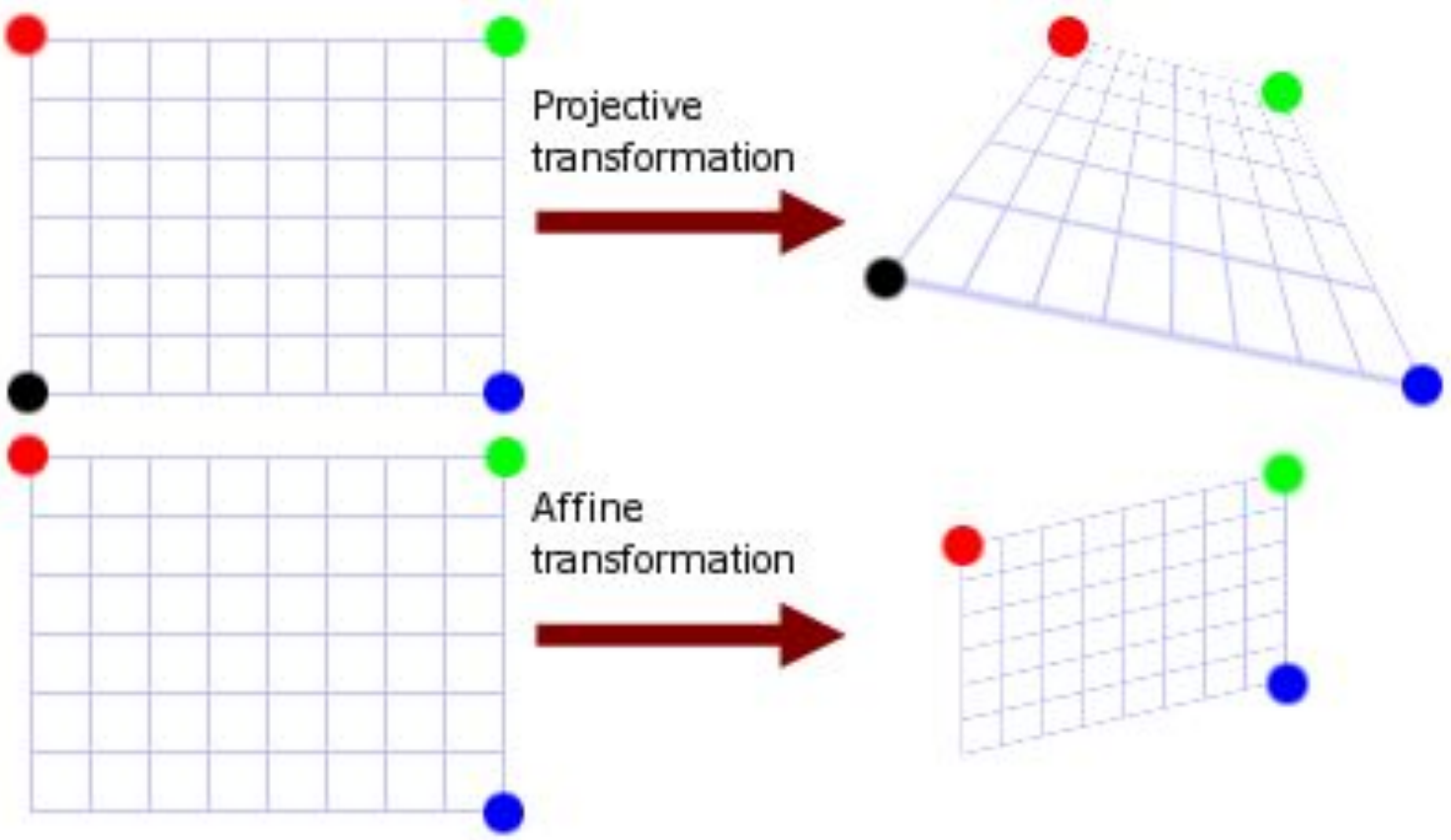
- Normalized Cross-Correlation
- Template matching tips&tricks
- Image Binarization
- Morphological operations
- **Nonlinear image transformation**

Linear transforms

Group	Matrix	Distortion	Invariant properties
Projective 8 dof	$\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$		Concurrency, collinearity, **order of contact**: intersection (1 pt contact); tangency (2 pt contact); inflections (3 pt contact with line); tangent discontinuities and cusps. cross ratio (ratio of ratio of lengths).
Affine 6 dof	$\begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$		Parallelism, ratio of areas, ratio of lengths on collinear or parallel lines (e.g. midpoints), linear combinations of vectors (e.g. centroids). The line at infinity, $\mathbf{l}_\infty$.
Similarity 4 dof	$\begin{bmatrix} sr_{11} & sr_{12} & t_x \\ sr_{21} & sr_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$		Ratio of lengths, angle. The circular points, $\mathbf{I}, \mathbf{J}$ (see section 2.7.3).
Euclidean 3 dof	$\begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$		Length, area

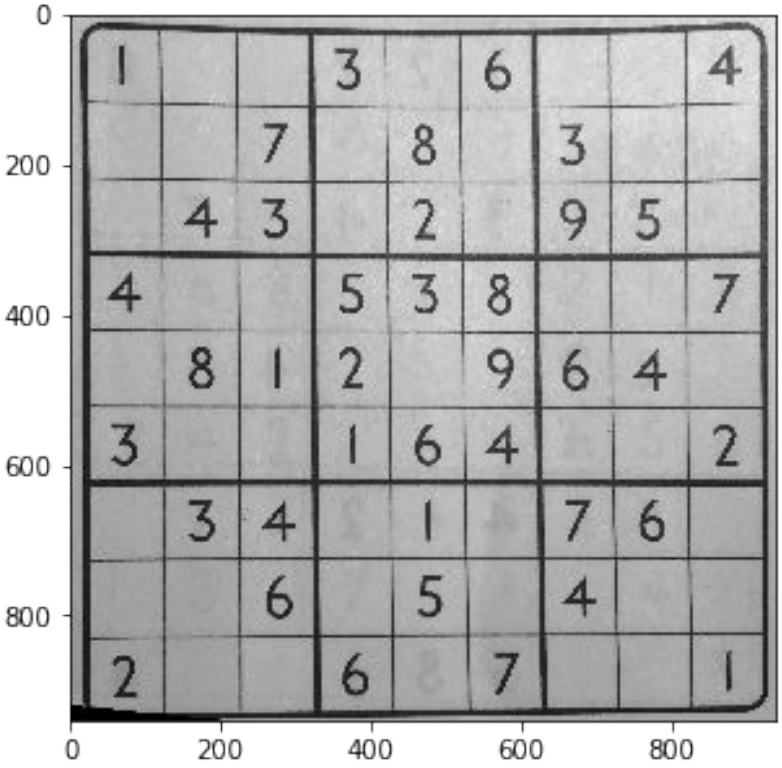
Source

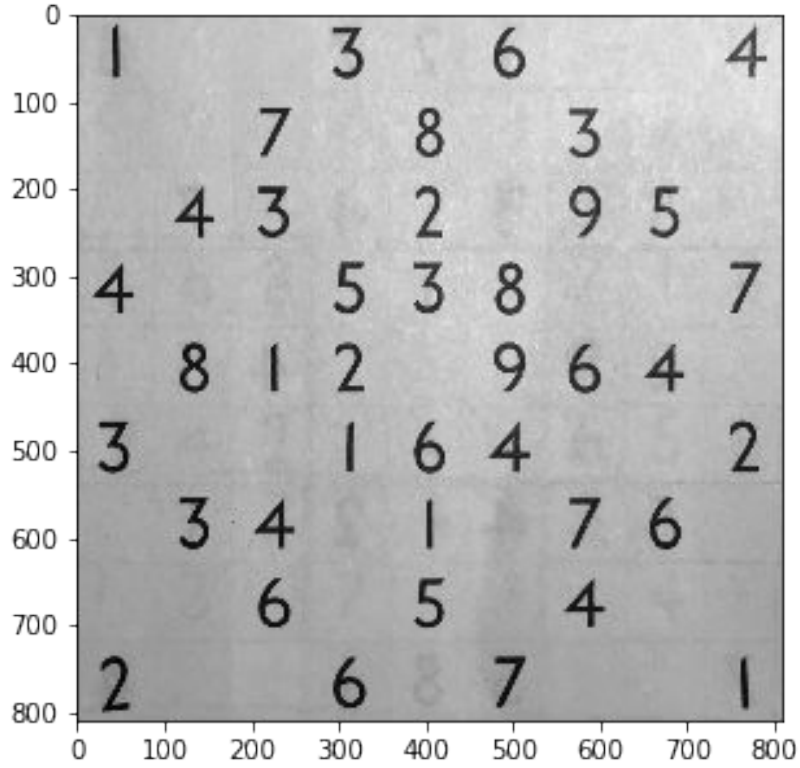
Projective transform



Is possible to fit a nonlinear transform?

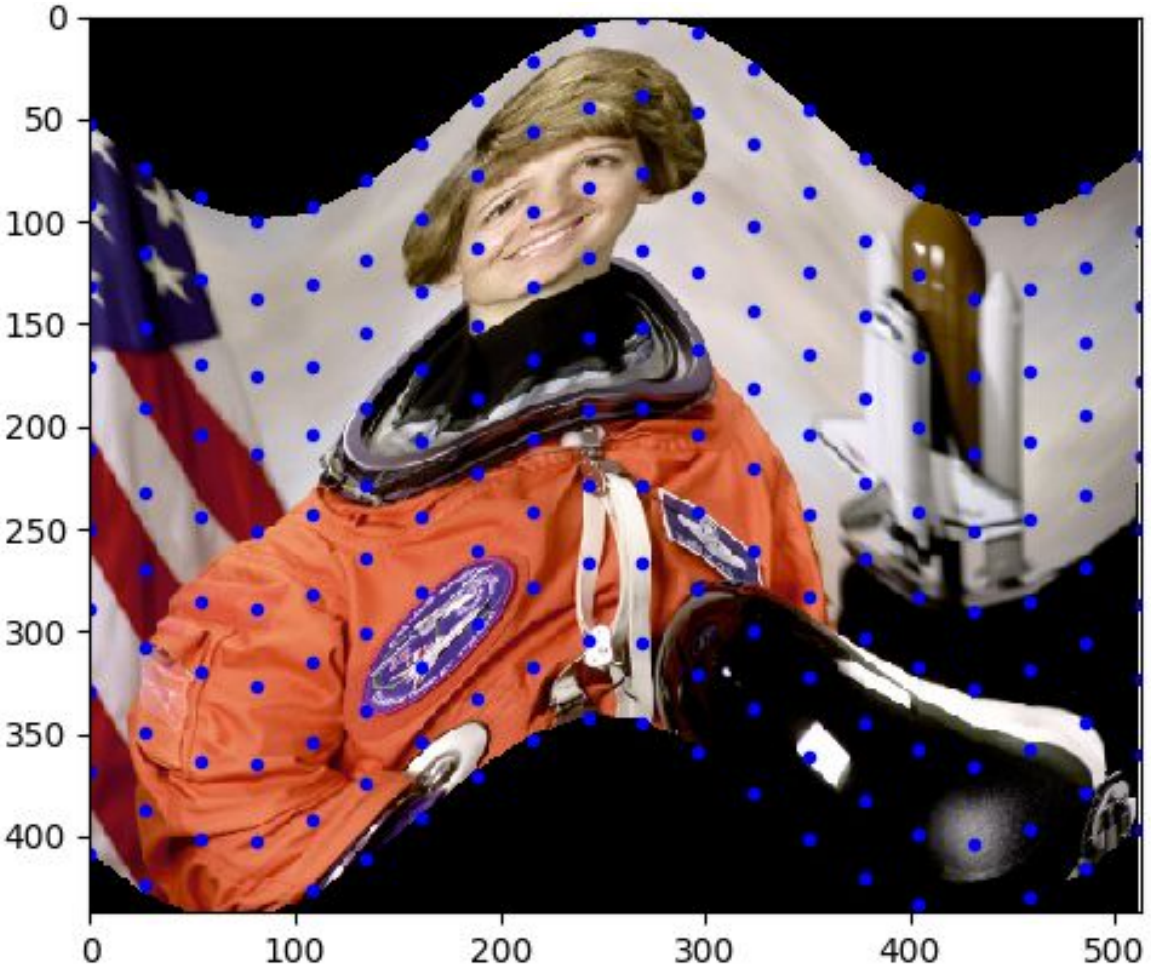
- The simplest solution is to use piecewise linear transforms



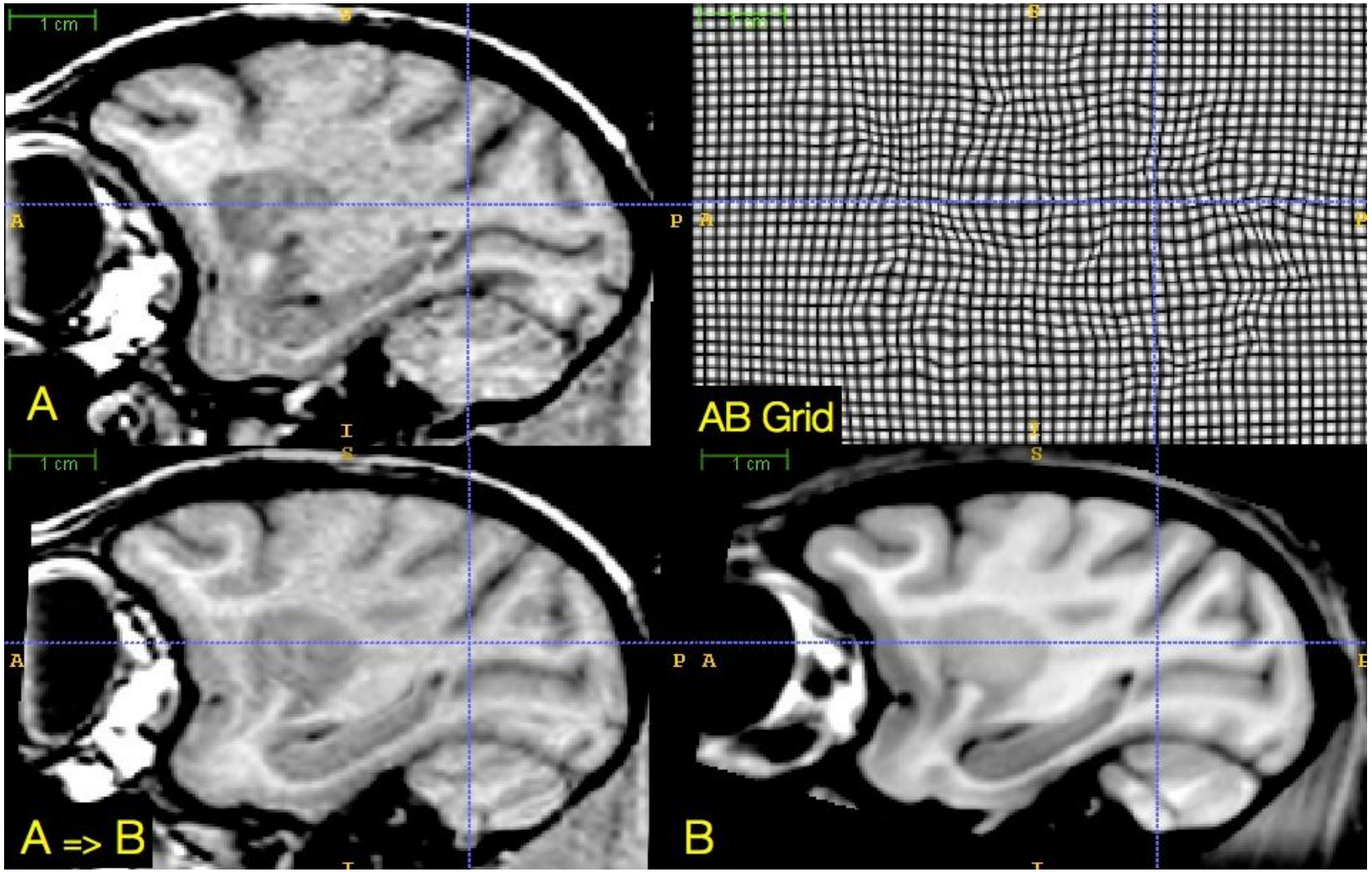


Is possible to fit a nonlinear transform?

- The simplest solution is to use piecewise linear transforms, see this skimage example

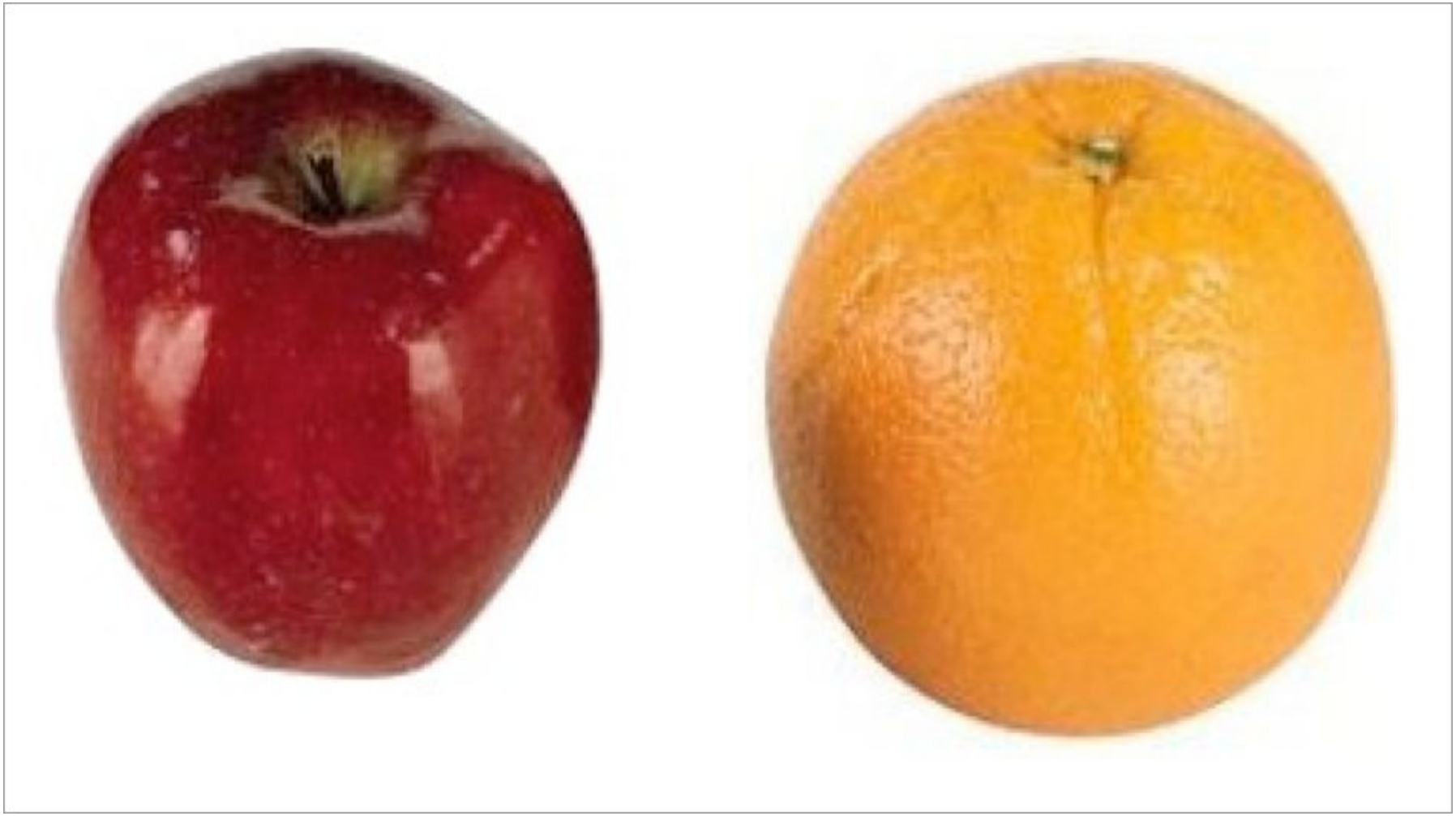


Is possible to fit a nonlinear transform?



Source: ANTs tutorial

Is possible to fit a nonlinear transform?



Source: ANTs tutorial

Is possible to fit a nonlinear transform?

Step 1: linear (affine, actually) transform

Source: ANTs tutorial

Is possible to fit a nonlinear transform?

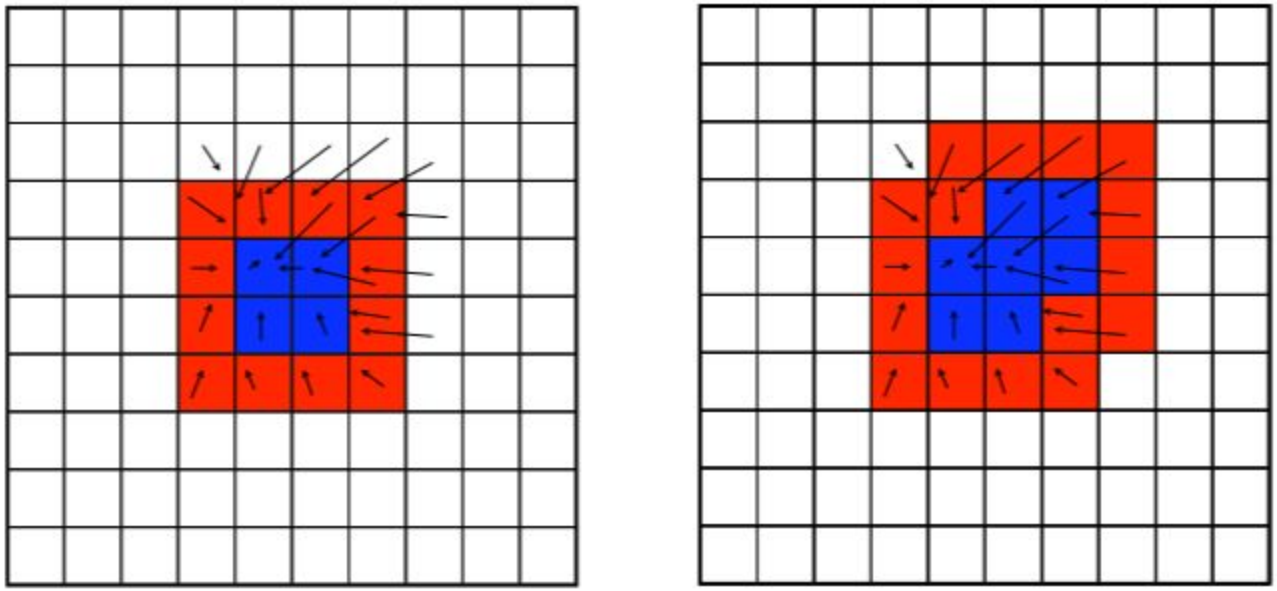
Step 2: nonlinear transform

Source: ANTs tutorial

Is possible to fit a nonlinear transform?

Step 2: nonlinear transform.

There are multiple options how to map images nonlinearly. But the most generic one is to use some diffeomorphisms (differentiable map with differentiable inverse).

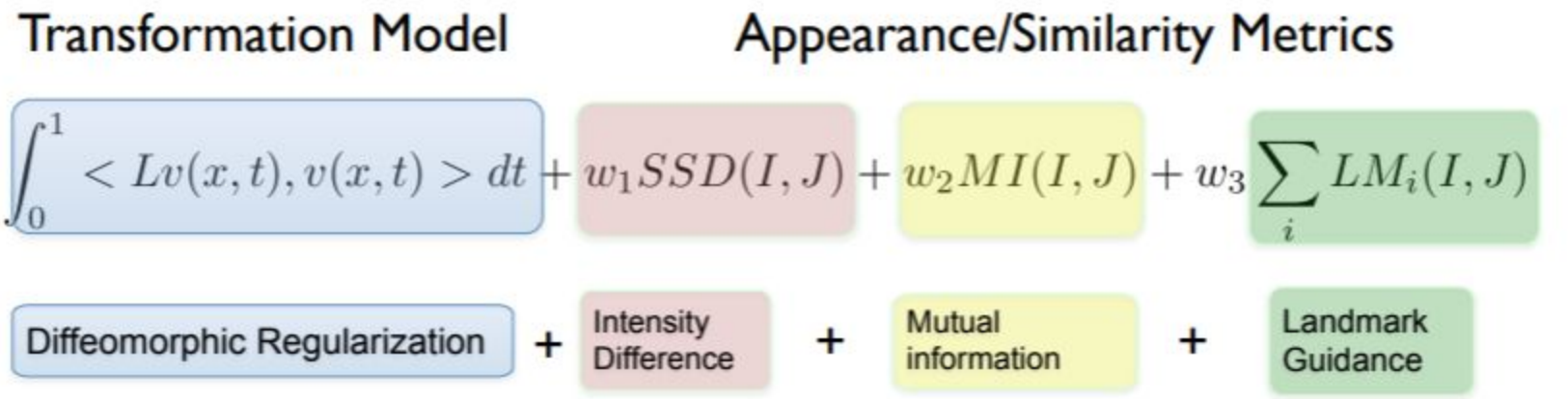


Source: ANTs tutorial

Is possible to fit a nonlinear transform?

Step 2: nonlinear transform.

The core idea is to parametrize the transform (e.g. via splines) and control its smoothness.

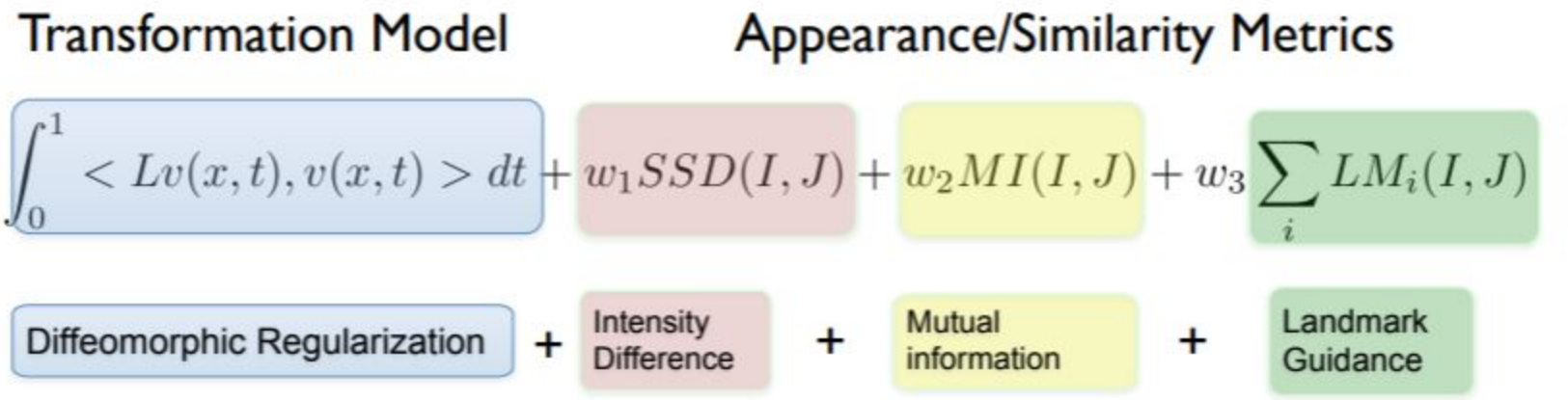


Source: ANTs tutorial

Is possible to fit a nonlinear transform?

Step 2: nonlinear transform.

The core idea is to parametrize the transform (e.g. via splines) and control its smoothness.



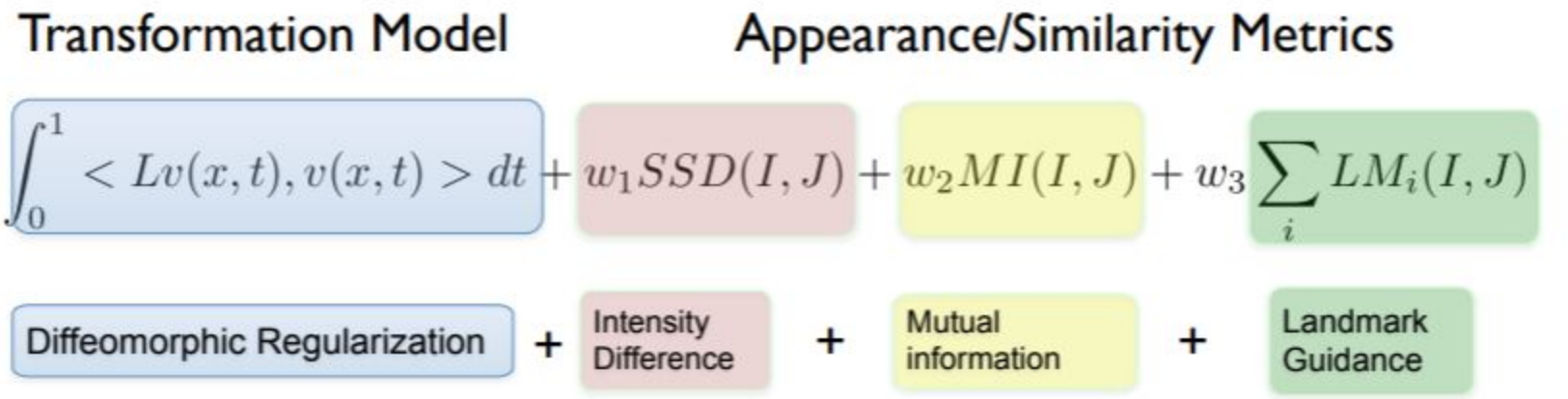
However, these methods are rarely used outside medical CV. Can you guess why?

Source: ANTs tutorial

Is possible to fit a nonlinear transform?

Step 2: nonlinear transform.

The core idea is to parametrize the transform (e.g. via splines) and control its smoothness.



However, these methods are rarely used outside medical CV. Can you guess why? It's too slow! Processing can takes hours for large images

Source: ANTs tutorial

Projective transform

Exercise 3:

- Find edges & lines
- Plot lines
- Select points to fit ProjectiveTransform via **RANSAC**

RANSAC ?!

RANSAC algorithm

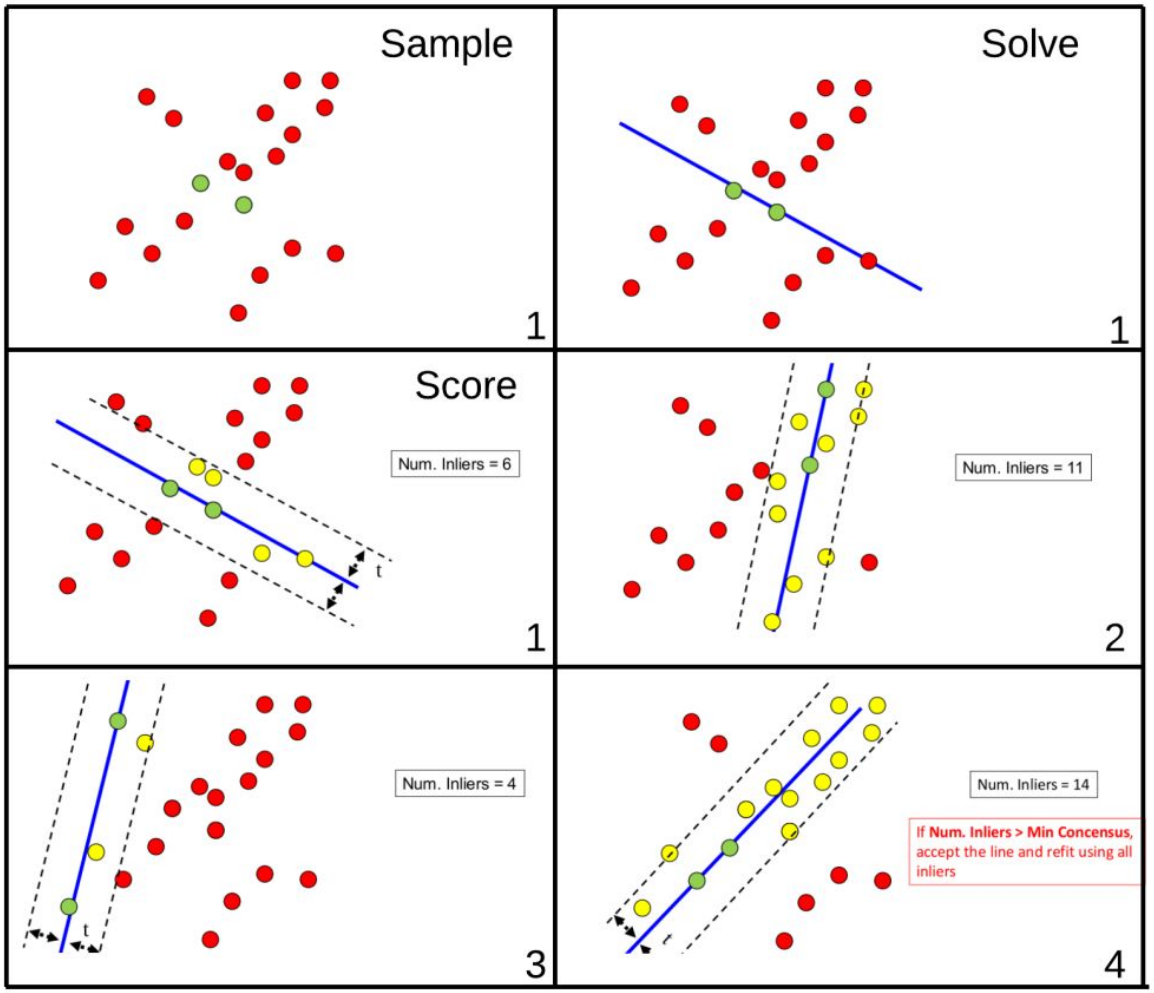
RANdom SAmple Consensus

RANSAC loop:

1. Randomly select a seed group of points on which to base transformation estimate (e.g., a group of matches)
2. Compute transformation from seed group
3. Find inliers to this transformation
4. If the number of inliers is sufficiently large, re-compute least-squares estimate of transformation on all of the inliers

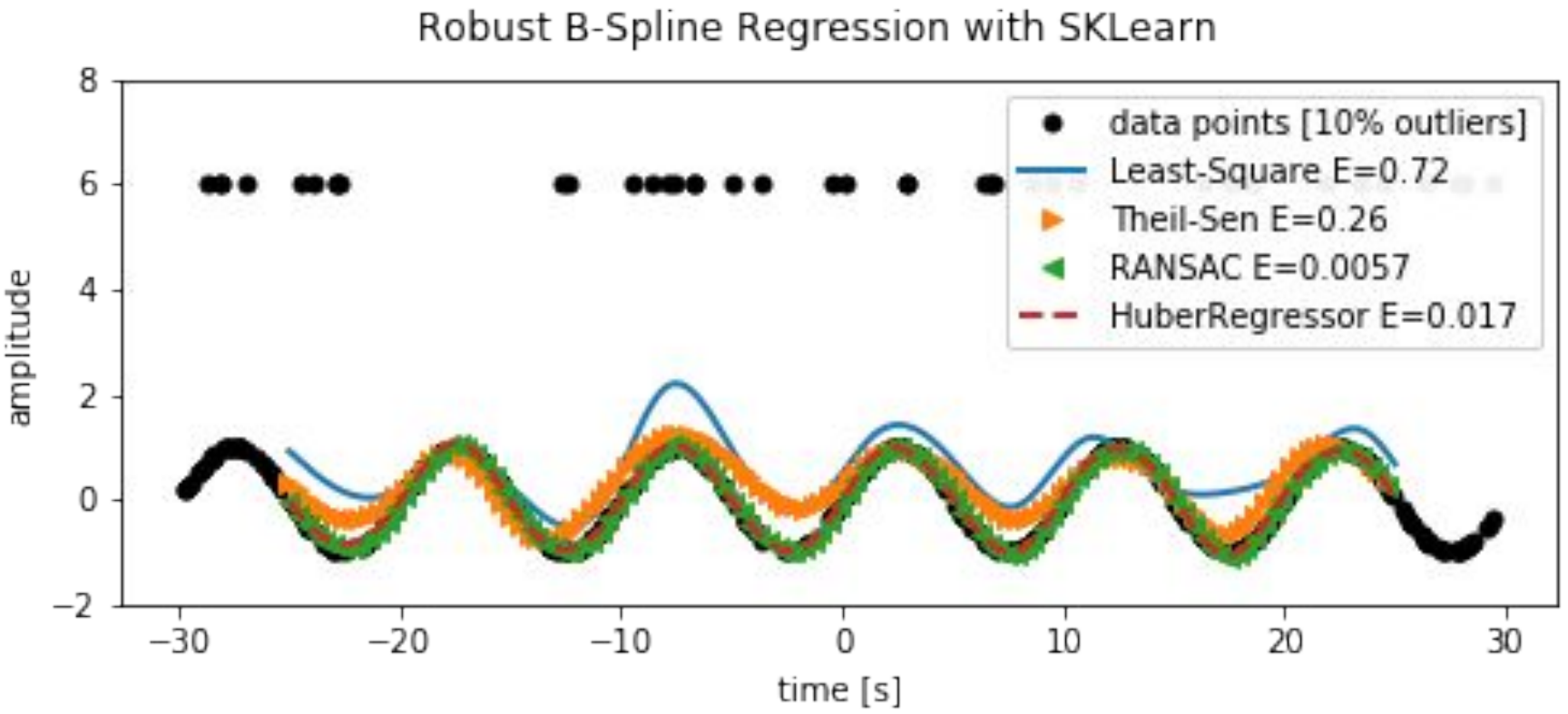
Keep the transformation with the largest number of inliers

RANSAC algorithm



Source: F. Moreno

RANSAC algorithm



Source: [this gihub gist]